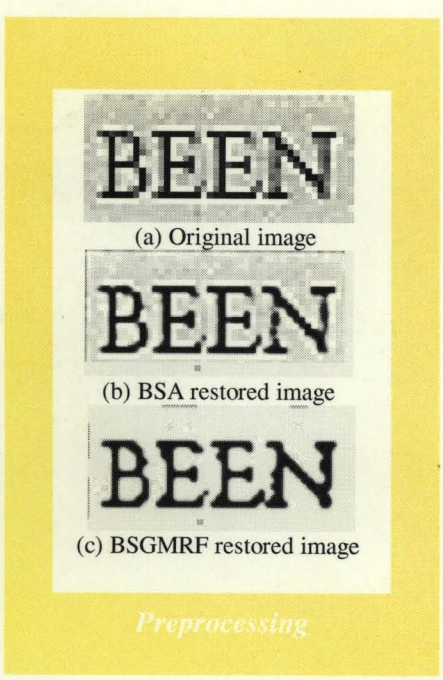
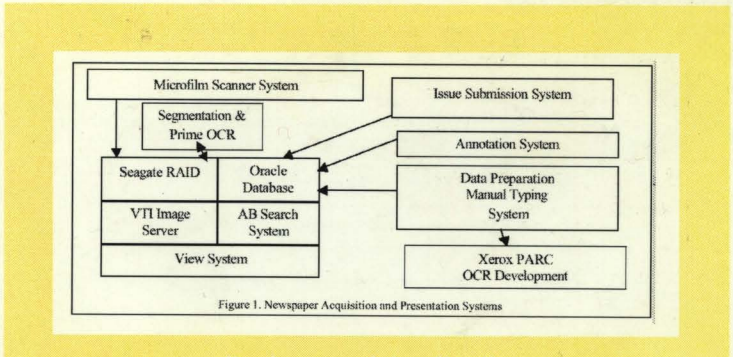
Proceedings

2001 Symposium on Document Image Understanding Technology



Preprocessing

Multimedia



Projects and Applications

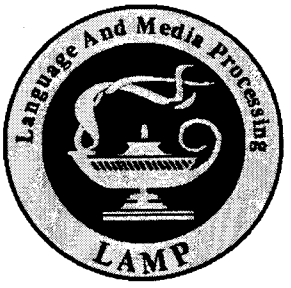
April 23-25, 2001
Sheraton Columbia Hotel
Columbia, Maryland

Proceedings

SDIUT01

The 2001 Symposium on Document Image Understanding Technology

Sheraton Columbia Hotel
Columbia, Maryland
April 23-25, 2001



Sponsored by: The United States Department of Defense
Organized by: The Laboratory for Language And Media Processing
Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742

A limited number of additional copies of these proceedings are available for $50 from:
University of Maryland
Institute for Advanced Computer Studies
College Park, MD 20742
Phone (301) 405-6444
Fax: (301) 314-9115
Email: sdiut01@lamp.cfar.umd.edu

Table of Contents

SESSION 3 PROJECTS AND APPLICATIONS

SESSION 4 PERFORMANCE EVALUATION

SESSION 5 FOREIGN LANGUAGE INFORMATION RETRIEVAL

SESSION 6 PAGE ANALYSIS AND CLASSIFICATION

SESSION 7 INDEXING AND RETRIEVAL

SESSION 8 TEXT RECOGNIZING AND PAGE ANALYSIS

ADDITIONAL SUBMISSIONS

Message from the Organizers

Welcome to the fourth bi-annual Symposium on Document Image Understanding Technologies (SDIUT). Over the past decade this symposium and its preceding Document Image Understanding Workshops have attempted to bring together researchers and research sponsors from government, academia and industry to explore trends in document image analysis research and to identify the areas of primary interest in the field. We have seen many topics addressed at many different levels, from OCR and page segmentation to image matching, indexing and retrieval. The progress of the field is evident in the evolution of specialized techniques and their applications to new and interesting problems. It is clear, however, that many challenges remain, especially in dealing with highly degraded and hand-written text, complex page layouts, and multilingual documents. An ever-expanding number of domains and applications are serving to drive interest in these problems and to bring new participants to the field. We hope that this forum can provide a means for sharing new ideas and defining directions for future work.

These Proceedings are intended to present a snapshot of the research and development activities that are of most interest to our government sponsors. This year, we have over 27 presentations, posters and demos from a number of different government agencies, academic institutions and corporations. We have seen an increase over the past several years in two areas in particular. The first is evaluation. It is clear that many approaches have evolved to the point where they are useful in a variety of applications. We need effective ways to evaluate the results of different algorithms in different domains so that we can determine if a given approach satisfies a given set of requirements, so that we can choose the best approach for a given problem and identify the areas which need improvement. A second area is multilingual document image analysis. In an ever-shrinking world, language continues to be a barrier that is not easily crossed. The analysis of multilingual document images is essential to follow-on processes such as indexing and retrieval or machine translation. Both of these areas are being addressed in detail this year.

For those of you who are participating in this year's symposium, we hope you enjoy the presentations and take an active part in the discussions, which develop throughout the meeting. For those who are reading these Proceedings, we encourage you to follow up with the authors and start dialogs about the problems you are trying to address. We feel that this symposium is best viewed as a catalyst for more in-depth offline discussions.

In closing, I would like to thank Ms. Denise Best for her endless hours of work on the facilities, registration, Proceedings and travel arrangements for our participants. As we all know, the amount of work that goes on behind the scenes to make such an event run smoothly is tremendous, and she has done a wonderful job.

Thank you for your participation.

The SDIUT '01 Organizers and Sponsors.

Keynote Speakers

Document Appliances in Practice

Kurt Piersol
Ricoh Silicon Valley, Inc.

Abstract

Ricoh has been creating document appliances, a class of device designed for very low maintenance users. These dedicated network devices are characterized by unattended operation, strict stability requirements, long-term high availability, a wide range of document inputs, and critical need for document security and recoverability.

Various current limitations of OCR and other image understanding and categorization software, as they apply to document appliances, will also be described.

I'll also discuss some experiences with users of such systems (both products and our research prototypes), and how the limitations imposed by the appliance idea, and the automatic capture of documents, often conflict with user expectations about document management. Unexpected uses arise as well, which point to areas where further research may prove useful.

Biographical Sketch

Kurt Piersol is the Chief Technologist at Ricoh Silicon Valley, Inc. He is the primary designer of eCabinet, a networked document appliance. Previously, Kurt worked at Apple Computer, Inc, where he served as the lead human interface designer on MacOS X, designed the AppleScript scripting language, the AppleEvents distributed computing system, and worked on other projects such as Hypercard, OpenDoc, the Macintosh Finder, and the system toolbox. Before Apple, Kurt worked at Xerox on some of the first commercial applications of object-oriented computing, as well as some of the first commercially available GUI based systems. He is a graduate of the University of Louisville's Speed School of Engineering.

Document Appliances in Practice

Observations from deployment of a new kind of document system

Kurt Piersol
Chief Technologist, Ricoh Silicon Valley Inc.
kpiersol@rsv.ricoh.com

Abstract

For the last few years, Ricoh has been developing *document appliances,* devices which can unconsciously capture documents, archive them, and make them available over a network. There have been surprising obstacles to their adoption, based on the preconceptions of users and the work patterns to which they have become accustomed. It appears that the customary ways of talking and thinking about documents and their management pose significant barriers to the use of this kind of device.

Introduction

Ricoh has been developing a new sort of product, called a *document appliance*. These devices, simply put, automatically archive many of the documents appearing in an office, capturing them as a side effect of normal work tasks. The documents are then made searchable and available through a web browser. Most people, when they hear of such devices, believe that they provide a significant value. However, there are surprising obstacles which arise as the devices are deployed, which point out interesting research and development problems.

Network appliances

The term network appliance has been used to describe a very wide array of devices. For the purposes of this paper, let's suggest that network appliances are dedicated hardware & software combinations which have a well defined purpose and which are attached directly to a network. For instance, a router might well be considered a network appliance, but a personal computer would not. A switch, a network attached storage device, a wireless access point, or a network attached webcam would all fit into the definition, although a general purpose computer performing the same functions would not. An IP phone is a network appliance, a telephony enabled PC is not.

Appliances may be arbitrarily hard to install, but tend to require little maintenance or administration once in place. This is quite similar to the situation of various home appliances. Refrigerators, stoves, and home theater systems are all fairly difficult to install, but most people would refer to them as appliances.

Another characteristic of an appliance is that it a user, in general, need not make extensive preparation in order to use one. Also, long-term discipline is not required to get benefit from the appliance. One needn't activate a food extraction protocol in order to obtain a soda from a refrigerator. Using a stove is as simple as activating the burner and placing a pan atop it.

A document appliance

Ricoh has been working to extend the idea of network appliances into a new area, that of document storage and retrieval. A document

appliance would be a network attached device which accepts documents from a wide array of sources, and then makes them available in a simple way to users.

Normal document management techniques involve having a person consciously insert documents into a repository, attach keywords or other content information, and choose a location in which the document will reside. Security restrictions are also placed on the document.

Such techniques clearly violate several of the principles of appliance design. Significant preparation and long term discipline are required to get benefits from such systems. Constant administration and maintenance are required. Typically, the software for such systems is installed on a general purpose operating system, and the maintenance and administration overhead of such systems is added to the cost of the total system.

Ricoh's attempts to address these issues resulted in a research project known as the Infinite Memory Multifunction Machine, or IM3 (pronounced 'I-M-cubed'). This prototype system was attached to receive documents from a copier which had been modified to retain images of every document which was copied. The goal was the *unconscious capture* of the copied documents. The images were then processed using optical character recognition, and then indexed using full text indexing techniques. The documents could be retrieved from any web browser, using a simple search interface.

The prototype was very successful, and provided valuable data to product development process which resulted in the Ricoh eCabinet product. This product extended the original idea to more kinds of device, including scanners, network printers, fax machines, and regular PCs and email servers. Security and backup, relatively minor considerations of the IM3 project, were significantly extended and refined in eCabinet.

Observations

As we have deployed document appliances in the field, a number of interesting facts have emerged. Examined independently, each observations seems relatively innocuous. Taken in combination, though, the observations have interesting implications. The observations themselves are based on watching customer focus groups, conversations with customers at installed sites, and field service calls. As such, the observations are not at all quantitative. However, the author and several of his colleagues have made similar observations over a period of three years, and these may prove fruitful areas to do some quantitative research in the future.

People don't have as many documents as they imagine.

Research on the IM3, as well as field deployments of eCabinets, have shown that a typical person produces about 8 documents each day, aside from email. This number is relatively consistent across different kinds of user in different facilities. A 20 person work group produces about 50,000 documents per year, including every copy, fax, scan, and print job. In general, users are extremely skeptical of this number, and consistently overestimate their requirements.

People don't trust search engines to find specific documents

Most users we talk to about eCabinets are familiar with search engines, such as Google, Excite, AltaVista, and so on. They are generally convinced that they can find anything they need on the web by using such

search engines. However, almost no one believes that this same technology will allow them to find their own documents. This appears to be based on an expectation that the set of documents produced in their own office closely resembles the mix which is produced by the internet at large. Thus, they imagine huge numbers of 'false positives' to wade through to find a specific document.

People believe that they can manage their own documents

Most people we talk to have a persistent belief that in the near future, they will somehow free enough time to thoroughly organize all of the important documents in their life, and will afterwards be disciplined enough to maintain this careful organization.

People want to hide documents

People have a strong desire to have documents secured from prying eyes. In general, they are convinced that *most* of their documents are quite private, and should be hidden from unauthorized users. If asked for details about how this should be done, almost no one can define a rule which would decide which documents were interesting and private.

People do not believe that other people can effectively manage documents

When asked, most persons will tell you that most documents are lost because they are put in the wrong place, by someone else. If asked for details, they generally note that particular individuals are the source of most problems, and that these individuals simply 'cannot follow the system'. If pressed further, most will admit that the individuals are probably capable of implementing the organizational system, but do not agree on the necessity of compliance.

People believe that documents can be automatically categorized

If asked whether it is possible to automatically 'file documents into the correct place', most users say that it is. They have never seen such software, but they believe that it can be done.

People are not willing to expend effort in advance to make documents easier to find later

This is a well known trade-off that a fairly small percentage of the population can successfully make over time. It applies to documents as it applies to health matters, savings, and any number of other areas requiring long-term disciplines.

People generally confuse location with categorization

Librarians have long noted that the location of a document has only a relatively small correlation with the categories into which a document falls. Cross reference systems and taggings of various kinds have provided solutions to the worst of these problems. The notion of applying tags to documents appears to be unsatisfying to many users, though. It is difficult to coax many people to speak in terms of tagging documents with category information. Instead, they prefer to think in terms of location. Even worse, they have trouble even talking about documents being in more than one location.

OCR engines often miss particularly important data

OCR engines have interesting properties which are not necessarily well suited to the needs of a document appliance. In particular, the most important terms for searching in

business documents are proper names, digit strings, and serial numbers. OCR engines often trade off accuracy on these terms for general word accuracy. For example, the dictionary checking often done by OCR engines, where low confidence words are checked against a dictionary, often introduces errors into proper names which might otherwise have been recognized. Often, the errors introduced are sufficient to defeat simple transposition checks or 1 character variance checks. In an appliance, where human interaction with each document is proscribed, this represents a serious obstacle.

Full text summarization is often confusing

Automatic summarization of documents, using relevance ranking, does not seem to help users recognize the document. After attempting to use this summarization, we switched to a short excerpt from the front of the document. However, this method has a serious disadvantage when standardized forms are used, because the initial excerpt is often exactly the same from one form to the next. In an appliance, this presents difficulties because the summaries are not generated by humans. No obvious method of detecting summarization patterns across large document sets, which is both computationally tractable and not prone to bias from the initial document set, is known to us.

Combinations

Now that we've listed our innocuous facts, let's take a look at how they interact. Several interesting sets of conflicts arise, some based on inconsistent belief sets, some based on technical limitations.

The conflict about filing

Everyone believes that they can manage their own documents, but that others cannot. A common story is one our team has come to call the 'life cycle of the file server'. In the story, the file server is nicely organized with a few categories when it is first brought on line. As time passes, people create new categories because they are either unaware of the existing ones or cannot decide where to put documents in the existing structure. Eventually, there are so many aspects to the structure that no one can handle it, and the file server becomes rapidly disorganized. Once this occurs, usage drops off as people move their working documents to a new, well organized file server. After a certain period of time, the unused file server is archived to a backup tape, and the now unused contents are erased and the space is reused for a new, pristine file server setup. The backup tape is placed on a shelf, and its location forgotten.

Historians may recognize this as a classic 'tragedy of the commons' situation. Many people regard it as unspoken common knowledge. People regard it as almost inevitable. Filing, as an organizing technique, doesn't scale very well. If multiple people are involved, the required discipline tends to exceed human capacity.

This 'story' may also suggest that, fundamentally, people cannot agree on how documents ought to be categorized. The fact that new categories inevitably arise is an integral part of the story. This is a very interesting notion, because it suggests that users unconsciously understand something they will not directly admit, that there is no 'correct' place to put documents.

Interestingly, many people appear to believe that machines will be able to do this where other humans cannot. This would suggest that most users have an expectation that computers are controllable in ways that people are not. The possibility that multiple machines might

have conflicting rules does not appear to occur to them.

The conflict about security

People seem to believe that it is possible to secure documents automatically, even though they cannot themselves imagine how it might be done. This idea may be related to the widely held notion mentioned above, that computer systems are inherently controllable.

It is interesting to note that many people seem to believe that the security problem is one of categorization. They constantly talk in terms of the device 'understanding' the document and successfully placing it into the 'correct' category. Categories of documents appear to be the preferred method of expressing ideas about security. One seldom hears a user talking about the security needs of an individual document. Users can usually decide immediately if another person should see a document, but may have a hard time deciding into which category it falls. Nonetheless, almost all speech about security involves document categories, not individual documents.

Appliances vs. direct management

Appliance operation precludes some kinds of cleanup and human interaction which are integral parts of existing document management technique. It may not be feasible to build appliances which can be used as replacements for traditional document management systems. Document understanding technology might be able to remedy some of these issues, but it currently does not address them. For example, the observations noted above with document summarization, OCR accuracy tradeoffs, and automatic categorization are all problems in this area.

Difficult but not insoluble problems

These conflicts suggest a need for further work. I would make the assertion that Ricoh, at least, has not produced a metaphor for discussing document appliances that makes sense to ordinary people. Further, as appliance-like computing becomes more prevalent, this will become a problem for other companies and organizations as well. There are fundamental problems with how we speak and think about documents, which may well preclude adoption in certain environments.

The observations indicate that the new metaphor, whatever it is, needs to resolve issues about categorization, security, and group interactions. Filing, tagging, and simple full text retrieval all appear to have problems with either internal consistency or with user satisfaction.

This is not to say that there are no technical problems to be overcome. Several issues examined in this paper may have feasible technical solutions. Certainly, ordinary users believe that they *ought* to be solvable by technical means.

In the final analysis, thinking and talking about understanding documents implies that we have something to do with that understanding. It is not yet clear that we have a working model of the kinds of understanding that will solve user problems.

Filling the Shelves of the Digital Library: A mass of books for a mass of users

Maria Bonn
University of Michigan

The Making of America is a digital library of about 10,000 19th century volumes (3,000,000 pages) that have been converted to digital form. The conversion of the MoA volumes is both a preservation effort and an attempt to make the content of these volumes more accessible to a wide variety of users. This talk will discuss the library principles behind MoA, the low-level conversion treatment applied to the volumes, and the possibilities for further and more sophisticated conversion. It will also report on uses and users of MoA and speculate on some ways in which document analysis technologies could better meet library needs.

Biographical Sketch

I am the Head of the Scholarly Publishing Office at the University of Michigan University Library, an office charged with exploring and developing the possibilities for electronic publishing in an academic setting. I have worked for several years with the UM Digital Library Initiative, an organization that works extensively with encoded text, OCR and other tools that promote text retrieval. I have just finished two years as the project manager on the Making of America IV, a project that digitized and put online about 7500 19th Century books in an eighteen-month period.

Preprocessing

Line by Line Script Identification
Document Image Understanding Technology 2001

Carson Cumbee
Department of Defense

Abstract

A method is introduced to quickly recognize the script of a line of machine printed text automatically segmented from a document image. This method normalizes the line of text, turns the image into a series of vectors, quantizes those vectors and performs an n-gram analysis on the quantized results. It takes roughly two seconds for the analysis of a full page of text and is 86.6% accurate in determining the nature of each line of text for the given test set of 13 scripts.

1.) Introduction

Script identification is a useful preprocessing step in automatic document recognition. Most optical character recognition (OCR) technologies are trained to recognize one set of scripts, so if it becomes necessary to OCR a set of unknown documents with many different scripts, script identification can successfully route those documents to the appropriate OCR technologies. Several techniques have been presented to deal with whole page classification of scripts. Hochberg et al. [1] , used a template matching algorithm based on the connected components found on a page of text. The templates were formed by clustering together similar components and assigning a reliability number to each template. Components in an unknown document would then be compared to this set, and the script which accumulated the highest score would then be declared the script for the page. Spitz [2] used topological features (principally concavity location) derived from components to group scripts into an Asian class and a European class, and used a further set of features to determine the language of the document. Recently [3], Verizon/BBN has demonstrated the utility of considering document images as a collection of lines of text. They adapted their speech recognition system to OCR by considering a line of text as a sequence of vectors, and applying hidden markov models (HMM) to analyze the image's contents.

Many documents also contain a mix of scripts, typically a foreign language and English (Latin) script. In these more complicated images, it is useful to find regions of text that contain only one type of script. This paper outlines such a method based on classifying the script of each individual line of text in a machine printed document.

2.) Line detection

One of the core issues with line by line script id is the detection of the lines in a document image. The results in this paper were derived from a technique that links connected components in an image to their nearest horizontal neighbor, and then iteratively merges these segments until all of the connected components on the line are found. This technique is proprietary but is similar to the technique found in Liang et al. [4].

3) Normalization

Once the region of the image with the line of text is cropped out of the image, it is deskewed by finding the best fit line through the pixels. After deskewing the line, a horizontal histogram is produced, and only the middle 95% of this projection is kept as figures 1 and 2 illustrate. This is to help normalize the line of text against spurious strokes, and help align the baseline of the scripts. Experience has shown that this recognition technique is very sensitive to the vertical alignment of text.

Figure 1: Horizontal histogram

Figure 2: Cropped image

The last normalization procedure is to downsample the image to make it 8 grayscale pixels tall, and to retain the aspect ratio. Each column of pixels is now treated as a vector, and the series of vectors that constitutes the image is the starting point for the script identification. Figure 3 shows the downsampled image where each pixel column should be thought of as a vector with 8 elements which are the pixel values.

Figure 3: Downsampled image

4.) Quantization

The set of vectors that represent the line are vector quantized, so a line of length N vectors would now be represented as N quantization values. The results described in this paper are for quantization to 64 unique codes. Quantization itself is done by finding which of 64 quantization vectors is closest to the unknown vector (by euclidean distance). Figure 4 illustrates this process.

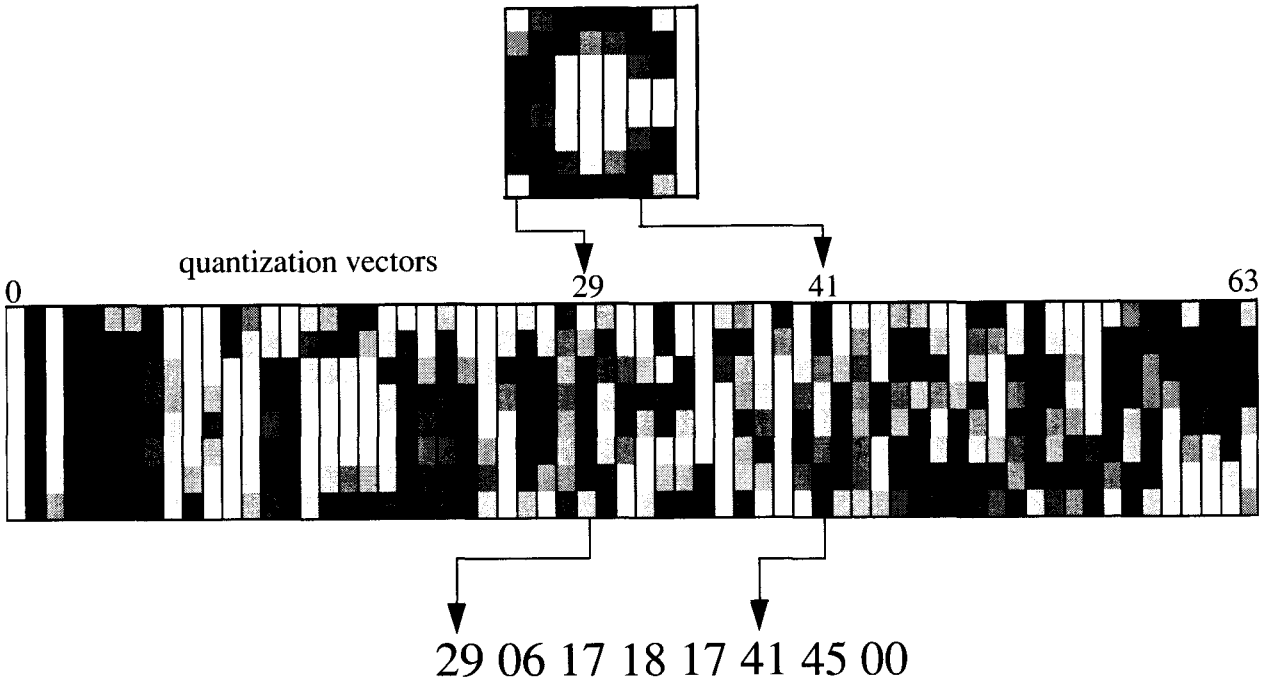


Figure 4: coded values from assigning closest quantization vector to each column of pixels

These quantization vectors were created by taking training data and using k-means clustering to find 64 centroids. Figure 5 shows the original downsampled image on top and the bottom image is the recontruction by the quantization vectors.

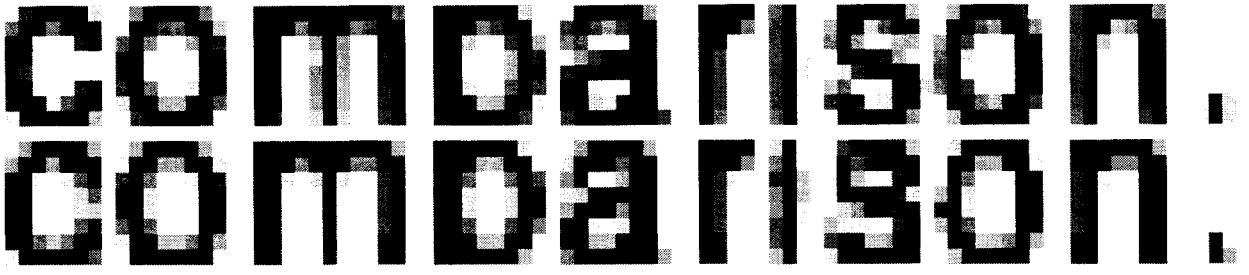


Figure 5: Original downsampled image top, compressed image on bottom

5.) N-Gram analysis

The sequence of codes that represent each line of text are now considered as bytes, where each byte contains the code's value from 0 to 63. This byte sequence is scored by analyzing 3-8 byte long windows. These n-grams slide along the sequence of bytes one byte at a time and increment a weight found in a hash table. Scores are accumulated by a weight associated with each n-gram and each language type. The script with the highest accumulation is the classification result. This can be expressed as **argmax(Wg)** where W is the weight matrix (each column representing an n-

gram and each row representing a script type) and g is the n-gram vector from a given line, where each element is the observed n-gram count.

Figure 6 shows "glyphs" created by the observed 8-grams in the Latin text lines. These were created by collecting the 8 quantization vectors together for observed 8 gram sequences. There are also corresponding images for 7 through 3-grams. Given an unknown line of text each n-gram is matched in near linear time to one of the n-grams representing these glyphs and accumulates weights represented by this n-gram (all of the weights in figure 6 are Latin 1.0).

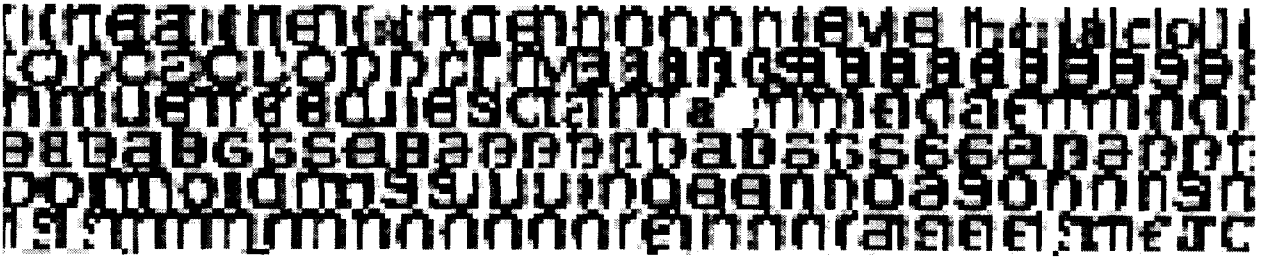


Figure 6: A sampling of the glyphs represented by the 8-grams from Latin

There are various methods of setting the weights in W. The accumulation operation is essentially a linear operation and hence several different weighting schemes can be used (linear support vector machines, log-odds weighting, trigonometric methods [5], and information based methods [6]). The following simple formula was used to set weights for this experiment .

$$W_j = \frac{\frac{1}{N_j}\sum_i G_{ij}}{\sum_j \left(\frac{1}{N_j}\sum_i G_{ij}\right)}$$

W_j is the n-gram weight for script j, G_{ij} is the normalized frequency of n-gram G in line i of script j, N_j is the total number of lines of script j. All of these weights are between 0 and 1. The majority of them are in fact 1. This happens when the given n-gram is observed in only one script. It should be noted that only n-grams occurring in more than one line of a particular script were used. While this method has not been shown to improve results, it does cut memory usage by a factor of 10 with only negligible effect on accuracy.

6.) Results

The average overall line by line accuracy was 86.6 %. The average page took 2 seconds to process on a 866 Mhz. Pentium 3. An additional experiment was performed for whole page script classification. This was done by concatenating all of the lines on a page. There were no whole page errors on this data set. Table 1 shows a confusion matrix with percent accuracy. As expected Cyrillic and Latin have strong confusions and Chinese and Japanese have strong confusions. It should be noted that many characters overlap between these scripts and a single line of text may in fact contain only characters found in the other script.

The algorithm was run against images collected at Los Alamos National Laboratory. This set contained images in the 13 scripts shown in table 1. All but one of the 195 train/test documents

were used. One document was thrown out because line finding could not find any lines on it. In the Los Alamos whole page classification experiment [1], there were also no errors. For purposes of this paper, the corpus was divided into two sets of 97 pages each, one for training and one for testing, then vice versa. The results of these two independent tests were then averaged and presented here. The accuracy of line finding on these images was not checked by hand. It is not certain that each line found on a page was indeed a line of text but whatever image came out of the line finding process was included in the training and testing. This ambiguity is a source of error in these experiments.

Table 1: Confusion matrix

Percent accuracy, normalized horizontally													total lines	script
99	0	0	0	0	0	0	0	0	0	0	0	0	1112	Amharic
3	87	0	3	0	1	1	1	2	2	1	0	0	598	Arabic
1	1	93	0	0	1	0	0	0	1	0	1	0	839	Armenian
1	2	0	85	1	0	1	1	1	2	2	0	2	723	Burmese
0	1	0	1	80	1	1	0	0	9	6	0	0	964	Chinese
1	0	1	0	0	90	0	1	0	0	0	6	1	998	Cyrillic
0	2	0	3	0	0	89	0	1	2	1	0	0	856	Devanagari
1	0	1	1	1	3	2	84	0	0	1	5	1	923	Greek
4	1	0	0	0	0	0	0	90	2	1	0	1	1148	Hebrew
0	2	0	1	3	1	0	0	1	88	3	0	0	733	Japanese
1	2	0	3	4	1	1	0	2	13	72	0	1	811	Korean
0	0	1	1	0	6	0	0	0	0	0	92	0	855	Latin
6	2	2	1	0	2	0	1	5	2	1	0	77	1288	Thai

Table 2 shows that the longer the line is the more likely it is to be classified correctly. Short lines are often not deskewed correctly, are poorly aligned, and do not contain as much information as longer lines. The line lengths are for the downsampled image in which a typical character is 8

pixels wide. If a segmented line has more than 25 characters this method is over 96 percent accurate.

Table 2: Accuracy Versus Line Length

line length range in pixels	percent of all lines	percent accuracy
13 - 50	8.04	52.03
51 - 100	13.50	71.93
101 - 201	25.33	90.27
201 - 300	23.99	96.55
301 - 1500	28.04	96.37

7.) Future work

This method may be extended to the identification of language, font, and handwriting script identification. It could also be applied to speech applications such as language and speaker identification. It is not obvious what exactly is the best weighting method. The author has developed some iterative optimization techniques that are useful for adapting the weights to errors, but they have not shown improvements in this experiment.

Feature vector selection is also important and should be reinvestigated. The crude features used for this report are sensitive to the vertical alignment step described in section 3. It may be beneficial to use a finer downsampling method, especially in extensions to other types of recognition.

Experience has taught that using more quantization levels doesn't help script identification significantly, but it would probably help in other uses. The biggest obstacle this method faces is in the actual line finding and vertical alignment. Noise and graphics cause problems for the line finding method used for this paper and columns of text also frequently cause problems. The horizontal histogram for short lines of text are often poor indicators of the baseline.

Acknowledgements

The author is grateful to Becker Drane at DoD for his document image processing algorithms, Paul England of Integrated Computer Concepts for his image processing libraries, and Lynn Golebiowski of Booz Allen Hamilton for her line finding algorithm. The author would also like to thank Steve Dennis and Thomas Drayer of DoD for technical direction.

References

[1] Hochberg, J., Kelly, P., Thomas, T., Kerns, L. (1997) Automatic Script Identification From Document Images Using Cluster-Based Templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 19, No. 2

[2] Spitz, A. L. (1997) Determination of the Script and Language Content of Document Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 19, No. 3

[3] Bazzi, I., Schwartz, R., Makhoul, J., (1999) An Omnifont Open-Vocabulary OCR System for English and Arabic. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 6

[4] Liang, J., Phillips, I., Haralick, R., (1999) A Unified Approach for Document Structure Analysis and its Application to Text-line Extraction. *Proceedings of the Symposium on Document Image Understanding Technology* (SDUIT '99), April 14-16, Annapolis, Maryland, pp. 32-41

[5] Damaschek, M. (1995) Gauging similarity with n-grams: Language-independent categorization of text. *Science*, Vol. 267, Februrary 10

[6] Shaner, R. (1999) US Patent: *Method of Identyfing data type and locating in a file* (5,991,714)

Gaussian Model-based Image Binarization for Text Extraction

Thomas H. Drayer, Ph.D.

Department of Defense, Fort Meade, MD 20755, USA

Abstract

This paper describes a new technique for converting gray scale document images into binary images for Optical Character Recognition (OCR). The new algorithm computes a histogram of all pixel intensities, which is then modeled with a Sum-of-Gaussian (SOG) representation. Domain knowledge of image quantization and spatial subsampling is used to define methods for estimating the number and parameters of the underlying Gaussian models. This unique algorithm provides both image binarization and spatial resolution expansion in a single integrated process. A goal-driven evaluation is used to measure performance of the new algorithm. Binarized images are input to OCR software to perform text extraction. The OCR character accuracy obtained using this method is compared to the accuracy obtained by binarizing and expanding images using existing algorithms from the current literature.

1. Introduction

Image binarization is the process of transforming a multi-level input image to a new bi-level image; one in which each pixel intensity is represented by a single bit variable with a value of either 0 or 1. Multi-level gray scale images may be represented by a single discrete value, with the range of possible values determined by the number of bits used for representation. Alternatively, multi-level images may be represented by a vector of values, such as the 3-dimensional vector typically used to represent pixel intensities in color images. This paper considers binarization algorithms that transform digital images with a discrete set of pixel values (n-bit precision $\rightarrow 2^n$ values) into a corresponding output image where each pixel is represented by a single bit value. In this paper we will consider discrete gray scale images only, the algorithms may be extended to process color pixel values in future work.

The intended use of this algorithm is the extraction of text from gray scale images for input to automated OCR software programs. Most OCR software programs only have the ability to process binary images, however some newer commercial products process gray scale or color input images [1]. Converting from gray scale or color to binary allows input to a broader range of OCR programs and may provide an alternative to a less effective binarization process included with a specific OCR software program.

Images of text should contain a significant percentage of pixel values that correspond to regions of textual characters in the image scene and other pixel values that correspond to regions of background in the image scene. For the domain of

document images, the background regions correspond to the near-uniform color of the document paper and the text regions correspond to regions where ink has been added to the page. An essential observation is that when the spatial resolution of an image is low, a significant percentage of input image pixel regions may correspond to *both* text *and* background along the borders of text characters.

2. Background

Several techniques have been used to perform image binarization, and a survey and evaluation of prior work has been provided by Trier [2]. Techniques intended for rendering image graphics for human viewing, such as those that employ dithering [3], are not appropriate for the intended use of this algorithm and will not be considered further.

Prior work on image binarization for text extraction can be divided into two classes of algorithms. Algorithms in the first class use spatial derivative information to classify output pixels as either text or background. These algorithms determine rising and falling edge pixels in the input image using the spatial derivatives, then classify all pixels between the falling and rising edges as text [4]. These techniques only work well when there is sufficient spatial resolution and image contrast.

Algorithms in second class determine a direct transformation of the input gray scale pixel values to the output binary value. These methods typically calculate statistics of the image in the form of a histogram of gray scale pixel intensities. This histogram is used as a model of the Probability Distribution Function (PDF) for gray scale pixel intensity values. The histogram may be calculated either globally [5] or within local regions of the image [6, 7, 8].

Typically, a threshold is selected such that all pixels in the input image with gray scale values at or above the threshold are defined to be background pixels in the output image and pixels below the threshold are determined to be text pixels. Figure 1 illustrates the application of a threshold ($Tval$ = 128) to transform gray scale pixel values to binary pixel values in the output image.

The algorithm presented in this paper is most closely related to algorithms in the second class, so the most significant of these are presented in the following:

1. Bernsen's Method [6]: This algorithm finds the maximum pixel value, I_H, and minimum pixel value, I_L, within a subregion of the image. A threshold value is computed as follows:

$$\left.\begin{array}{l} Tval = (I_H - I_L)/2 \\ Tval = I_L \end{array}\right\} \begin{array}{l} if\,(I_H - I_L) > l \\ else \end{array} \quad (1)$$

   where the value of l defines a maximum tolerance on the variation in pixel values (thus indicating the presence of text). Otherwise, the threshold is set to the minimum to assign all input pixels the value for background.

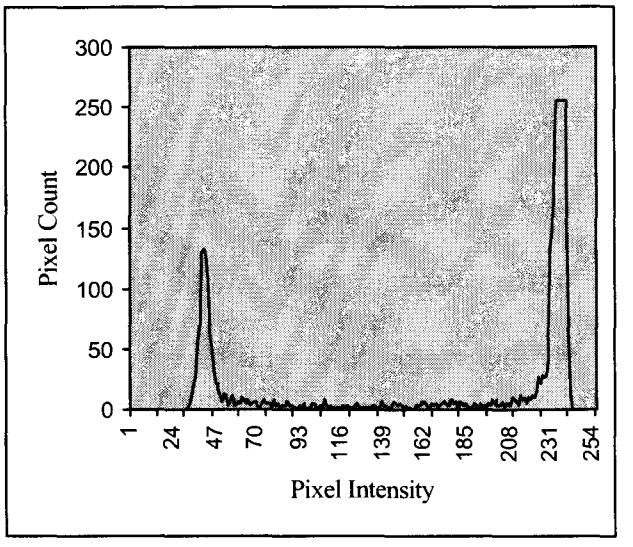
2. Niblack's Method [7]: This algorithm calculates the mean, μ, and standard deviation, σ, of pixel values within a subregion of the image. A threshold value is computed as follows:

$$Tval = \mu + k\sigma \quad (2)$$

   Values of –0.2 for k and a subregion size of 15x15 are suggested in [2].

(a) Original Image

(b) Binary Image



(c) Image Histogram

Figure 1. Binarization of image with bi-modal histogram using a threshold of 128

3. Chow and Keneko's method [8]: This algorithm tests the histogram from non-overlapping input image subregions for bi-modality (two dominant peaks as in Figure 1) and models the histogram with the sum of two Gaussian distributions. A threshold is computed for all regions that are determined to be bi-modal. For regions that are not bi-modal, a threshold is interpolated from the thresholds of surrounding bi-modal regions. The individual thresholds are smoothed to eliminate outliers.

4. Taxt's Method [9]: This algorithm is similar to Chow and Kenko's algorithm as it attempts to approximate the histogram of non-overlapping image subregions with the sum of two Gaussian distributions. However, Taxt's method uses an Expectation-Maximization (EM) algorithm to converge an initial guess of the Gaussian model parameters to the estimated solution and solves for the output binary pixel values using the quadratic Bayes' classifier.

The limitation of previous methods is that their model of two Gaussian distributions is inadequate for accurate description of the underlying physical process of gray scale image formation. One solution to this problem is to extend the representation by modeling the histogram with a larger number of Gaussian distributions.

3. The New Binarization Technique

This new algorithm is similar to the algorithms in class two as it attempts to model the histogram of pixel values with a sum of Gaussian distributions. The probability distribution function, $f(z|\phi)$ of the gray level intensity values of the input image (or a subregion of the image) is modeled by a sum of several individual

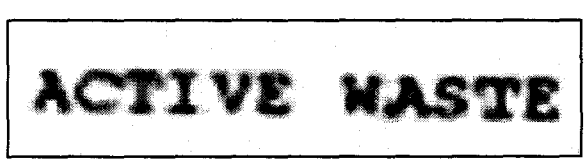
underlying Gaussian distribution functions as defined below:

$$f(z \mid \phi) = \sum_{k=1}^{m} \alpha_k f(z \mid \phi_k), \quad x \in \Re^d \qquad (3)$$

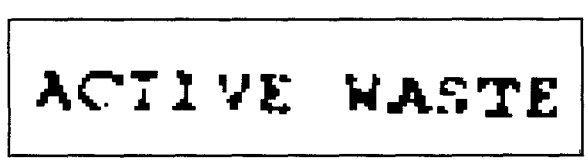
where, m and d are known positive integers for the number of Gaussian models and number of color channels respectively, the α_k are LaGrange multipliers, and $f(z/\phi_k)$ are the m individual underlying Gaussian distribution functions. Each of the Gaussian models is defined by its parameters ϕ_k. The gray level z at any location x,y in the input image is defined as $z = I(x,y)$.

Prior approaches [8, 9] attempt to model the PDF of gray scale images with the sum of only two Gaussian distributions (i.e. m = 2 and d = 1). One Gaussian distribution corresponds to the text regions and another to the distribution of pixel values of background regions. These previous methods only work well when the histogram is clearly bi-modal, as shown in Figure 1. However, individual gray scale pixel values do not always simply correspond to areas of background or text in images of low spatial resolution. Pixels at the borders of characters will correspond to regions of *both* text *and* background. In cases of low spatial resolution, the histogram is more closely represented by the image and uni-modal histogram shown in Figure 2.

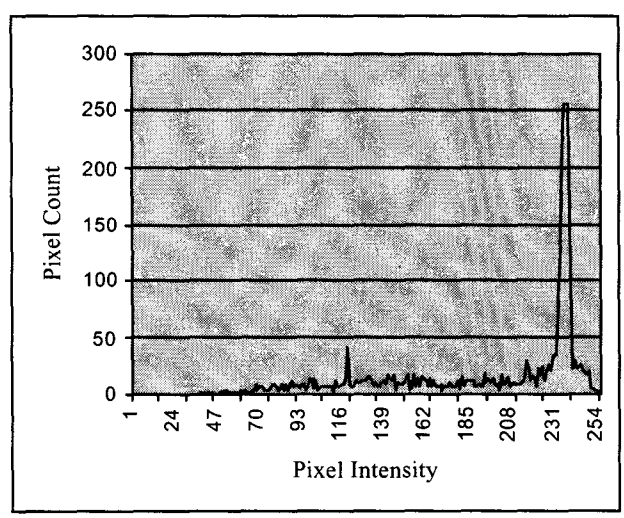
The new method differs from previous approaches by modeling the global image histogram with the sum of five Gaussian distributions (e.g. m = 5 in eqn. 3). The new method also doubles both the horizontal and vertical spatial resolution of the output image. Therefore, there are four corresponding binary pixels in a 2x2 region of the output image, $I'(x, y)$, for each gray scale pixel in the input image. Each of the five Gaussian models corresponds to the *number* of text pixels in the four new pixels of the spatial resolution expanded output image. There may be m = 0, 1, 2, 3, or 4 text pixels in the corresponding 2x2 region of the output image, and the distribution of each of these cases is modeled by a Gaussian distribution with parameters ϕ_k. The overall composite model probability distribution $f(z/\phi)$ is therefore the sum of the individual underlying Gaussian distributions $f(z/\phi_k)$ scaled by its LaGrange multiplier α_k.



(a) Original Image



(b) Binary Image



(c) Image Histogram

Figure 2. Binarization of image with uni-modal histogram using a threshold of 128

The task of estimating the parameters of five Gaussian distributions is computationally demanding. This requires the calculation of five separate means and variances. Additionally, four LaGrange multipliers are required. Prior methods use EM to perform the search for five parameters required for two Gaussian models [8]. This algorithm uses domain knowledge in two heuristic search methods to find the five Gaussian models used in the binarization process (or possibly provide an initial state for EM in future work). The two heuristic search methods are defined below.

A) Two-Peak Method

We assume that the background pixels create a dominant peak in the histogram. After finding the maximum value of the histogram, it is assumed that almost all of the points above this peak should belong to the Gaussian model for background pixel values. The histogram values above the dominant peak are used to calculate an estimate of the variance for the background Gaussian model. This is done by creating a new distribution by reflecting the values above the peak below the peak. The maximum value, mean, and variance of this distribution provides the model for the pure background ($f(z/\phi_0)$, the model for zero output text pixels and α_0, its LaGrange multiplier). The estimated model parameters are used to calculate the first Gaussian model, which is then subtracted from the total histogram to allow estimation of the remaining models.

To find the second peak, the remaining histogram is multiplied by an exponential function of the distance from the first peak, creating a weighted histogram to emphasize peaks that are farther away from the first peak. The search for a second maximum is restricted to points from the histogram value of zero up to one standard deviation below the dominant peak, as determined in the procedure above. The maximum of the remaining weighted histogram defines the mean and LaGrange multiplier of the second model ($f(z/\phi_4)$, the model for four output text pixels). The variance is found by reflecting the remaining values in the histogram below this second peak in an analogous manner to the method used to calculate the variance for the first peak above. As before, this model is subtracted from the total histogram for the remainder of the processing.

Having modeled and removed the pure background and text models, the remaining histogram is estimated by a uniform distribution. This uniform distribution is then modeled by three Gaussian distributions with means evenly spaced between the means of the upper and lower peaks. The variance and LaGrange multipliers for the three Gaussian distributions are assumed to be equal and are selected to force the sum of three Gaussians models to be equal to the uniform distribution at the peaks and at the midway points between the peaks.

B) One-Peak Method

The one-peak method begins by finding and modeling the main peak in the same manner as the two-peak method defined above. The remainder of the histogram is estimated by a uniform distribution. Lower bounds of the uniform distribution are determined by the first occurrence of a count above ½ the average value of the remaining

histogram. This uniform distribution is then modeled by four Gaussian distributions with means evenly spaced between the main peak and the lower bound of the uniform distribution. The standard deviation and LaGrange multipliers for the four Gaussian models are assumed to be equal and are selected to force the sum of the four Gaussian models to be equal to the uniform distribution at the peaks and at the midway points between the peaks.

The sum of the five individual Gaussian models found in the two heuristic search methods above provides two separate composite histogram models. Each of the two composite histograms is compared with the input histogram, and the method with the lower error is selected. The error is computed as the sum of absolute differences between the modeled and the input histogram at each gray scale level in the histogram.

The five underlying Gaussian distributions of the selected composite model correspond to the distribution functions of the five possible numbers of text pixels in the corresponding 2x2 region of the output image. Therefore, for an input gray scale pixel value $z = I(x,y)$, the model with the highest value at z determines the number of text pixel values in the four output pixel values as defined below:

$$N(x,y) = \underset{k=0}{\overset{4}{MAX}}(\alpha_k f(z,\phi_k)) \qquad (4)$$

A method is required to determine the location of the k text pixels in the 2x2 region of the output image, except in the trivial cases of four or zero output text pixels. This new method uses an estimate of the pixel value from the original image. Four linear predictors, one for each distinct location in the 2x2 array of output pixel values, are used to predict the text pixel locations. The four separate 3rd order, 2-dimensional predictors are defined as follows:

$$\hat{I}_{a,b}(x,y) = \frac{I(x+a,y) + .7 \cdot I(x+a,y+b) + I(x,y+b)}{3} \qquad (5)$$

The four predictors are defined for the possible combinations of $a = \pm 1; b = \pm 1$.

The final step is to combine the number of text pixels, k, with the four predicted values from Equation 5. The k predictors with the lowest value are assigned the bit value for text and the $4-k$ remaining pixels are assigned the background value. Figure 3 illustrates the use of these predictors to assign binary pixel values in the output image.

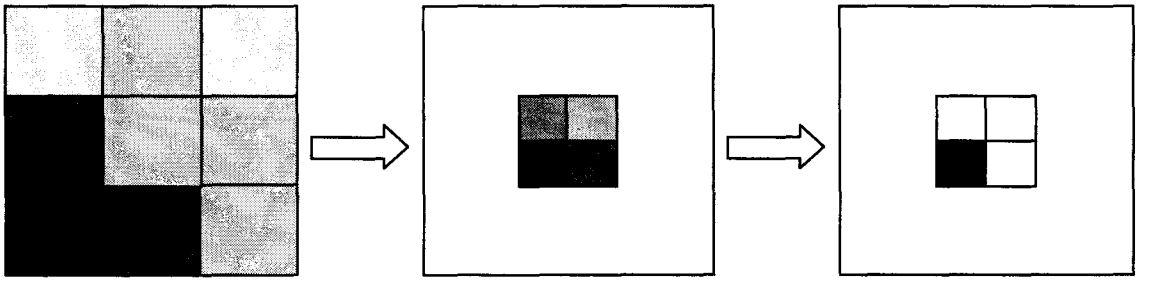
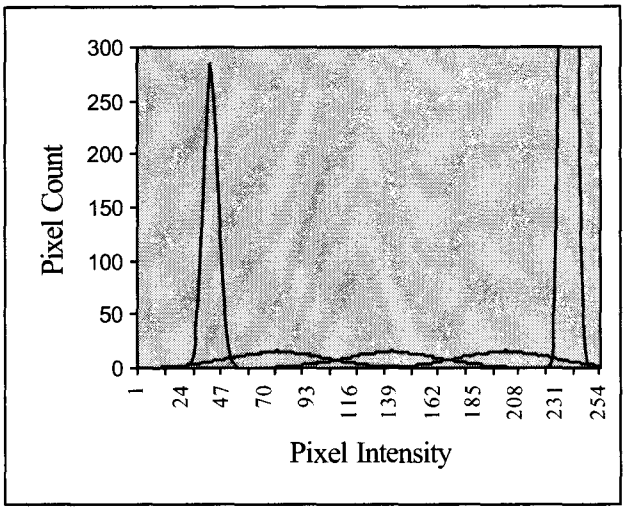


Figure 3: Transformation from input gray scale to output binary image

(a) Original Image

(b) Binary Image



(c) Modeled Histogram

Figure 4. Binarization of image with bi-modal histogram using a Gaussian modeling

In Figure 3, the gray scale value of $I(x,y)$, the center pixel in this example, is input to Equation 4 to determine the number of text pixels in the 2x2 output region $I'(x,y)$. The eight surrounding pixels are used in the four linear predictors to calculate the 2x2 array of prediction values in $\hat{I}(x,y)$. These determine the actual locations of the k output text pixels (in this example, k=1). The application of this new method of Gaussian modeling to the input image of Figure 1 is illustrated in Figure 4.

4. Results

To evaluate the effectiveness of the new Gaussian modeling approach it is compared to a number of different binarization and resolution expansion techniques from the current literature. Different methods are used to transform a large set of gray scale images to output binary images. The output binary image is then processed by commercial OCR software [1] to create a transcript file of the recognized text. A common set of OCR performance metrics are used to evaluate the binarization performance by comparing the OCR output with manually edited ground truth files for each of the images.

The first metric used for evaluation is the Normalized Edit Distance (NED). The difference between the total number of characters in the ground truth and the total number of errors (sum of insertion, deletion, and substitution errors) is calculated, then divided by the number of characters in ground truth. The Character Accuracy Rate (CAR) is defined in the same manner as the NED, but ignores insertion errors. The Word Accuracy Rate (WAR) is the percentage of words in the ground truth that appear correctly in the OCR output. Finally, the Non-StopWord Accuracy Rate (NSWAR) is the percentage of "important" words in the ground truth that appear correctly in the OCR output. Finally, the processing time for execution is measured on a Sun Ultra 2.

Method	NED (%)	CAR (%)	WAR (%)	NSWAR (%)	Time (s)
Niblack	28.0	29.1	12.5	4.7	13
Replication-Niblack	58.5	58.6	39.2	29.9	95
Spline-Global Niblack	78.1	79.8	65.6	56.7	150
Gaussian Modeling	80.9	82.6	70.4	61.4	65
Spline-Niblack	83.3	83.5	72.0	65.1	158
BSA-Niblack	85.6	87.0	76.7	70.6	12300

Table 1: Image Binarization performance on 59 gray scale document images.

The test data set consists of cropped regions of text from the University of Washington data set. The images are clean and from the domain of scientific journal articles. The source images are at lower resolution than the original data set, having been printed and rescanned at 75 pixels/inch. The test images were cropped from originals to ensure that all text occurs in a single column format. This reduces the probability of error from the automated character accuracy measurement tools. This test set contains 59 cropped images containing 128,984 Latin characters.

In the evaluation of Trier [2], Niblack's method was found to perform the best on a very limited test set. Therefore, the performance of Niblack's method was evaluated using the test images at their original spatial resolution was evaluated. These same images are processed by the new Gaussian modeling technique.

Because the new Gaussian modeling algorithm combines spatial resolution expansion with binarization in a single integrated process, it is unfair to compare against other binarization algorithms without expanding the image spatial resolution as well. The performance of Niblack's method was also evaluated after resolution expansion of the gray scale input image using pixel replication, using BSA expansion [13], and using bicubic spline interpolation. In all cases above, the spatial resolution was increased by two in both dimensions to correspond with the expansion provided by the Gaussian modeling approach.. Finally, Niblack's method was applied globally after bicubic spline interpolation, to evaluate how much benefit was provided by the local adaptation. Results of these experiments are summarized in Table 1.

5. Conclusions

These tests show that enhancing the spatial resolution of the 75 pixel/inch input images provides a significant improvement to all character and word accuracy measurements, even when simply replicating pixel values. The most significant improvement is provided by the new Gaussian modeling approach or by using Niblack's method after bicubic spline interpolation or BSA expansion. Of these methods, the BSA algorithm requires an excessive amount of processing time for large images. Small gains above the Gaussian Modeling approach (in both character and word accuracy) may be obtained by combining bicubic spline interpolation and Niblack's method. However, this improvement is

gained at the expense of over twice the computation time. Additionally, much of the improvements in Niblack's method may be due to its adaptive nature, as illustrated by the drop in performance of Niblack's method when applied globally. Overall, the new Gaussian modeling method provides a significant improvement with a reasonable amount of processing time.

Several opportunities exist for improving the existing approach of Gaussian modeling. First, the two composite histogram models created by heuristic methods could be further refined using EM techniques. Second, the algorithm is fast enough to be implemented on a moving window of the image. This would make the algorithm adaptive to local changes in background level (similar to Niblack's method). Finally, features already extracted from the image histogram could be used to select a number of morphological image post-processing steps to enhance the output binary image.

References:

[1] Caere/Scansoft, Caere Developer's Kit 2000, 1998

[2] Trier, Oivind Due, and Jain, Anil K. "Goal Directed Evaluation of Binarization Methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol 17, No 12 Dec 1995, pp. 1191-1201.

[3] Gonzlez, R.C. and Woods, R.E., *Digital Image Processing*, Addison Wesley, September 1993, pp. 663-669

[4] Parker, J.R., "Gray Level Thresholding in Badly Illuminated Images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol 13, No 8, 1991, pp. 813-819

[5] Otsu, N., "A Threshold Selection Technique from Grey-level Histograms," *IEEE Trans. Systems, Man, and Cybernetics*, Vol. 9, No. 1, pp. 62-66, 1979.

[6] Bernsen, J., "Dynamic Thresholding of Grey-level Images," *Proc. Eighth Int'l Conf. Pattern Recognition*, pp.1251-1255, Paris, 1986.

[7] Niblack, W., *An Introduction to Digital Image Processing*, pp. 115-116, Englewood Cliffs, N.J., Prentice Hall, 1986.

[8] Chow, C.K. and Kaneko, T., "Automatic Detection of the Left Ventricle from Cineangiograms," Computers and Biomedical Research, Vol. 5, pp.388-410, 1972.

[9] Taxt, T., Flynn, P.J., and Jain, A.K., "Segmentation of Document Images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol 11, No 12, 1991, pp. 1322-1329

[10] Eikvil, L., Taxt, T., and Moen, K., " A fast adaptive method for binarization of document images," *Proc. First Int'l Conf. Document Analysis and Recognition*, pp.435-443, Saint-malo France, 1991.

[11] Jelinek, F., Statistical Methods for Speech Recognition, Massachusetts Institute of Technology, pp. 147-163, 1997.

[12] Titterington, D.M., Smith, A.F.M., and Makov, U.E., *Statistical Analysis of Finite Mixture Distributions*, New York: John Wiley & sons, 1985.

[13] P.D. Thouin and C.-I Chang, "A method for restoration of low-resolution document images", *International Journal of Document Analysis and Recognition*, Volume 2, Number 4, June 2000.

Low Resolution Expansion of Gray Scale Text Images Using Gibbs-Markov Random Field Model

Paul David Thouin
Department of Defense
Fort Meade, MD 20755 USA

Yingzi Du and **Chein-I Chang**
Remote Sensing Signal and Image Processing Laboratory
Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County
Baltimore, MD 21250 USA

Abstract

Image resolution expansion becomes increasingly important in restoration of text and document images. In order to take advantage of the text image properties such as bimodal, and smooth in background and foreground with sharp transitions only occurring at the edge, an algorithm, called bimodal-smoothness-average (BSA) method, was recently developed by Thouin and Chang and has shown success in low resolution expansion of text images. In this paper, an alternative version of the BSA method, referred to as bimodal-smoothness-Gibbs-Markov random field (BSGMRF), is presented. It replaces the average score function used in BSA with 27 cliques derived from the Gibbs-Markov random field. Each of these 27 cliques is a triplet with each component assigned by either black, gray, or white. They are particularly designed to capture transitions occurring in the 4-neighbor connectivity in text images. These 27 cliques provide possible resolution expansion for a given low resolution text image. Since the cliques only allow certain patterns to occur when a low resolution image is expanded, the resulting expanded image generally has clean background compared to the gray background obtained by the BSA method. During initial experiments on a small number of images, BSG performed slightly better than BSA using OCR accuracy as a measure. Additional experiments need to be conducted to determine if there is a measurable improvement.

1. INTRODUCTION

Enhancement of text images continues to be an important research area in both the document and video recognition fields. Restoring text from video surveillance imagery is often crucial to law enforcement agencies. Digital video compression algorithms can benefit from successful text resolution expansion techniques. In order to derive an effective method of expanding low-resolution text images we must custom design algorithms that can take advantage of the text image properties such as bimodal, and smooth in background and foreground with sharp transitions only occurring at the edges. An algorithm, called bimodal-smoothness-average (BSA) method, was recently developed by Thouin and Chang in [1] to capture these characteristics and has shown success in low resolution expansion. In this paper, we present an alternative version of the BSA method which replaces the average score function (A) used in the BSA method with a set of 27 cliques that are derived from the Gibbs Markov random field (GMRF). The proposed method, referred to as bimodal-smoothness-Gibb-Markov random field (BSGMRF), has two advantages over the BSA method. The average score function used by the BSA was designed to preserve the property that the averaged gray-level resolution of an expanded block from a pixel of a low resolution text image equal to the low gray level resolution of the pixel. The BSGMRF takes a different view point. Since a text image generally has text on a white background, there are only certain transitions which may occur within a 3x3 neighboring window. In order to describe this characteristic statistically, the GMRF model was first proposed in [2] where 27 cliques were constructed. Each of these 27 cliques is a triplet with each component assigned either black, gray, or white. They use particular patterns of white-gray-black combinations to provide possible resolution expansion for a given low resolution text image. Another advantage is that by means of the 27 cliques the BSGMRF-expanded image generally has clean background compared to the gray background obtained by the BSA method. As a result, the OCR accuracy is slightly better than that achieved by the BSA method despite that the images produced by both BSGMRF and BSA are very close with little visual difference. The experiments conducted in this paper will demonstrate

that in most cases, the BSGMRF method improves the BSA method by increasing the OCR accuracy.

2. BSA METHOD

The BSA method was developed to take advantage of text image properties for resolution expansion. It used three criteria, bimodal (B), smoothness (S) and average (A), as the goodness of fit of an expanded high resolution image from a low resolution text image. Each of these three criteria introduced a score function to measure how well a potential expanded image exhibit a particular property. Using these three score functions a constrained optimization problem can be formulated for resolution expansion by

$$BSA(x) = \lambda_B B(x) + \lambda_S S(x) + \lambda_A A(x) \qquad (1)$$

where x is a block of pixels and λ_B, λ_S, and λ_A are Lagrange multipliers to be determined. Now our goal is to design a BSA-based algorithm which can iteratively solve for a block of pixels x that minimizes the *BSA(x)* score given by Eq.(1). In order to solve Eq. (1), the initial values of the high-resolution image are set using pixel replication. Every value within the high-resolution neighborhood is identical to the corresponding low-resolution pixel. Each block of data is updated iteratively using optimization techniques to solve for the block which minimizes the $BSA(x)$ score. In what follows, we briefly describe each of the three score functions.

2.1. Bimodal Score Function: B(x)

The bimodal score is defined by

$$B(x) = \sum_{r,c} (x_{r,c} - \mu_B)^2 (x_{r,c} - \mu_W)^2 \qquad (2)$$

where r and c are the row and column indices within the block x being evaluated. μ_B and μ_W are the peaks in the histogram for black and white. When a pixel value within the block x is close to either μ_B and μ_W, its contribution to $B(x)$ is minimal. When $B(x) = 0$ it implies that the image is perfectly bimodal. Solving for the pixels in the block x that minimizes $B(x)$ produces a strongly bimodal image, which is one of the desired properties of this text restoration technique. The first partial derivative of this bimodal score with respect to pixel $x_{r,c}$ can be calculated by:

$$\frac{\partial B(x)}{\partial x_{r,c}} = 4x_{r,c}^{\ 3} - 6(\mu_B + \mu_W)x_{r,c}^{\ 2} + 2(\mu_W^{\ 2} + 4\mu_W\mu_B + \mu_b^{\ 2})x_{r,c} - 2\mu_W\mu_B(\mu_W + \mu_B) \qquad (3)$$

2.2 Smoothness Score Function: S(x)

With the exception of edges, text images tend to be very smooth in both the foreground and background regions which result in neighbors with similar values. A smoothness score, $S(x)$ is introduced to measure this feature. It is given by

$$S(x) = \sum_{r,c} [(x_{r-1,c} - x_{r,c})^2 + (x_{r+1,c} - x_{r,c})^2 + (x_{r,c-1} - x_{r,c})^2 + (x_{r,c+1} - x_{r,c})^2] \qquad (4)$$

and is computed for pixels in each block x. When $S(x)$ achieves the minimum value 0, it implies that all pixels have identical values. The first partial derivative of this smoothness score function with respect to pixel $x_{r,c}$ can be calculated by

$$\frac{\partial S(x)}{\partial x_{r,c}} = -2(x_{r-1,c} + x_{r+1,c} + x_{r,c-1} + x_{r,c+1}) + 8x_{r,c}. \qquad (5)$$

2.3. Average Score Function: A(x)

The average score function $A(x)$ is used to measure how well the restored high-resolution pixels meet the average constraint imposed by the corresponding low-resolution pixels.

We expand the original image to a high resolution image with the expansion factor q. The $q \times q$ group of high-resolution pixels that are being restored from pixel μ_1 are represented by $\{x^{(1)}_{r,c}, 1 \le (r,c) \le q\}$. The average score for this 2×2 block is expressed by

$$A(x) = \sum_{i=1}^{4} [\mu_i - \frac{1}{q^2} \sum_{r=1}^{q} \sum_{c=1}^{q} x^{(i)}_{r,c}]^2 \qquad (6)$$

where i is used to index the low-resolution pixels, μ_i is the value of the i^{th} low-resolution pixel, and $x^{(i)}_{r,c}$ are the restored high-resolution pixels corresponding to pixel μ_i. The initial high-resolution image is first formed by using pixel replication and always has an average score of zero because it satisfies the constraint. The first partial derivative for the group of high-resolution pixels corresponding to pixel μ_i is given by

$$\frac{\partial A(x)}{\partial x^{(i)}_{r,c}} = \frac{2}{q^2} [\frac{1}{q^2} \sum_{r=1}^{q} \sum_{c=1}^{q} x^{(i)}_{r,c} - \mu_i]. \qquad (7)$$

2.4 BSA Score Function: BSA(x)

Substituting $B(x)$ in Eq. (2), $S(x)$ in Eq. (4) and $A(x)$ in Eq. (6) into the BSA function in Eq. (1) results in the BSA scoring function, BSA(x). The BSA function can be represented by its Taylor series approximation a small distance away from x:

$$BSA(\vec{x}+\vec{\delta}) \approx BSA(\vec{x})+[\nabla BSA(\vec{x})]\vec{\delta} \quad (8)$$

where the Taylor series approximation was used for a small distance away from $\vec{x}$, $\vec{\delta}$ is a small change in the image vector $\vec{x}$ and $\nabla BSA(\vec{x})$ is the gradient. The image change at iteration i is computed by $\vec{\delta}_i = -\nabla BSA(\vec{x})$. The image is then updated using

$$\vec{x}_{i+1} = \vec{x}_i + \vec{\delta}_i. \quad (9)$$

The iterative process continues until $\vec{\delta}_i \approx \vec{0}$. Based on the experiments conducted in [1], the relative weights of λ_B, λ_S, λ_A were chosen empirically to be $\lambda_B = 1$, $\lambda_S =$ 10,000, and $\lambda_A = 1{,}000{,}000$.

3. GIBBS-MARKOV RANDOM FIELD (GMRF)

In [2] a Gibbs-Markov random field (GMRF) was developed for restoration of DCT-compressed text images. The GMRF given in [2] is defined on the image x by

$$P(x) = \frac{1}{K} e^{-\sum_{\forall C} E_C(x)} \quad (10)$$

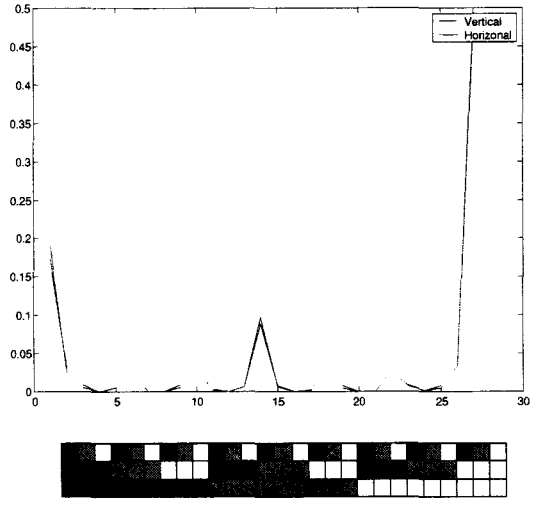


Figure 1. 27 clique triplets distribution

where $Ec(x)$ is the potential associated with clique C and K is a normalizing constant. To further reduce computations, we are only interested in 3×1 vertical triplets and the 1×3 horizontal triplets. We assume that there are only three regions of interest within text images, background (white), transition (gray), and foreground(black), and that such a triplet is sufficient to characterize text properties. A triplet specified by black, gray and white is sufficiently enough to characterize the text properties. Each pixel within a clique triplet can be considered as one of the background, transition, and foreground regions found in text images. Therefore, there are 27 triplet combinations of interest. As an example 27 cliques are shown at the bottom of Fig. 1 and the solid and dotted lines represented the frequency of occurrence of the vertical triplets and horizontal triplets respectively.

Now a criterion is created to measure the distance from each clique in the image to the 27 cliques described above in Fig. 1. It is determined by the horizontal and vertical distance equations formed by means and variances combined with the pixel values,

$$D^H_{abc} = \frac{1}{2\sigma_a^2}(x_{r,c-1} - \mu_a)^2 + \frac{1}{2\sigma_b^2}(x_{r,c} - \mu_b)^2 + \frac{1}{2\sigma_c^2}(x_{r,c+1} - \mu_c)^2 \quad (11)$$

$$D^V_{abc} = \frac{1}{2\sigma_a^2}(x_{r-1,c} - \mu_a)^2 + \frac{1}{2\sigma_b^2}(x_{r,c} - \mu_b)^2 + \frac{1}{2\sigma_c^2}(x_{r+1,c} - \mu_c)^2 \quad (12)$$

where each of the parameters a,b,c can have three values, white, gray and black. Using Eqs. (11) and (12) we can define a Gibbs-Markov text (GMT) score function $G(x)$ by the following formula,

$$G(x) = \sum_{pixels} \sum_{a=1}^{3}\sum_{b=1}^{3}\sum_{c=1}^{3} H_{abc} e^{-D^H_{abc}} V_{abc} e^{-D^V_{abc}} \quad (13)$$

where H_{abc} and V_{abc} are the weights to be determined. It should be noted that $G(x)$ is a function of the pixel x, means, μ_a, μ_b, μ_c and variances, $\sigma_a^2, \sigma_b^2, \sigma_c^2$ of the background, foreground, and transition distributions, and the weights of each of the 27 clique triplets. Differentiating the G(x) with respect to pixel $x_{r,c}$ results in

$$\begin{aligned}\frac{\partial G(x)}{\partial x_{r,c}} = \sum_{a=1}^{3}\sum_{b=1}^{3}\sum_{c=1}^{3} C^H_{abc}\Big[&-\frac{1}{\sigma_a^2}(x_{r,c}-\mu_a)e^{-\frac{1}{2\sigma_a^2}(x_{r,c}-\mu_a)^2-\frac{1}{2\sigma_b^2}(x_{r,c-1}-\mu_b)^2-\frac{1}{2\sigma_a^2}(x_{r,c-2}-\mu_c)^2}\\ &-\frac{1}{\sigma_b^2}(x_{r,c}-\mu_b)e^{-\frac{1}{2\sigma_a^2}(x_{r,c-1}-\mu_a)^2-\frac{1}{2\sigma_b^2}(x_{r,c}-\mu_b)^2-\frac{1}{2\sigma_a^2}(x_{r,c+1}-\mu_c)^2}\\ &-\frac{1}{\sigma_c^2}(x_{r,c}-\mu_c)e^{-\frac{1}{2\sigma_a^2}(x_{r,c-2}-\mu_a)^2-\frac{1}{2\sigma_b^2}(x_{r,c-1}-\mu_b)^2-\frac{1}{2\sigma_a^2}(x_{r,c}-\mu_c)^2}\Big]\\ +C^V_{abc}\Big[&-\frac{1}{\sigma_a^2}(x_{r,c}-\mu_a)e^{-\frac{1}{2\sigma_a^2}(x_{r,c}-\mu_a)^2-\frac{1}{2\sigma_b^2}(x_{r-1,c}-\mu_b)^2-\frac{1}{2\sigma_a^2}(x_{r-2,c}-\mu_c)^2}\\ &-\frac{1}{\sigma_b^2}(x_{r,c}-\mu_b)e^{-\frac{1}{2\sigma_a^2}(x_{r-1,c}-\mu_a)^2-\frac{1}{2\sigma_b^2}(x_{r,c}-\mu_b)^2-\frac{1}{2\sigma_a^2}(x_{r+1,c}-\mu_c)^2}\\ &-\frac{1}{\sigma_c^2}(x_{r,c}-\mu_c)e^{-\frac{1}{2\sigma_a^2}(x_{r-2,c}-\mu_a)^2-\frac{1}{2\sigma_b^2}(x_{r-1,c}-\mu_b)^2-\frac{1}{2\sigma_a^2}(x_{r,c}-\mu_c)^2}\Big].\end{aligned} \quad (14)$$

4. BSGMRF METHOD

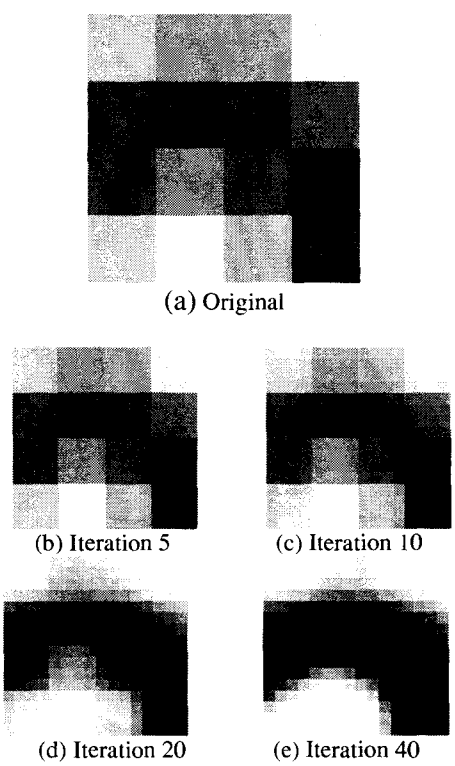
In this section, we propose a method, referred to as Bimodal-Smoothness-GMRF (BSGMRF), that replaces the average score function $A(x)$ in Eq. (6) with the GMRF score function $G(x)$ in Eq. (13). The resulting *BSGMRF* score function can be calculated by

$$BSGMRF(x) = \lambda_B B(x) + \lambda_s S(x) - \lambda_G G(x) \qquad (15)$$

where $B(x)$ and $S(x)$ are the same $B(x)$ and $S(x)$ used in *BSA* method, and $G(x)$ is the same one used in [2]. It is worth noting that we use "-" to control the $G(x)$ in Eq. (14). This is because if the image is more like a text image, the score of $G(x)$ will be higher, thus $-G(x)$ will be smaller. The three parameter values: $\lambda_B = 1$, $\lambda_S = 10{,}000$, and $\lambda_G = 3{,}000{,}000$ used in Eq. (15) were selected empirically based on our experiments. Using the Taylor series approximation, we can approximate the BS*GMRF*(x) within a small distance away from $\vec{x}$ by

$$BSGMRF(\vec{x}+\vec{\delta}) \approx BSGMRF(\vec{x}) + [\nabla BSGMRF(\vec{x})]\vec{\delta} \qquad (16)$$

where $\vec{\delta}$ is a small change in the image vector $\vec{x}$ and $\nabla BSGMRF(\vec{x})$ is the gradient. The image change at iteration i is computed by $\vec{\delta}_i = -\nabla BSGMRF(\vec{x})$ and the expanded image is then updated using Eq. 9.



(a) Original

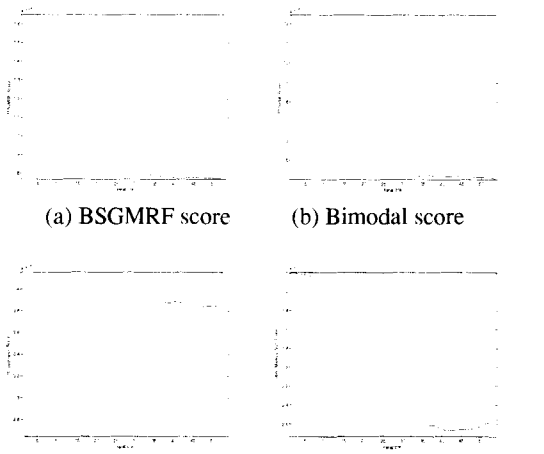
(b) Iteration 5 (c) Iteration 10

(d) Iteration 20 (e) Iteration 40

Figure 2. An example of resolution expansion of 4×4 block of pixels

The iterative process continues until $\vec{\delta}_i \approx \vec{0}$. Fig. 2 shows an original 4×4 block of pixels expanded by a factor of 4 with iterations 5, 10, 20, and 40.

Fig. 3 shows the scores generated by *BSGMRF(x), B(x), S(x),G(x)* respectively for the block shown in Fig. 2.



(a) BSGMRF score (b) Bimodal score

(c) Smoothness score (d) Gibbs Markov Text score

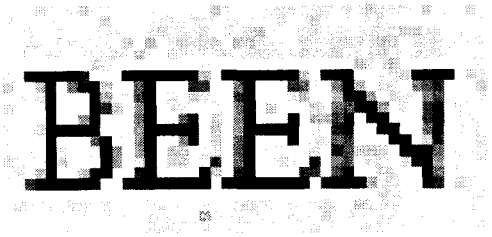
Figure 3. BSGMRF score minimization

The BSGMRF score function decreased very rapidly at the beginning before 30 iterations, and then slowed down. The bimodal score decreased very rapidly before it reached 35 iterations, but unlike the BSGMRF score the bimodal score function experimented a small concave, after that it remains almost unchanged until after 53 iterations, it dropped a little bit. But because the GMT score and Smoothness scores were increased at this time, the further decrease of the bimodal score is limited. Interestingly, the smoothness score function also decreased very rapidly during the first 15 iterations, but it suddenly increased its values very rapidly between 16 to 35 iterations. After 35 iterations, it then decreased again with a slower rate. After 40 iterations, it almost unchanged. For the GMT score function, the values dropped very rapidly before 33 iterations, then the GMT score experimented a small concave. After 40 iterations, it became stable and flat. Beyond 48 iterations, it began to increase. From the scores in Figs. 3(a)-3(d) we can see that the bimodal score produced the largest values in the first 35 iterations, while the GMT score produced the smallest value. But after 35 iterations, the smoothness score produced the largest values. In the first 35 iterations, the bimodal score would dominate the BSGMRF score, then the smoothness score took over to determine the BSGMRF score. In either case, the GMT score did not affect the BSGMRF score since it produced very small values. This makes sense since the bimodal score function tries to eliminate noise using bimodal distribution in the first place, then the smoothness score function intended to smooth transitions between foreground and background. Finally, the GMT score

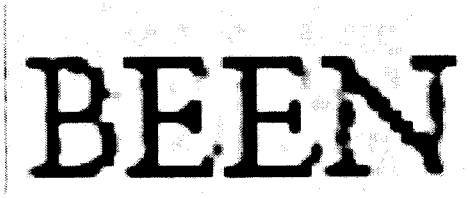
function enforces the restored image to satisfy the desired text characteristics by using 27 clique triplets. The reason that the GMT score function did not show impact on the BSGMRF score in this particular example is because the processed image block already satisfied the desired text properties. As a matter of fact, the GMT score function is crucial in the final stage of restoration processing.

5. EXPERIMENTAL RESULTS

In this section, we present experiments to demonstrate the performance of the proposed BSGMRF method. Fig.4 shows the results obtained by the BSA and BSGMRF methods where Fig. 4(a) is an original text image which was scanned by a 75-dpi resolution and Figs. 4(b) and 4(c) were restored by the BSA and BSGMRF methods as 300-dpi images respectively.



(a) Original image

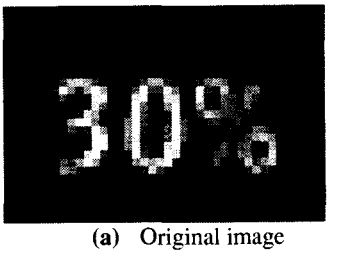


(b) BSA restored image

(c) BSGMRF restored image

Figure 4. Example text image restored by BSA and BSGMRF methods

As shown in Fig. 4, both restored images were better than the original image. However, the BSGMRF-restored image produced a better image than the BSA-restored image in the sense that the text image background has been cleaned up and no clear foreground-background transitions were observed. As we compare Fig. 4(b) to Fig. (c), we notice the effects of the GMT score function shown on the curve of the letter "B", the vertical line segments and ends of both letters "E" and the letter "N" where the BSGMRF intended to use 27 cliques to restore these 4 letters. As a result, in the BSA-restored image the gray regions in the text image background have been removed and broken line segments in the ends of letters "E" and the vertical segments of the letter "N" were connected.



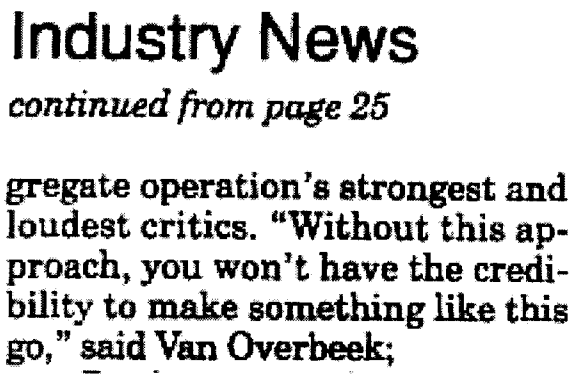
(a) Original image

(b) BSA restored image **(c)** BSGMRF restored image

Figure 5. Example a video frame restored by BSA and BSGMRF methods

Fig. 5(a) shows a section of a video frame image and Figs. 5(b) and 5(c) are images restored by the BSA and the BSGMRF methods respectively. Compared to the BSA-restored image, the background of the BSGMRF-restored image is very clean. The "30" in the BSGMRF-restored image looks more like the text. The two circles of the percentage restored by the BSGMRF were recovered nearly perfectly compared to that restored by the BSA method. This is due to the fact that the effects of foreground-background transitions in the BSA-restored image was removed by the BSGMRF method. Interestingly, the slash "/" of the percentage in the BSGMRF-restore image was broken into two parts. This is because the 27 cliques used in the BSGMRF method are designed to restore vertical and horizontal line segments but not diagonal or anti-diagonal line segments such as a slash"/".

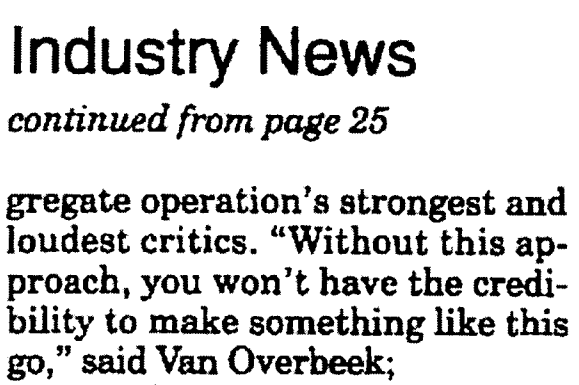
Fig. 6 was used to evaluate the performance of OCR accuracy produced by the BSA and the BSGMRF methods. The used original low resolution image in Fig. 6(a) was a 100-dpi scanned text image. The OCR result was shown in Fig. 6(b) where the Caere's OminiPage Pro 10.0 commercial OCR package was used for recognition. As we can see, there were many errors. The OCR result produced by the BSA-restored image in Fig. 6(c) is shown in Fig. 6(d), which was significantly better than that Fig. 6(a). Only one error was made. The OCR result produced by the BSGMRF-restored image shown in Fig. 6(e) was perfect and is shown in Fig. 6(f).



(a) Original image

Industry News
continued from pagre 25
gngate operntian o strongest and loudest critics. "Without this approach, you won't havcthecredibliity to make something like this go"'sadVanOverbmi,

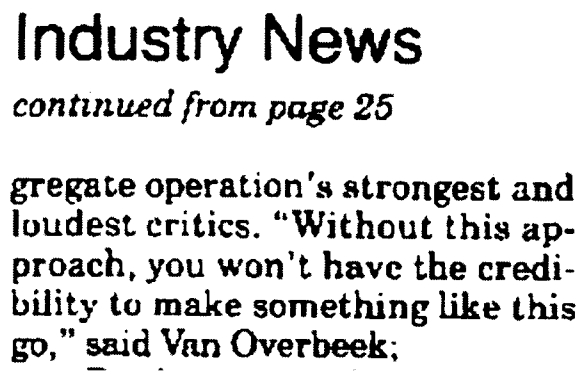
(b) Original image OCR results



(c) BSA restoration

Industry News
continued from page 25
gregate operation's strongest and loudest critics. "Without this approach, you won't have the credibility to make something like this go," said fan Overbeek;
V

(d) BSA-restored OCR results



(e) BSGMRF restoration

Industry News
continued from page 25
gregate operation's strongest and loudest critics. "Without this approach, you won't have the credibility to make something like this go," said Van Overbeek;

(f) BSGMRF-restored OCR results

Figure 6. Example Text Image restored by BSA and BSGMRF and their OCR recognitions.

In addition to the experiments presented in this paper, many more images have been conducted for performance evaluation. On a whole, the BSGMRF-restored images look more like text images with cleaner image background compared to the BSA-restored images. On some occasions, the BSA cannot restore very noisy text images which can be still restored by the BSGMRF method for OCR processing.

6. CONCLUSION

In this paper, we present a new image restoration method, bimodal-smoothness-Gibbs-Markov random field (BSGMRF), which expands a low resolution text image to a high resolution image while preserving text image characteristics. The BSGMRF method is derived from the BSA method by replacing the average score function with the GMRF, which can produce clean image background compared to images produced by the BSA method. The experiments seem to demonstrate that in some cases, the BSGMRF method combines the strengths of the BSA method and the Gibbs-Markov random field to achieve better OCR accuracy. As a final remark, it should be noted that the experiments conducted in this paper only made comparison between the BSA and BSGMRF methods. This is because in [1] the BSA method has been shown to perform significantly better than the commonly used methods such as bilinear and the spline methods. So, no experiments on comparative study among these methods are included.

ACKNOWLEDGMENT

The third author would like to thank Department of Defense to support his work through the contract number MDA-904-00-C2120.

References:

[1] P. Thouin and C.-I Chang, "A method for restoration of low-resolution text images," *International Journal on Document Analysis and Recognition*, Vol. 2, No. 4, pp. 200-210, June 2000.

[2] P. Thouin and C.-I Chang, "New technique restores text from 8x8 block DCT-compressed gray-scale images," *Special Section on J. Electronic Imaging*, *OE Report*, no. 179, p. 11, November 1998.

Bilevel Image Degradations: Effects and Estimation

Elisa H. Barney Smith
EBarneySmith@boisestate.edu
Electrical and Computer Engineering Department
Boise State University, Boise, Idaho 83725, USA
Phone: 208-426-2214

Abstract

The two most significant parameters affecting degradations of bilevel images are the point spread function (PSF) width and the binarization threshold. Each pair of these values will affect an image differently. However, several combinations of these parameters will affect images in a similar fashion. This paper looks at two aspects of image degradation: the displacement of an edge, which determines stroke width, and the erosion of a corner, which affects crispness. The relationship between the PSF width and the binarization threshold and these two effects will be described. Sample characters, first with similar edge displacement and second with similar corner erosion, will show the effect of estimating the broader degradation versus the exact system parameters. Methods of estimating these degradations will also be briefly discussed.

1. Introduction

Bilevel processes such as scanning, photocopying, faxing, and printing cause many degradations to document images. These processes are characterized by spatial and intensity quantization, which changes the appearance of the image content, such as characters and line drawings. The ability to characterize the degradations that are introduced when a document passes through a bilevel process is an important step toward improving recognition accuracy. This paper discusses bilevel degradations in the context of the scanning process.

Baird developed a degradation model that contained 10 parameters: resolution, blur, threshold, sensitivity, jitter, skew, width, height, baseline, and kerning [1]. Ho and Baird compared the OCR accuracy under different values of blur, thresholding, and pixel sensitivity and determined that PSF width and binarization threshold are the two most significant parameters [11]. Figure 1 shows a model of the production of bilevel digitized images using only the PSF and thresholding parameters. Each combination of PSF width, w, and binarization threshold, Θ, produces a different digitized image. The degradations in a scanned image are due to these two

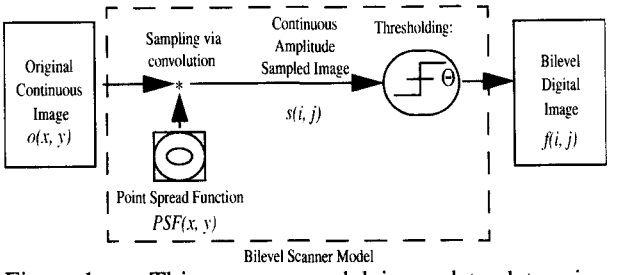


Figure 1: This scanner model is used to determine the value of the pixel *(i,j)* centered on each sensor element.

parameters. The degradation type, more than the individual parameter values, is used to describe the image quality.

This paper describes two degradation types. It describes how these degradations are related to the parameters w and Θ. Characters synthetically generated with width and threshold values that produce a common degradation are shown for comparison. Methods for estimating the degradations are briefly discussed, followed by a discussion of when knowledge of the degradation amount is adequate or when we must go beyond this and estimate values for w and Θ as well.

2. Degradation Types

When a character image is degraded, the two most

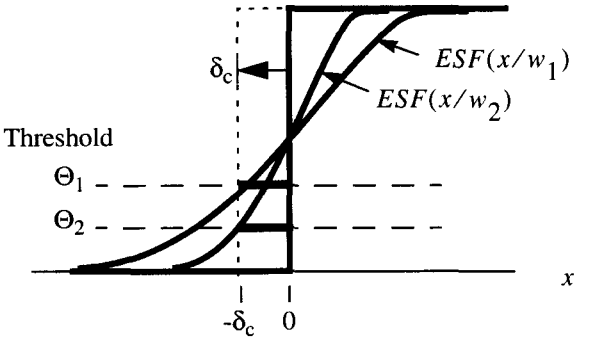


Figure 2: Edge after blurring with a generic PSF of two widths, w. Two thresholds are shown that produce the same edge shift δ_c.

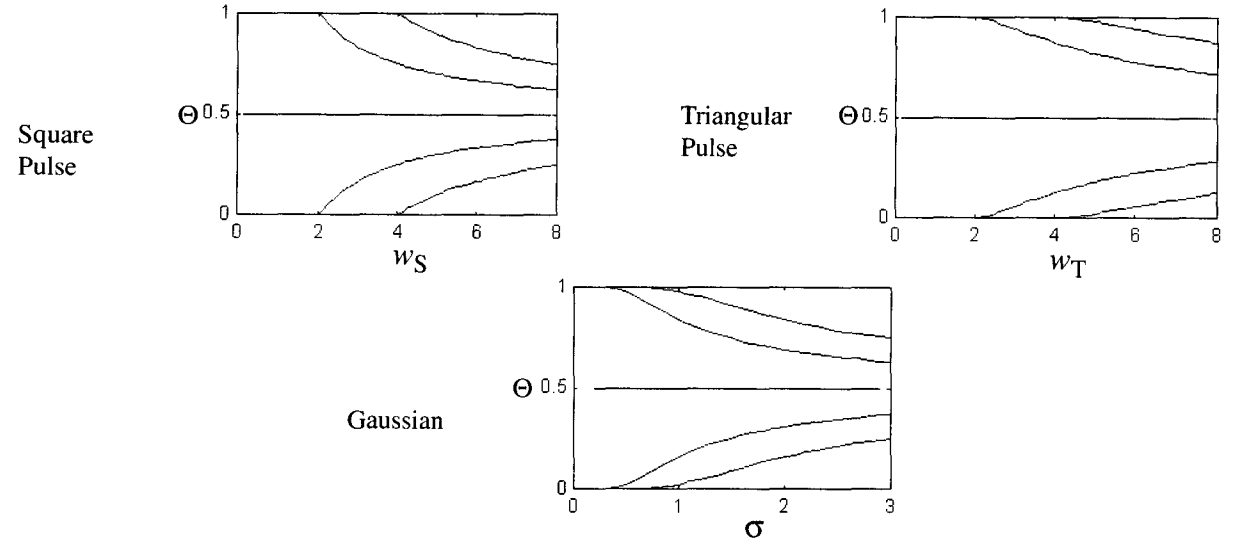


Figure 3: Contours showing constant edge spread of δ_c =[-2 -1 0 1 2] (from top to bottom) for three PSF functions.

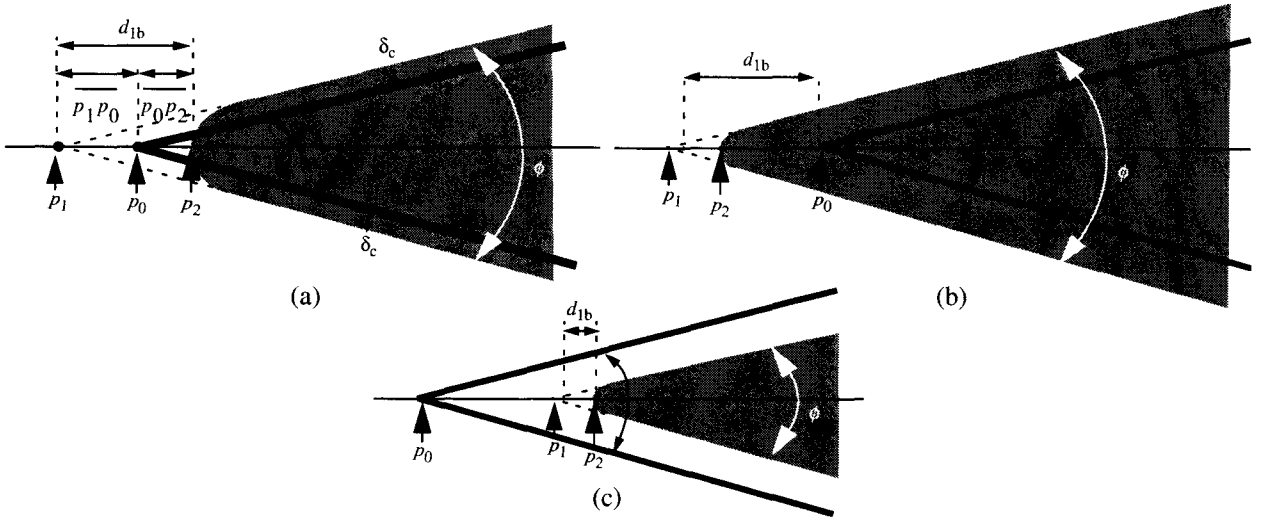


Figure 4: Three possible ways in which a blurred corner (grey area) may be displaced from the original position (black lines).

noticeable effects are a change in the stroke width and a change in the shape of the corners. Both of these degradations are caused by a joint effect of the point spread function width, w, and the binarization threshold, Θ. This relationship can be quantified to describe the amount of the degradation.

2.1 Edge Displacement

The stroke width is determined by the location of the edges of the stroke. The stroke width will change as the edge locations move. During scanning, the edge changes from a step to an edge spread function, ESF, through convolution with the PSF. This is then thresholded to reform a step edge, Figure 2. The amount an edge was displaced after scanning, δ_c, was shown in [3] to be related to w and Θ by

$$\delta_c = -w\, \mathrm{ESF}^{-1}(\Theta)\,. \qquad (1)$$

The distance that an edge is displaced depends on the threshold, the PSF width and the PSF functional form. An infinite number of (w, Θ) values could produce any one δ_c value. Eq. (1) holds when edges are considered in isolation, for example when the edges are separated by a distance greater than the support of the PSF. Figure 3 shows how the values of (w, Θ) vary for 5 different constant δ_c values for each of four PSF shapes. A positive threshold value will produce a negative edge displacement. The curves for δ_c and $-\delta_c$ are symmetric around the Θ=1/2 line. If Θ=1/2, δ_c=0 for all values of w.

2.2 Corner Erosion

The other major degradation important to bilevel

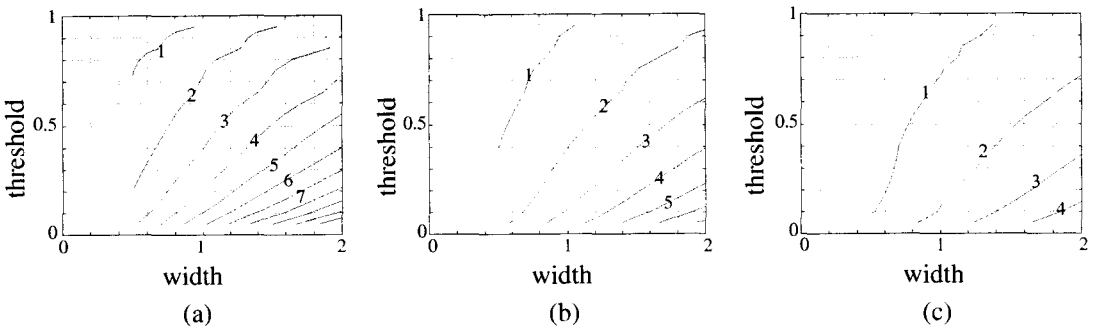


Figure 5: Observable erosion contours for constant d_{1b} loci for a Gaussian PSF (a) $\phi=\pi/6$ (b) $\phi=\pi/4$ (c) $\phi=\pi/3$

images is the shape of a corner [5]. At a distance greater than one half the support of the PSF from the corner, only edge spread effects are present. Nearer to the intersection of the two edges, a degradation is caused by the interaction of the two edges together. The degradation of a corner can occur in any of the three forms shown in Figure 4. The point p_0 is the apex of the original corner. The point p_2 is the point along the angle bisector of the new rounded corner where the blurred corner equals the threshold value. The point p_1 is the point where the new corner edges would intersect if extrapolated. The distance that a corner is eroded from the original apex point p_0 depends on the threshold, the PSF width and the functional form similar to the edge displacement above.

One common aspect of these degradations is the amount of the corner that is eroded away, shown as the distances $\overline{p_0p_2}$ and d_1 in Figure 4. The equation for the amplitude of the blurred corner along its line of symmetry can be written as a function of w and ϕ,

$$f_b(d_{0b};w,\phi) = \int_{x=0}^{x=\infty} \int_{y=-x\tan\frac{\phi}{2}}^{y=x\tan\frac{\phi}{2}} PSF(x-d_{0b},y;w)\,dy\,dx\,, \quad (2)$$

in which case

$$\overline{p_0p_2} = f_b^{-1}(\Theta;w,\phi)\,. \quad (3)$$

Measuring distance $\overline{p_0p_2}$ requires knowledge of the original location of the corner, which is not easily found on its own.

Further from the corner, the edges do not interfere with each other during blurring and the edge spread effect discussed earlier is present. The distance between point p_1 and point p_0 is a function of edge spread and the corner angle:

$$\overline{p_1p_0} = \frac{\delta_c}{\sin(\phi/2)} = \frac{-wESF^{-1}(\Theta)}{\sin(\phi/2)}\,. \quad (4)$$

The scanned edges will be parallel to the original edges, allowing the angle ϕ to be measured. The point p_1 and p_2 can be located easily. The distance between points p_1 and p_2 is the sum of the two distances since the points are co-linear, therefore

$$d_{1b} = \overline{p_1p_2} = \overline{p_1p_0} + \overline{p_0p_2} = \frac{-wESF^{-1}(\Theta)}{\sin(\phi/2)} + f_b^{-1}(\Theta;w,\phi)\,. \quad (5)$$

While this is not the erosion from the original corner location, it does represent the degradation actually seen on the corner. A given amount of corner erosion can also occur for an infinite number of (w, Θ) values. Samples of constant d_{1b} for three angles ϕ are shown in Figure 5.

The three corner erosion cases in Figure 4 are determined by the corner angle and the threshold. They are generally independent of the PSF shape and width, w. The cases in Figures 4a and b have $\delta_c>0$ and thus occur when $\Theta<0.5$. The case in Figure 4b will occur for extremely small values of Θ. The erosion case in Figure 4c has an edge displacement of $\delta_c<0$ which will occur only when $\Theta>0.5$. As the corner angle ϕ increases, the case in Figure 4b occurs for a larger range of Θ.

For a white corner on a black background, the corner will also be eroded, but in an opposite manner from the black corners. The amount of erosion has the relationship

$$d_{1w} = \frac{-wESF^{-1}(1-\Theta)}{\sin(\phi/2)} + f_b^{-1}(1-\Theta;w,\phi)\,. \quad (6)$$

Samples of this are shown in Figure 6. It can be observed that the contours for black and white corners are symmetric to each other about the $\Theta=1/2$ line, similar to the relationship between δ_c and $-\delta_c$.

For both black and white corners, the angle between the edges affects the range of d_1 that will occur for a given PSF width. Larger angles, ϕ, will show less erosion for the same range of w.

3. Sample Characters

These two types of degradation, edge displacement and corner erosion, are present in various combinations for all scanned characters. To illustrate how much each affects characters, 12-point sans-serif font characters e, m, x and z are synthetically blurred. The characters are created at 600dpi "scanning" resolution and are shown at twice their standard size. These characters are created

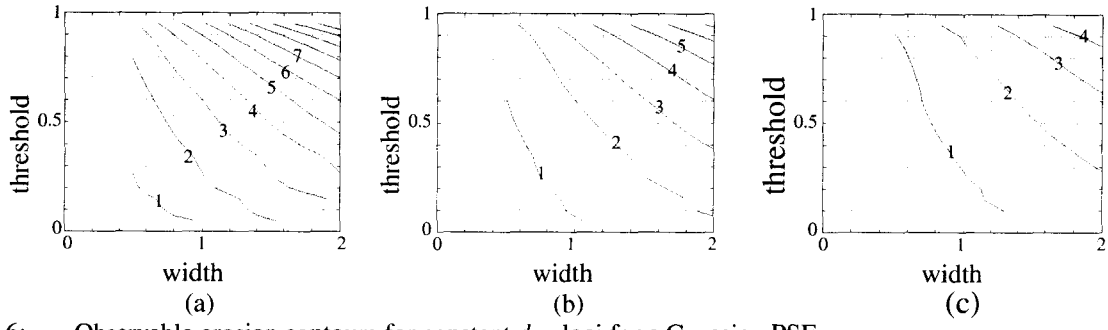


Figure 6: Observable erosion contours for constant d_{1w} loci for a Gaussian PSF (a) $\phi=\pi/6$ (b) $\phi=\pi/4$ (c) $\phi=\pi/3$.

	Edge Displacement:	$\delta_c = -2$	$\delta_c = -1$	$\delta_c = 0$	$\delta_c = 1$	$\delta_c = 2$
Synthetic Characters	Square Pulse	emxz emxz emxz	emxz emxz emxz	emxz emxz emxz	emxz emxz emxz	emxz emxz emxz
	Triangular Pulse	emxz emxz emxz	emxz emxz emxz	emxz emxz emxz	emxz emxz emxz	emxz emxz emxz
	Gaussian	emxz emxz emxz	emxz emxz emxz	emxz emxz emxz	emxz emxz emxz	emxz emxz emxz

Figure 7: Comparison of characters with common δ_c values. Three different (w, Θ) values produced samples for each PSF. (Note: characters as seen in the proceedings have extra degradation from printing of this paper.)

with (w, Θ) values to give constant edge spread and constant corner erosion. As each of these degradations can occur for multiple thresholds, PSF widths, and functional forms, each of these parameters are varied to show their effects.

3.1 Edge Displacement

Five edge spread values, -2, -1, 0, 1 and 2, were shown in Figure 3. Three (w, Θ) values were selected for each of these δ_c values for three PSF shapes and synthetic characters were generated, Figure 7. The range of the (w, Θ) values is large, but the characters appear quite similar due to the constant δ_c. Differences can still be seen among characters with a constant δ_c, particularly where the lines intersect. This is because the δ_c calculation is only valid when the edges are isolated from other edges. Please note that the characters, as seen in the printed proceedings, show additional degradations from the printing of the manuscript and the printing of the proceedings. This also holds for the characters in Figures 8 and 9.

3.2 Corner Erosion

For constant corner erosion, the angles present in the characters affect the corner erosion. The locus of (w, Θ) points that give constant corner erosion of d_1 for an angle ϕ_1 will not give a constant corner erosion for an angle ϕ_2. Figure 8 shows characters created with three (w, Θ) values selected to give the corner erosion values of d_{1b}= 1, 2 and 3 on the outer most black corners of the letter x. These corners have an approximate measure of 0.95 radians. In Figure 9, the (w, Θ) values were chosen to give corner erosions d_{1w}=1, 2, 3 for the white corners in the letter z, which are also approximately 0.95 radians.

The black corners in the x and the white corners in the z have approximately the same angle measure. Therefore the amount of erosion on the black corners of the x and white corners of the z have an inverse relationship. Characters in Figure 9 were created with the same PSF width as the characters in the corresponding locations in Figure 8, but the threshold was 1-Θ. As the amount of erosion in the black corners increase, some of the white corners increase and some do not. This is because the contours for constant corner erosion in white wedges are

Black Corner Erosion		$d_{1b} = 1$	$d_{1b} = 2$	$d_{1b} = 3$
Synthetic Characters	Square Pulse			
	Triangular Pulse			
	Gaussian			

Figure 8: Comparison of scanned and synthetic characters based on constant d_{1b} values based on the acute black corners on the tips of the letter x. (Note: characters as seen in the proceedings have extra degradation from printing of this paper.)

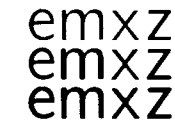
White Corner Erosion		$d_{1w} = 1$	$d_{1w} = 2$	$d_{1w} = 3$
Synthetic Characters	Square Pulse			
	Triangular Pulse			
	Gaussian			

Figure 9: Comparison of scanned and synthetic characters based on constant d_{1w} values based on the acute white corners on the interiors of the letter z. (Note: characters as seen in the proceedings have extra degradation from printing of this paper.)

the at the locations of the black corners flipped about the Θ=1/2 line. Therefore when following a contour for a black corner erosion, the white corner erosion will grow or shrink depending on the direction you move along the black contour.

Within the groups of characters with constant corner erosion, some characters have a wide stroke and some have a narrow stroke. This gives the characters a considerable variation in appearance between characters with the same corner erosion, much more than was seen for characters with equal edge spread and varied corner erosion. The feature which people notice most easily is the stroke width. When this is constant, characters appear similar. Characters with constant corner erosion do not usually have the same edge displacement, as can be seen in Figures 8 and 9.

4. Estimation Methods

While many methods are available to estimate the scanner characteristics from a grey-level scan [6-10,12-15], very little research has been completed on estimating scanner parameters from bilevel scans [2,4]. The scanner calibration methods that use grey-level information either directly or indirectly consider the profile of the blurred edge. Only the location of the edges is available in bilevel images. Bilevel images require new PSF width estimation techniques because the edge profile is no longer available. Here we are concerned with estimating δ_c and d_1 more than w and Θ, although knowledge of w and Θ can be used to estimate δ_c and d_1 via Eqs. (1), (5)

and (6).

4.1 Edge Displacement

In [3], I described a method of estimating the edge spread. This involved using star sector test charts. The edge spread is related to the number of pixels that are black in annular bands at a given radius relative to the number of pixels that were black in the original star image. If the original star image has bands of equal black and white of width $\tau(r)$, then

$$\delta(r) = \left(fr(r) - \frac{1}{2}\right) \tau(r) \ , \qquad (7)$$

where $fr(r)$ is the fraction of pixels in an annulus at radius r that are black. If the sector edges are separated by more than the support of the PSF, this amount will be constant, δ_c.

The edge displacement can then be estimated by counting the number of black and white pixels at each radius starting at the center of the star. When the quantity $\delta(r)$ becomes constant, δ_c is known.

In practice, this estimation method gives a relatively good estimate of δ_c. It is, however, sensitive to the quality of the star chart. If the star chart does not have equal sectors of black and white, the estimate of δ_c will not be accurate unless the variation is known and accounted for by modifying the formula. Star charts enable measurement of the edge spread δ_c, but star charts aren't available in most documents. In practice, this can be overcome partially by scanning a star chart before a batch of documents. This would not work when this analysis is extended to include multiple bilevel processes in series like the ones occuring when printing then scanning, or photocopying, unless the star chart was present in the original document. Edge displacement could be estimated from other features readily available in documents, such as characters, if the exact stroke width present before scanning is known. It is however, quite unlikely that, for a general document, the exact stroke width of any feature would be known *a priori*.

4.2 Corner Erosion

In [5], a method of estimating the erosion of a corner was presented. This was for the observed erosion and not the erosion from the original corner location. This also contained, at least partially, the edge spread estimate.

Measuring the distance between the extrapolated vertex point p_1 and the apex point p_2 will give us the erosion distance, d_1. The value of d_1 is specific to each corner angle. That angle, ϕ, can be measured from the image.

When ϕ and d_{1b} are measured from an image, a locus

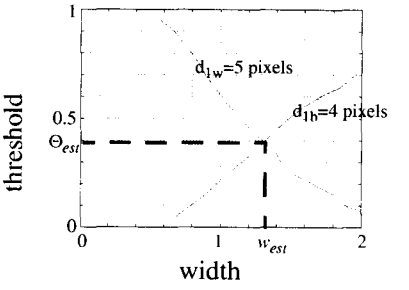


Figure 11: The system parameter estimate is where d_{1b} and d_{1w} loci intersect.

of (w, Θ) points can be found, like in Figure 5. The w and Θ values depend on the measured erosion distance, d_1. Small angles will have d_1 contours spaced close together so an error in measuring d_1 will make only a small change in the estimate of w and Θ. Larger angles are much more common in characters than small angles, but the d_1 loci are widely spaced for pairs of d_1 distances that differ by 1 pixel. Thus, an error of 1 pixel caused by noise on the apex pixel or phase effects will greatly affect the estimates for large angles.

When at least one black and one white corner are available, the difference in orientation of the constant erosion d_{1b} and d_{1w} loci for black and white corners is utilized to estimate the parameters. The intersection of the d_1 loci from black and white corners should occur at the (w, Θ) value for that scanner (Figure 11). This method will still work when d_1 data is collected from black and white corners of different angle measures, increasing the amount of data that can be used on a given page of text or in a given line drawing. Corners are readily available in most images to be used for estimation of the scanner parameters w and Θ.

5. Discussion

Two bilevel image degradations were introduced: edge displacement, δ_c, and corner erosion, d_1. These were related to the bilevel process variables PSF width, w, and binarization threshold, Θ. Each of these degradations can be caused by an infinite number of w and Θ values. Corner erosion has a different relationship to w and Θ if the corner is white on a black background as opposed to a black corner on a white background.

These degradations affect how a character looks after scanning. Characters with a constant edge displacement will have a similar look, even for different w and Θ values. Characters with a constant corner erosion will have a different appearance. This can be attributed to character similarity being gauged mostly by the stroke width.

Methods of measuring the amount of edge displacement and corner erosion have been described. Estimation of edge displacement requires a special test pattern, or at least one that is specified in great detail. Estimation of corner erosion requires only knowing that the corner

was originally made by the intersection of two straight lines. Therefore, the estimation of corner erosion is easier on common images and shows great promise for use in document analysis.

Both edge displacement and corner erosion can occur for an infinite set of w and Θ values. When estimating edge spread, it is less critical to estimate both w and Θ to characterize the system for the purpose of generating sample outputs from that system. When estimating the corner erosion, a good estimate of the PSF width and the binarization threshold are needed to generate synthetic characters corresponding to that system. The parameters w and Θ can be estimated by combining the corner erosion estimates from a black and a white corner.

A large number of pixels are used when estimating δ_c from star charts. The estimate of d_1 often relies on a single pixel at the tip of the rounded corner to influence the estimate. A large number of corners is needed to get an estimate with the same amount of averaging present in estimates of δ_c.

These two degradation parameters have distinctly different characteristics in how they affect an image and how they can be estimated with current techniques. There is a need to integrate the estimation of the two parameters. As corner erosion contains the edge displacement in its calculations, there is potential in continuing research in that direction. Documents usually are subjected to multiple bilevel degradations, so significant advantages will result from expanding this analysis to other bilevel processes such as printing, photocopying and faxing, as well as multiple combinations of these processes.

6. Acknowledgement

I would like to thank George Nagy for his useful ideas and comments that encouraged this work.

7. References

[1] Henry S. Baird, "Document Image Defect Models," Proc. IAPR Workshop on Syntactic and Structural Pattern Recognition, Murry Hill, NJ, June 1990, pp. 13-15. Reprinted in H. S. Baird, H. Bunke, and K. Yamamoto (Eds.), Structured Document Image Analysis, Springer Verlag: New York, 1992, pp. 546-556.

[2] Henry S. Baird, "Calibration of document image defect models," *Proc. of Second Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, Nevada, April 1993, pp. 1-16.

[3] Elisa H. Barney Smith, "Characterization of Image Degradation Caused by Scanning," *Pattern Recognition Letters*, Volume 19, Number 13, 1998, pp. 1191-1197.

[4] Elisa H. Barney Smith, *Optical Scanner Characterization Methods Using Bilevel Scans*, Doctoral Thesis, Rensselaer Polytechnic Institute, December, 1998.

[5] Elisa H. Barney Smith, "Estimating Scanning Characteristics from Corners in Bilevel Images," Proc. SPIE Document Recognition and Retrieval VIII, San Jose, CA, 21-26 January 2001, pp.176-183.

[6] M. Michael Chang, A. Murat Tekalp, A. Tanju Erdem, "Blur Identification using the Bispectrum," *IEEE Trans. Signal Processing*, Vol. 39, October 1991, pp. 2323-2325.

[7] F. Chazallet, J. Glasser, "Theoretical bases and measurements of the MTF of integrated image sensors," *Proc. SPIE Image Quality: An Overview*, Vol. 549, Arlington, VA, 9-10 April 1985, pp. 131-144.

[8] Luigi P. Cordella and George Nagy, "Quantitative Functional Characterization of an Image Digitization System," *6th International Conference on Pattern Recognition*, Munich, Germany, 19-22 October 1982, pp. 535-537.

[9] D. B. Gennery, "Determination of Optical transfer function by inspection of the frequency domain plot," *Journal of the Optical Society of America*, Vol. 63, No. 12, December 1973, pp. 1571-1577.

[10] C. A. Glasbey, G. W. Horgan, D. Hitchcock, "A note on the grey-scale response and sampling properties of a desktop scanner," *Pattern Recognition Letters*, Vol. 15, No. 7, 1994, pp. 705-711.

[11] Tin Kam Ho and Henry S. Baird, "Large-Scale Simulation Studies in Image Pattern Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 10, October 1997, pp. 1067-1079.

[12] S. E. Reichenbach, S. K. Park, R. Narayanswamy, "Characterizing digital image acquisition devices," *Optical Engineering*, Vol. 30, No. 2, March 1991, pp. 170-177.

[13] R. M. Simonds, "Two-dimensional modulation transfer functions of image scanning systems," *Applied Optics*, Vol. 20, No. 4, February 1981, pp. 619-622.

[14] P. L. Smith, "New Technique for Estimating the MTF of an Imaging System from its Edge Response," *Applied Optics*, Vol. 11, No. 6, June 1972, pp. 1424-1425.

[15] H. Wong, "Effect of knife-edge skew on modulation transfer function measurements of charged couple device imagers employing a scanning knife edge," *Optical Engineering*, Vol. 30, No. 9, 1991, pp. 1394-1398.

Multimedia

A Framework for Reliable Text Based Indexing of Video

R. Kasturi **S. Antani** **D. J. Crandall** **V. Y. Mariano**
Department of Computer Science & Engineering
The Pennsylvania State University
University Park, PA 16802
kasturi@cse.psu.edu

Abstract

In this paper we describe our recent research efforts towards reliable and automatic generation of indices for use in content understanding of video. Following our earlier research in temporal shot segmentation of video, we have developed a comprehensive system framework for segmenting an unconstrained variety of text from general purpose broadcast video. In addition, the framework also contains a novel tracking and a binarization algorithm. Also developed to be a part of the above framework are other modules, viz. a novel scene text segmentation method, and a novel text segmentation method which extracts uniform colored text from still video frames. We have thoroughly evaluated the methods which form a part of our framework against a fairly large dataset. The framework applies a battery of methods for reliable localization and extraction of text regions. Towards this, we have developed methods for fusing the results from different methods. More recently, we have extended our interest to localizing and extracting stylized text from video and determining the lifetimes of the video text events. Results from the above research are presented in this paper.

1 Introduction

The use of digital video is becoming increasingly ubiquitous. Today, many homes are maintaining personal media archives. Large media archives are also being maintained by several organizations with an interest in commerce, entertainment, medical research, security, etc. Additionally, there has been a growing demand for image and video data in applications, due to the significant improvement in the processing technology, network subsystems and availability of large storage systems. Use of digital video involves capture, compression, archival, indexing, retrieval, querying, browsing, transmission, and viewing. While capture, compression and archival issues are being addressed by better hardware, transmission is the subject of recent advances in communication technologies and viewing is something we are all used to. Indexing, retrieval, querying and browsing, on the other hand, will require automated methods to understand the content of digital video. Content-based information retrieval from such digital video databases and media archives is a challenging problem and is rapidly gaining widespread research and commercial interest.

For retrieval purposes the video may be either annotated and indexed manually or be indexed using automated content description methods. Not only is manual indexing a challenging and cumbersome task, but also suffers from possibly incomplete, subjective and summarial content descriptions. The latter method can solve many of these problems. Several automated methods have been developed which attempt to access image and video data by content from media databases [1]. Providing semantic access to visually rich and temporally linear information is a challenging task. A popular approach to address this problem has been to temporally segment video into subsequences separated by shot changes, gradual transitions or special effects such as fade-ins and fade-outs [2, 3]. A story board of events that occurred in the video can thus be created by selecting a key frame from each (or significant) subsequence. The video can now be queried with visual queries that use color, texture or activity. This is a pseudo-semantic approach to video content description, wherein the human interpretation of color, texture and/or motion define the content. The next step in content-based indexing of digital video is to localize, extract, and recognize objects contained in it. An example of such an object is the visual text appearing in the video data.

1.1 Text as a Video Index

There is a considerable amount of text occurring in video that is a useful source of information. The presence of text in a scene, to some extent, naturally

describes its content. If this text information can be harnessed, it can be used along with the temporal segmentation methods to provide richer content-based access to the video data. The text in video frames can be classified broadly into two large categories - *caption* or *artificial, overlay* text and *scene* text. Caption text comprises of text strings that are generated by graphic titling machines and composited on the video frame during the editing stage of production. This text could also be graphical elements with text contained within them or graphic effects using text. It is placed intentionally by the program editor to provide information of the subject being discussed. Examples of such text are found as credit titles, ticker tape news, information in commercials, etc. Scene text, on the other hand, occurs naturally in the scene being imaged. The image of this text may be distorted by perspective projection, be subject to the illumination conditions of the scene, be susceptible to occlusion by other objects, suffer from motion blurring etc. It can also be on planar as well as non-planar surfaces such as the text on soft drink cans.

A number of research efforts are on to create video storyboards or abstracts in a digital library context. Such efforts naturally concentrate on artificial text since they deal with video from content creators or producers, whose structure and intent are well known [4, 5]. Artificial text can hence serve as a key to the visual content. In addition, any scene text is also likely to be highly correlated with the story being depicted. Thus, the detection and recognition of text from unconstrained, general-purpose video is an important research problem. An indexing system that seeks to comprehensively label or index video by detecting, localizing and recognizing text in the frame must handle both kinds of text in digital video.

In this paper, we describe our research in indexing video through reliable localization and extraction of text in video. We have developed a multi-threaded framework for this purpose [6, 7]. This framework applies a battery of text extraction methods, on MPEG-1 video and JPEG images, in order to add reliability in segmenting text from video. Some of these methods are novel methods developed by us, some contain enhancements made by us on algorithms published in the literature, while others are our implementations of original work by other authors. We have also developed a novel scene text extraction algorithm [8] and a novel algorithm for detection of uniform colored text from video [9]. In addition, the framework also contains a novel tracking and a binarization algorithm [10]. We have evaluated the methods which form a part of our framework against a fairly large dataset [11]. From our evaluation, we conclude that the results of the algorithms vary greatly. Common elements affecting the results are the size of text, the contrast between the text and the background, stroke width, the background image of which the text is a part, etc. Such a scenario points towards developing methods for combining the strengths of a variety of text extraction methods for achieving better results. We have developed methods for fusing the outputs of various text methods and are in the process of enhancing it further [12].

Thus far, the video text segmentation methods have used the assumption that the size of the text remains rigid. The methods also assume that the text blocks move in a predictable fashion. There is a large amount of text appearing in video that does not comply with these temporal assumptions. Moving text effects are often used with caption text to attract viewer attention. Such effects may cause text to change size, perspective, inter-character (word) distance, or color over time. Text strings may rotate or spin. All of these effects would cause text extraction systems to fail. A large amount of scene text also violates these assumptions. We call such text as *stylized* text. It is clear that in order to handle such a wide variety of scene and caption text, more sophisticated text segmentation and tracking algorithms are required. We are in the process of developing methods to address such text. In addition, a system to extract text from video must also determine the *lifetime* or extent of the text event over time. The lifetime of the text event will mark the first and last frame at which the text appeared. This will enable the system to index the video better. We are in the progress of developing methods for determining these.

The remainder of the paper is organized as follows. Section 2 highlights other attempts for extracting text from video. In Section 3, we describe the system framework. The text localization methods are described in Section 4. Tracking and binarization methods are described in Section 5. Section 6 presents the performance evaluation results. Finally, we conclude with Section 7.

2 Related Work

This section presents methods for extracting text from images and video that are published in the literature. There has been a growing interest in the development of methods for detecting, localizing and segmenting text from images and video. There has been relatively more work done on the detection and recognition of artificial text, but even here the literature is sparse on work that deals with video. More work has been done on the extraction of text strings from images and many of the schemes for artificial

text recognition in video are modifications of work originally done for static images.

2.1 Caption Text Extraction

Yeo [13] has proposed a method for detecting caption text that involves computing differences between *a priori* selected corresponding regions of consecutive frames. Changes in this region are assumed to be due to caption events, large shot changes having been filtered out. Sato et al [14] describe a system for performing OCR on video caption text in the context of a digital news archive. Text is localized by looking for clusters of edge pixels that satisfy aspect ratio and other criteria.to increase its resolution. Messelodi and Modena [15], extract text from book cover images. They use simple homogeneity properties to separate text from other image components and further correct their extraction through estimation of orientation and skew correction of text lines. Li and Doermann [16] extract text from digital video keyframes. They use the heuristic that the texture for text is different from the surrounding background to identify text regions. Wavelets are used for feature extraction and a Neural Network is used for decisions. They also present an algorithm for tracking moving text. The tracker assumes that text is mostly rigid and moves in a simple, linear manner. Wu et al [17] describe a scheme for finding text in images. They use texture segmentation to localize text, edge detection to detect character strokes and join strokes to form text regions. Chaddha et al [18] have developed a method to detect text from JPEG images. The sum of the absolute values of a set of DCT coefficients is computed. This measure reflects the high spatial frequency content of blocks containing text. Zhong et al [19] like Chaddha et al use the DC coefficients available in the MPEG I-frames. Those coefficients that highlight the vertical and horizontal frequencies are summed and then thresholded to detect text blocks.

Shim *et al* [20, 21] present a method to detect caption text from MPEG compressed video frames by identifying homogeneous regions in intensity images, forming positive and negative images by double thresholding and applying heuristics based on text characteristics for eliminating non-text regions. The text regions are validated using the temporal redundancy property of text in video. In [22, 23] methods for locating text in complex color images is presented. They have proposed a method for quantizing the color space using peaks in the histogram before performing segmentation. Their other method uses the heuristic of high horizontal spatial variance to localize text. Kim [24] also proposes text localization method for video images similar to the spatial variance method of Zhong [22]. Suen and Wang [25] propose a method for segmenting uniformly colored text from a color graphics background. They assume that all characters have the same color, an assumption that does not hold in general. Hase et al [26] propose an extraction algorithm for character strings. Their approach is directed towards binary document images. Lopresti and Zhou [27] use a similar method to segment text from images found on the Internet. Hauptmann and Smith [4] localize text in video using the heuristic that text regions consist of a large number of horizontal and vertical edges in spatial proximity. Lienhart and Stuber [28] describe a system for automatic text recognition in digital video that works on pre-title sequences, credit titles and closing sequences with title and credits. LeBourgeois [29] presents a system for multifont OCR from gray level images. The method is a modification of run length smearing to segment and recognize text. Mitrea and de With [30] propose a simple algorithm to classify video frame blocks into graphics or video based on the dynamic range and variation of gray levels within the block. Gargi et al [6] describe an algorithm for localizing text in a video frame. The method localizes bounding boxes of horizontal text strings, assuming that each character is composed of a number of segments and that the characters within the string are separated. Other approaches for detecting text in images and video are found in [31–36].

2.2 Scene Text Extraction

Ohya et al [37] describe a method to recognize characters in scene images. They use local gray level thresholding to segment the image and localize text regions by looking for high contrast, uniform gray level of a character, and uniform width. They also use the results of the OCR stage to improve their extraction result—if the Chinese OCR algorithm they use does not find a good enough match, they reject the character candidate region. There is work on the recognition of vehicle license plates [38, 39] from video which shares some of the characteristics of scene text. However, these approaches make restrictive assumptions on the placement, contrast or format of the license plate characters. Cui and Huang's approach [39] takes the advantage of using the information from multiple frames and also correcting for perspective projection distortion. Winger et al [40] discuss the segmentation and thresholding of characters from low-contrast scene images acquired from a hand-held camera. Their data set includes images with low contrast, poor and uneven illumination. Our implementation of the algorithm does not find it to perform well in general purpose video. Communications with the authors suggests that the parameters were fine tuned to individual images on a small dataset.

3 System Framework

A study of the literature reveals that no complete video text extraction system has been developed. Additionally, it is seen that no single algorithm is robust for detection of an unconstrained variety of text appearing in the video [11]. Most methods have been developed to extract text from complex color images and have been extended for application to video data. However, these methods do not take advantage of the temporal redundancy in video. Further study of the methods presented in the literature, presented in detail in Section 2, reveals that the methods assume that the text regions are in high contrast with the background, are composed of one consistent color or gray-level or form a major component in the image. In general, the use of a few rigid assumptions about the nature of text in video forms a weak heuristic. This study of the state of the art was the motivation behind the development of the framework for reliable extraction of text from video.

The video text extraction problem is divided into four main tasks, viz. detection, localization, tracking and binarization. The detection and localization tasks have been merged because there is a significant overlap between the detection and localization processes. A spatio-temporal algorithm fusion module has been proposed for aggregating the decisions of the multiple localization algorithms over multiple frames. The tracking stage is used for temporally validating the text localization. Also, the caption text and scene text segmentation tasks are separated. Extraction of scene text uses an algorithm developed by Gandhi [41] that uses the assumption that scene text lies on a plane in the 3-D world and that camera motion exists. Our approach is to use a battery of different methods employing a variety of heuristics for detecting, localizing and segmenting both caption and scene text. The system also takes advantage of the temporal nature of video and uses the fact that the text data lasts over several frames for providing robust text detection, specifically by performing algorithm fusion in a spatio-temporal manner. The system framework is described in the following section.

An object oriented approach has been adopted in the design of the software prototype. A multi-threaded design has been adopted to allow maximum flexibility. The reasoning stems from the desire that the detection and localization processing of new frames not stall because of a time consuming tracking or segmentation of older frames. The POSIX standard pthreads are used which are lightweight and portable between IRIX, Solaris and other flavors of UNIX. Figure 1 shows the design of the framework. The main components are:

- The **control thread** is the overall data and process control center. It creates a **Frame Queue** and spawn a **readframe** thread. This thread is also responsible for spawning the desired number of **detection-localizion threads**, one per algorithm, the **fusion thread**, the **tracking thread**, and the **binarization thread**.
- The **Frame Queue** contains all the information for both unprocessed and partially processed frames. It provides a data repository for all the threads in the system while maintaining the temporal seqentiality of the data.
- The **readframe thread** reads frames from video stream and adds it to the **Frame Queue**. It provides an abstraction to the data stream type. It presently can read in MPEG-1 bitstreams and JPEG images and can be extended to handle Motion JPEG compressed video.
- The **detection/localization threads** are the implementations of the detection and localization algorithms selected from the literature and the novel algorithms developed here.
- The **tracking thread** tracks a given bounding region over a set of frames.
- The **segmentation thread** binarizes a localized text instance to make it suitable for OCR.
- The spatio-temporal **algorithm fusion thread** fuses the results of various methods to result in a single text instance.
- The **output thread** uses the information in the **Frame Queue** to write output in one of multiple formats: a binarized image suitable for OCR, the original frame with localized text marked by a box, or a ViPER[1] compatible ASCII data file which lists the bounding boxes for each element, etc. ViPER (Video Performance Evaluation Resource) is a Java based tool developed at the laboratory for Language and Media Processing in Center for Automation Research at the University of Maryland for ground-truthing of video and evaluating the performance of content extraction methods.

4 Text Localization Algorithms

Of the methods seen in the literature, only those methods which we judged to be promising were selected. The selection was based on their applicability to general purpose video, use of features, ease of

[1] ViPER: `http://documents.cfar.umd.edu/LAMP/Media/Projects/ViPER`

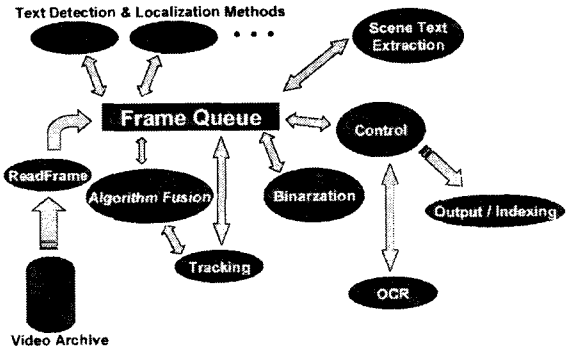


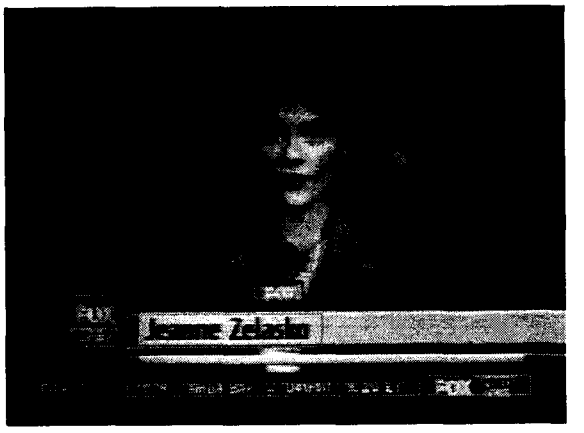
Figure 1: Text Localization/Extraction Framework

implementation and speed of detection. In addition to work done by others, we also include algorithms developed by us for evaluation. The algorithms chosen for evaluation are: **Method A** [6], **Method B** [29], **Method C**: based on initial idea published in [30], **Method D**: enhanced from initial idea published [18], and **Method E** [9]. Details on other methods can be found in the original publications cited above. We include details on the modifications here. Sample text localization results are presented in Figure 2.

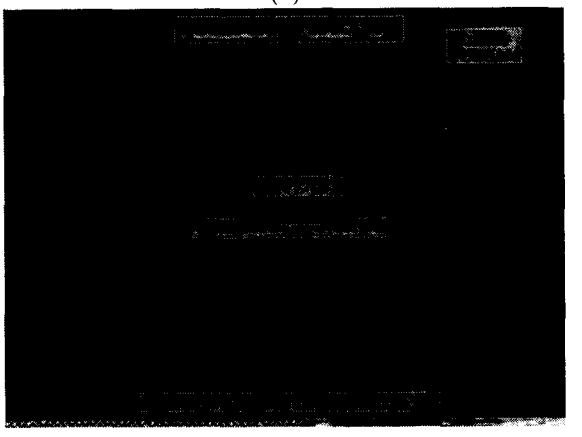
4.1 Modified Algorithm : Method C

A simple algorithm [30], originally proposed to classify video frame 4x4 pixel blocks into graphics or video based on the dynamic range and variation of intensity within the block. This method was developed to achieve higher compression for TV picture signals. The method operates on the premise that in the graphic regions in the frame, many adjacent pixels have the same luminance values or have regions of very high dynamic range. The dynamic range of a block is defined as the absolute difference between the maximum and the minimum intensity in a 4x4 block.

This method is modified to classify blocks as text or non-text. As with the original method the number of pixels in a 4x4 block that have similar gray levels is counted and the dynamic range is computed. If the number of gray level blocks is less than a parameter and the dynamic range of the block is either greater than or a distinct thresholds or is 0, the block is classified as a text block. This change in condition follows from the fact that text has a high number of edges. Thus, the number of pixels with similar intensity levels will be small and the dynamic range will be zero only on the character stroke which typically has near uniform intensity. Additionally, the modified method excludes the boundary regions from its operating space on the frame image.



(a)



(b)

Figure 2: Sample Text Localization Results

4.2 Modified Algorithm : Method D

This method [18] was originally proposed for classifying JPEG image blocks as text or non-text. It has been modified to work on MPEG-1 I-, B-, and P-frames and determined appropriate threshold empirically on our data. The method has been further refined to use an iterative thresholding scheme to reduce the number of false alarms. MPEG B- and P-frames need to be decompressed before Discrete Cosine Transform (DCT) can be reapplied to them for use with this method. For this the `fastct`[2] algorithm [42] was used.

The method uses texture energy to classify 8x8 blocks as text or non-text and works as follows. A subset of the 64 possible DCT coefficients produced during the MPEG encoding process is chosen. For each block, the sum of the absolute values of these coefficients is compared to a threshold to categorize it as text or non-text. Using these blocks as seed blocks, a series of decreasing thresholds is iteratively applied from high (150) to low (30) and the appearance of more and more text blocks as the threshold is lowered is observed. Blocks that are classified as

[2]fastct: `http://dmsun4.bath.ac.uk/dcts/fastdct.html`

text at a particular threshold are left in if they also have a 8-neighbor that was classified as text at the previous higher threshold. The motivation for doing this is that text regions usually have at least one of their component blocks detected at 150. So the text region can be enlarged by lowering the threshold without creating as many false positives. Any blocks with no neighbors on the left or right are removed. This is from the heuristic that the text is horizontal or if vertical, is fairly wide. Any other blocks which appear to be due to a sharp luminance change between two large homogeneous regions are discarded. This is done by computing the mean of average luminance given by the DC term of the DCT coefficients three blocks on either side of a target block. The mean energy of these blocks is also computed. If the average luminance of the three on the right is greater than that of the left by a certain threshold, and the energies of the blocks are below a threshold, we conclude that this block was found because of a sharp luminance cliff and it is discarded. Finally the aspect ratio constraint is used to filter out false alarms.

4.3 Novel Method : Method E

The algorithm visits every *Interval* rows in the image. *Interval* is set small enough to be able to detect very small text and large enough so as not to consume too much time. In our experiment we set $Interval = 3$. Given a row R on the image, we want to determine whether or not R passes through the middle of a text region.

Clustering in $L^*a^*b^*$ space: The pixels of R are transformed and clustered in the perceptually uniform $L^*a^*b^*$ color space using hierarchical clustering. The algorithm first assigns each pixel as a cluster and the distance of pairs of clusters are stored in an array. Two clusters A and B are merged if $||\mu_A - \mu_B||$ is minimum and for each pixel p in $A \cup B$, $||p - \mu_{A \cup B}|| < MaxClusterRadius$, where μ_Z is the mean $L^*a^*b^*$ vector of cluster Z and $||\cdot||$ is the weighted Euclidean norm. The weighted norm was used to achieve a slight invariance to lightness (weights: $L^* = 0.8, a^* = 1.1, b^* = 1.1$). In our experiments, we set $MaxClusterRadius = 10$ (ranges: $L^* = 0 \ldots 100$, $a^* = -97 \ldots 88$, $b^* = -100 \ldots 88$). Merging continues until no two clusters can be merged.

4.3.1 Determining bounding rows

Each cluster C is tested to see if it contains pixels belonging to text. Locating the bounding rows (top and bottom rows of text) is the first step (Fig. 3). The cluster points are marked back on row R to create streaks $S_i, i = 1 \ldots N_s(numberofstreaks)$ of pixels in the row R. Then all pixels in the entire image are examined and each pixel with a value within the range of values represented in the cluster are colored with a value of T. All other pixels are marked T'.

We now try to find out if there are bounding rows above and below R which may contain horizontal text. Given a pair of adjacent streaks S_i and S_{i+1}, we find R_a - the first row above R in which the segment covering S_i and S_{i+1} is colored T'. We also find R_b - the first row below R in which the segment covering under S_i and S_{i+1} is colored T'. The R_a of each pair of adjacent streaks is computed and collected in an alignment histogram H_a, where the bins are the rows of the image. H_b is computed in the same way by taking all the R_b's. We declare the existence of a bounding row B_a if at least 60% of the elements in H_a are contained in three or fewer adjacent histogram bins. B_b's existence is computed in the same way from H_b. If B_a and B_b exists, *height* is defined as their difference.

If the cluster C contains text pixels, then B_a and B_b would mark the text block's upper and lower row boundaries, and *height* would define its vertical dimension. Figure 3 illustrates the computation of B_a, B_b and *height*.

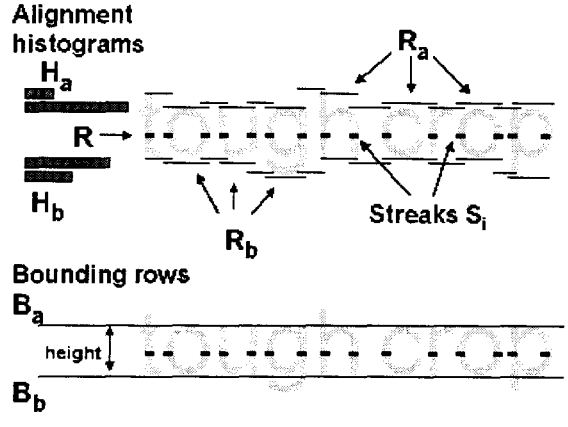


Figure 3: Computing the bounding rows. One of the color clusters in row R are marked as short streaks and pixels of the text "tough crop" lie within the range of values of the cluster. Each segment of the bounding top (R_a) and bottom (R_b) rows are shown separated for clarity even when they are actually on the same row.

Finding text blocks: We look for text blocks using heuristics on *height* and the lengths and gaps of the short streaks. Streaks longer than *height* are discarded and added to the gaps. Gaps longer than *height* are considered not part of a text block. The remaining regions are now smaller blocks with short streaks. If a block's width is greater than $1.5 * height$

and the number of short streaks inside is greater than 3, then it is considered a text block, otherwise it is discarded. Finally, the text block is expanded a few pixels to the left and right to ensure full coverage of the characters at the ends.

Figure 4 shows how the text block "For generations" is detected. The pixels of row R (passing through the middle of text) are clustered in color space. One of the color clusters is marked black. Pixels in the image having similar color as the black ones are marked white. On the left side of the image, the two alignment histograms H_a (above R) and H_b (below R) are used to mark the bright bounding rows B_a and B_b. The short streaks marked black and the *height* between the bounding rows are used to find the text block. The two black streaks on the right were not included in the text block because their gap from the other streaks is greater than *height*.

Figure 4: Analysis of a video frame and detected text block.

Fusing the detected text blocks: It was observed that other color clusters were caused by the presence of text. The characters' color "shadows" and the pixels in the transition from text foreground to background result in other detected text blocks which largely overlap with the foreground text block. All the detected text blocks are fused (set union) to come up with the final regions of text.

4.4 Localization of Stylized Text and Event Determination

It is observed that while the size, orientation, color, *etc.* of a text event may change over time, the basic shape of its characters remains constant. This property can be exploited to determine whether two text boxes correspond to the same text event. We analyze two consecutive frames at a time. First, the text box localization algorithm, Method D, described above is applied to each frame. Oriented text instances are made horizontal by applying a simple rotation transformation. A text binarization algorithm is next applied on each text instance. We used the binarization algorithm developed in our earlier work [10]. This algorithm is tailored for the special challenges of binarization of text in video frames, including low resolution, complex background, and unknown text color. Connected component analysis is performed on the binarized text to locate individual characters. The contour of each character is traversed and stored as a chain code. Each chain code is then parameterized as two 1-D functions $\theta(t)$ and $r(t)$, that represent the angle and distance of each point on the boundary from a reference point, respectively. A Gaussian filter is then applied to both functions,to smooth out any noise introduced by imprecise binarization. The resulting functions represent a signature of the shape of a given character. From this shape, feature points are extracted. We use the points of maximum curvature (critical points) as our features. Zhu and Chirlian's critical point detection algorithm [43] is used in our implementation.

The row and column coordinates of features points within each detected text rectangle are normalized. Text boxes within the two consecutive frames are then analyzed as follows. For each text box in the first frame, its normalized feature point locations are compared to the feature point locations of every text instance in the second frame. The feature point location error between each pair is computed, and the pair with the lowest error is chosen. If the lowest error is below a threshold, the two text instances are declared to belong to the same text event. Otherwise, the text event's lifetime is assumed to end with the current frame. Any text instances left unpaired in the second frame are assumed to be the start of a new text event.

5 Tracking and Binarization

5.1 Binarization Module

This section describes the binarization module of our system. The goal of the binarization module is to separate the pixels of a localized text region into categories of text and background. The output of the module is a binary image of the localized region suitable for input into an OCR system. For document images, simple thresholding is typically sufficient to convert a gray scale image into a binary image suitable for OCR. This technique assumes that the text and background colors are uniform (typically black on white) so that the image histograms are bimodal. In video, however, text often appears against complex, nonuniform backgrounds. The text color may also vary due to uneven illumination of scene text, antialiasing, or due to bleeding caused by video com-

pression. These problems are further compounded by the low resolution of video images, in which character strokes may be two pixel or less in width. Due to these factors, it was found that algorithms which rely on histogram bimodality [29, 17, 44, 45]) are generally unsuccessful for video images.

As with the detection and localization modules, the binarization module uses a number of different algorithms. After a preprocessing step, an initial binarization of the region is created. The binarization is then refined by examining other properties of the region, including stroke width, color, character size, character spacing, gray scale topography, and shapes. Following is a detailed description of each step.

- **Preprocessing**: Given a localized text box, the region is first pre-processed by stretching the gray scale contrast [29]. This allows binarization to succeed with low contrast text.

- **Binarization**: Logical level binarization algorithm proposed by Kamel and Zhao [46] has proven to be fairly successful for this step. The logical level algorithm was developed to extract character strokes from complex backgrounds in document images (for example, cash amounts from noisy check images). Upon experimentation, it was discovered that it also works well for extracting character strokes from gray scale images of video frames, provided that the text gray levels are darker than the the background. If the text is lighter than the background, the inverse of the frame must be taken before applying this algorithm.

  Unfortunately, determining whether the text is lighter than or darker than the background is a nontrivial problem. Other systems have handled this problem by making assumptions about the text color [47], by examining the pixels along the edge of the text box and assuming they are the background colors [37], and by examining both light and dark strokes and keeping those whose orientation and connected component size fit the characteristics typical of text strings. Methods described in [23, 17] perform poorly for text appearing against complex backgrounds. A variant of the method published in [15] is selected. The method performs logical level binarization on both the positive and the negative of the video frame, and the decision of determining the correct polarity is delayed until later. Connected components are then found in both binarized images.

  In general, logical level creates good binarizations of localized text. However, it can miss character strokes which have a very low contrast with the background, and it can include non-text pixels which exhibit some characteristics of character strokes. Color, size, spatial location, topography, and shape are used to refine the result of the binarization.

- **Color Clustering**: It is reasonable to assume that the characters of a text string are of uniform color. However, due to bleeding effects caused by low-resolution capture and compression, the actual pixel colors may vary significantly. Color clustering is used to allow for these effects. The text region is first quantized to reduce the color space, and then the complete-link algorithm [48] is used to cluster in the $l^*a^*b^*$ color space. The results of clustering may be used to refine the output of the logical level algorithm. Currently, components containing many different clusters are considered noise and removed [49]. Results of clustering could also be used to add or remove pixels from a given component.

- **Size filtering**: Connected components are filtered based on their size and aspect ratio. Very small components (with area less than about 12 pixels) are eliminated since they typically represent noise, or, if text, are too small to be recognized by an OCR system. Very large components and components with extreme aspect ratios are also removed.

- **Positive or negative image selection**: As mentioned earlier, it is not known *a priori* whether the text is lighter than or darker than the background in a gray scale video frame. Therefore the binarization and filtering steps were conducted on both the original video frame and its inverse. This approach has also been taken by [23, 17, 15]. These methods take the union of both image polarities (after applying heuristics to reduce noise) obtained as a result of binarization. Unfortunately, this results in too many false alarms. However, unlike the other systems using this approach, this method includes a localization module separate from the binarization module. It is reasonable to assume that all text in a localized bounding box is either darker than or lighter than the background. The binarized images are examined and a choice is made based on statistical information computed from the components of each image, such as the similarity in character height, character aspect ratio, vertical position, and horizontal spacing. The image with the more text-like features is then chosen.

- **Spatial location filtering**: Non-text components present in the binarization can be further reduced by introducing spatial constraints about character location. It is observed that most text in video is oriented horizontally, so components that are not located along horizontal lines with other components are eliminated. Components are clustered based on vertical position, and clusters with few components are then eliminated. This could be generalized to allow for non-horizontal text by, for example, using the Hough transform [50] or Messelodi and Modena's slope histogram method [15].

- **Topographical analysis**: While the logical level algorithm works well for most font sizes, it fails to capture the detail of very small fonts (with stroke widths near 1 pixel) effectively. For these fonts, we a topographical analysis algorithm [51] that operates on the gray scale image region is applied. Pixels that were chosen by the logical level algorithm and that correspond to a peak or ridge in the topography are used as the final binarization. This method serves to thin the binarization and produces cleaner output for small fonts.

- **Shape analysis**: It is observed that some common non-text objects satisfy the heuristics used by the detection and localization algorithms, such as uniformity of size, color, stroke width, spacing etc. Using the similarity of the shapes of connected component within a region serve as an indicator of text. It is noted that regions with nearly identical shapes are usually not text. However, the shapes of a given script are expected to be somewhat similar, so an area with very diverse shapes are also unlikely to have text. Currently these comparisons are based on simple statistics of the shapes, such as number and density of critical points along the contours. A simple contour-following algorithm is used to find the outline of each connected component, and this data is parameterized to polar coordinates. Zhu and Chirlian's algorithm [43] is used to locate the critical points of the contours.

5.1.1 Results

The results of binarization of localized text regions are shown in Figure 5.

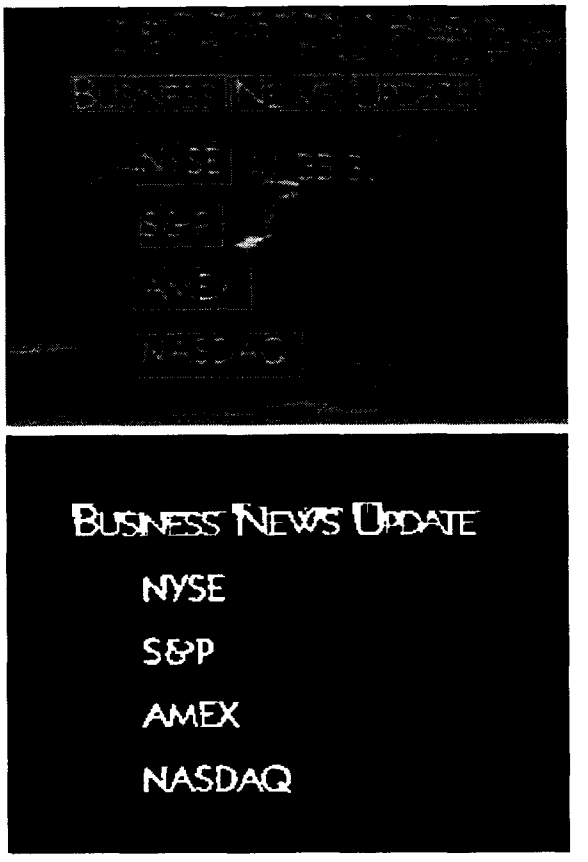


Figure 5: Sample Binarization Result

5.2 Text Tracking Module

Given the goal of automatically extracting text from video, it is not immediately apparent that a tracking component should be present in the system. However, it is needed as a verification of the localization algorithms. A detected text box that is changing in shape or that is moving (especially moving non-uniformly) may be eliminated as being due to noise unless its motion is verified by tracking. For scene text, zooming and rotation of the camera may alter the size and orientation of the text box. Erroneous discarding of localized regions can be avoided by using this knowledge. Finally. the intended application of this system may not always be to run in a completely automatic mode. In some scenarios, e.g., a ground truthing one, it may be desirable to have a human mark initial text boxes and have the system merely track it with time. The approach used for developing a tracker is based on methods described by Nakajima *et al* [52] and Pilu [53] with substantial modifications. Unlike the algorithm described by Li and Doermann [16] which uses only correlation within a search window (template matching) to track text, with the consequence that tracking is slower than localization, the adopted method also uses the motion vectors in the MPEG-1 bitstream.

The tracking algorithm assumes the availability of an initial horizontal bounding rectangular region to track. Multiple regions may also be specified. If the input is an MPEG bitstream, the method skips to the next available P-frame and extract the motion vectors for macroblocks that point to any

macroblock partially or completely within the initial box. Some macroblocks may be intra-coded in which case they cannot be used. Only those vectors that are 2 pixels or larger are considered. Next, spatial constraints are applied. This step eliminates stray motion vectors retaining only those that are similarly oriented. The motion vectors that point to the initial text box region are clustered and the largest cluster is assumed to belong to the new position of the moving text box. Further, flat vectors are deleted. This is achieved by computing edges using a kernel. Edgels[3] are statically thresholded and a count of remaining significant edgels is made. A macroblock is considered flat if it has less than 4 such significant edgels. In [52], the authors compute the absolute sum of the first 20 DCT coefficients and the last 60 coefficients and retain a block only if either of these is greater than an empirically determined threshold. This approach failed to provide satisfactory results.

Assuming that the text box undergoes rigid movements, the average motion vector for the region is computed. A correlation match is performed over a small neighborhood of the predicted text box region (the region in the P-frame). To ensure that the text is matched and not the background, the luminance gradients at each pixel (as computed by the Sobel operator) are compared against those in the initial text box. With subsequent P-frames an estimate of the block motion velocity is computed. The velocity is used to predict the position of the text box in the current P-frame. This region is used if it has a better correlation match than the region found using the motion vectors. If the correlation results in a high value then the neighborhood area is relaxed and the step is repeated. At the next I-frame, a predicted text box is obtained by averaging the motion from the last P-frame and the next P-frame, and then a correlation search is performed to find the exact position of the moving text box.

It is important to note that text may leave the frame or an initial text string may grow as more text belonging to the same string enters the frame. In the former case, the text box is resized appropriately. The latter situation is detected in [52] by looking for intra-coded macroblocks at the edge of the frame. If these are present, the authors hypothesize an object entering the frame. However, that paper deals with large, solid objects whereas we are tracking relatively small text regions with a possibly static background showing through. A different approach is adopted in this module. The number of edgels found along the edge of the frame is compared to the number found inside the text box. If these counts are comparable, text is assumed to be entering the frame, and the text box is grown to accommodate them. The tracker discards a block on complete failure of determining a suitable candidate region.

[3]Edgel is an edge pixel.

6 Performance Evaluation

Unlike the evaluation of automated methods for detection and localization of video events and objects contained within the imaged scene, the evaluation of text detection and localization methods presents interesting challenges. For example, when evaluating video shot change events [3], it is sufficient to detect at which frame a shot change (or other video transition event) occurred. In case of localization of vehicles, faces or other objects a tightly fitting bounding region is usually enough to perform a fair evaluation.

In case of automated text detection and localization methods, however, the degree of correctness is difficult to determine. This is because the the intent of text detection and localization is to recognize it for indexing, retrieval and other purposes. Also, humans tend to identify the text contained in the video as characters and words along a line, sentences, and paragraphs. Unfortunately, the algorithms that detect "text-like" regions within the video frame do not take this approach into consideration when applying the heuristics. They detect small regions that contain text and the size of the region (tightness of fit) is dependent on the size of the operating element used by the algorithm. For example, algorithms that operate on MPEG DC coefficients, will result in regions along 8x8 block boundaries, while those that use horizontal windows will have other boundaries. In order to obtain a commonality for evaluation, we evaluate the methods at the lowest common denomination, i.e. at the pixel level. Every pixel belonging to a text region, as detected by the algorithm as well as in the ground truth, is labeled as a such. All other pixels are labeled as non-text pixels. Evaluating the performance of the methods at the pixel level eliminates any issues related to the size operating elements of each method.

Unfortunately, the ground truth is usually marked by rectangular bounded regions which include the inter-character and sometimes inter-word non-text pixels. Also, non-text pixels surrounding the characters but within the ground-truth bounded region are considered as text pixels. Thus, if an algorithm is very accurate and detects the text but not the surrounding or inter-character pixels, it suffers a penalty for being very precise in the form of a low recall (higher missed detections). Conversely an algorithm which operates on large blocks actually detects the text correctly but has a looser region

boundary (due to operating block size) suffers the penalty in the form of low precision (higher false alarms). Thus, in a sense, the algorithms are being evaluated unfairly. It is necessary to allow a degree of subjectivity in evaluating these methods, which is to evaluate them based on their ability to detect each text event. We are in the process of developing such an evaluation method.

6.1 Test Data

Our test consists of 15 MPEG-1 video sequences totaling 10299 frames. The sequences were captured at 30 frames per second and encoded in MPEG-1 with a 352x240 frame size. The sequences are portions of news broadcasts and commercials from various countries. The test database is challenging due to the poor quality and low contrast of these broadcasts. Text appears in a variety of colors, sizes, fonts, and language scripts.

The ground truth was performed frame-by-frame by humans using the ViPER tool. Bounding text box size, position, and orientation angle were specified to pixel-level accuracy. All regions distinguishable as text by humans were included in the ground truth, including text too small or fuzzy to be actually read but nevertheless identifiable as characters. Closely spaced words lying along the same horizontal were considered to belong to the same text instance. Separate lines of text were kept separate. The ground truth contains a total of 133 temporally-unique caption text instances (36302167 ground-truth pixels) and 79 scene text instances (57532887 ground-truth pixels). There are 212 text events in total.

6.2 Evaluation criteria

The ground truth defines tightly-bound text boxes for each frame. A good detection/localization algorithm would (ideally) produce similarly tight boxes. To evaluate the accuracy and tightness of fit of an algorithm's output, the pixel areas of the text regions in the ground truth are matched with the detected text regions. The evaluation is a frame-by-frame, pixel-by-pixel comparison of algorithm output with the ground truth. In case of non-horizontal oriented scene text, All pixels within the oriented bounding region are considered. During evaluation, each pixel in the test database is placed into one of three categories:

- **Detection:** Pixels belonging to text regions in the ground truth and regions identified as text by the localization algorithm.
- **False Alarm:** Pixels identified by the detection algorithm but not belonging to text regions in the ground truth.
- **Missed Detection:** Pixels belonging to the text regions in the ground truth and not identified by the algorithm.

The performance of an algorithm is quantified by its recall and precision as defined in Equation 1. Note that this pixel-level evaluation is very strict. Most actual applications would not require such precise localization. However our pixel-level criteria provides an easily measurable basis by which the relative performances of algorithms may be compared.

$$Recall = \frac{detects}{detects + missed\ detects}$$
$$Precision = \frac{detects}{detects + false\ alarms} \quad (1)$$

6.3 Results and Discussion

This section presents the results of the performance evaluation of the selected text detection and localization algorithms. Most of the parameters for the methods were kept as described in the original publication. Only those parameters which were highly dependent on the dataset were tuned on a small subset of the test dataset (approx. 1000 frames).

Table 1 presents the caption text detection and localization performances, while Table 2 shows evaluation results for scene text, for the five algorithms on the entire test dataset. The table shows the raw numbers of total number of text pixels in the ground truth, the detected, false alarm, and missed detected pixels, along with computed recall and precision rates.

The results show that for caption text, overall Method D produces the highest precision rate of the individual algorithms, while the precisions of the other algorithms are comparably similar. Method E shows the highest recall. For scene text, Method D has the highest precision followed by Method E. Other methods have comparably similar results. Method E also has the highest recall for scene text. The test database contains some very challenging scene text instances. For applications in surveillance and navigation, detecting scene text would be important. In other applications, such as video indexing, detecting scene text may not be important or even useful. Therefore scene text and caption text were evaluated separately. All of the algorithms perform better for caption text than the scene text.

The recall and precision rates of the algorithms in our evaluation are relatively low and perhaps highlight the need for better text detection and localization algorithms. A solution to improving the precision and recall values of the methods is to apply algorithm fusion to combine the outputs of multiple existing algorithms to produce better outputs.

Algorithm	Text Pixels	Detects	FAs	MDs	Precision	Recall
Method A	36302167	14461593	62125359	21840574	39.84%	18.88%
Method B	36302167	14894707	45627542	21407460	41.03%	24.61%
Method C	36302167	22663915	156512965	13638252	62.43%	12.65%
Method D	36302167	26955906	119769022	9346261	74.25%	18.37%
Method E	36302167	17534049	35101417	18768118	48.30%	33.31%

Table 1: Overall Detection/Localization Performance : Caption Text

Algorithm	Text Pixels	Detects	FAs	MDs	Precision	Recall
Method A	57532887	10016556	66570396	47516331	17.41%	13.08%
Method B	57532887	7278171	53244078	50254716	12.65%	12.03%
Method C	57532887	27062384	152114496	30470503	47.04%	15.10%
Method D	57532887	22207563	124517365	35325324	38.60%	15.14%
Method E	57532887	13878758	38756708	43654129	24.12%	26.37%

Table 2: Overall Detection/Localization Performance : Scene Text

Method	Frames/sec.	Sec./frame
A	0.64	1.56
B	3.1	0.32
C	5.8	0.17
D	2.3	0.44
E	0.01	100

Table 3: Approximate algorithm running time.

Each algorithm uses an independent set of features and heuristics and so a fusing of outputs of multiple algorithms is likely to be beneficial.

6.4 Running time

Table 3 gives approximate running times for our implementation of each of the algorithms on an dual 270MHz. IP30 R12000 MIPS processor SGI Octane workstation. The times include overhead resulting from I/O and MPEG stream decompression. The times reported above can be improved since our implementations have not been fully optimized. Our implementation of the Method D operating on I, P and B frames was found to be the fastest (2.3 frames/sec.) as shown in the table. This method applied to I frames only clocked at 10.9 frames/sec. The increase in the processing time is because P and B frames need to be completely decompressed before Discrete Cosine Transform can be applied to them.

7 Conclusions

We have developed a system for extracting and segmenting an unconstrained variety of text from general purpose broadcast video. We have thoroughly evaluated the methods which form a part of our framework against a fairly large dataset. We have developed methods for fusing the results from different methods. More recently, we have extended our interest to localizing and extracting stylized text from video and determining the lifetimes of the video text events. We plan on developing methods for matching localized text regions based on shape and color properties. We also plan on developing methods for recognizing the localized text events to enable retrieval based on search strings.

References

[1] S. Antani, R. Kasturi, and R. Jain. Pattern Recognition Methods in Image and Video Databases: Past, Present and Future. In *Joint IAPR International Workshops SSPR and SPR*, number 1451 in Lecture Notes in Computer Science, pages 31–58, 1998.

[2] U. Gargi, R. Kasturi, and S. Antani. Performance characterization and comparison of video indexing algorithms. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 559–565, 1998.

[3] U. Gargi, R. Kasturi, and S. H. Strayer. Performance Characterization of Video-Shot-Change

Detection Methods. *IEEE Transactions on Circuits and Systems for Video Technology*, 10(1):1–13, 2000.

[4] A. Hauptmann and M. Smith. Text, Speech, and Vision for Video Segmentation: The Informedia Project. In *AAAI Fall 1995 Symposium on Computational Models for Integrating Language and Vision*, 1995.

[5] M. A. Smith and T. Kanade. Video Skimming for Quick Browsing based on Audio and Image Characterization. Technical Report CMU-CS-95-186, Carnegie Mellon University, 1995.

[6] U. Gargi, S. Antani, and R. Kasturi. Indexing text events in digital video databases. In *Proc. International Conference on Pattern Recognition*, volume 1, pages 916–918, 1998.

[7] U. Gargi, D. Crandall, S. Antani, T. Gandhi, R. Keener, and R. Kasturi. A system for automatic text detection in video. In *International Conference on Document Analysis and Recognition*, pages 29–32, 1999.

[8] T. Gandhi, R. Kasturi, and S. Antani. Application of planar motion segmentation for scene text extraction. In *Proc. International Conference on Pattern Recognition*, volume 3, pages 445–449, 2000.

[9] V.Y. Mariano and R. Kasturi. Locating Uniform-Colored Text in Video Frames. In *Proc. International Conference on Pattern Recognition*, volume 4, pages 539–542, 2000.

[10] S. Antani, D. Crandall, and R. Kasturi. Robust extraction of text in video. In *Proc. International Conference on Pattern Recognition*, volume 3, pages 831–834, 2000.

[11] S. Antani, D. Crandall, A. Narasimamurthy, Y. Mariano, and R. Kasturi. Evaluation of Methods for Extraction of Text from Video. In *IAPR International Workshop on Document Analysis Systems*, pages 507–514, 2000.

[12] S. Antani, D. Crandall, V. Y. Mariano, A. Narasimhamurthy, and R. Kasturi. Reliable extraction of text in video. Technical Report CSE-00-022, Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16801, November 2000.

[13] B.-L. Yeo and B. Liu. Visual Content Highlighting Via Automatic Extraction of Embedded Captions on MPEG Compressed Video. In *SPIE/IS&T Symposium on Electronic Imaging Science and Technology: Digital Video Compression: Algorithms and Technologies*, volume 2668, pages 38–47, 1996.

[14] T. Sato, T Kanade, E.K. Hughes, and M.A. Smith. Video OCR for Digital News Archive. In *IEEE International Workshop on Content–Based Access of Image and Video Databases CAIVD'98*, pages 52–60, January 1998.

[15] S. Messelodi and C.M. Modena. Automatic Identification and Skew Estimation of Text Lines in Real Scene Images. *Pattern Recognition*, 32(5):791–810, May 1999.

[16] H. Li, D. Doermann, and O Kia. Automatic Text Detection and Tracking in Digital Video. *IEEE Transactions on Image Processing*, 9(1):147–156, 2000.

[17] V. Wu, R. Manmatha, and E. Riseman. Textfinder: An automatic system to detect and recognize text in images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(11):1224–1229, November 1999.

[18] N. Chaddha, R. Sharma, A. Agrawal, and A. Gupta. Text Segmentation in Mixed–Mode Images. In *28th Asilomar Conference on Signals, Systems and Computers*, pages 1356–1361, October 1995.

[19] Y. Zhong, H. Zhang, and A.K. Jain. Automatic Caption Localization in Compressed Video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(4):385–392, 2000.

[20] Dimitrova, N. and Agnihotri, L. and Dorai, C. and Bolle, R. MPEG-7 *Videotext* description scheme for superimposed text in images and video. *Signal Processing: Image Communication*, 16:137–155, 2000.

[21] J.-C. Shim, C. Dorai, and R. Bolle. Automatic Text Extraction from Video for Content-Based Annotation and Retrieval. In *Proc. International Conference on Pattern Recognition*, pages 618–620, 1998.

[22] Y. Zhong, K. Karu, and A.K. Jain. Locating Text in Complex Color Images. *Pattern Recognition*, 28(10):1523–1536, October 1995.

[23] A.K. Jain and B. Yu. Automatic Text Location in Images and Video Frames. *Pattern Recognition*, 31(12):2055–2076, 1998.

[24] H.-K. Kim. Efficient Automatic Text Location method and Content–Based Indexing and

Structuring of Video Database. *Journal of Visual Communications and Image Representation*, 7(4):336–344, December 1996.

[25] H.-M. Suen and J.-F. Wang. Segmentation of Uniform-Coloured Text from Colour Graphics Background. *IEE Proceedings: Vision, Image and Signal Processing*, 144(6):317–322, December 1997.

[26] H. Hase, T. Shinokawa, M. Yoneda, M. Sakaiand, and H. Maruyama. Character String Extraction by Multi–stage Relaxation. In *International Conference on Document Analysis and Recognition*, volume 1, pages 298–302, 1997.

[27] J. Zhou, D. Lopresti, and Z. Lei. OCR for World Wide Web Images. In *Proceedings of SPIE Document Recognition IV*, volume 3027, pages 58–66, 1997.

[28] R. Lienhart and F. Stuber. Automatic Text Recognition in Digital Videos. In *Proceedings of SPIE*, volume 2666, pages 180–188, 1996.

[29] F. LeBourgeois. Robust Multifont OCR System from Gray Level Images. In *International Conference on Document Analysis and Recognition*, volume 1, pages 1–5, 1997.

[30] M.v.d. Schaar-Mitrea and P.H.N. de With. Compression of Mixed Video and Graphics Images for TV Systems. In *SPIE Visual Communications and Image Processing*, pages 213–221, 1998.

[31] B.T. Chun, Y. Bae, and T.-Y. Kim. Text extraction in videos using topographical features of characters. In *IEEE International Fuzzy Systems Conference*, volume 2, pages 1126 –1130, 1999.

[32] O. Hori. A video text extraction method for character recognition. In *International Conference on Document Analysis and Recognition*, pages 25–28, 1999.

[33] H. Kasuga, M. Okamoto, and H. Yamamoto. Extraction of characters from color documents. In *Proceedings of IS&T/SPIE Conference on Document Recognition and Retrieval VII*, volume 3967, pages 278–285, 2000.

[34] Y.-K. Lim, S.-H. Choi, and S.-W. Lee. Text extraction in mpeg compressed video for content-based indexing. In *Proc. International Conference on Pattern Recognition*, volume 4, pages 409–412, 2000.

[35] M. Sawaki, H. Hurase, and N. Hagita. Automatic acquisition of context-based images templates for degraded character recognition in scene images. In *Proc. International Conference on Pattern Recognition*, volume 4, pages 15–18, 2000.

[36] K. Sobottka, H. Bunke, and H. Kronenberg. Identification of text on colored book and journal covers. In *International Conference on Document Analysis and Recognition*, pages 57–62, 1999.

[37] J. Ohya, A. Shio, and S. Akamatsu. Recognizing Characters in Scene Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:214–224, 1994.

[38] P. Comelli, P. Ferragina, M. N. Granieri, and F. Stabile. Optical Recognition of Motor Vehicle License Plates. *IEEE Transactions on Vehicular Technology*, 44(4):790–799, November 1995.

[39] Y. Cui and Q. Huang. Character Extraction of License Plates from Video. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 502–507, 1997.

[40] L.L. Winger, M.E. Jernigan, and J.A. Robinson. Character Segmentation and Thresholding in Low-Contrast Scene Images. In *Proceedings of SPIE*, volume 2660, pages 286–296, 1996.

[41] T. Gandhi. *Image Sequence Analysis for Object Detection and Segmentation.* PhD thesis, The Pennsylvania State University, University Park, PA 16802, 2000.

[42] B.G. Sherlock and D.M. Munro. Algorithm 749: Fast Discrete Cosine Transform. *ACM Transactions on Mathematical Software*, 21(4):372–378, 1995.

[43] P. Zhu and P.M. Chirlian. On Critical-Point Detection of Digital Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):737–748, August 1995.

[44] J.N. Kapur, P.K. Sahoo, and A.K.C. Wong. A New Method for Gray-Level Picture Thresholding Using the Entropy of the Histogram. *Computer Vision, Graphics, and Image Processing*, 29(3):273–285, March 1985.

[45] A.K.C. Wong and P.K. Shaoo. A Gray-Level Threshold Selection Method Based on Maximum Entropy Principle. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(4):866–871, July 1989.

[46] M. Kamel and A. Zhao. Extraction of Binary Character/Graphics Images from Grayscale Document Images. *Computer Vision, Graphics, and Image Processing*, 55(3):203–217, May 1993.

[47] J.-H. Jang and K.-S. Hong. Binarization of noisy gray-scale character images by thin line modeling. *Pattern Recognition*, 32(5):743–752, 1999.

[48] A.K. Jain. *Algorithms for clustering data.* Prentice Hall, Englewood Cliffs, NJ, 1988.

[49] S.D. Yanowitz and A.M. Bruckstein. A New Method for Image Segmentation. *Computer Vision, Graphics, and Image Processing*, 46(1):82–95, April 1989.

[50] L.A. Fletcher and R. Kasturi. Automated Text String Separation from Mixed Text/Graphics Images. Technical Report TR-86-001, Department of Electrical and Computer Engineering, Penn State University, University Park, PA, 1986.

[51] S.-W. Lee and Y.J. Kim. Direct Extraction of Topographical Features for Gray Scale Character Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(7):724–728, July 1995.

[52] Y. Nakajima, A. Yoneyama, H. Yanagihara, and M. Sugano. Moving Object Detection from MPEG Coded Data. In *Proceedings of SPIE*, volume 3309, pages 988–996, 1998.

[53] M. Pilu. On Using Raw MPEG Motion Vectors to Determine Global Camera Motion. In *Proceedings of SPIE*, volume 3309, pages 448–459, 1998.

Digital Camera for Document Acquisition

Francis (Pete) Fisher
U.S. Army Research Laboratory
(pfisher@arl.army.mil)

Abstract

A variety of military operations utilize information collected from documents in the field. These documents, collected or captured through various means, may be in foreign languages and encompass a wide range of document types and sizes. Army Research Laboratory (ARL) has developed the FALCon system (Forward Area Language Converter) to permit non-linguists to assist translators and linguists by triaging foreign language documents and prioritizing those documents for translation and evaluation. One difficulty that users reported in pilot field tests of FALCon was that the sheet-fed scanner incorporated in the FALCon system was not suitable for certain document types. Documents that were very small, stapled or bound, or printed on stiff or poor quality paper could not be scanned into the FALCon system. In order to expand the types of documents that can be processed using FALCon, ARL is evaluating commercial digital cameras as a possible replacement for the sheet-fed scanner. Document images captured using the digital camera are passed through the FALCon process in a manner similar to that for scanned document images. We are evaluating digital cameras with respect to document imaging capability, ease of use, ease of image transfer, and perceived survivability in field environments. This paper will describe our digital camera evaluation, the digital camera selected for integration with FALCon, the integration of the digital camera into the FALCon system, and the final system capabilities.

1 Introduction

During military actions on foreign soil soldiers can capture large quantities of foreign language documents. Most soldiers involved in such operations are not likely to be able to read and understand these documents. The military maintains linguists, each trained in one or more languages, to evaluate these captured documents. The problem that can arise, particularly for operations in urban environments, is that solders in the field can capture documents in much greater quantities than can be evaluated by the limited number of linguists that are available. Army Research Laboratory (ARL), in conjunction with other military and government agencies, has developed a portable system called FALCon (Forward Area Language Converter) to assist soldiers in the field with the evaluation and triage of foreign language documents.

The FALCon system, which can be operated by non-linguists, provides the user with an English conversion of foreign language documents and an automated key word search. Users can rapidly evaluate the intent of a document that they originally could not read, permitting them to support linguists in the field by prioritizing those documents for translation and evaluation. This permits the limited resources of the linguists to be focused on documents deemed most important to the mission.

Figure 1 shows a block diagram of the FALCon process. The FALCon system consists primarily of a scanner and personal computer (PC) with four software modules. The scanner software stores document images and permits users to edit those document images to remove unwanted content. Users select the image that they want converted and then click the FALCon button to start the process. The scanner software then passes the document image to the OCR (optical character recognition) software where it is converted from an image file to a foreign language text file. The OCR software then passes the foreign language text file to the MT (machine translation) software where the foreign language text is converted to text in English. The resulting English text is then scanned for keywords in order to measure the relevance of the document to the specified keyword list. At the conclusion of the process the PC displays a window showing the foreign language text (OCR output), a window showing the English text (MT output) with found keywords marked in red, and a window showing the number of keyword hits. Users can then browse the English result and check the keywords in order to access the importance of the document. If desired the user can then save files from any or all of the process steps for further evaluation by a trained linguist. A user interface software module is included to simplify the process of setting language parameters in the three primary software modules. Because users may already have a PC at their location, FALCon is also available as software only. This permits FALCon to be added to a wide variety of systems for evaluation and use by military personnel.

As part of the initial integration effort, ARL provided seven prototype systems to Army users for pilot field-testing in Bosnia in 1997. Feedback from this pilot field test included several reports of documents that could not be evaluated using the FALCon system because users were unable to scan these documents using the

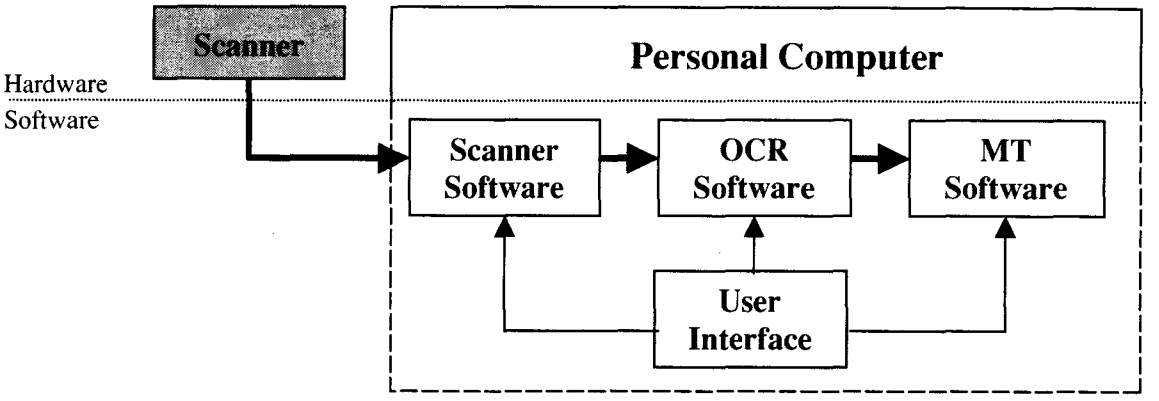


Figure 1; FALCon Process

sheet-fed scanner included as part of the FALCon system. Examples of problem document types included small and/or stiff documents (ID cards), flimsy documents, and documents that were stapled or bound. Soldiers noted that much of the paper being used in Bosnia for printing at that time was of low quality due to the embargo on imports.

Based on this user feedback I proposed the use of a digital camera as an alternative document capture front-end for FALCon. The digital camera provides several benefits beyond supporting a wider variety of document types. It can be used to capture text from a variety of targets other than documents. Examples include maps, signs on buildings and walls, and road signs.

Replacing the scanner with a camera can also increase document throughput. Two operators can run the system, one capturing document images with the camera while another processes the images on the computer. The camera can also be used to capture pictures of people and places for reference purposes.

Options to use a flatbed scanner were also considered. One major problem is the size of a flatbed scanner. While new units are currently available in very thin profiles (about 1 inch in some cases), they are still larger then most sheet fed scanners. These units also appear to be more fragile as compared to sheet fed scanners or digital cameras. Flatbed scanners have a lid that must be raised for loading of documents. The hinges used to mount the lid may not hold up to rigorous field use. Also, all units that I have seen to date have some form of manual locking mechanism to protect the scan head during transport. Users would be required to operate this lock prior to equipment transport. Failure to do so would likely result in failure of the flatbed scanner. Earlier tests of hand-held scanners and line scanners also revealed a variety of problems with document image acquisition.

2 Initial Evaluation

The first step in our camera evaluation was to establish initial specifications for the cameras so that we could select units to evaluate. One major specification to establish was the minimum required resolution for the camera. From past experience we have found that low-resolution images such as faxes generally produce poor quality OCR results. When considering the use of a digital camera for document capture we knew that camera resolution would be a major issue. Other specifications considered included size, weight, and cost of the camera, and the interfaces provided for transferring images to the PC.

To support our testing process I established a set of four test documents with ground truth files. Half of these documents are Croatian text in Latinic font and half are Serbian text in Cyrillic font, each in 10 point and 12 point font size. These languages (fonts) were selected for evaluation due to their relevance to the current military presence in the Balkans region. The test documents were assembled from documents used for testing during the integration of the FALCon system.

In order to establish the minimum dots-per-inch (DPI) requirement for the digital camera application we needed to define a process of evaluation independent of the digital camera. To this end we scanned test documents using a flatbed scanner at resolutions from 100 DPI (dots per inch) to 400 DPI. While this method ignores distortions introduced by the digital camera, such as changes in image brightness, optical distortion, and image degradation caused by compression, it does provide a good measure of the "best case" OCR capability for a given DPI value. This in turn establishes a minimum limit on the required camera resolution for a given document size.

In order to test for worst-case pixel rates we scanned the documents at DPI values that were not regularly spaced along the range of resolution values. This was done to reduce the possibility of false results that might occur if we tested only at DPI rates that easily scaled to the values that the software was trained on. The scanned images were then processed to text using the FALCon OCR software and the resulting text files were compared character-by-character to the original source files. The OCR accuracy for the different DPI rates is

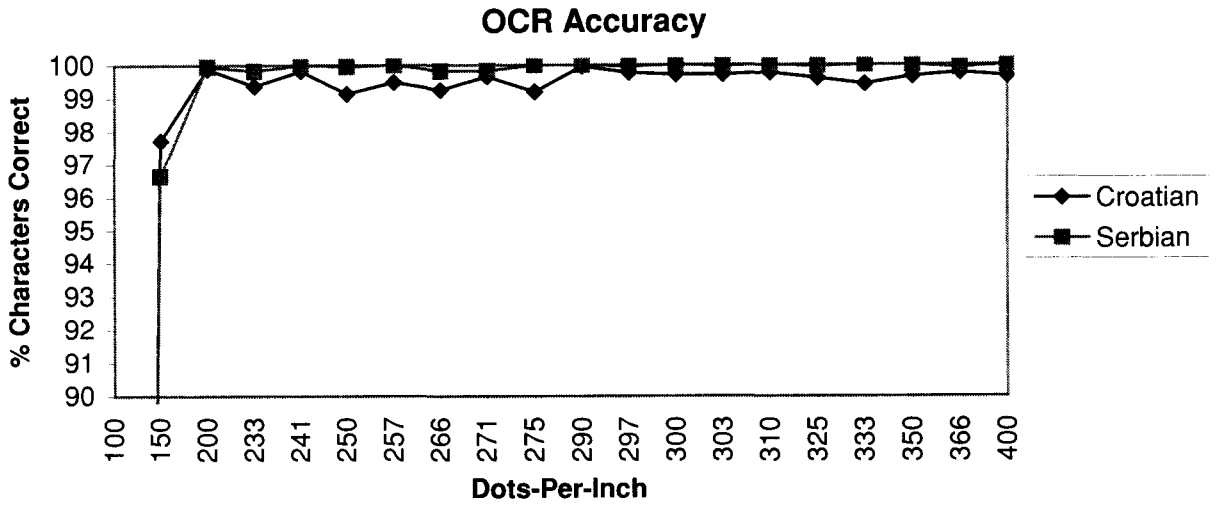


Figure 2; OCR Accuracy versus Document Image Resolution

plotted in Figure 2. This test shows that there is a precipitous decrease in OCR accuracy for DPI rates below 200 DPI.

Having established the need for a minimum of 200 DPI on the document image, we then proceeded to calculate the maximum area that could be spanned using typical high-resolution commercial digital cameras. The resolution for these products is typically 2048 by 1536 pixels. Since the dots-per-inch specification for a scanned document is equivalent to the pixels-per-inch for an image acquired using a digital camera, we calculate the maximum permissible document size as follows:

Maximum Height
2048 pixels * (1 DPI/1 PPI) / 200 DPI = 10.24 inches

Maximum Width
1536 pixels * (1 DPI/1 PPI) / 200 DPI = 7.68 inches.

Since many documents of interest have a 1-inch margin and are roughly 8.5" x 11", these results show that a commercial digital camera with a minimum resolution of 2048 by 1536 pixels should be suitable for this application. Again, remember that this ignores any added image degradation introduced by the camera itself.

With our specifications in hand we performed a market survey collecting information on a wide variety of commercially available digital cameras that had sufficient resolution for the document-processing task. We were able to identify 21 digital cameras that would be suitable for this application. No information was available from manufacturers on possible future cameras with higher resolutions. Assuming that the smallest and lightest camera would be the most portable, and thus the best choice for this application, we selected and purchased the Fuji FinePix 4700 and the Canon Power Shot S20. These cameras were considerably smaller than other products with similar capabilities. While the Fuji FinePix had a higher resolution as compared to many of the other cameras, 4.3M pixels versus 3.3M pixels for the others, we were concerned because this higher resolution is obtained through interpolation. We were unsure what effect this would have, positive or negative, on the final image quality with respect to the operation of the OCR software. Some cameras with resolutions up to 3072 by 2048 pixels were identified in the market survey. These units were not considered for this application due to high cost and the need for operation using a fixed mount such as a stand or tripod.

3 Initial Camera Evaluation

With digital cameras in hand we set out to acquire and evaluate document images. That is when the first problems became apparent. The cameras selected are so small that it becomes difficult to operate them. We found ourselves covering up sensors and flash units with our fingers while trying to take pictures. After a little effort we learned how to hold the cameras in a manner that did not block vital functions. The second problem identified was short battery life. In order to minimize the size of these cameras manufactures have reduced the size of the battery set. This in turn reduces the time that the camera will operate before the batteries need recharging.

For our camera evaluation we took five pictures of each of the four test documents using both cameras. Initial results from the evaluation of the document images were mixed. On the plus side we found that anticipated problems with the document images being distorted into a keystone shape were minimal. This type of distortion results when the camera is not

perpendicular to the image plane of the document. Initial assumptions were that users would find it difficult to adjust the camera to the proper orientation. All of the digital cameras that we evaluated included an LCD (liquid crystal display) on the digital camera that is used as a through the lens viewfinder. Operators can view the document image on this display during document image acquisition and align the document text to the outside edge of the display. When the camera is in the proper orientation the edges of the document text are aligned with the edges of the display. As a training aid I generated documents with line frames around the outside of the text. During training users can align this frame to the outside edge of the camera display in order to align the camera to the document. After the user understands the process of camera alignment they can then use the edges of the text to align the camera.

On the negative side, evaluation of the document images revealed that the OCR quality was far below what we anticipated. While we had no hard specification on minimum OCR quality we estimated that at least 90% accuracy would be required for the digital camera to be of any use at all. Any errors introduced into the FALCon process affect all remaining steps of the process. If the OCR software recognizes a character in a word incorrectly then the word is misspelled. A misspelled word will either not translate, or the spelling error could result in a real but incorrect word. One way information is lost. The other way false information in provided. From this point of view we needed the best OCR quality that we could achieve. Even with the OCR accuracy at 90% there can still be large numbers of words that do not translate due to spelling errors.

After some initial evaluation we concluded that distortion in the document images was leading to the low OCR accuracy. For document images taken at the extreme limits of the zoom range we found that the lines of the frame surrounding the text area were no longer straight as can be seen in the image shown in figure 3. There was a corresponding distortion of the text near the text frame lines. This problem may be due to either the small size of the lenses on these cameras, or to the automatic settings used with the lenses.

After several rounds of testing in the lab we found that the cameras had "sweet spots" in the zoom adjustment that minimized the optical distortion. Even with this adjustment there were still wide variations in OCR quality across different document images acquired using the same camera settings. For the most part results looked promising.

4 Camera Evaluation

Using the information obtained in the initial camera evaluation we set out to reevaluate our camera selection criteria. A second market survey added one more camera to the list of possible choices. We relaxed the requirement for minimum camera size and started looking for information on the battery type and quantity and the operating time on a set of batteries. Optical image distortion due to zoom adjustment was a much harder parameter to characterize. After some thought I

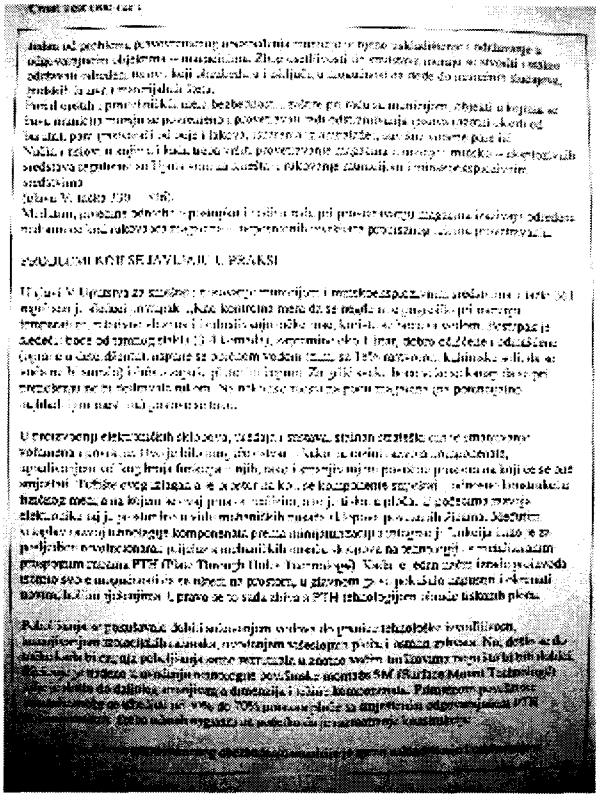


Figure 3; Document Image at Wide Zoom Setting

spent an afternoon at a local electronics store where a salesman was kind enough to show me every high-resolution digital camera that they carried. I used a sheet of paper as a target and pre-evaluated all of the cameras in an attempt to identify those cameras with the least optical distortion. Back at the lab I correlated the list of best optical choices against the size, cost, and battery parameters of the cameras. I then selected and purchased a Ricoh RDC-7, Casio QV-3000EX, and Sony Cyber-Shot DSC-S70, and borrowed a Nikon CoolPix 990.

With the new cameras in the lab we processed a limited number of documents on each camera for different zoom settings in order to get a baseline on optical quality. Again we found that most of the cameras had particular zoom settings that produced good OCR results. In order to minimize the camera test set I decided that we would process document images for each camera at zoom settings of wide, middle, and narrow. We captured and evaluated five images of each of the four test documents for all of the cameras. With six cameras, four documents, three zoom settings, and five pictures per document we captured a total of 360 images.

This second round of camera testing revealed several new features of the digital cameras. For four of the six cameras the lens assembly protrudes from the camera body during operation, and retracts into the camera body when the camera is turned off. This provides a potential path for dirt to enter the camera. Further, if the lens protrudes from the camera body then it may be damaged in field use.

		Document Type/Font Size (% Accuracy averaged over 5 documents)			
Camera	**Zoom Setting**	**Croatian 10 pt**	**Croatian 12 pt**	**Serbian 10 pt**	**Serbian 12 pt**
Canon	Wide	65.8%	79.9%	68.7%	70.7%
	Middle	72.4%	77.6%	68.9%	81.8%
	Narrow	70.3%	81.7%	54.3%	92.5%
Casio	Wide	78.3%	53.8%	47.5%	76.0%
	Middle	94.8%	88.6%	81.9%	74.5%
	Narrow	38.1%	56.1%	41.5%	85.4%
Fuji	Wide	54.5%	67.5%	46.6%	72.9%
	Middle	82.6%	90.9%	86.0%	98.4%
	Narrow	58.1%	51.3%	53.9%	70.0%
Nikon	Wide	73.1%	72.9%	76.6%	76.4%
	Middle	99.3%	99.4%	97.9%	98.2%
	Narrow	72.0%	75.0%	38.7%	71.3%
Ricoh	Wide	92.1%	79.2%	85.0%	64.7%
	Middle	93.2%	98.9%	92.2%	96.1%
	Narrow	78.7%	53.9%	83.6%	68.0%
Sony	Wide	80.8%	78.1%	84.2%	91.0%
	Middle	98.4%	99.0%	96.8%	99.7%
	Narrow	96.9%	95.4%	96.7%	99.9%

Table 1; OCR Accuracy for Digital Cameras

Several of the cameras use custom battery packs that would not be readily available for field replacement. In my limited experience with soldiers in field exercises I found that AA batteries were always available. Those cameras that use AA batteries would then be more desirable for this application because soldiers could find replacement batteries when needed.

While all of the cameras included a USB (universal serial bus) connection for image transfer, I found it easier to simply remove the image storage card and mount it in the PC using a PC-Card adapter. This worked well for Compact Flash memory and for the Sony Memory Stick. The Smart Media card used by the Fuji and Ricoh cameras proved difficult to handle in this operation.

After capturing the document images we converted them to text using the OCR software and compared the resulting text files to the ground truth documents to measure the OCR accuracy. All changes in page format were ignored because they do not change the content of the document. The results of this testing, averaged for each set of five test documents, are shown in Table 1. Most of the cameras worked best with the zoom set to the middle of the range of adjustment. Several of the cameras had greater than 90% OCR accuracy for this case, and the Sony and Nikon cameras had greater than 96% accuracy. Also, most of the cameras suffered severe degradation in OCR quality when the zoom was set to the wide or narrow setting. The Sony camera was the exception in the narrow zoom setting. For most zoom settings the OCR quality for the 12-point text is better than that for the 10-point text. This result was anticipated because the 12-point text has more pixels on each character. This in turn provides more information to the OCR software for recognition. Exceptions occurred mostly in cases where the OCR quality was below 90%. Again the Sony camera was the exception.

As in the initial testing we found that OCR results for some of the cameras varied considerably from image to image for the same document with the same zoom, flash, and white balance settings. This variation is illustrated by the data shown in Table 2, which lists the OCR error counts for each of the five document images taken at each zoom setting for the Canon camera. To better understand this problem we repeated the testing process on some of the cameras and obtained similar results. We then examined the document images using image-processing software. Using only visual inspection we found minimal variation between images that had good OCR results and images that had bad OCR results. After some processing we found that many of the images with bad OCR results had substantial variations in illumination across page. Figure 4 shows a document image that has been posterized, or converted to a fixed number of image intensity levels, in order to show the variation in illumination.

Assuming that we could attribute some of the OCR error to improper document illumination we proceeded to adjust the flash level and image brightness on the cameras with those options in order to try to optimize image quality. We repeated the image acquisition and evaluation process for selected zoom settings on selected cameras. In some cases these adjustments improved OCR accuracy while in others they decreased it. Also, the number of OCR errors remained more consistent from image to image, even for those cases

Table 2; OCR Character Error Counts for Canon Camera

	OCR Character Error Count				
Zoom	**Image**	**Croatian 10 pt**	**Croatian 12 pt**	**Serbian 10 pt**	**Serbian 12 pt**
Wide	1	2286	595	1046	1588
	2	1602	855	2149	501
	3	1791	608	1024	1313
	4	1385	732	1104	894
	5	1736	749	2303	872
Middle	1	3475	689	1037	220
	2	1129	2548	791	608
	3	479	404	2330	1381
	4	800	192	921	696
	5	1218	109	2516	298
Narrow	1	1924	83	2501	611
	2	2696	260	2652	611
	3	840	1082	893	120
	4	769	414	2475	59
	5	1432	1389	2634	120

where OCR accuracy was not improved. Unfortunately, it would not be reasonable to expect operators in the field to make a variety of adjustments to the digital camera in order to obtain suitable results. We needed other options.

One option considered was to correct the image intensity by thresholding. With thresholding the pixels of the image are converted to either the maximum or minimum possible value based on their value as compared to the threshold level. Figure 5(a) shows a document image that has been processed in this manner. Note that the corners of the image are darkened. This occurs because the threshold level was set to obtain good text clarity at the center of the page. If the threshold is set low enough to remove the corner darkening then the text at the center of the page will be washed out.

A coworker suggested correcting for the flash variation by performing a scaling correction proportional to the distance from the center of the image. In order to test this possibility I used two commercial software packages to generate a correction image that was the inverse of the lighting pattern on the document images. I then multiplied that image by one of the document images. Evaluation of the resulting image showed a substantial increase in OCR accuracy. For our case we found that the center of the camera flash pattern moved on the page as the distance from the camera to the document changed. Changes in the distance from the camera to the document also resulted in different rates of change of the illumination across

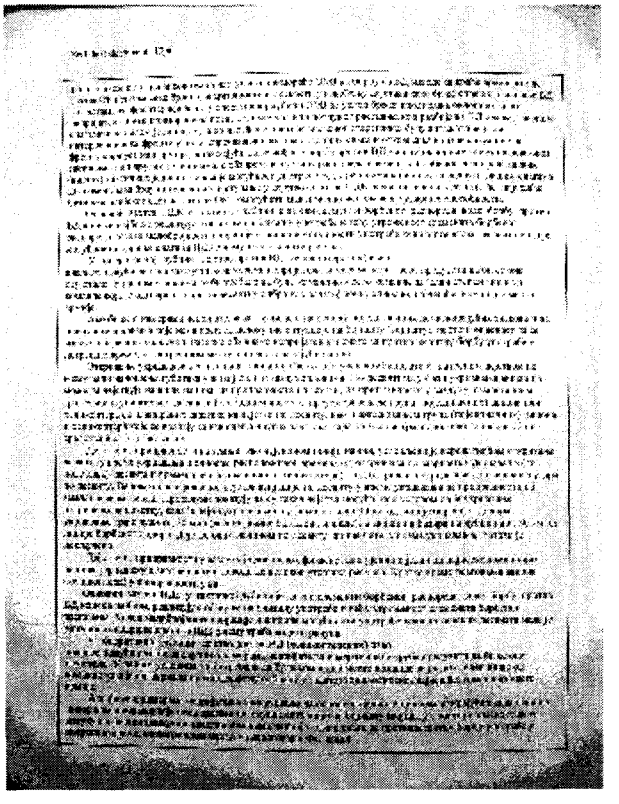
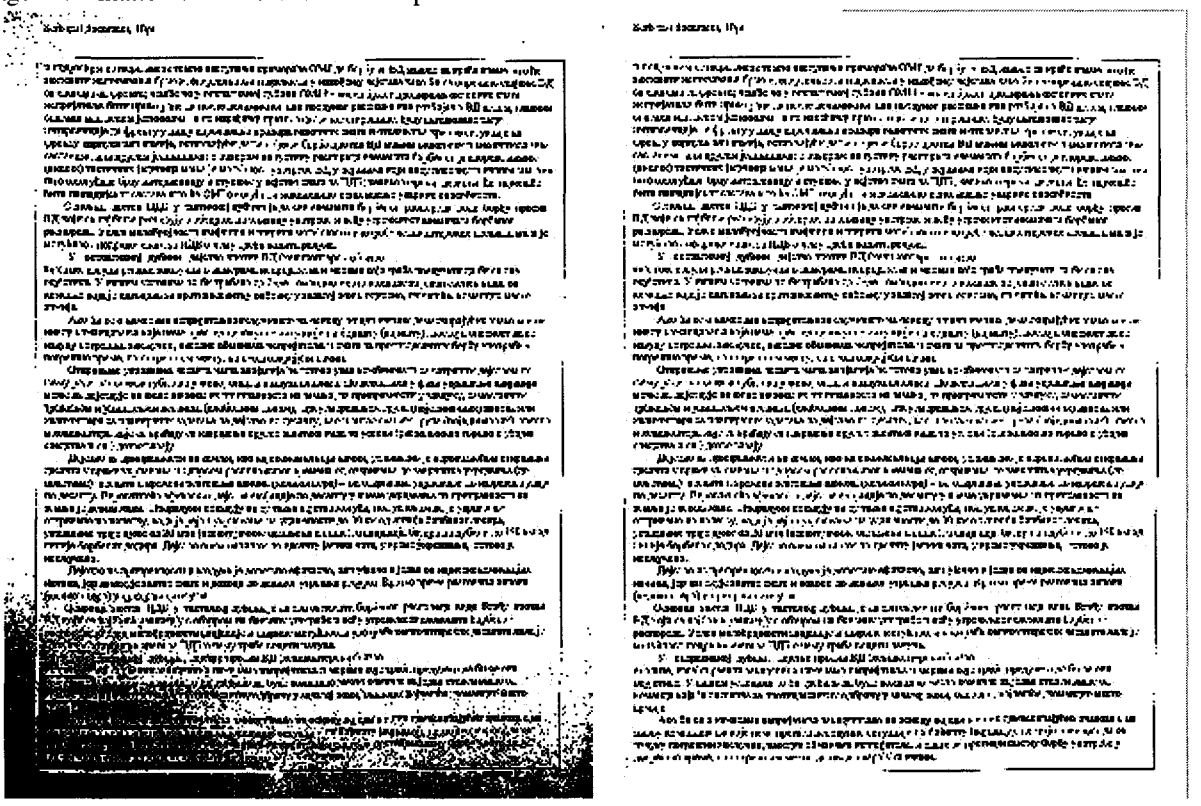


Figure 4; Posterized Document Image Showing Changes Illumination

the page. This would require the illumination correction across the page to be calculated for each document

image processed. While it sounds like a good idea this process does not seem feasible for our application.

As an alternative I proposed performing a threshold process on sub-blocks of the image. By dividing the image into sub-blocks we minimize the change in image illumination across the area to be processed and eliminate the problem of text in one part of the image being lighter than paper in another part of the image. In order to prove this concept I manually processed an image by thresholding sub-blocks of the image. This process substantially reduced the OCR error.



(a) (b)

Figure 5; (a) Full-Page Threshold Document Image, (b) Sub-Block Threshold Image

Figure 5 shows the full-page threshold image on the left and the sub-block threshold image on the right. An algorithm was then developed to perform the sub-block threshold automatically. One interesting aspect of the sub-block threshold process was the selection of the number of sub-blocks into which the image would be divided. I selected the number of sub-blocks to process by working with a worst case image obtained by taking a picture of a document placed against a page in an open book. This document image was severely distorted both optically and in image intensity due to the curved surface of the book pages due to the binding. After several processing steps I settled on 256 sub-blocks in a 16 by 16 block pattern.

Using the sub-block threshold algorithm we reprocessed the document images acquired using the Nikon and Casio cameras. The OCR accuracy results are shown in table 3. Again these are averages of the results over 5 document images. Note that for most cases the OCR accuracy has increased to greater than 90%. We also found that the resulting OCR character errors were relatively consistent from image to image for a given document, camera, and zoom setting. Any remaining error is most likely attributable to the optical distortion caused by the camera lens. Using the sub-block threshold we found similar results with a partial test of the document images from the other cameras.

As part of this process we identified another interesting feature of the digital camera images. Most of these cameras store their images in JPEG format with no reference for the image dimensions. This makes sense because the camera does not know how large the target object is. The OCR software used with FALCon interprets these camera images as very large documents at 72 pixels-per-inch. This causes two problems. First, the text output from the OCR has a very large font.

Table 3; OCR Accuracy for Sub-Block Threshold Document Images

		% Accuracy/% Improvement (averaged over 5 documents)			
Camera	**Zoom**	**Croatian 10**	**Croatian 12**	**Serbian 10**	**Serbian 12**
Nikon	Wide	84.7%/+11.6%	90.6%/+17.7%	90.3%/+13.7%	98.6%/+22.2%
	Middle	99.3%/0%	99.5%/+0.1%	99.6%/+1.7%	99.9%/+1.7%
	Narrow	97.5%/+25.5%	98.8%/+23.8%	98.9%/+60.2%	99.8%/+28.5%
Casio	Wide	91.9%/+13.6%	84.3%/+30.5%	95.5%/+48%	92.7%/+16.7%
	Middle	99.3%/+4.5%	99.4%/+10.8%	99.3%/+17.4%	99.9%/+25.4%
	Narrow	90.2%/+52.1%	93.8%/+37.7%	97.1%/+55.6%	98.0%/+12.6%

Second, the standard FALCon process cannot pass the image. We have modified the sub-block threshold process so that it sets the pixels-per-inch setting to 200 in order to correct this problem.

5 Alternative Lenses

Since the camera optics appeared to affect the OCR process we considered possible options to improve the quality of the optics. I ordered a close up lens for the Nikon camera for evaluation. I selected the Nikon for this test because it is the only unit being evaluated that has a threaded mount to accept accessory lenses. After receiving the lens we mounted it on the camera and started looking at documents. Unfortunately, the lens blocks the flash output making it unusable in our application. Also, while the optical distortion was reduced in some areas of the image, it was not reduced in others. Overall I concluded that the close up lens was not suitable for this application.

6 Camera Selection

With the development of the sub-block threshold process we demonstrated the capability to compensate for OCR errors resulting from variation in document image brightness. Table 3 shows that two of the cameras have greater than 90% OCR accuracy for the narrow zoom setting and greater than 99.3% accuracy for the middle zoom setting. A partial evaluation of document images from the other four cameras evaluated yielded similar results. We were unable to reevaluate all of the document images for all cameras prior to the final camera selection or the writing of this paper due to the loss of support staff.

With the OCR accuracy testing complete we needed only to make our final selection of a digital camera for our application. The initial selection criterion was to get the best possible OCR accuracy. After working with the digital cameras in the lab we placed a much greater emphasis on the need to have a camera that was easy to operate and that we considered would have minimal problems in the field. Table 4 shows the general evaluation criteria for camera selection.

Based on our experience in the lab we reviewed the camera size specifications and decided that selecting the smallest cameras would not be the best option. If the smallest cameras were difficult to use it the lab the problem could only be worse in a military field environment.

Next we eliminated some of the camera by looking at the requirement for field use. We eliminated those cameras with lenses that protrude from the camera body during operation. This would reduce the possibility of camera failures in the field due to dirt infiltration and physical impact to the lens assembly. With two cameras remaining, the Ricoh and Nikon, we looked at the battery and memory card types. The Nikon uses the Compact Flash card and AA-batteries, both pluses for field use. The Ricoh uses the Smart Media card and a Lithium-ion battery pack. We found the Smart Media card difficult to handle in our camera evaluation, and as mentioned previously it would probably be difficult to locate a replacement Lithium-ion battery pack in the field.

Camera cost was a consideration in the selection of our cameras for evaluation. Several very high-resolution cameras were not considered due to the high cost. For the six cameras evaluated the variation in cost was small. As a result cost was not considered in the final camera selection.

Based on our updated selection criteria and the results of the OCR evaluation we selected the Nikon CoolPix 990 as our camera of choice for this application.

7 User Evaluations

Soldiers of the U.S. Army will test the digital camera extension for FALCon in Advanced Concepts Technology Demonstration (ACTD) field exercises in 2001. ARL staff will train users in the application of the digital camera extension for FALCon. We hope to get user feedback on the viability of this concept along with suggested system improvements

As a preliminary user test I had one of my coworkers, picked in part because he was not a camera expert, use the Nikon digital camera to acquire document images. I provided five minutes of training, explaining how to operate the camera and how to take pictures of the documents. With that my coworker was on his own to acquire five document images for each of the four test documents. The OCR accuracy for these document images was only 1% lower than results that I obtained

after hours of practice capturing document images in the lab.

Table 4; Camera Selection Criteria

	Selection Criteria					
Camera	**Size (cu. in.)**	**Weight**	**Lens Pop Out**	**Memory Type**	**Cost**	**Battery Type**
Casio	36.15	0.7 lbs.	Yes	Compact Flash	$999	AA x 4, Ni-MH
Canon	14.39	0.7 lbs.	Yes	Compact Flash	$1030	Ni-MH pack
Fuji	15.31	0.56 lbs.	Yes	Smart Memory	$999	AA x 2, Ni-MH
Nikon	27.44	0.81 lbs.	No	Compact Flash	$999	AA x 4, Ni-MH
Ricoh	15.37	0.59 lbs.	No	Smart Memory	*$800	Li-ion pack
Sony	31.82	0.96 lbs.	Yes	Memory Stick	$899	Li-ion pack

* = Price quoted after other products, prices decrease over time

8 Conclusion

While user evaluation has not yet been performed, it is apparent that high-end commercial digital cameras can be used in a lab environment to capture document images for processing purposes. We can capture an 8.5" x 11" text document with one image using a 3.3-mega pixel digital camera and obtain reasonable OCR results. Future increases in digital camera resolution will lead to better OCR results for this application.

Given the proper set of adjustments, most of the digital cameras we evaluated appear capable of supporting document capture for evaluation using the FALCon system. The zoom settings for these cameras can be adjusted to minimize optical distortion. Unfortunately, some of these adjustments make it difficult to acquire document images by placing the target document far from the user. The distance to the document amplifies any slight motion on the part of the user while taking the document image, possibly resulting in image blur and improper framing of the document.

Many of the document images captured using the digital cameras that we evaluated suffered some degradation of the OCR accuracy due to changes in illumination across the image. We were able to compensate for this by running a sub-block threshold process on the document images. This sub-block threshold process resulted in substantial improvement in OCR accuracy.

For the digital camera extension for FALCon we selected the Nikon CoolPix 990 based on our evaluation of the Nikon's perceived usability in a military field environment, and on our evaluation of OCR accuracy for digital cameras.

9 Acknowledgements

We would like to express our sincere gratitude to the Human Intelligence Support Tools Advanced Concepts Technology Demonstration (ACTD) staff for funding this effort. I would also like to thank Richard Chang, a Computer Science student from the University of Maryland, for assisting in the data collection and evaluation.

RECOGNITION OF TEXT IN 3-D SCENES

Gregory K. Myers, Program Director
Robert C. Bolles, Program Director
Quang-Tuan Luong, Computer Scientist
James A. Herson, Senior Computer Scientist
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025
myers@erg.sri.com

ABSTRACT

Video is an increasingly important source of information to the intelligence analyst. Recognizing text that appears in real-world scenery is potentially useful for characterizing the contents of video imagery. Previous research in text recognition for both printed documents and other sources of imagery has generally assumed that the text lies in a plane that is oriented roughly perpendicular to the optical axis of the camera. However, text such as street signs, name plates, and billboards appearing in captured video imagery often lies in a plane that is oriented at an oblique angle. SRI International (SRI) is developing an approach that takes advantage of 3-D scene geometry to detect the orientation of the plane on which text is printed. The text recognition process will then be able to transform the video image of the text to a normalized coordinate system before performing OCR, yielding more robust recognition performance. Our approach applies full-perspective projections and image-to-image homographies that capture the appearance of a plane viewed through perspective optics. We describe our approach and present some preliminary results.

PROBLEM STATEMENT

Video is an increasingly important source of information to the intelligence analyst, and the volume of collected multimedia data is expanding at a tremendous rate. A capability to automatically identify the contents of video imagery would enable videos to be indexed in a convenient and meaningful way for later reference, and would enable actions (such as automatic notification and dissemination) to be triggered in real time by the contents of streaming video. Methods of realizing this capability that rely on the automated recognition of objects and scenes directly in the imagery have had limited success because (1) scenes may be arbitrarily complex and may contain almost anything, and (2) the appearance of individual objects may vary greatly with lighting, point of view, etc. The recognition of text is easier than the recognition of objects in an arbitrarily complex scene, because text was designed to be readable and has a regular form that humans can easily interpret.

Our effort is focused on scene text, such as street signs, name plates, and billboards, that is part of the video scene itself. Most previous text recognition efforts in video and still imagery [Jain and Bhattacharjee 1992; Ohya, Shio, and Akamatsu 1994; Zhong, Karu, and Jain 1995; Smith and Kanade 1995; Lienhart 1996; Yeo and Liu 1996; Wu, Manmatha, and Riseman 1997; Jain and Yu 1998; Sato et al. 1998; Li and Doermann 1998a; Li and Doermann 1998b] have assumed that the text lies in a plane that is oriented roughly perpendicular to the optical axis of the camera. Of course, this assumption

is valid for scanned document images and imagery containing overlaid text captions, but is not generally true for scene text. Figure 1 shows an image, captured from a video camera, of a café scene in which the name of the café is viewed from an oblique angle. Such a configuration is quite common when the main subject of the scene is not the text itself, but such incidental text could be quite important (for example, it may be the only clue to the location of the captured imagery).

To address the problem of recognizing text that lies on a planar surface in 3-D space, we note that the orientation angle of such text relative to the camera can be modeled in terms of three angles, as shown in Figure 2:

- θ, the rotation in the plane perpendicular to the camera's optical axis
- φ and γ, the horizontal (azimuth) and vertical (elevation) components, respectively, of the angles formed by the normal to the text plane and the optical axis.

Figure 1. Café Scene

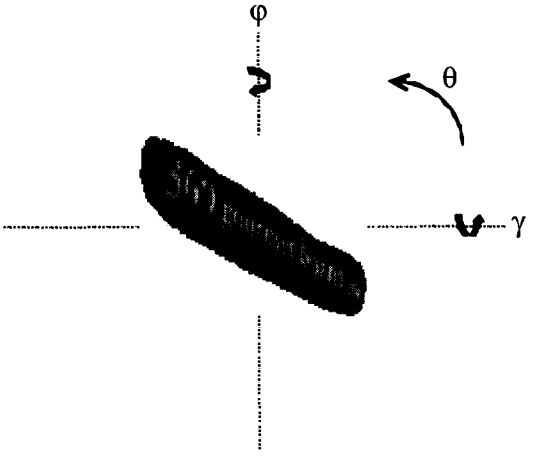


Figure 2. Orientation Angles of Text

The three angles represent the amount of rotation that the text plane must undergo relative to the camera in each of its three axes to yield a frontal, horizontal view of the plane in the camera's field of view. When θ and γ are zero and φ is nonzero, the apparent width of the text is reduced, resulting in a change in aspect ratio and a loss of horizontal resolution. Similarly, when θ and φ are zero and γ is nonzero, the text appears to be squashed vertically. The severity of perspective distortion is proportional to D/Z, where D is the extent of the text parallel to the optical axis (its "depth") and Z is the distance from the text to the camera. When the text is not centered at the optical axis or both φ and γ are nonzero, the text appears to be rotated in the image plane (see Figure 3). If the text were rotated to remove this apparent angle by a text recognition process that mistakenly assumed the text is fronto-parallel, the characters would become sheared (see Figure 4). When both φ and γ are nonzero and perspective distortion is significant, the shearing angle varies from left to right within the text region. OCR engines perform poorly if the shearing causes characters to touch or to be severely kerned (overlapped vertically).

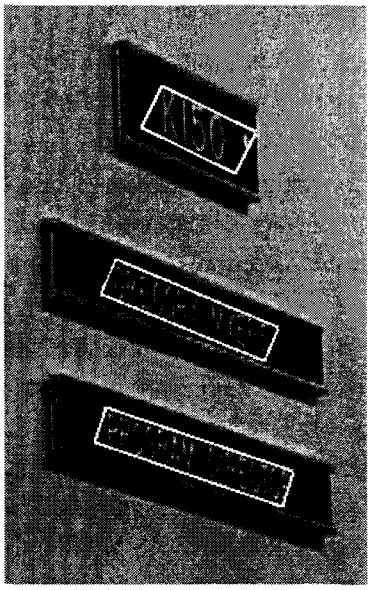


Figure 3. Image Showing Apparent Rotation in the Image Plane

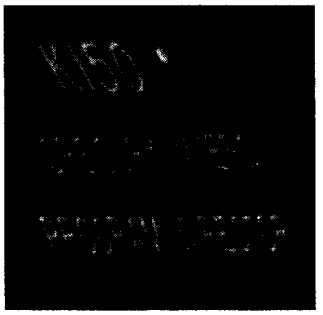


Figure 4. Image from Figure 3, After In-Plane Rotation

TECHNICAL APPROACH

In our approach we take advantage of 3-D scene geometry to detect the orientation of the plane on which text is printed. The text recognition process can then transform the video image of the text to a normalized coordinate system before performing OCR. There are two ways to estimate the parameters of a plane containing text. The first uses the shape and orientation of the text and the plane in a single image. The second examines the motion of the text and plane through a sequence of video image frames. We plan to develop techniques of both types and combine them to form a robust estimation procedure that takes into account the full perspective projection involved in the imaging process. In this paper we report some preliminary results from single-image analysis.

When the plane that contains the text is at an angle relative to the image plane, several types of distortions can be introduced that make it difficult to read the text. In the most general case, the distortion is described as a projective transformation (or homography) between the plane containing the text and the image plane. We can correct this distortion by applying the appropriate "corrective" projective transformation to the image. That is, we can rotate and stretch the original image to create a synthetic image, which we call a "rectified image," in which the projective distortion has been removed.

In general, a two-dimensional projective transformation has eight degrees of freedom. Four correspond to a Euclidean 2-D transformation (translations along two axes, a rotation, and an isotropic scale factor); two correspond to an affine transformation (a shear and a nonisotropic scaling of one axis relative to the other); and the remaining two degrees of freedom represent a perspective foreshortening along the two axes.

From an OCR point of view, some of the eight parameters produce changes that are harder to handle than others. In particular, the two translations are not a problem, because they simply produce an image shift that is naturally handled by OCR systems. Similarly, the two scale factors are not a problem, because the OCR systems typically include mechanisms to work at multiple scales. The Euclidean rotation is important, but is easily computed from a line of text. Therefore, three critical parameters produce distortions that are difficult for OCR systems to handle: the two perspective foreshortening parameters and the shearing.

In our single-image analysis approach, estimates of the plane parameters are computed from the orientations of the lines of text in the image and the borders of planar patch, if they are visible. To remove a projective distortion, we need to compute the three critical degrees of freedom associated with the plane on which the text is written. In general, we can do this by identifying three geometric constraints associated with the plane. For example, we can compute necessary parameters, given two orthogonal pairs of parallel lines, such as the borders of a rectangular sign or two parallel lines of text and a set of vertical strokes within the text. The three constraints deriveable from these sets of lines are two vanishing points (one from each set of parallel lines) and an orthogonality constraint between the sets of lines.

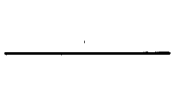
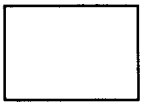
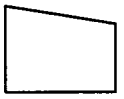
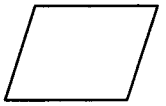
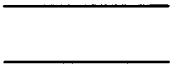
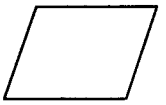
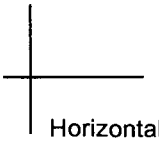
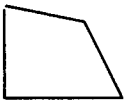
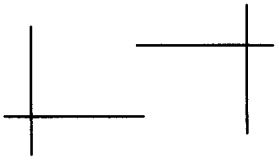
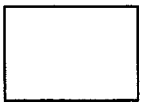
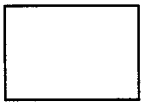
Sometimes, however, such linear properties are difficult to detect. In such cases, we can estimate the parameters by making assumptions about the camera-to-plane imaging geometry that are often true. For example, people normally take pictures so that the horizon is horizontal in the image. In other words, they seldom rotate the camera about its principal axis. In addition, they often keep the axis of the camera relatively horizontal. That is, they do not tilt the camera up or down very much. When these two

assumptions apply and the text lies on a vertical plane, such as a wall of a building or a billboard, the projective distortion is only along the X axis of the image. The perspective foreshortening in that direction can be computed from one constraint, such as a pair of horizontal parallel lines.

Another assumption that often holds is that the perspective effects are significantly smaller than the effects caused by the out-of-plane rotations. This is the case if the depth variation in the text is small compared with the distance from the camera to the plane. In this case, the perspective distortion is reduced to an affine shear and the projection is described as a weak perspective projection.

Table 1 summarizes the degrees of freedom that remain uncorrected after different sets of linear features are found and different assumptions are made about the plane-to-camera geometry.

Table 1. Degrees of Rectification

Geometric Relations Identified	Vertical Alignment		General Position	
	Weak Perspective	Full Perspective	Weak Perspective	Full Perspective
 Horizontal Line	 Fully Rectified	 Foreshortening in X	 Shear	 Foreshortening in X Foreshortening in Y Shear
 Parallel Horizontal Lines	 Fully Rectified	 Fully Rectified	 Shear	 Foreshortening in Y Shear
 Horizontal Line Vertical Line	 Fully Rectified	 Fully Rectified	 Fully Rectified	 Foreshortening in X Foreshortening in Y
 Parallel Horizontal Lines Parallel Vertical Lines	 Fully Rectified	 Fully Rectified	 Fully Rectified	 Fully Rectified

Given these relationships, our general strategy is to identify as many properties of a region of text as possible, and then compute a corrective transformation, using as few assumptions as possible. Initially, we use information derived independently from each individual line of text. Next, we combine information from multiple text lines after partitioning them into sets of lines that lie within a common

plane. We then further augment the process by detecting auxiliary lines that can provide horizontal and vertical cues. These can include lines in the same plane as the text (such as sign borders), and extraneous lines (e.g., building edges). Finally, depending upon our success in finding these features, we can either make assumptions to substitute for missing constraints (and then compute a transformation that corrects for a full perspective projection) or compute a transformation that does not completely remove all degrees of freedom. This approach is more general than the method described by Clark and Mirmehdi [2000], which requires the text to lie within a quadrilateral whose edges must be found; this quadrilateral is then transformed to a rectangle under a weak perspective assumption.

PRELIMINARY EXPERIMENTS

Thus far we have implemented only the first part of our general strategy—rectifying each text line in a single image independently. After possible lines of text are detected, various features of each text line are then estimated. These include the top and base lines, and the dominant vertical direction of the character strokes. The rectification parameters for each text line are computed from these characteristics. Each text line is then rectified independently and sent to an OCR engine.

The text detection and location process, somewhat similar to those described by Smith and Kanade [1995] and by Wu, Manmatha, and Riseman [1997], detects vertically oriented edge transitions in the gray-scale image, and links those that are compatible in size and relative position to form lines of text. A rectangle is then fitted to each line of detected text. Figure 5 shows a test image of a poster containing text that was captured at an azimuth angle of 70 degrees; the rectangles that have been fitted to each detected text line are shown in overlay. (Some of the rectangles do not look to the eye like true rectangles because of the perspective view of the image contents). Computing the best-fitting rectangle for each text line is an expedient way to approximate the location and extent of the text, but the top and bottom of the text are not accurately computed when significant perspective distortion is present.

A top line and base line for each line of text are estimated by rotating the text line at various angles and then computing a series of horizontal projections over the vertical edge transitions. (When the text consists of predominantly lower-case characters, the "top" line actually corresponds to the "midline" of the text that touches the tops of lower-case characters, excluding their ascenders.) The best estimate of the top line should correspond to the rotation angle that yields the steepest slope on the top side of the horizontal projection; the best estimate of the base line is similarly computed. Figure 6 shows an example of this procedure.

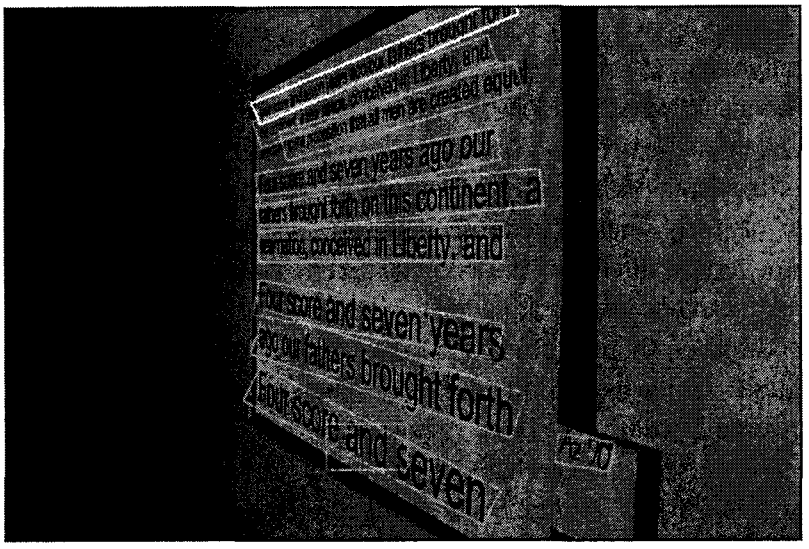


Figure 5. Rectangular Bounding Boxes Overlaid on a Test Image (Azimuth 70 Degrees)

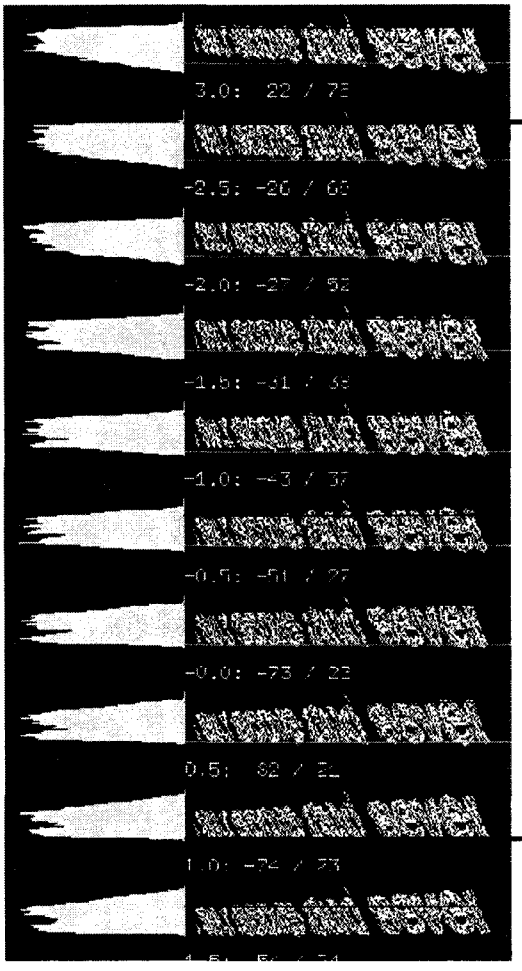


Figure 6. Estimation of Top and Base Lines

In addition to computing two horizontally oriented lines, we would like to find and measure the angles of two vertically oriented lines to use in the computation of the rectification parameters. Unfortunately, an individual line of text does not have much vertical extent, and it is difficult to determine which parts of the text could be used as vertical cues. However, the height of the text is not usually a significant fraction of the depth of the text in 3-D space, so that the perspective foreshortening in the Y dimension should be relatively small. Therefore, in the absence of any other reliable vertical cues, we compute the dominant vertical direction (shear) of the text by computing a series of vertical projections over the vertical edge transitions after rotating the text line in 2-degree increments. The best estimate of the dominant vertical direction should correspond to the angle at which the sum of squares of the vertical projection is a maximum (on the assumption that the projection of true vertical strokes is greatest when they are rotated to a vertical position). Figure 7 shows an example of shear computation. The deshearing process can be somewhat unreliable, because it assumes that a significant fraction of the characters contain vertical strokes. Figure 8 shows the refined bounding boxes based on the top and base lines and on the dominant vertical direction. Figure 9 shows the warped text lines (a) after the initial rectangular bounding box is deskewed; (b) after the baseline is refined (without including the top line in the dewarping computation) and then deskewed; and (c) after the lines are desheared.

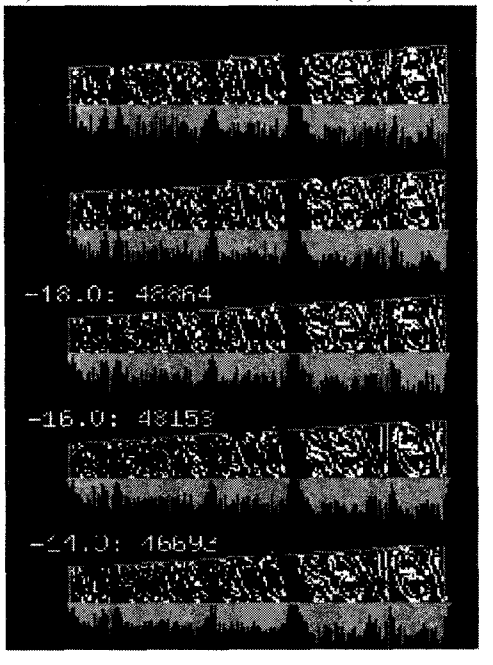


Figure 7. Estimation of Shear

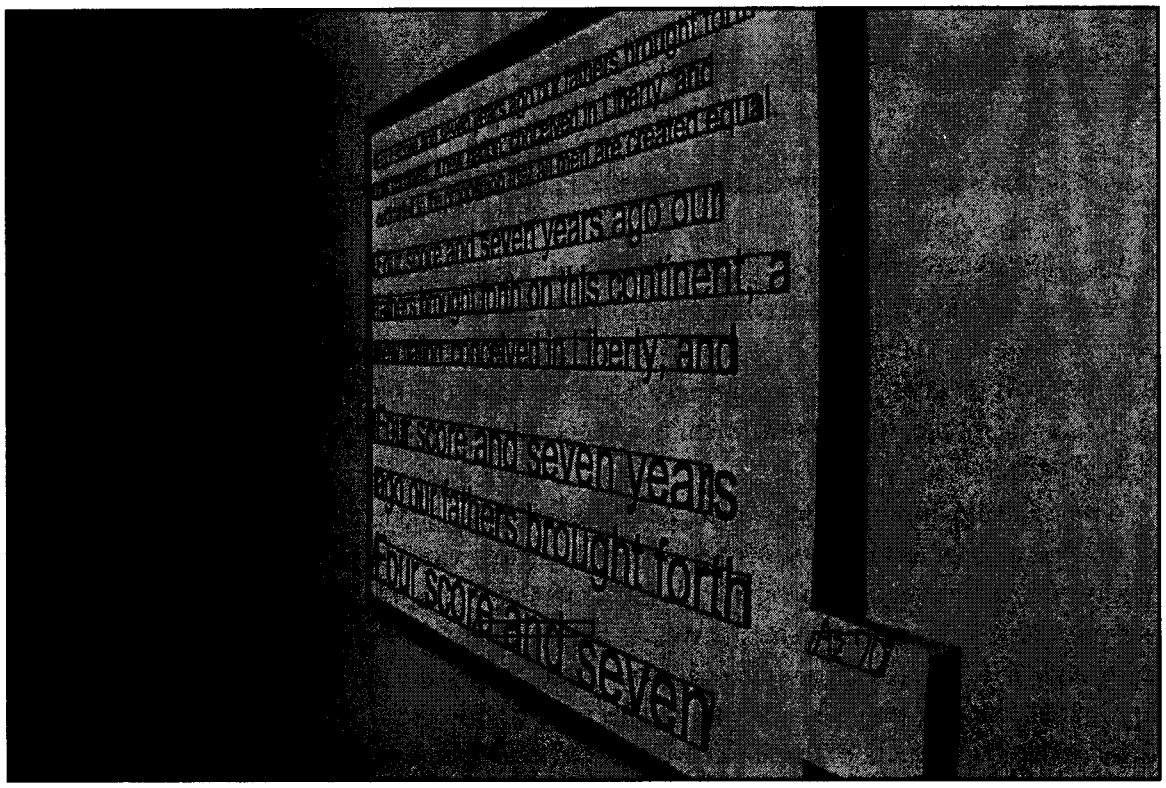
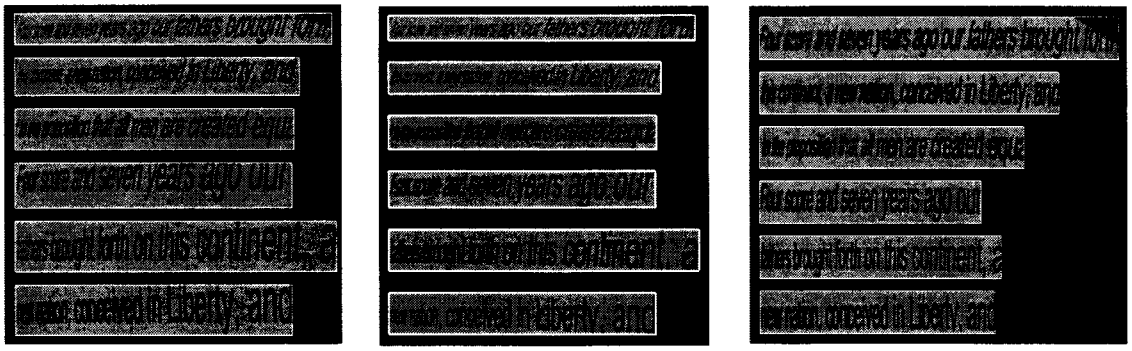


Figure 8. New Boxes from Top Lines, Base Lines, and Shear



(a) Initial Bounding Boxes **(b) Refined Baseline** **(c) After Deshearing**

Figure 9. Warped Text Lines

The transformation used to rectify the image of each text line, L_j, occuring in an obliquely viewed image, O_i, is a projective transformation, T_{ij}, of the text plane. This transformation is described by

$$m' = Hm \quad ,$$

where H is a 3×3 matrix that maps the homogeneous coordinates $m = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ in O_i to the homogeneous

rectified coordinates $\boldsymbol{m'} = \begin{bmatrix} sx' \\ sy' \\ s \end{bmatrix}$ in a normalized image N_i. The horizontal and vertical vanishing points are mapped to the points at infinity in the horizontal $\left(\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}\right)$ and vertical $\left(\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}\right)$ directions. This process takes care of the perspective foreshortening in both directions, as well as the skew and rotation. The remaining four degrees of freedom correspond to the origin and scale factors that place the line in the normalized image N_i. The image N_i, which contains all of the rectified lines from image O_i, is then sent through the OCR process. (We are currently using the Scansoft, Inc. DevKit2000 OCR package.) Figure 10 shows, for the 70 degree azimuth test image, the recognition results overlaid on the normalized image.

To measure the improvement in recognition performance due to the rectification process, we ran our process on a set of test images of a poster containing text viewed at various angles. The evaluation was performed semiautomatically by the process shown in Figure 11. We generated a ground truth data set (including the bounding boxes as well as the identities of the characters) by running the text detection and OCR process on a reference image $\boldsymbol{R}$, and manually correcting any recognition errors. For our reference image we used a fronto-parallel view of the poster. In each of the test images $\boldsymbol{O_i}$, the positions of the four corners of the poster were automatically detected and used to compute a transformation $\boldsymbol{E_i}$ that maps a pixel position in $\boldsymbol{O_i}$ into a corresponding position in $\boldsymbol{R}$. By applying $\boldsymbol{T^{-1}_{ij}}$ and then $\boldsymbol{E_i}$, we mapped the OCR results for line $\boldsymbol{L_j}$ from normalized coordinate space into the coordinate space of $\boldsymbol{R}$. We expect that the lines and characters of text in correctly rectified images will coincide with those in a true fronto-parallel image. An automated process compares the recognized results to truth data on a line-by-line and character-by-character basis. Figure 12 shows the reference image overlaid with the ground truth data. Figure 13 shows the OCR results of Figure 10 after the inverse mapping back into the coordinate system of reference image **R**.

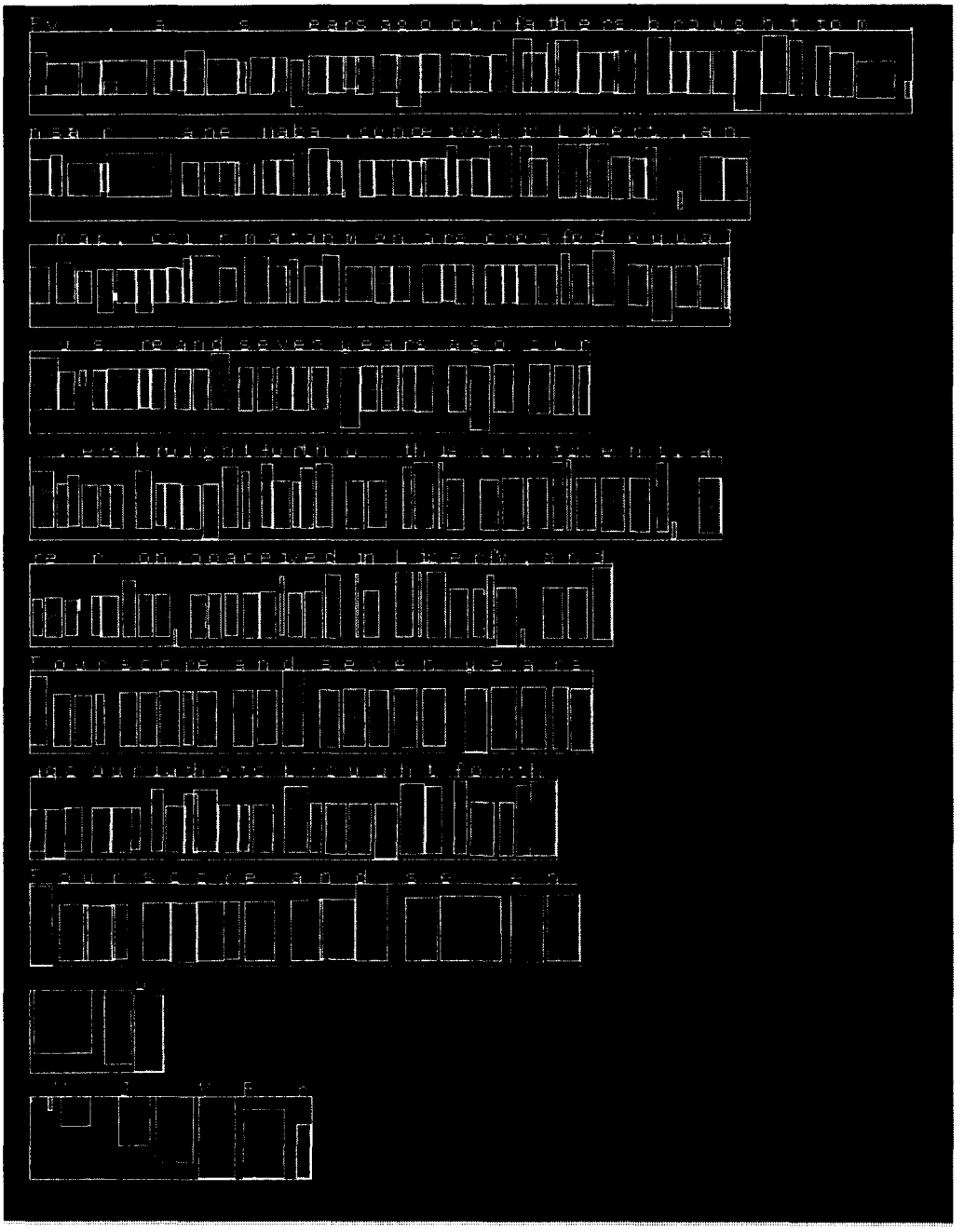


Figure 10. OCR Results Overlaid on the Normalized Image

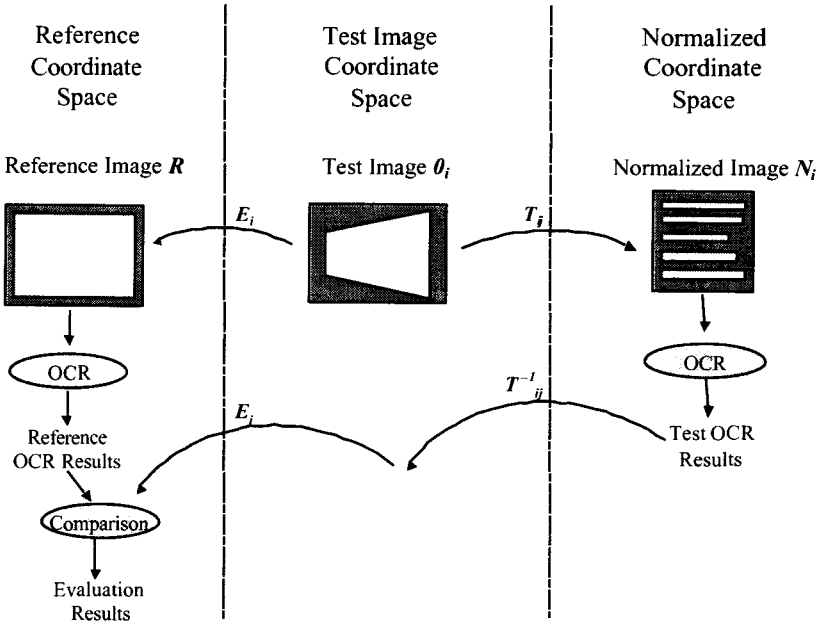


Figure 11. Evaluation Procedure

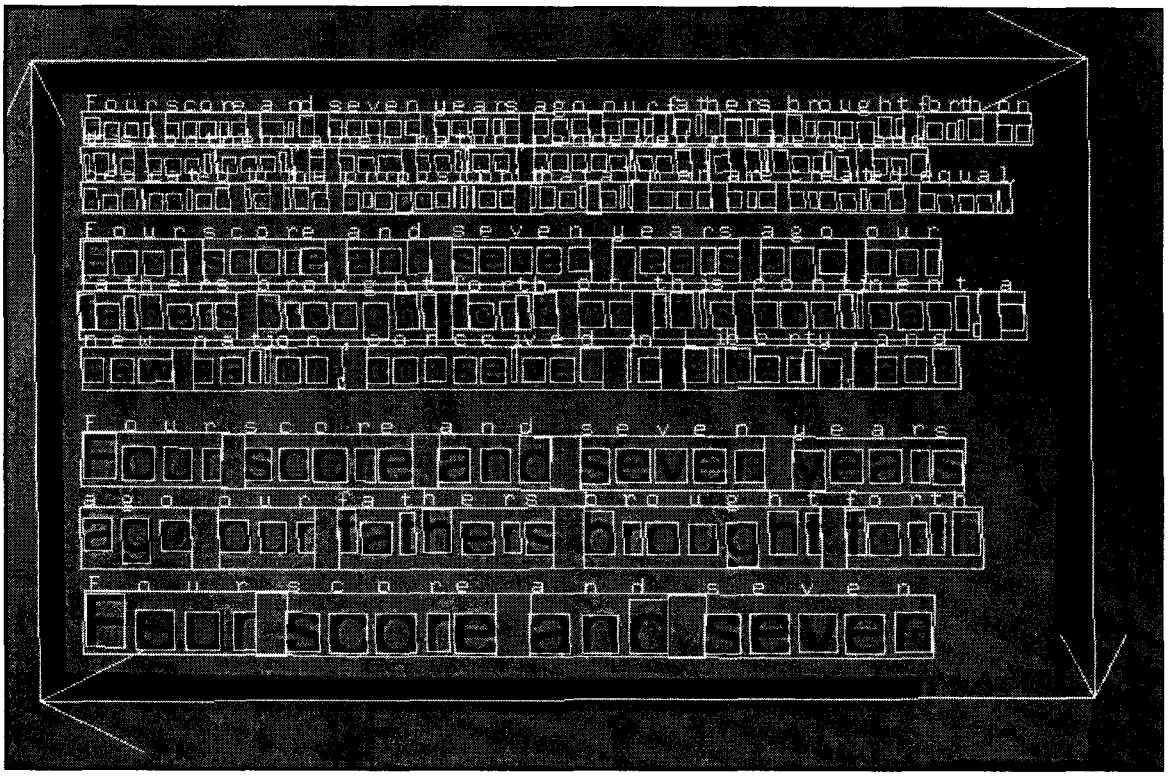


Figure 12. Reference Image and Ground Truth Data

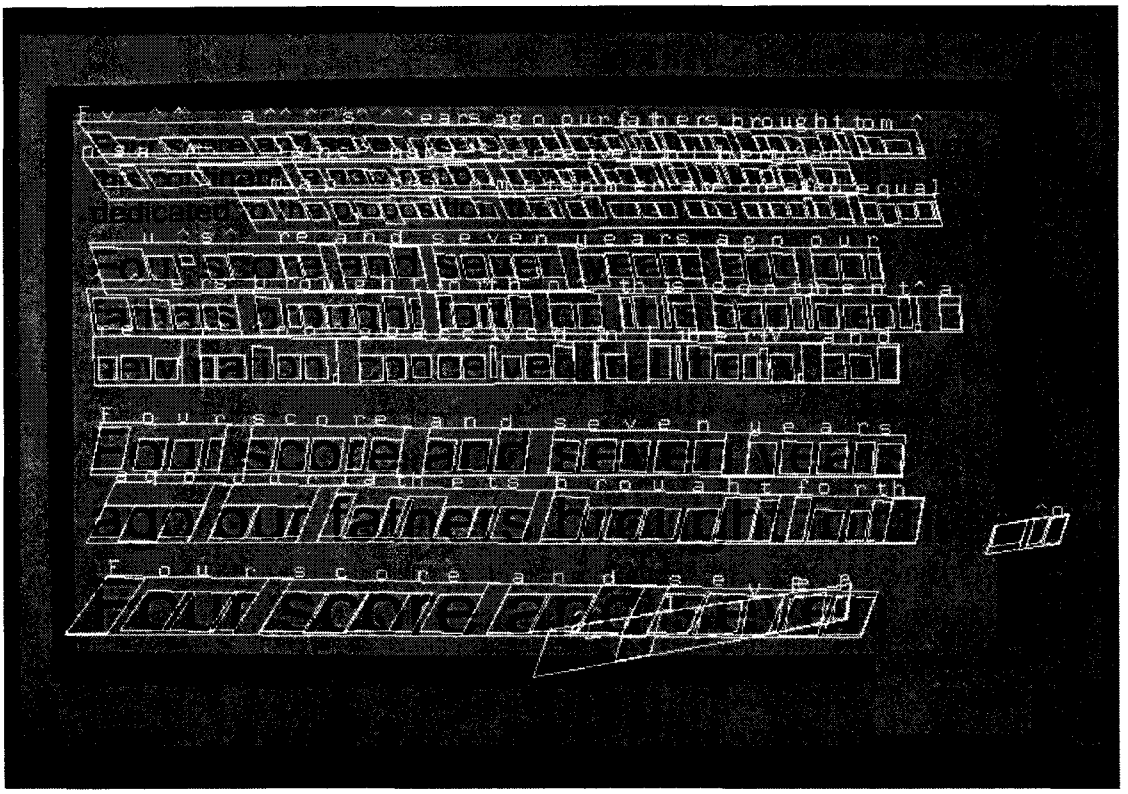


Figure 13. Test Results Overlaid on the Reference Image

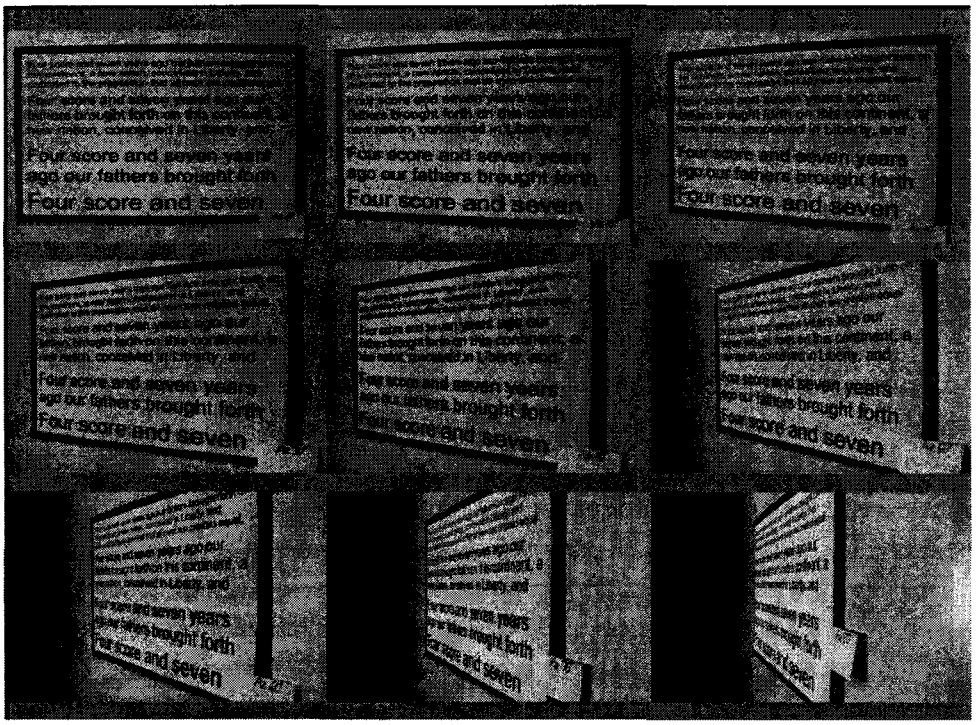


Figure 14. Azimuth Test Image Set

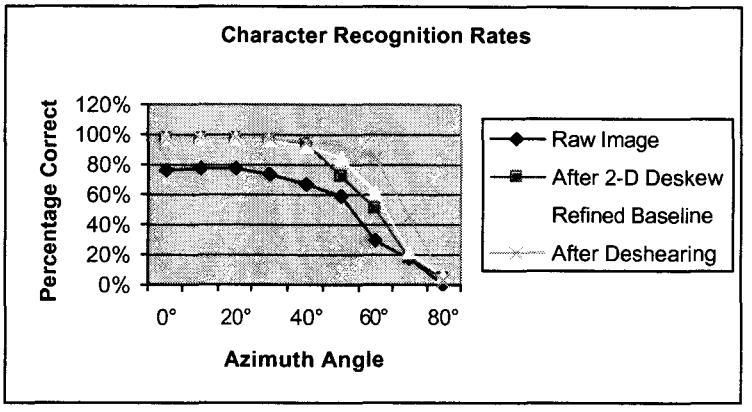


Figure 15. Test Results

Figure 14 shows one series of test images where the azimuth angle varies in increments of 10 degrees. Figure 15 shows the character recognition results as a function of azimuth angle for the various version of rectification. "Percentage Correct" means the number of characters recognized correctly, divided by the number of characters in the reference ground truth data set. As expected, the performance drops as the azimuth angle increases. At the most oblique angles, when the character stroke width and/or spacing between characters becomes one pixel or less in the original image, the resolution available for the interpolation process during rectification is not sufficient to adequately preserve the character features. The graph shows that each of the three processing steps contributes to the increase in performance. These improvements are greatest at the more oblique angles.

CURRENT AND FUTURE WORK

We are currently developing methods that will automatically compute the rectification parameters for all text lines in a single image simultaneously. This process is expected to yield more reliable estimates of the rectification parameters, especially for short lines of text. Part of this process will automatically determine which sets of lines (the top and base lines from lines of text, and other significant lines in the image) have a common vanishing point and are therefore truly parallel and lie within a single plane.

Because lens distortion can affect the accuracy of our estimation of straight lines, we are implementing a preprocessing step that characterizes and corrects for lens distortion in the imagery. We estimate the parameters of our lens distortion model by providing the sequence of pixels along four or more straight lines detected in the set of image frames in a video sequence captured with the same lens setting.

In the future we plan to use the information from multiple images in the video sequence to produce a more robust estimate of the rectification parameters. Methods we will consider include exploiting the consistency across frames of the rectification parameters themselves, tracking features of the text regions such as baselines and sign borders, and deducing the orientation of text planes by tracking the displacement of individual points through the image sequence.

ACKNOWLEDGEMENT

This work was supported in part by the Advanced Research and Development Agency (ARDA).

REFERENCES

Clark, P. and M. Mirmehdi. 2000. "Location and Recovery of Text on Oriented Surfaces," *SPIE Conf. on Document Recognition and Retrieval VII*, pp. 267–277 (January).

Jain, A., and S. Bhattacharjee. 1992. "Text Segmentation Using Gabor Filters for Automatic Document Processing," *Machine Vision and Applications*, Vol. 5, pp. 169–184.

Jain, A., and B. Yu. 1998. "Automatic Text Location in Images and Video Frames," *Proc. ICPR*, pp. 1497–1599.

Li, H., and D. Doermann. 1998a. "Automatic Identification of Text in Digital Video Key Frames," *Proc. Intl. Conf. on Pattern Recognition*, pp. 129–132.

Li, H., and D. Doermann. 1998b. "Automatic Text Tracking In Digital Videos," *Proc. IEEE 1998 Workshop on Multimedia Signal Processing*, pp. 21–26.

Li, H., D. Doermann, and O. Kia. 1998. "Text Extraction and Recognition in Digital Video," *Proc. Third IAPR Workshop on Document Analysis Systems*, pp. 119–128.

Li, H., D. Doermann, and O. Kia. 1999. "Automatic Text Detection and Tracking in Digital Video," *IEEE Trans. Image Processing—Special Issue on Image and Video Processing for Digital Libraries*, pp. 147–155.

Lienhart, R. 1996. "Indexing and Retrieval of Digital Video Sequences based on Automatic Text Recognition," in *4th ACM International Multimedia Conference*, Boston (November).

Ohya, J., A. Shio, and S. Akamatsu. 1994. "Recognizing Characters in Scene Images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 2, pp. 214–220.

Sato, T., K. Takeo, E. Hughes, and M. Smith. 1998. "Video OCR for Digital News Archive," *Proc. 1998 Intl. Workshop on Content-Based Access of Image and Video Databases (CAIVD '98)*, Bombay, India, IEEE Computer Society, ISBN 0-8186-8329-5 (3 January).

Smith, M.A., and T. Kanade. 1995. *Video Skimming for Quick Browsing Based on Audio and Image Characterization*, Technical Report CMU-CS-95-186, Carnegie Mellon University (July).

Wu, V., R. Manmatha, and E. Riseman. 1997. "Automatic Text Detection and Recognition," in *Proc. Image Understanding Workshop*, pp. 707–712.

Yeo, B.-L., and B. Liu. 1996. "Visual Content Highlighting via Automatic Extraction of Embedded Captions on MPEG Compressed Video in Digital Video Compression: Algorithms and Technologies," in *Proc. SPIE 2668-07.*

Zhong, Y., K. Karu, and A. Jain. 1995. "Locating Text in Complex Color Images," in *Proc. Third Intl. Conf. on Document Analysis and Recognition*, Montreal, Canada (14–16 August).

MediaBrowse: a Workbench for Multimedia Information Fusion

Jisheng Liang **Giovanni B. Marchisio**
Insightful Corporation
1700 Westlake Ave N, Suite 500
Seattle, WA 98109, USA

Abstract

MediaBrowse is a research prototype for multimedia information retrieval. It integrates existing information extraction technologies in video processing with advanced information retrieval methods. MediaBrowse can extract multimedia attributes from digital video streams, index and store video attributes and metadata in a database system, browse and retrieve video sequences based on a combination of multimedia feature vectors and attributes. Currently, video attributes include: 1) key frames; 2) captions; 3) audio; 4) text from video and 5) presence, location and number of faces in a frame. The Information Retrieval (IR) modules combine these multiple (and potentially noisy) sources of information more effectively when compared to retrieval from a single source. The GUI module allows the user to express multimedia queries based on these attributes.

The system's architecture is designed to allow integration of additional functions for content extraction and information mining. We developed a modular framework for supporting pluggable information extraction (IE) and indexing modules. We designed and implemented a multimedia database API that supports both video and multipage scanned documents. The search engine is based on Insightful's proprietary search technology for Latent Semantic Retrieval (LSR). We designed a query syntax which supports modular IR engine pluggability and general fusion logic for multimedia queries. We tested the retrieval and browsing function of MediaBrowse on a video databases of news broadcasts.

1 Overview

The goal of this research is to provide intelligence analysts with a system that allows intelligent retrieval and browsing of large digital video databases containing commercial broadcasts (particularly news broadcasts). The two basic major challenges are (1) extraction of meaningful information from video data, and (2) information retrieval methods for this data that are robust to errors of interpretation. Our research addresses aspects of both aforementioned problems. In the context of information extraction, we integrated existing methods to extract and index textual captions and image objects from a given video. In the context of information retrieval, we developed robust indexing and search methods, and methods to combine multiple sources of information. Finally, we developed a prototype graphical user interface-based Mediabrowse system to demonstrate the feasibility of our ideas.

MediaBrowse is a research prototype that allows intelligent retrieval and browsing of large (up to approximately 1 million key frames) digital video databases containing news broadcasts. It integrates existing technologies in video processing with advanced information retrieval technology to produce a complete system for video browsing and retrieval. The functionality of MediaBrowse can be divided into the following categories:

- Extraction of monomedia attributes from digital video
- Storage of monomedia attributes and metadata in a database system
- Retrieval of video sequences based on a combination of monomedia feature vectors and attributes
- Visual management of indexing, querying, and IR functions

Figure 1 shows the architecture of the MediaBrowse system. The process begins by running Information Extraction (IE) modules to extract multimedia video attributes. Currently, video attributes include: 1) key frames; 2) captions; 3) audio; 4) text from video and 5) presence, location and number of faces. These attributes are entered into a MediaBrowse database. Indexing modules for the Information Retrieval (IR) engines then index the extracted information in order to support efficient retrieval. The IR modules combine these multiple (and potentially noisy) sources of information more effectively when compared to retrieval from a single source, e.g., audio. The GUI module allows the user to express multimedia queries based on these attributes, and to access the database (through the Server) in order to browse the contents and display IR

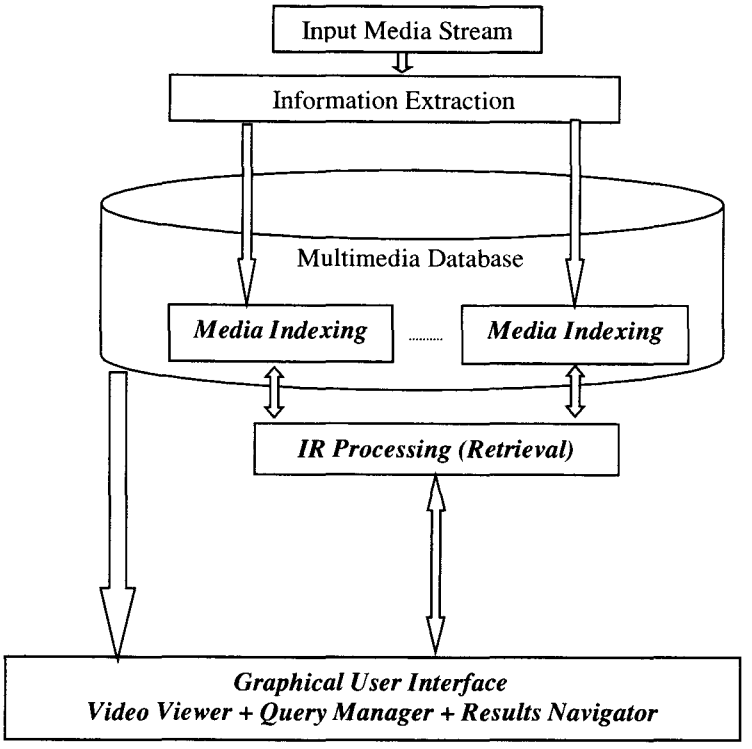


Figure 1: general architecture of MediaBrowse system

results. The system's architecture is designed to allow integration of additional functions for content extraction and information mining.

We designed MediaBrowse to be backward compatible with our DocBrowse system for document imaging [Bruce97]. We designed an extension to the DocBrowse query syntax which supports modular IR engine pluggability and general fusion logic for multimedia queries. We designed a modular framework for supporting pluggable IR engines and indexing. We designed and implemented a multimedia database API to be used by the IE and indexing modules. It supports both video and multipage scanned documents. We integrated the existing DocBrowse information extraction functionality with the new framework. The search engine is based on MathSoft's proprietary search technology for Latent Semantic Retrieval (LSR). The MediaBrowse GUI and database server can run either on a Windows or UNIX environment. The video segmentation and information extraction modules run on UNIX. MediaBrowse supports video stream encoded in MPEG-1 format.

This paper is organized as follows. Section 2 outlines database architecture. Section 3 describes the various modules for information extraction that we have integrated in MediaBrowse. Section 4 describes the indexing and search schemes. Section 5 describes the Graphical User Interface (GUI). Finally, in Section 6, we summarize and list possible future work.

2 Multimedia Database

The MediaBrowse database API serves as the interface between the information extraction process and the rest of the information retrieval/browsing system. The extraction process populates the database with a description of each extracted entity. The database is organized around the hierarchical structure of the medium the database is to contain. The hierarchical structure is shown in Figure 2.

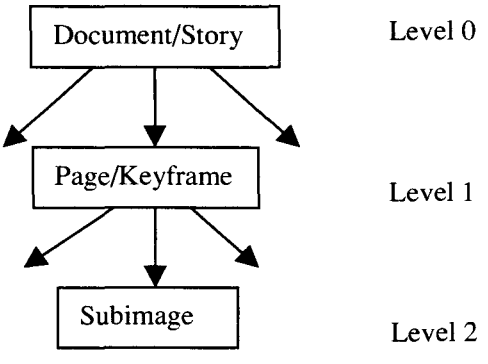


Figure 2: hierarchical representation of video or document images

The MediaBrowse GUI can treat subimages either as text (e.g. a word bounding box) or as an image (e.g. a logo or face). Entities that are neither textual nor iconic can be associated with a document/story or page/keyframe. The MediaBrowse GUI uses external modules to display such entities and enter them into queries. Attributes (name-value pairs) may be associated with entities at all three levels. Subimage entities may be given an engine designator to specify which information retrieval engine should handle the entity when it is specified as a query term. Designators are mapped to IR engines by a configuration file.

The information extraction and indexing processes use the data subdirectory to store any additional information that may be needed by the IR modules or foreign entity display modules. The content and format of any such information must be specified by the individual modules.

3 Information Extraction

The inputs to the MediaBrowse IE module (called MediaLoad) are video programs. The first step in the MediaLoad process is to segment video programs into the MediaBrowse story:shot data structure. Users should be able to select which segmentation modules to use and specify their parameters.Story segmentation currently is based on information available in the closed-caption text. We assume a new story begins every time when the news anchor speaks.

3.1 Story Segmentation

Two alternative methods may be used when the closed-caption information is not available: 1) a super-histogram method performs shot segmentation first, then groups the detected shots into clusters based on the color histogram of the shots; 2) an equal-space method simply cuts a video program into segments with equal length.

In the long term we could address story segmentation by robust cross-indexing techniques which use additional multimedia features and attributes, such as speaker recognition, audio event classification, etc.

3.2 Shot Segmentation

Shot segmentation module takes as input a video stream, segments the stream into sequences by detecting significant changes between frames, and selects one frame (usually the first frame) to represent each sequence. We integrated and evaluated two different methods for shot segmentation.

Color histogram-based method

Color histogram-based methods employ histogram differences to temporally segment video sequences. The input data is converted to one of a number of different color space representations. A histogram in one or more dimensions is computed using the resulting data. Difference measures are applied to a uniformly sub-sampled sequence of computed frame histograms to measure the corresponding changes. A large difference indicates a possible shot change. The video sequence is then temporally segmented into subsequences based on "sufficiently" large difference values.

We generate a histogram for the Y,U,and V color space of the frame every ten frames, and compare it (by summing the absolute difference) to the previous keyframe's histogram. If this difference is greater than a certain threshold, we output the current frame as a keyframe.

Motion-based method

Motion-based methods use motion features computed from video data to identify shot changes in video sequences. The motion vector information is available in MPEG bitstreams [MPEG]. We use the MPEG Encoded Retrieval and Indexing Toolkit (MERIT) developed at the University of Maryland (UMD) [Kobla96] to parse MPEG-encoded video clips into shots. The analysis is performed in the compressed domain using available macroblock and motion vector information, and if necessary, DCT information. MERIT is fast because it works in the compressed domain, and it is the method of choice for a preliminary selection of keyframes which can be decompressed for further analysis.

Apart from cuts, other types of edits between shots include fades, dissolves, wipes, etc. These special effects cannot be detected with the use of macroblocks alone, since they occur gradually over a series of frames and the macroblocks tend to remain bidirectionally predicted or intra-coded over this span. We detect these transitions by clustering frames after mapping each frame to a point in a low-dimensional space using a technique called FastMap [Kobla97]. Gradual transitions in FastMap space appear as sparsely threaded trails. We use the VideoTrails program developed by UMD to detect additional shot boundaries at gradual transitions.

Finally, the list of shot boundaries is the union of those produced by story segmentation, motion-based shot segmentation, and gradual transition detection. We assign detected segments to the hierarchy as level-1 entities where each segment is associated with indices of its starting and ending frames, and a still image of the starting keyframe.

3.3 Text Recognition

The video OCR module consists of text block detection,

enhancement and recognition. Compared with OCR of scanned documents, text recognition in digital video presents several new challenges. First, the text resolution is often so low that commercial OCR software cannot recognize it reliably. Second, text is often embedded in a complex background, so text separation from the background is difficult.

We employ the text detection, tracking and enhancement modules developed by the UMD [Li99a, Li99b]. Text block detection module extracts image regions containing text from video frames. Text tracking tracks the detected text in consecutive frames based on a multi-resolution SSD (Sum of Squared Differences) measurement. The text enhancement module interpolates a text image to increase the resolution and smoothes the text foreground and background through filtering and correlation of text blocks between frames. We then apply Caere's OCR DevKit to convert extracted and enhanced text block image into character strings. Then we attach the recognized character strings to the corresponding shots.

UMD's detection (TextDetect), tracking (TextTrack) and enhancement (TextEnhancemnet) are separate modules. In order to detect text regions from video sequences and feed them into the OCR engine, we have integrated all of the above modules as follows:

1. Run TextDetect every N frames

2. For every text region detected on a frame k,

   a) Run the TextTrack on the frame sequence {k, k + 1, ..., k + N – 1}
   b) If the SSD is larger than a certain threshold, stop tracking
   c) Run TextEnhancement on the text region sequence
   d) Output each enhanced text region
   e) Perform binarization on each output text region

3. Feed each binary image piece to Caere's OCR engine.

3.4 Face Detection

We have integrated the face detection module developed by CMU [Rowley98] into MediaBrowse. Given an image, a neural network examines small windows of the image, and decides whether each window contains a face.

We apply the face detection algorithm to every keyframe produced by the shot segmentation module. We assign each detected face to the indexing hierarchy as a level-2 image zone represented by its bounding box. Then, the faces are classified into categories based on their location (center, top-left, top-right, bottom-left, and bottom-right) and size (large, small, medium). Category labels are assigned to the faces as level-2 attributes. The system allows users to search for faces based on the number of faces in a keyframe, their position and size.

3 Indexing and Search Modules

MediaBrowse supports several types of searchable information (text, tag, image, image template, subimage) and several information retrieval engines for each type of multimedia information (currently LSR, DupDoc [Rogers99], QBIE, tag). The IR module (shown in Figure 3) consists of:

- A client GUI for query entry and for managing or viewing IR results.
- Dedicated IR engines for text and face search
- An IR server for dispatching queries to the appropriate IR modules and combining search results
- A modular architecture for plugging in IR modules (CORBA API)

We use Insightful's proprietary LSR algorithms to index and search words recognized with commercial speech recognition and OCR engines. These algorithms are based on the ideas underlying Latent Semantic Indexing (LSI); however, computationally and conceptually, they represent a quantum leap with respect to previous work [Berry95]. LSI-based techniques try to overcome the problem of query and document matching by using statistically derived conceptual indices instead of using individual keyword terms.

LSI is based on a matrix factorization method such as Singular Value Decomposition (SVD) and is an optimal special case of multidimensional scaling. Let m denote the number of terms and let n be the number of documents. Given an $m \times n$ matrix A and $rank(A) = r$, the SVD of A is defined as

$$A = U\Sigma V^T$$

where

$$U^T U = V^T V = I_n \text{ and } \Sigma = diag(\sigma_1, \ldots, \sigma_n)$$

are the singular values of A. There matrices reflect a breakdown of the original relationships into linearly independent vectors. The use of q vectors or q-largest singular triplets is equivalent to approximating the original matrix by

$$A_q = U_{m \times q} \Sigma_{q \times q} V_{q \times n}^T$$

where A_q is the best rank-q approximation to A, U and V are considered the term and document vectors, respectively. The truncated SVD captures most of the important underlying structure in the association of

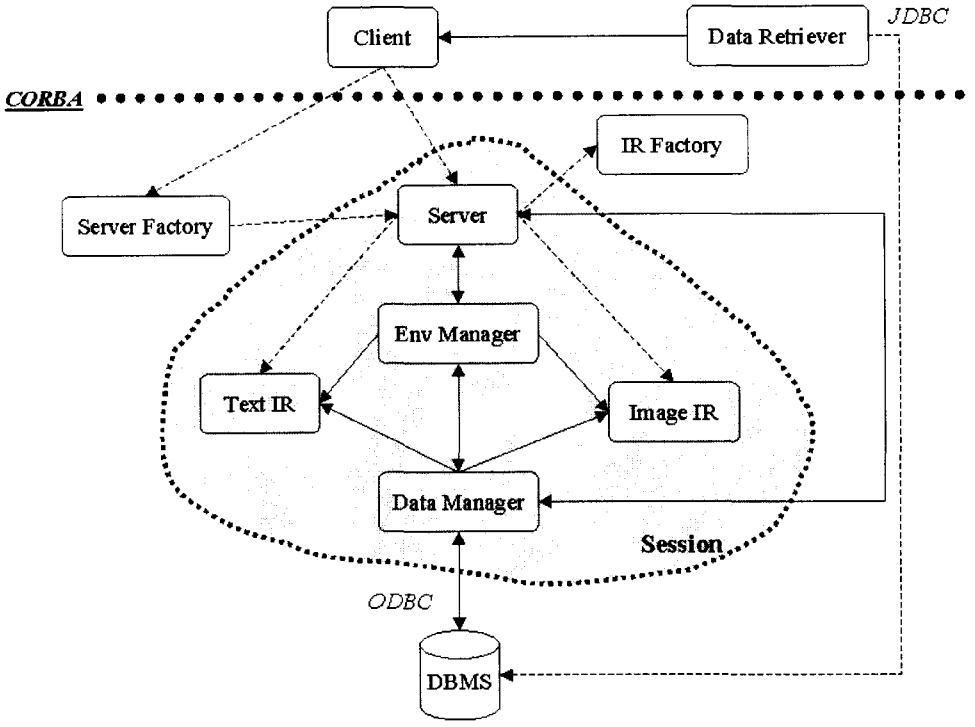


Figure 3: server/client architecture of MediaBrowse IR module

features and symbols, yet at the same time removes the noise or variability in feature values. For purpose of search relevant documents, an input query x is represented as a vector in q-dimensional space:

$$\hat{x} = x^T U_{m\times q} \Sigma^{-1}_{q\times q}$$

The vector $\hat{x}$ can then be compared to all existing document vectors, and the documents ranked by their similarity (nearness) to the input pattern. While theoretically appealing, this approach has serious limitation in speed and scalability. We have developed a theoretically and computationally superior algorithm. Our implementation allows for the inclusion of a large training sample, and supports fast search of similar patterns.

We have shown that the measurement of the similarity between query and document content can be cast as a query optimization (or inverse) problem. This provides many advantages over the unsupervised classification process implied by LSI. The first is a formalism to assess the effect of dimensionality reduction in multivariate transform space on the process of fitting a "noisy or imprecise" query. The second is that our optimization techniques can increase computational efficiency dramatically, increasing, in turn, the amount of information that is available for learning by several orders of magnitude. The third advantage is that the ability to introduce constraints in the optimization can translate into previously unknown IR functionality, including latent keyword feedback and entity tracking. This search methodology also presents the advantage of being robust to speech recognition or OCR errors.

We combine query results from heterogeneous IR engines by weighted linear combination or predictive regression. The two approaches are very similar in philosophy since they both aim at producing a weighted linear combination of the results. A set of training queries is submitted to a combination of IR engines, using a set of documents for which binary relevance judgements (Y/N) or ranking weights with respect to each query are known. One can optimize the combined performance of the IR engines by minimizing a penalty function which is a measure of the absolute relevance or nonrelevance of each document to a query. Alternatively, using the training data, one can construct a classifier that for each query takes as its input variables the binary scores that the n "experts" (i.e., IR engines) assign to each of the documents, and provides as output a binary relevance variable, Y. The classifier is then used to predict the performance of future queries. In the future we will introduce the functionality to update the classifier interactively, using relevance feedback information from the user.

5 Graphical User Interface

The MediaBrowse GUI serves two functions: 1) it provides support for multimedia query entry; 2) it provides tools for displaying and organizing query results.

The GUI consists of three modules: (1) a video viewer; (2) a visual programming interface that permits the user to compose queries; and (3) a query result browser. The video viewer includes functionality to view video frames in a playback mode, browse through key frames and play back individual scenes of interest. The viewer also provides functionality to display an entire video scene as thumbnails of the individual frames.

MediaBrowse users interact with the GUI to create and submit multi-media queries. Queries may consist of textual and graphical terms. The query syntax allows the user to specify which information retrieval engine(s) should process the query, and how the engine results should be combined.

The query syntax is of the form:

- *Query : Query op Term | Term*
- *Term : Engine | text | (Query)*
- *Engine : designator#identifier | designator#(text)*

where

- *op* is a data fusion operator (and, or, min, max, sum, etc.)
- *designator* is a string ALPHA (ALPHA|DIGIT)* indicating which engine should handle the term. Specific IR engines are mapped to designators by a configuration file.
- *identifier* is an entity identification number
- *text* is free-form text to be sent to the default or designated text IR engine.

The information extraction/indexing process associates a designator with each extracted and indexed entity. When a user double-clicks an entity, the entity is added as a term to the current query with its associated designator. The user may edit the query to submit the entity to a different IR engine.

The visual programming interface for queries currently allows the user to combine heterogeneous queries (e.g., keywords and faces) using Boolean operators. The face query includes:

- Number of faces (one, two, three or more)
- Face size (small, medium, large)
- Face position (top-left, top-right, bottom-left, bottom-right, center).

The query-result browser displays keyframes from stories that match an input query and allows the user to choose a story and play the entire video sequence, if so desired. Figure 4 shows the query result browser. Figure 5 displays all the representative keyframes for a given story. Finally, Figure 6 shows a snapshot of the video player and the meta tag and feature viewer. This GUI component allows the user to inspect indexing output such as video OCR and ASR. Shown here is the raw output of the video OCR module.

We developed the entire MediaBrowse user interface in Java so that it is platform independent and can be run through a Web-browser. This Java front-end client application communicates with a processing server and a database server via standard CORBA interfaces.

6 Conclusion and Future Work

We performed a number of qualitative multimedia information retrieval experiments on a smaller database of 12 hours of digitized video. This database was useful primarily for developing the MediaBrowse indexing infrastructure, but it was not ideal for multimedia retrieval experiments. Nevertheless, we demonstrated that under noisy and error-prone conditions, the combination of information from multiple sources, i.e., captions, audio, and images, has the potential for more effective information retrieval, when compared to retrieval from a single source of information. Our results are limited by two factors: 1) the shortcomings of some of the multimedia extraction techniques that we integrated in the system; 2) the small size of our test database. For instance, a qualitative assessment of results indicates that video OCR achieves an average precision of only 20%, while face detection is successful in about 50% of cases. Our inductive learning algorithm requires a much larger corpus for training.

Having established a framework for combining multiple sources of information for information retrieval, we will explore a few research areas in greater detail in future work.

Video Segmentation

In the long term we will address story segmentation by robust cross-indexing techniques which employ additional multimedia features and attributes, such as speaker recognition, audio event classification, etc. In particular, we propose to develop a mechanism for event profiling in time, based on the dynamics of clusters of multimedia attributes. We will base this approach to event profiling on our fast and scalable algorithm for Latent Semantic Retrieval.

Text Recognition

Text captions typically consist of characters that form regions of high intensity contrast against the background. Characters usually have a homogeneous color. Typical captions are oriented horizontally and the spatial frequency of regions containing text segments is relatively high. Text also shows spatial cohesion – characters of the same text string are of similar heights, orientation and spacing. By detecting these properties of text in video, we can extract regions from video frames that contain textual information. We will explore a method to segment a video frame into text and non-text areas based on localized texture analysis, where the key observation is that the texture characteristics of text regions are typically very different from those of the non-text regions. We will explore the use of wavelets and wavelet packets for texture characterization and segmentation at a range of different scales. The wavelet transform can be interpreted as a multi-scale edge detector that represents the singularity content of an image at multiple scales and different orientations. We will use the hidden Markov tree (HMT) model as the classifier to distinguish between text and non-text textures, since the HMT is well suited to images containing singularities (edges and ridges) [Choi00]. The HMT is a tree-structured probabilistic graph that captures the statistical properties of the coefficients of the wavelet transform.

Text detection, tracking and enhancement are performed sequentially and separately in the existing systems, even though they are clearly dependent on each other. Instead of using some simple heuristics, we propose to develop a coherent scheme for integrating the above tasks by extending the idea of hidden Markov tree model from a single frame (2-D) to a sequence of frames (3-D). When a detected text block does not have an equivalent in subsequent frames, or its tracked blocks in subsequent frames show a significant difference, we are less certain that the block should be classified as text. Equivalent text strings in consecutive frames also provide redundant information that can be used to refine text coordinates.

In addition to the process of segmenting and recognizing individual characters, we will explore methods for detecting entire words or phrases as single entities directly in video frames. Words have more features than isolated characters; thus, the recognition of whole words is faster and more dependable than character recognition, especially in the presence of image noise. We propose to develop a word image spotting algorithm that is language independent and not restricted to a pre-determined alphabet. We generate the query keyword from cutting-and-pasting of sub-images from video frames or with input from a word processor. We first apply the algorithm for detection of text regions to video frames. We then segment the detected text regions into text-lines. We compute localized features by applying wavelet transforms and morphological operations to the input keyword image. We compute the same set of features on images of the extracted text-lines. A probabilistic signature-matching algorithm is then used to find the most probable location of the keyword in the text-line images.

Graphic Object Detection

For segmentation and indexing of graphic object such as logos and flags, we will again use the aforementioned segmentation algorithm. From each segmented object on the image, which has not been identified as text, we will extract features, and index these features in the database. The features we have found most useful for the characterization and indexing of logos in the presence of various types of distortions are the wavelet transform coefficients of the region [Bruce97].

Information Retrieval

IR research will consider:

- Extension of text retrieval component to cross language retrieval. This will allow the intelligence analyst to establish semantic links across video text indices (as derived from ASR, closed caption or video OCR) from multiple language sources. The approach of using Machine Translation (MT) is clearly unrealistic for large multilingual video databases. By identifying and aligning principal axes for the various languages, the LSR algorithm correlates clusters of documents across the various language subspaces.
- Extension of latent semantic analysis to fusion of multimedia attributes. This will be couple with training of multimedia event labels with user relevance feedback in a probabilistic framework.
- Dynamic clustering of database indices in semantic space with respect to a conditioning variable such as time, with the ability to track time evolving patterns and concepts.

Acknowledgements

The authors would like to thank Dr. David Doermann for providing us with the video segmentation and text extraction software and for his help in obtaining digitized video data. This research was funded by the US Department of Defense under contract (MDS904-99-C-2653).

References

[MPEG] ISO/IEC 11172-2:1993 Information technology -- Coding of moving pictures and associated audio for digital storage media at up to

about 1.5 Mbit/s.

[Kobla96] V. Kobla, D. Doermann, and A. Rosenfeld, Compressed domain video segmentation, Technical Report, Center for Automation Research, University of Maryland, 1996.

[Kobla97] V. Kobla, D. Doermann, and C. Faloutsos, "VideoTrails: Representing and visualizing structure in video sequences," *Proc. of the ACM Multimedia Conference*, 1997.

[Rowley98] H.A. Rowley, S. Baluja, and T. Kanade, Neural network-based face detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 20, number 1, pages 23-38, January 1998.

[Berry95] M. Berry, S. Dumais, G. O'Brien, "Using linear algebra for intelligent information retrieval," *SIAM Review*, vol. 37, pp. 553-595, 1995.

[Bruce97] A. Bruce, V. Chalana, M. Jaisimha, and T. Nguyen, "The DocBrowse system for information retrieval from document image data," *Proc. 1997 Symposium on Document Image Understanding Technology*, Annapolis, MD.

[Rogers99] R. Rogers, V. Chalana, A. Bruce, and T. Nguyen, "Exact and near duplicate document detection in DocBrowse," *Proc. 1999 Symposium on Document Image Understanding Technology*, Annapolis, MD.

[Choi00] H. Choi and R. Baraniuk, "Multiscale document segmentation using wavelet-domain hidden Markov models," *Document Recognition and retrieval VII, Proceedings of SPIE*, 2000.

[Li99a] H. Li, D. Doermann, and O. Kia, "Automatic Text Detection and Tracking in Digital Video," *IEEE Transactions on Image Processing - Special Issue on Image and Video Processing for Digital Libraries*, pages 147-155, 1999

[Li99b] H. Li and D. Doermann, "Text enhancement in digital video using multiple frame integration," *Proceedings of SPIE - Conference on Document Recognition and Retrieval VI*, pages 2-9, Jan 1999.

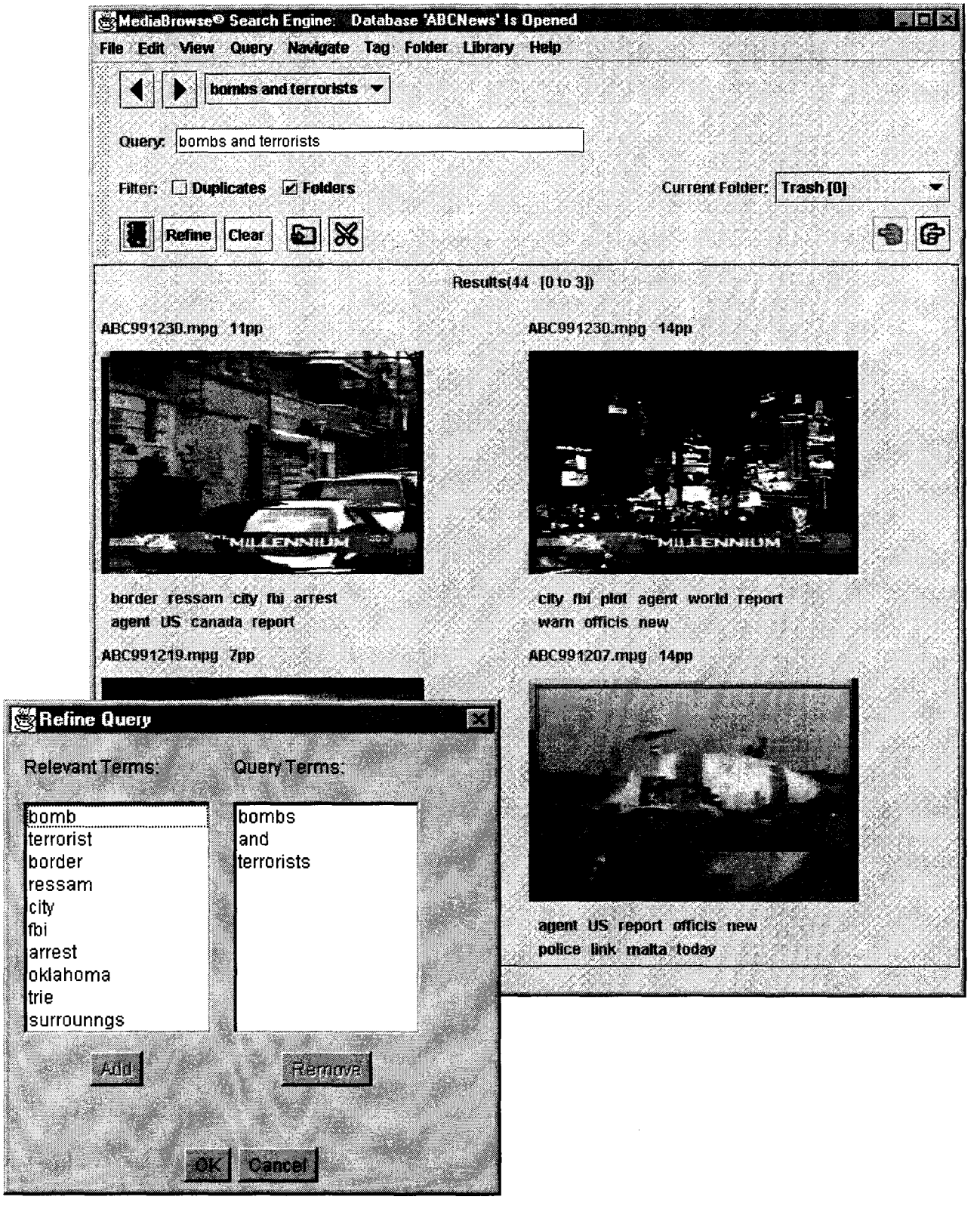


Figure 4: MediaBrowse query result panel and semantic feedback tool

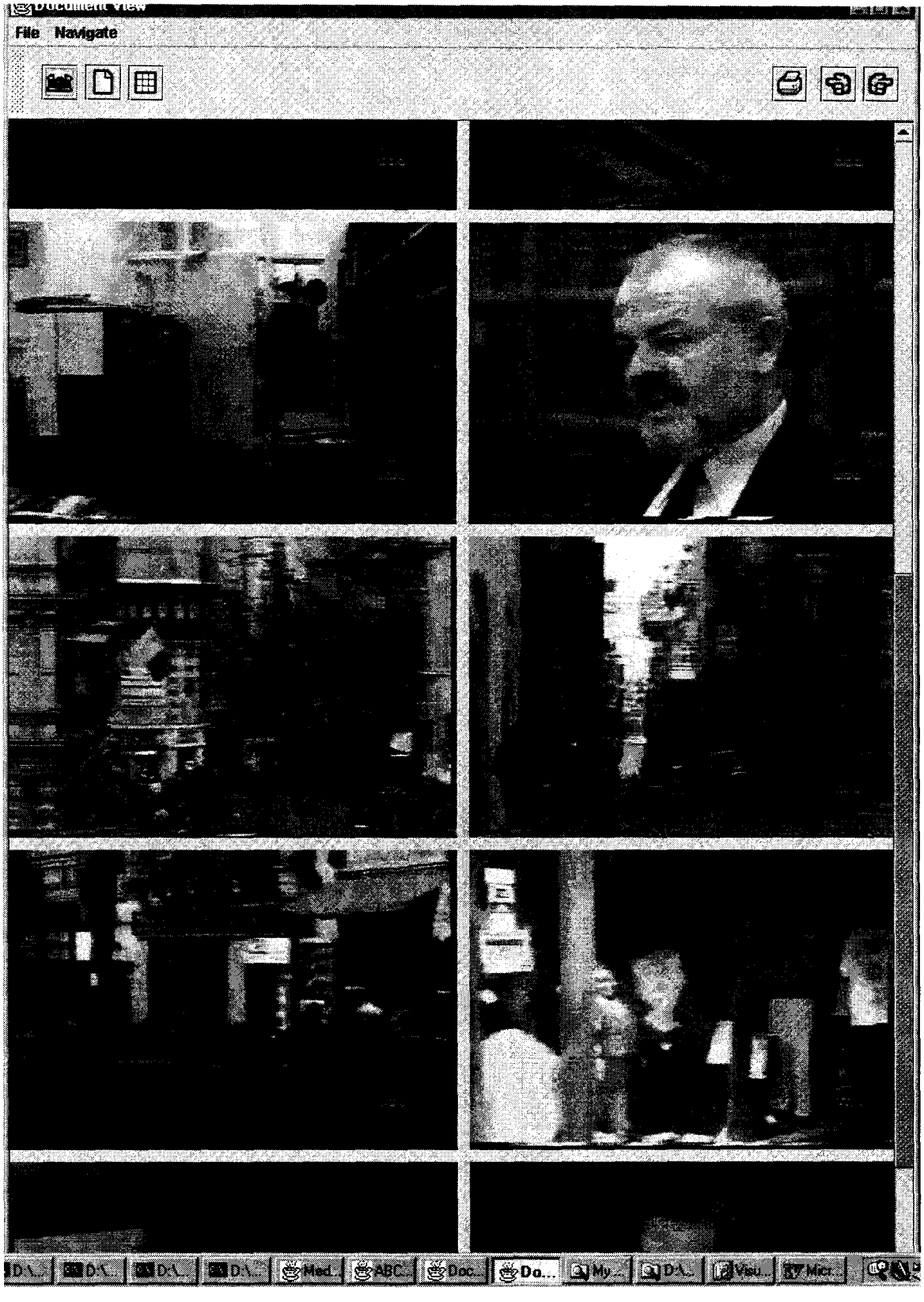


Figure 5: Diagnostic keyframes for a story

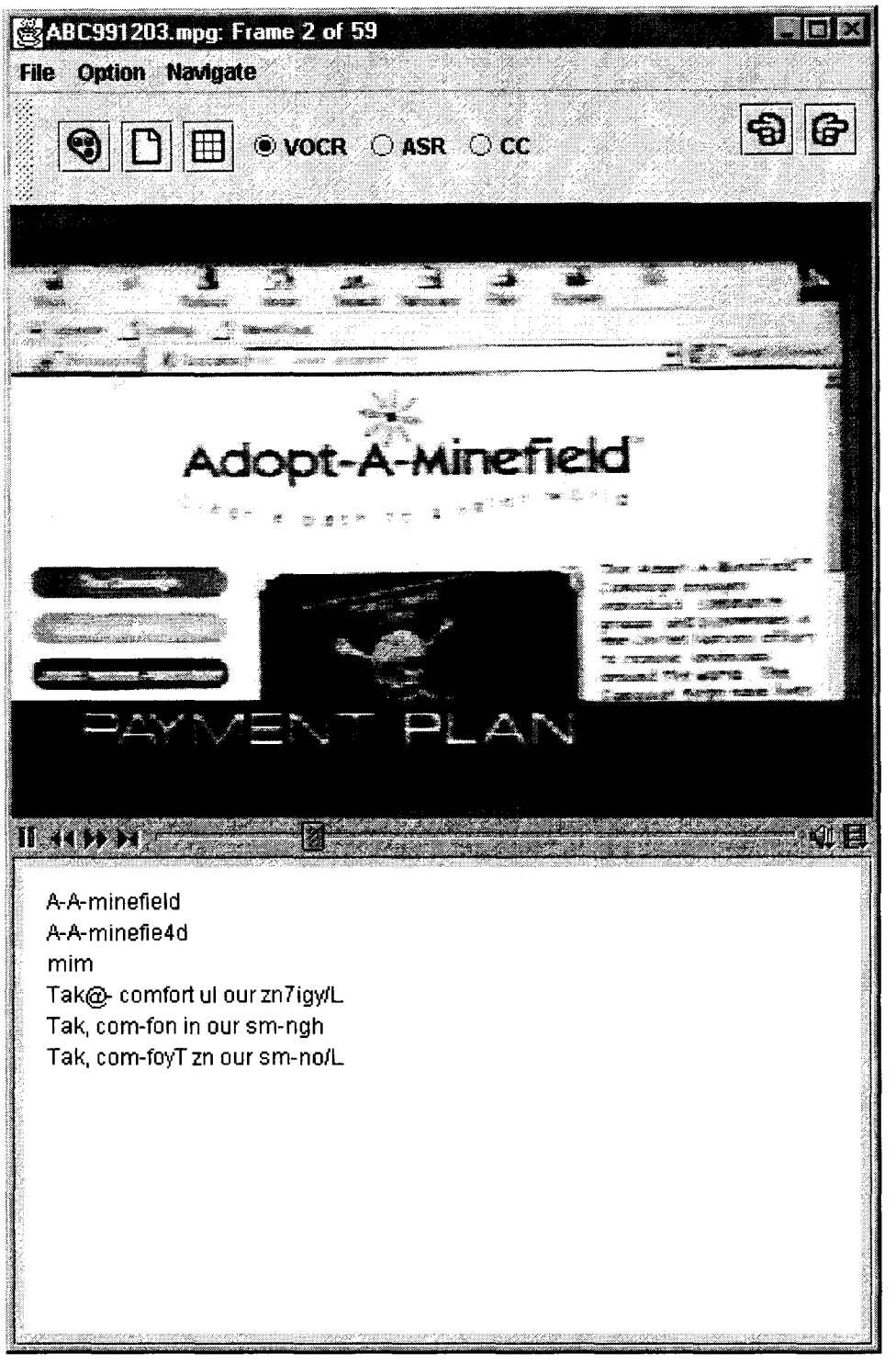


Figure 6: MediaBrowse video player allows the user to inspect the output of the video OCR and ASR.

Projects and Applications

The CMU-Seagate Historical New York Times Project

Robert Thibadeau **Chris DeWan**
Joel Young **Dennis Marous**
Internet Systems Laboratory
Robotics Institute
Carnegie Mellon University
Pittsburgh PA 15213
rht@cs.cmu.edu

Abstract

The abstract should summarize the context, content and conclusions of the paper in 150–200 words. It should not contain any references or displayed equations. Typeset the abstract in 10 pt Times Italic with baselineskip of 11 pt, and indentation of 1 em for each paragraph.

1 Introduction

As part of the effort in online digital libraries, we have undertaken to digitize the microfilm record of the New York Times between its inception in September 1851 and the last issue in the public domain, December 1923. By empirical sampling, this constitutes a corpus of approximately 500,000 newspaper images and has an estimated storage requirement of about .75 terabyte based on approximately 200 pixels per inch at the original newspaper surface.

The purpose of the effort is to create ready access to this entire corpus on the Web in a commercial model that provides free reading but low cost, access-controlled, subscription to enhanced Internet services.

2 The Fidelity Matrix Design Tool

In an earlier effort with the National Academy Press, we designed a system where 1700 books were made available for free reading while visitors were encouraged to buy hard copies (www.nap.edu). This has resulted in higher rates of hardcopy sales than had previously been obtained.

We term this a free-and-fee model that employs a Fidelity Matrix where the rows of the matrix describe the "stuff" offered, the columns describe the subscription conditions, and the cells are checked or not checked. For the National Academy Press, at www.nap.edu, the design Fidelity Matrix is shown in Table 1. For the New York Times Project the proposed Fidelity Matrix is shown in Table 2 on the next page.

We have completed the system for the first two columns of the Matrix (free access and basic access) as well as digitizing the Historical New York Times for the periods of the Civil War (1860-1865) and the Turn of the 20th Century (1895-1905). The results can be observed at www.nyt.ulib.org. To accomplish this work, we had to develop both the data acquisition from raw microfilm and the data presentation capabilities. A great deal of attention was given to automating as much as possible in order to make further capture and presentation as low cost as is possible considering the material available.

Table 1. Fidelity Matrix for www.nap.edu, The National Academy Press web site.

Stuff	Free	Fee
Color Cover Image	X	
Search Title/Author/Category	X	
Table of Contents Hyperlinked	X	
Low Resolution Page Images	X	
Page-Forward/Back Button	X	
Go-to-Page-Number Button	X	
Full-text Search	X	
Print One Page Low Quality	X	
Hard Copy of Book		X

3 System Considerations

The *free* system had to provide for low cost content

Table 2. Fidelity Matrix for www.nyt.ulib.org.

Stuff	Free Access	Basic Access	Advanced Access (planned)	High Performance Access (planned)
	Low Fidelity	*Medium Fidelity*	*High Fidelity Remote*	*High Fidelity Proximal*
Display Tiff-Gif	X	X	X	X
Search Date/Page	X	X	X	X
Search Keyword Full Text		X	X	X
Titles			X	X
Major Themes	X	X	X	X
Detailed Themes			X	X
Browse Picture			X	X
ASCII Text			X	X
Formatted Articles			X	X
Machine XML Access			X	X
High Access Speed				X
Online Ref Librarian			X	X
Picture Browse			X	X

acquisition, basic indexing by newspaper issue date, page and column. Additionally, the *fee* system had to provide for full text keyword search utilizing the most cost effective and rapid means possible. For this we developed automated column segmentation and the unattended application of commercially available OCR. Finally, we incorporated a research component in providing for ground-truth data to be employed for experimentation with new OCR techniques in association with Henry Baird at Xerox PARC. So, while there is a web site that is generally viewed at www.nyt.ulib.org, there are another four web sites that have been developed to enable the maintenance of this one. The principal purpose of this paper is to describe how the various systems interact and also to show acceptance of this approach to document image management by a view of the logs on site usage.

Figure 1 shows all the systems.

The process illustrated in Figure 1 is as follows:

1. **The Microfilm Scanner System** utilizes the Mekel M525 automated microfilm scanner and a Windows NT workstation. Considerable work was necessary to configure this for microfilms of historical newspapers, in part because of poor quality material and large image size variation particular on Sunday editions. Individual, but successively orderd, newspaper images were put to a working directory on the Seagate Xiotech RAID storage device.
2. **The Issue Submission System** is a web site which brings up the successively ordered images in order to assign a newspaper issue date and a page number to each image. This system is optimized so that most of the time all the operator has to do is look at a page, confirm that the date and page number is correct, and hit an "accept" button. Since the microfilm images are in order, this process is

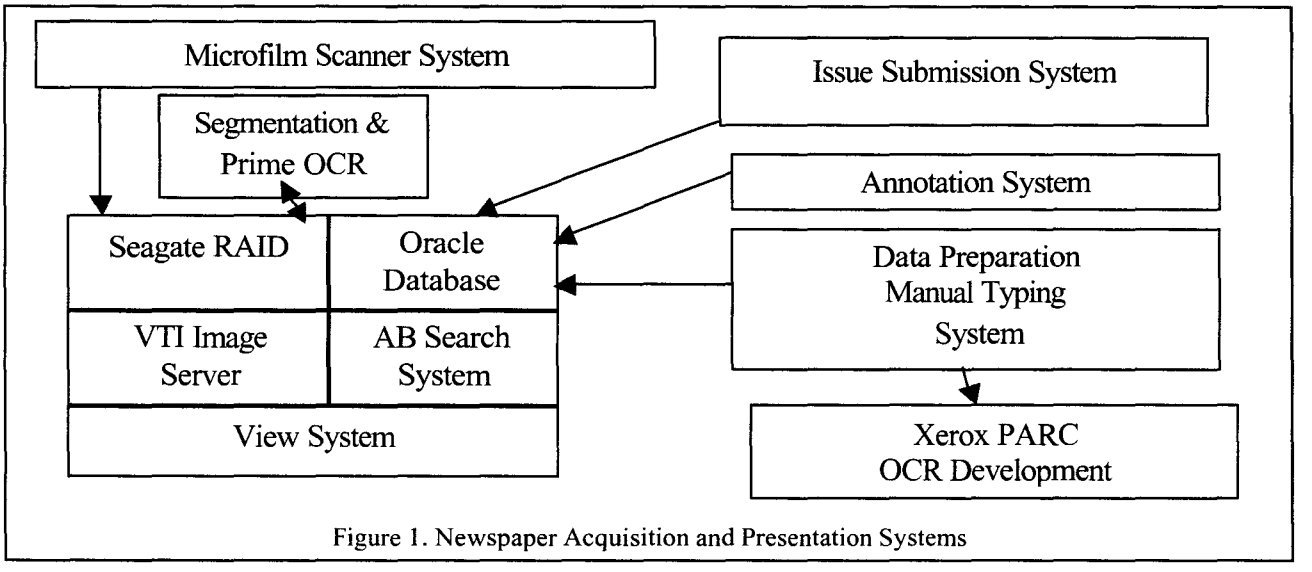


Figure 1. Newspaper Acquisition and Presentation Systems

very quick and also provides a check on the scan quality. Another automated feature of the Issue Submission System is that the image files are moved to the correct directories of the Seagate RAID and the Oracle Database is updated with the appropriate metadata.

3. **The Newspaper Processing System** is automatically applied to newspaper images to compute the columns of each newspaper page and submit and receive the optical character recognition on those columns. The segmentation information becomes a part of the oracle system and the text results from the optical character recognition goes to appropriate directories. Finally, the master full-text retrieval index is updated with the text from the OCR output. This step requires no manual intervention.
4. **The Annotation System** is another web site, and this is always available to any editor with appropriate authorization. The editor may browse newspaper columns by date, page number, and column. He may, for any column he browses, indicate a title, description, and one or more keyword topic areas beginning at that column. The result is that the main Viewing site is automatically updated to provide easy access to this special annotation. On the www.nyt.ulib.org web site, the results of the annotation system is manifest in the topic search and lists of titles and descriptions.
5. **The Data Preparation System** is yet another web site that was specially developed to enable the efficient entry of ground-truth data or the manual entry of the full text. In contrast to most ground-truth systems, we did not attempt to segment every character. On poor quality microfilm, this is infeasible. Rather, we segmented lines of text out of each column. The typist could, on the web site, correct the line segmentation and also type in the text seen for each line. Provision was made to mark unreadable regions. Approximately 100,000 words of ground truth data have been delivered to PARC for further research on OCR.
6. **The View System** is the main web site viewable by visitors. It incorporates access to the images of the newspaper pages utilizing a TIFF-to-GIF image server. The visitor can select a date and read the newspaper from that date. He can also go to one of the themes and get a hyperlink directly to a column. Finally, he can subscribe to the full text search service and utilize our unique search system.
7. **The Search System** is specialized to finite corpus archives. If a search is successful, one or more pages from the newspaper archive become available with hyperlinks to the viewable column of interest. In order to make the search always successful, we utilized a novel technique for dynamically eliminating search developed partly for another site by the principal author, www.antiquebooks.net [1]. Basically a list of all the words appearing in a year of the newspaper is shown and the subscriber simply clicks on a word of choice. Then, immediately (through Javascript), the word list is changed to list only the words that appear on all the pages that have this first word. The user can then select another word to add, and so on, until he has exactly one page and column to look at. He can, at any time, also look at all the page hyperlinks for a particular search, and he can also manually type in arbitrary Boolean expressions (using and, or, not) in conjunction with the dynamically eliminating search. We find this search technique is much faster than alternative methodologies because it allows for the rapid refinement of effective queries. Until the subscription service begins on the search section, it can be found for inspection at http://www.nyt.ulib.org/phase2 (check to see if if "registered" is turned on).

The free system provides for free reading: This means that it simulates the presentation available with a microfilm reader, but you could use a web browser anywhere in the world. You can find the date and page and read it. Technically, this means that we present the images of the page on the Internet. The basic fee system provides for full text search. Technically this means that we employ optical character recognition to obtain the full text from the scans of the microfilm. In both cases a great deal of care also had to be put in creating a genuinely good experience with this material and a genuinely good experience with the full-text search of this material on the Web (since the intent is to charge for this). The experiment is in maximizing digital benefits while minimizing costs of conversion from the film media. We have enough data to judge the benefit and are awaiting further exercise of the system to report costs in terms of both document intake cost and storage and maintenance costs.

4 Acceptance

The web provides a convenient means of testing whether a document image solution is acceptable: we can watch the traffic and the comments on it. Unsolicited email has been highly encouraging particularly among historians who are used to working with optical microfilm readers. The lack of perfect search is a lot better than no rapid search at all. Obviously this is limited value, but the intent in our

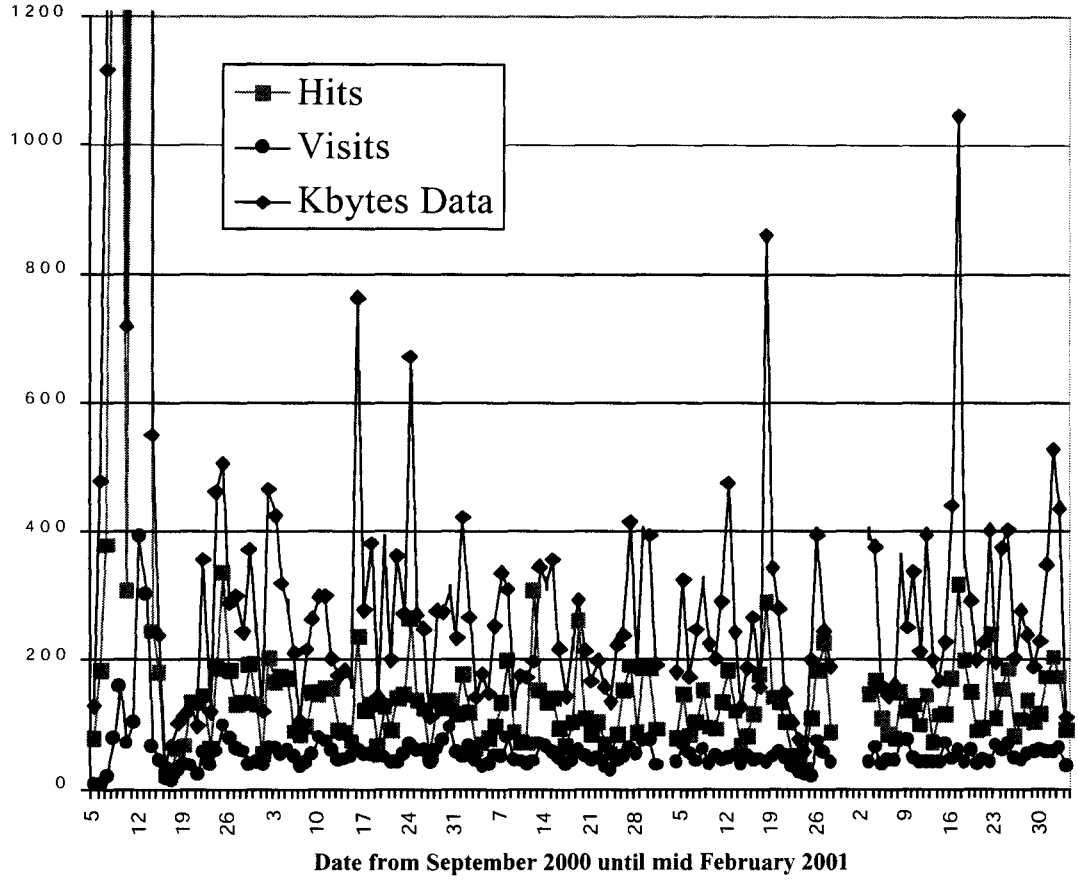


Figure 2. Usage Statistics for NYT.ULIB.ORG, free reading mode only. Note, there are roughly 60 Visits Per Day

design is also to keep the cost low enough that the value is perceived as a fair value.

The usage results are shown in Figure 2. The release date of the free-to-read site was September 5, 2000. Note that the subscription search site has not yet been released to the public. The initial site covers the period of the Civil War. For about two weeks after the site was released we sent email to sites that collected links about the Civil War. This accounts for the initial burst of interest on the Internet. Of more interest has been the sustained patronage on the site. With about 60 unique visitors a day sustained, except for brief periods at the beginning of December and after the New Year when the server was inadvertently down, we can regard this as a fairly popular site. Only rarely are more than a handful of columns read, as indicated by the number of average number of kilobytes associated with a visit. Considering the poor visual quality of the historical record, this is not at all surprising.

Acknowledgements

Seagate Technology LLC provided the support for this work as an investment in creating infrastructure which would enable other newspapers and other microfilm material to be efficiently repurposed to web use. The VTI Image Server is from Visus Technologies (www.vtiscan.com). The archival search is a variant on that developed for Antique Books at www.antiquebooks.net.

References

[1] R. Thibadeau and E. Benoit. Antique Books. D-LIB Magazine, July 1997. http://www.dlib.org/dlib/september97/thibadeau/09thibadeau.html.

Overview of the DjVu Document Compression Technology

Yann LeCun, Leon Bottou, Patrick Haffner, Jeffery Triggs
AT&T Labs - Research, Middletown, NJ
Bill Riemers, Luc Vincent
LizardTech, Inc., Seattle, WA
Contact: lvincent@lizardtech.com

Despite the growing importance of multimedia content, much of the knowledge, culture, and educational material in existence today is still available only in paper form. Bringing this wealth of information into the digital realm in a form that is faithful to the original, easily accessible, and searchable, is an essential step towards making the Internet the World's Universal Library.

DjVu (pronounced "deja vu") provides a way to do all this, and more. It was developed at AT&T Labs over the past several years and purchased by LizardTech in early 2000. DjVu is a compression technique, a file format, and a delivery platform that is specifically designed to enable the creation of digital libraries of printed documents (scanned from paper or digitally produced). It relies on a number of advanced content analysis techniques to achieve high compression ratios, low memory requirements, very fast rendering and indexing pf the material. These techniques have been thoroughly documented in the papers listed in the bibliography, and additional information is readily available on the web (`http://www.djvuzone.org` and `http://www.djvu.com` are good starting points).

A typical page from a book, magazine, or ancient document scanned in color at 300dpi contains on the order of 8 million pixels, and occupies 24MB uncompressed. Traditional compression techniques such as JPEG are notoriously inefficient on several counts:

- typical file sizes for a page will be between 400KB and 2MB at best, which is totally impractical for remote access.
- sharp edges (such as character outlines) are the cause of numerous wasted bits and/or unpleasant ringing artifacts.
- such large images are very slow to render, require a very large memory buffer for the decompressed image in the client, and are not easily zoomable or panable with current web browser technology.
- the text is not normally separated from the image, and therefore cannot be OCRed, indexed, or searched.
- no provision is made for multipage documents, unless one encapsulates the images into a container format such as PDF, thereby adding additional layers of inefficiencies.

The DjVu system alleviates these problems and can handle bitonal documents, low-color (palettized) images, photos and other continuous-tone images, scanned color or grayscale documents, as well as digitally produced documents (from PostScript or PDF).

Bitonal documents are encoded with a technique dubbed JB2, which builds a library of repeating shapes in the document (such as characters), and codes the locations where they appear on each page. Low-color images are compressed the same way, with the addition of a color palette, and a color index for each shape. Continuous-tone images are compressed with a progressive wavelet-based method dubbed IW44 that is on par with JPEG2000 in terms of signal to noise ratio, but whose decoder/renderer is very memory efficient, and extremely fast (3 times faster than the fastest JPEG-2000 mode).

Scanned color documents are decomposed into a foreground plane and a background plane. The foreground plane contains the text and the line drawings compressed as a bitonal or low-color image at maximum resolution (using JB2), thereby preserving the sharpness and readability of the text. The background plane contains the pictures and paper textures compressed at reduced resolution with IW44. Areas of the background covered by foreground components are smoothly interpolated so as to minimize their coding cost. The foreground/background segmenter first detects sharply contrasted areas, and then filters them with several criteria, such as their color uniformity, their geometry, and an estimation of their coding cost.

Digitally produced PDF or PostScript documents are turned into a list of low-level drawing commands using the popular tool GhostScript. This list is then translated into a list of non-overlapping shapes which are subsequently classified into the foreground or the background layer using a number of heuristics. The layers are then compressed as with scanned documents.

Bitonal documents in DjVu typically occupy 5 to 30KB per page at 300dpi, which is 3 to 8 times smaller than Group 4 (used in Fax machines, in TIFF files, and in PDF files). Low-color images such as icons are typically 2 times smaller than with GIF, but can be up to 10 times smaller if they contain lots of text. Photos are about 2 times smaller than JPEG, and about the same as fast modes of JPEG-2000 for the same SNR. An interesting aspect of IW44 wavelet codec is that it is optimized to allow on-the-fly decompression/rendering of the area visible in the display window (and not more) as the user zooms and pans around. This allows to keep the images in compressed form in the RAM of the client machine, and allows to display very large images without excessive memory requirements. Scanned color and grayscale documents in DjVu are typically 30 to 100KB per page at 300dpi, which is 5 to 10 times smaller than JPEG, and about 2-3 times smaller than MRC/T.44 or TIFF/FX. Digitally produced documents with mostly text are typically 2 times smaller than PDF or gzipped PostScript originals at 300dpi, but can be considerably smaller if the documents contain pictures.

DjVu documents are displayed within web browsers through a very compact plug-in (available for all major platforms). Everything in the design of DjVu was optimized to reduce the delay between the user's decision to view a page, and the display of that page on the screen. A multithreaded software architecture with smart caching allows individual document components to be loaded and pre-decoded on-demand. Pages are loaded on demand, allowing random access without prior download of the entire document, and without the help of a byte server. Page components (foreground layer, background chunks,...) are downloaded in sequence and rendered by a separate thread as soon as they are complete. This allows progressive rendering and refinement of the images. The page that follows the page currently being displayed is pre-loaded, pre-decoded and cached automatically

thereby reducing the page-flipping delay. The DjVu viewer has a "modeless" graphical user interface that allows fast zooming, panning, and page flipping with a single mouse operation or keystroke.

The foreground layer can be OCRed and the result embedded back into the DjVu file as a searchable "hidden text" layer. Tools are available to extract that text and translate it into an XML format that includes each word, together with its bounding box coordinates on the page, and the document structure (pages, columns, paragraphs, lines, words). Hyperlinks, annotations, page thumbnails, and other metadata can also be embedded into DjVu documents.

Server-side full-text search can easily be provided using free indexing tools and a few Perl scripts. Large collections have been or are being put on the Web in DjVu with full-text search capabilities, including the NIPS Proceedings (13 volumes, 14,000 pages at 400dpi, 191MB), the Century Dictionnary (8 volumes), along with several national library collections and content from commercial providers around the world. DjVu is currently used by thousands of users to publish and exchange scanned documents on the Web.

DjVu can be seen as a general open platform for document delivery. Much of the code including the full IW44 codec, the palettized image compressor, and the multithreaded decoder/renderer (but not including the best segmenter and the best bitonal compressor) is available as open source under the General Public License (GPL) and can be used as a platform for research on new codecs, segmentation schemes, delivery mechanisms, viewing interfaces, and content analysis systems. More information, source code, benchmarks, and examples can be obtained at `http://www.djvuzone.org`. Plug-ins, compressors, and SDKs can be downloaded from `http://www.lizardtech.com`.

References

[1] L. Bottou, P. Haffner, P. Howard, P. Simard, Y. Bengio, and Y. LeCun. Browsing through high quality document images with DjVu. In *Proceedings of IEEE Conference on Advanced in Digital Libraries*, 1998.

[2] L. Bottou, P. Haffner, P. Howard, P. Simard, Y. Bengio, and Y. LeCun. High quality document image compression with DjVu. *Journal of Electronic Imaging*, 7(3):410–428, 1998.

[3] L. Bottou, P. Haffner, Y. LeCun, P. Howard, and P. Vincent. Un système de compression d'images pour la distribution réticulaire de documents numérisés (DjVu: An image compression system for distributing scanned document on the internet). In *Proceedings of CIFED, Conférence Internationale Francophone sur l'Ecrit et le Document*, Lyon, France, July 2000.

[4] L. Bottou, P. Howard, and Y. Bengio. The Z-coder adaptive binary coder. In *Proceedings of IEEE Data Compression Conference DCC'98*, pages 13–22, Snowbird, UT, Mar. 1998.

[5] L. Bottou and S. Pigeon. Lossy compression of partially masked still images. In *Proceedings of IEEE Data Compression Conference, DCC'98*, Snowbird, UT, Mar. 1998.

[6] P. Haffner, L. Bottou, P. Howard, and Y. LeCun. DjVu : Analyzing and compressing scanned documents for internet distribution. In *Proceedings of International Conference on Document Analysis and Recognition*, Bangalore, India, Sept. 1999.

[7] P. Haffner, Y. LeCun, L. Bottou, P. Howard, and P. Vincent. Color documents on the web with DjVu. In *Proceedings of IEEE International Conference on Image Processing*, Kobe, Japan, Oct. 1999.

[8] Y. LeCun, L. Bottou, , P. Haffner, and P. Howard. DjVu: a compression method for distributing scanned documents in color over the internet. In *Proceedings of Color 6, IST*, 1998.

[9] Y. LeCun, L. Bottou, A. Erofeev, P. Haffner, and B. Riemers. DjVu document browsing with on-demand loading and rendering of image components. In *Proceedings of SPIE, Internet Imaging II*, San Jose, CA, Feb. 2001.

OCR Accuracy of Three Systems on English and Russian Documents of Highly Varying Quality

Kristen Summers
Highland Technologies, Inc.
4831 Walden Lane
Lanham, MD 20706
ksummers@htech.com

Abstract

This paper describes experiments that compare the performance of three leading commercial OCR packages on document images of varying quality, in English and Russian. It considers the performance of OCR developers' kits from ScanSoft, Abbyy, and International Neural Machines on a collection of document images from several sources. It also reports on experiences with preprocessing input to one OCR package with QUARC, a document image restoration package from Los Alamos National Labs.

1 Introduction

This paper describes experiments that compare the performance of three leading commercial OCR packages on document images of varying quality, in English and Russian. The OCR packages include Caere Developers Kit 2000 (version 8) from ScanSoft, FineReader 4.0 from Abbyy, and NeuroTalker 4.1 from International Neural Machines. It also reports on the effects of preprocessing a set of input to CDK 2000 with QUARC [1], an image restoration package. Section 3 describes QUARC. The corpus of documents includes three collections of English document images, and one collection of Russian images, with variations. Section 2 describes the collections. Only CDK 2000 and FineReader 4.0 processed the Russian corpus, since NeuroTalker 4.1 does not support Cyrillic characters.

Each system ran with its own standard preprocessing, without spelling correction. Since the goal was not to evaluate automatic zone detection, we ignored the issue of zoning. In the University of Washington corpus, we applied OCR to the zones provided in the ground truth. In the other corpora, we treated entire pages as single, predefined zones; our hand-entered ground truth was designed to mimic the physical layout of the page as much as possible. We also performed a separate instance of OCR on the declassified government documents, using CDK 2000 together with QUARC.

Precision and recall measure the OCR output accuracy. Precision (P) and recall (R) at the character level are given by

$$P = \frac{\textit{number of correct characters}}{\textit{number of characters in OCR output}} \quad (1)$$

$$R = \frac{\textit{number of correct characters}}{\textit{number of characters in ground truth}} \quad (2)$$

A string edit algorithm determines the correct characters, as described in Section 4. Section 5 presents the results.

2 Corpora

2.1 English Corpora

The English corpora include

- 1147 journal page images from the University of Washington corpus [2]
- 946 page images from a set of government documents that have been declassified and released
- 540 page images from documents produced in discovery in the recent tobacco litigation [3]

We obtained the declassified government documents from a private archiving company, and we downloaded the tobacco litigation documents from the web.[1] The University of Washington provides ground truth for its set; for the other two, ground truth was entered by hand. In these cases, the ground truth entry matches the physical appearance of the page as closely as possible. For instance, consider a stamp on a document. If a group of characters from the stamp align with a line of printed text on the page, the ground truth file includes the stamp characters on that same line (in the order in

[1] We downloaded the full original set of documents, which consists of about 39,000 documents made up of over 150,000 images, but we only acquired ground truth for a subset of 540 page images.

There are then the matters of motivation and organization. The folly of dumping a ready-made water scheme or, for that matter, irrigation or fishery project upon a primitive community is now well recognized. Aid projects in the past often took little account of the needs and aspirations of the users or of their traditions, habits and culture. Worse still, little or no effort was put into teaching them how to operate and maintain the new assets.

Figure 1: Sample image fragment from the University of Washington corpus

which they appear). If a group of stamped characters appears between the lines formed by the printed text of the page, these stamped characters appear in the ground truth as a separate line, between the surrounding lines of printed text.

The quality of the images varies within each set, and the type of documents varies across the corpora. The University of Washington images include pages from various technical journals. Figure 1 shows a sample image fragment. The tobacco documents include many letters and billing statements. Figure 2 shows a sample image fragment. The declassified government documents include agendas, cables, letters, memos, reports, and telegrams; they originate from several different government agencies. Figure 3 shows a sample image fragment.

2.2 Russian Corpus

The Russian documents originated as 118 distinct Unicode text files. Ground truth is thus directly available for these documents.

To create page images, we used Microsoft Word to format the files with various fonts, printed them, and scanned in the printed pages at 300 dpi. For a subset of 26 images, we also created degraded versions, to simulate noisy documents. This process creates a full set of highly varying quality, and it also allows the evaluation of the effects of particular types of degradation on OCR results. Figure 4 shows a sample fragment of a clean, original image, and Figures 5 through 7 show sample fragments of the same image with artificial noise added. The methods of adding noise were as follows.

Scanning at reduced resolution We created a set of images scanned at 200 dpi and a set scanned at 100 dpi.

Adding speckle We copied speckle from genuinely noisy documents and added it to the original clean documents.

Adding black edges Many documents acquire black borders as a result of photocopying or other processing (such as microfilming). We copied a small set of such edges from documents that acquired them "naturally" and added these to the clean Russian page images.

Adding skew We applied skew to the images, ranging from 2° to 12°.

Faxing We faxed the pages once, to create one set of degraded documents. We then faxed these degraded documents a second time, for a second set of more degraded documents.

Photocopying through a filter In order to create low-contrast images, we photocopied the pages through a filter of colored plastic. We scanned these pages with and without scanning correction. Figure 5 shows a sample image fragment with this type of noise, scanned with correction.

Photocopying through clear plastic In order to create images with "faded" areas, we photocopied the pages through multiple layers of clear plastic. We produced one set by photocopying through 4 layers and another set by photocopying through 5 layers. Figure 6 shows a sample image fragment that was photocopied through 4 layers of clear plastic.

Reducing and Re-enlarging In order to create fuzzy images, we photocopied the pages at a reduced scale and then photocopied the reduced versions at an enlarged scale that yielded the original size. We produced one set by reducing the images to 50% and re-enlarging, a second set by reducing the images to 33% and re-enlarging, and a third set by reducing the images to 25% and re-enlarging. Figure 7 shows a sample image fragment that was reduced to 33% and then re-enlarged to its original size.

3 QUARC Pre-Processing

QUARC (QUality Assessment and Restoration for OCR) is a system for pre-OCR image cleanup that was developed at Los Alamos National Labs. It is

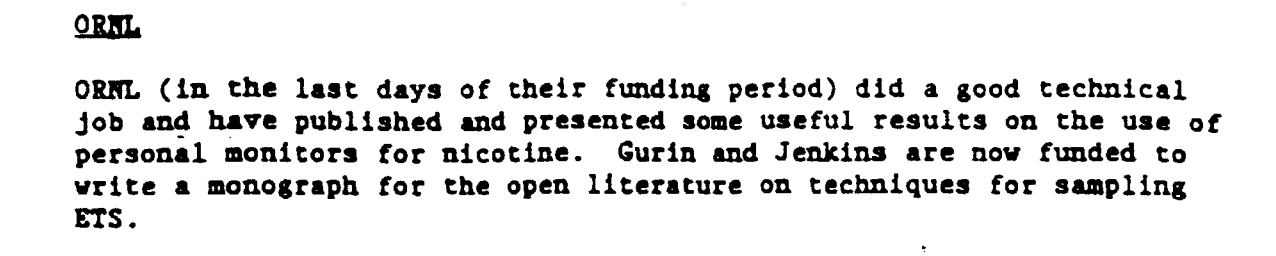


Figure 2: Sample image fragment from the tobacco litigation corpus

designed for use with very noisy documents in fixed-width fonts of a single size per image. Very old typewritten pages offer a prime example of this type of document.

QUARC is based on the insight that different restoration methods are appropriate for cleaning images with different characteristics. It defines 6 image quality measure factors, whose possible values fall in the range $[0, 1]$. These measures reflect characteristics such as the quantity of large speckle, marks that appear to be groups of touching characters, etc. It also includes a set of different image restoration algorithms. QUARC trains on a set of images and corresponding ground truth text files. It finds the characteristics of each image and the best restoration method for each. From this data, it learns the best restoration methods for different combinations of quality measures. In its application phase, it finds the quality measures for an image and then selects the appropriate restoration method and applies it.

We applied QUARC to our set of declassified and released government documents. Because QUARC requires a large training set that is representative of the test set, we performed 5-fold cross-validation. That is, we divided the corpus into 5 approximately equal subsets, and we trained and tested QUARC 5 times. Each time, we held out a different subset for testing, and we trained on the remaining 4 subsets.

4 Evaluation

The measures of precision and recall at the character level, as defined in the Introduction, provide a means of evaluating the systems' output. That is, suppose there are n characters in the ground truth for a file, and suppose the OCR process finds m characters. Suppose also that of these m characters, p are correct. Then $Precision = \frac{p}{m}$ and $Recall = \frac{p}{n}$. Since the ground truth for our corpora does not indicate the *location* of each character, finding the value of p requires determining how to align characters in the OCR results with characters in the ground truth.

The University of Washington provides an evaluation program [4] that produces a confusion matrix, which we used to find the correct characters for that corpus. This program uses a line-based string comparison approach to aligning the characters. It is designed for data that is represented in the form its corpus uses, with zone delimiting and with LaTeX-like representations for symbols. The program allows the specification of the costs of character insertion, deletion, and change. We set all non-zero costs to 1.[2]

The ground truth for the other corpora is represented as plain text, without markup, and the Russian documents use Unicode rather than ASCII. As a result, we used our own program to evaluate these corpora. It also uses a line-based string comparison to align the characters. Specifically, it finds the string edit distance between the sequences of lines in the ground truth file and the OCR result file. For this purpose, the cost of inserting a line of characters is the cost of inserting all characters in the line, the cost of deleting a line of characters is the cost of deleting all characters in the line, and the cost of changing a line is the character-based edit distance between the source line and the target line. This approach is the same as one of the possibilities for text-based duplicate detection discussed in [5]. The line edit sequence provides an alignment, and the character edit sequence indicates which characters in the OCR results are matches to characters in the ground truth.

5 Results

The results over a collection of documents can yield two distinct averages. The *document average* values are the average document precision and the average document recall. We calculate them by calculating

[2]We set the cost of changes between certain pairs of indistinguishable symbols to 0, and for certain symbols that might legitimately be considered "graphics," we set the cost of deletion to 0.

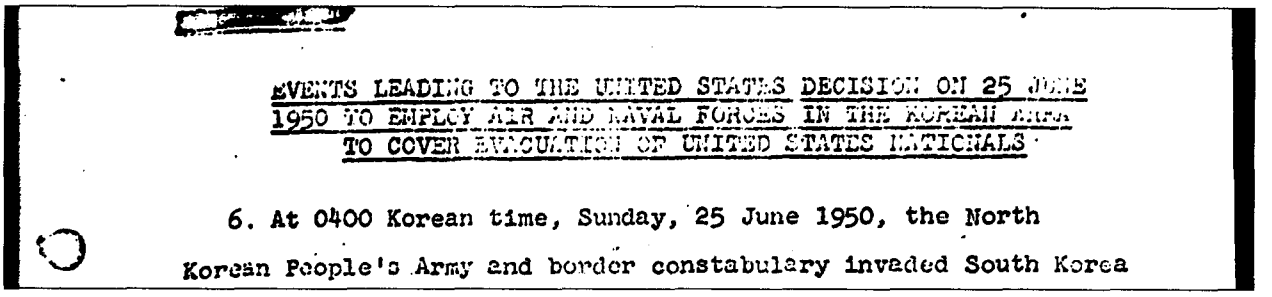


Figure 3: Sample image fragment from the declassified and released government document corpus

precision and recall for each document and taking the average. The *overall average* values are the precision and recall values that result from treating the entire collection as a single unit. That is, these values result from calculations that use the total number of matched characters in the collection, the total number of characters in the OCR results for the entire collection, and the total number of ground truth characters in the entire collection.

Table 1 presents the document average results of the three OCR packages on the English corpora. Table 2 presents the overall results of the three packages on the English corpora. CDK 2000 and FineReader 4.0 consistently outperformed NeuroTalker, and in general CDK 2000 slightly outperformed FineReader 4.0. In the case of the declassified government documents, however, FineReader 4.0 exhibited somewhat superior recall to that of CDK 2000.

Table 3 presents the document average results for the Russian documents, categorized by the type of noise introduced. FineReader outperformed CDK 2000 in almost every case. CDK 2000 yielded better results on skewed documents, however. Neither package produced useful results on the documents that were photocopied through colored plastic filters. The precision of CDK 2000 on the colored filter documents without scanning correction is inflated by documents in which no text was found and the precision was therefore 100%.

Table 4 presents the document and overall average results for the declassified and released government documents, using CDK 2000 with QUARC preprocessing and without QUARC preprocessing. Preprocessing with QUARC yielded only a tiny improvement, which was not statistically significant, in the average precision and recall. It did, however, substantially improve performance on *some* documents. It also degraded performance on several documents. For example, QUARC increased precision by more than 10% on 77 documents and decreased precision by more than 10% on 38 documents. However, it increased precision by more than 50% on 8 documents and decreased precision by more than 50% on 15 documents. Its effect on recall was similar.

Identifying common characteristics of the images that QUARC degrades could probably lead to improvements that would allow QUARC's overall effect to reflect the value it holds for many documents. For instance, additional quality measures may be necessary in order to capture accurately the characteristics of a collection as diverse as this set.

In summary, CDK 2000 and FineReader 4.0 consistently outperformed NeuroTalker 4.1 on English documents, and CDK 2000 usually outperformed FineReader 4.0, with some exceptions. FineReader 4.0 consistently outperformed CDK 2000 on Russian documents. QUARC seems highly promising but is not yet consistently helpful when applied to a diverse corpus.

References

[1] Michael Cannon, Judith Hochberg, and Patrick Kelly. QUARC: A remarkably effective method for increasing the OCR accuracy of degraded typewritten documents. In *Proceedings: 1999 Symposium on Document Image Understanding Technology*, pages 154–158, Annapolis, Maryland, April 1999.

[2] Su Chen, M. Y. Jaisimha, Jaekyu Ha, Robert M. Haralick, and Ihsin T. Phillips. Reference manual for UW English document image database I: Version 1.2. Available on CD from the University of Washington, August 1993.

[3] Committee on commerce tobacco documents. `http://www.house.gov/commerce/TobaccoDocs/documents.html`.

[4] Su Chen. OCR performance evaluation software user's manual: Version 2.0. Available on CD from the University of Washington, September 1993.

[5] Daniel P. Lopresti. String techniques for duplicate detection. In *Proceedings: 1999 Symposium on Document Image Understanding Technology*, pages 101–112, Annapolis, Maryland, April 1999.

Corpus	CDK 2000		FineReader 4.0		NeuroTalker 4.1	
	Precision	Recall	Precision	Recall	Precision	Recall
Tobacco	91.65%	91.73%	80.98%	87.61%	80.80%	80.51%
U. Washington	98.53%	74.75%	97.73%	74.66%	94.85%	72.81%
Declassified	76.99%	75.58%	74.86%	80.64%	71.54%	61.94%
English Total	89.38%	78.53%	86.08%	79.47%	78.60%	70.49%

Table 1: English Document Average Results

Corpus	CDK 2000		FineReader 4.0		NeuroTalker 4.1	
	Precision	Recall	Precision	Recall	Precision	Recall
Tobacco	87.79%	91.63%	79.87%	88.74%	79.94%	80.09%
U. Washington	98.72%	77.43%	97.80%	77.19%	95.28%	75.51%
Declassified	81.40%	78.07%	76.31%	84.03%	68.32%	68.55%
English Total	92.38%	79.18%	88.78%	80.20%	85.50%	74.28%

Table 2: English Overall Average Results

Image Type	CDK 2000		FineReader 4.0	
	Precision	Recall	Precision	Recall
Clean (300dpi)	95.71%	95.90%	99.50%	99.37%
200 dpi	87.05%	88.16%	98.21%	97.24%
100 dpi	58.68%	55.04%	84.87%	84.87%
Fax, 1 generation	79.79%	80.79%	94.59%	94.17%
Fax, 2 generations	63.35%	65.25%	86.93%	87.03%
Reduced to 1/2, re-enlarged	89.96%	89.86%	96.96%	96.33%
Reduced to 1/3, re-enlarged	75.61%	74.61%	92.41%	92.06%
Reduced to 1/4, re-enlarged	48.45%	45.32%	79.10%	78.32%
Clear Plastic, 4 layers	84.43%	84.06%	92.64%	92.21%
Clear Plastic, 5 layers	60.36%	58.57%	72.71%	72.07%
Black Edges	95.34%	95.53%	96.01%	98.47%
Speckle	78.41%	90.38%	81.34%	96.06%
Skew	78.48%	79.06%	36.03%	35.69%
Colored filters, scanning correction	2.78%	3.73%	6.02%	7.92%
Colored filters, no correction	23.08%	0.00%	1.24%	0.21%
All	72.04%	72.58%	79.38%	80.05%

Table 3: Russian Document Average Results

	CDK 2000 (no QUARC)	CDK 2000 + QUARC
Doc. Avg. Precision	76.99%	77.20%
Doc. Avg. Recall	75.58%	76.43%
Overall Precision	81.40%	81.53%
Overall Recall	78.07%	78.87%

Table 4: Declassified Document Average Results: CDK 2000 With and Without QUARC

ДЕЙСТВУЮЩИЕ ЛИЦА
Серебряков Александр Владимирович, отставной профессор.
Елена Андреевна, его жена, 27 лет.
Софья Александровна (Соня), его дочь от первого брака.
Войницкая Мария Васильевна, вдова тайного советника, мать первой жены профессора.
Войницкий Иван Петрович, ее сын.

Figure 4: Sample fragment of clean Russian image, scanned at 300 dpi

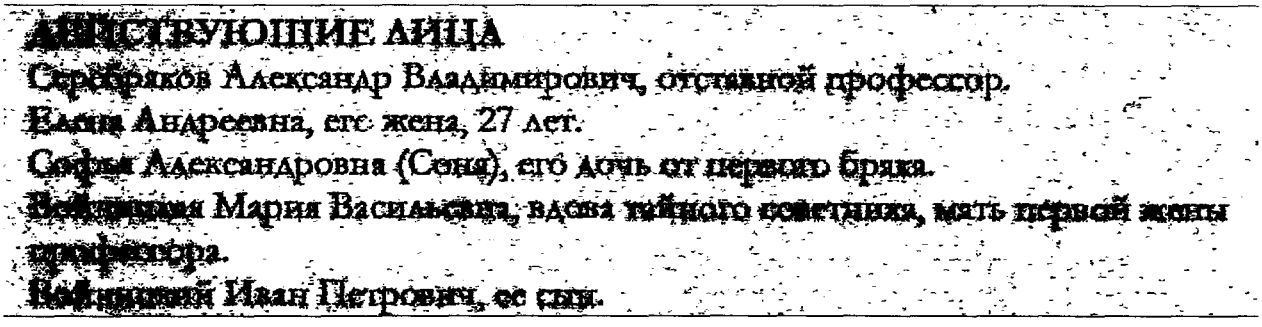


Figure 5: Sample fragment of Russian image photocopied through colored filters, scanned with corrections

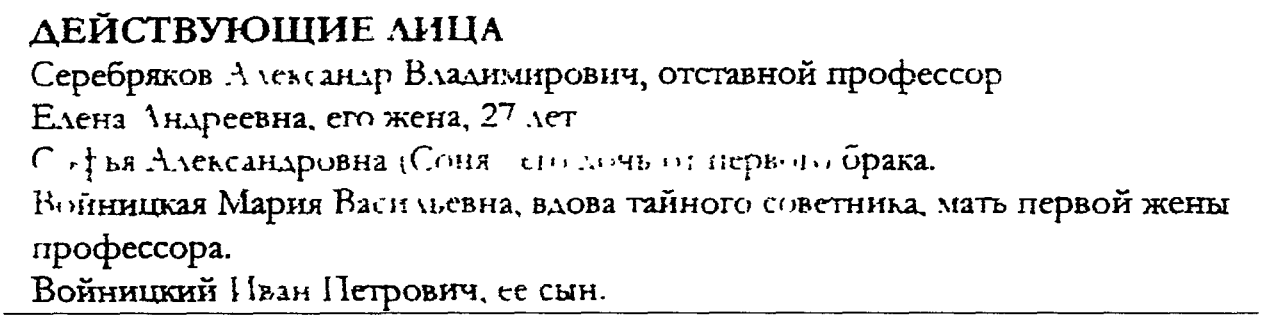


Figure 6: Sample fragment of Russian image photocopied through 4 layers of plastic

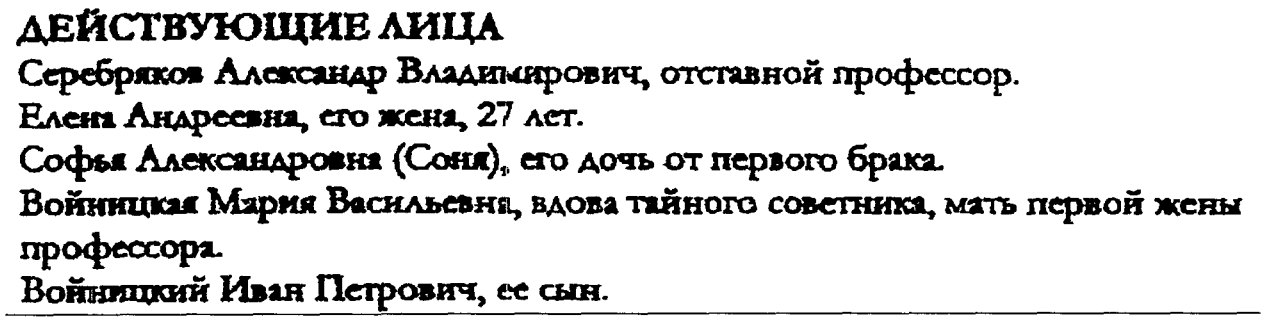


Figure 7: Sample fragment of Russian image reduced to 33% and re-enlarged to original size

Performance Evaluation

Truthing, Testing and Evaluation Issues in Complex Systems*

Srirangaraj Setlur Venu Govindaraju

Sargur Srihari

Center of Excellence for Document Analysis and Recognition, Amherst NY , USA

email:{setlur, govind, srihari}@cedar.buffalo.edu

Alfred Lawson

United States Postal Service, Merrifield VA, USA

email:alawson@email.usps.gov

Abstract

This paper describes the issues involved in the design of a system for evaluating improvements in the performance of a real-time address recognition system being used by the United States Postal Service for processing mail-piece images.

Evaluation of the performance of recognition systems is normally carried out by measuring the performance of the system on a representative sample of images. Designing a comprehensive and valid testing scenario is a complex task that requires careful attention.

Sampling live mail-stream to generate a deck of images representative of the general mail-stream for testing, truthing (generating reference data on a significant number of images), grading and evaluation, and designing tools to facilitate these functions are important topics that need to be addressed. This pa-

*This work was supported by contracts from the USPS

per describes the efforts of the United States Postal Service and CEDAR towards developing an infrastructure for sampling, truthing and testing of mail-stream images.

1 Introduction

The United States Postal Service has invested significantly towards automated processing of mail-pieces to speed up sorting and reduce labor costs.

The letter mail automation program of the United States Postal Service (USPS) utilizes recognition software developed by a number of vendors. A brief introduction to the scale and nature of the Postal recognition systems program is essential to appreciate the issues involved in the development of a comprehensive system for evaluating the performance of these real-time large scale automation systems.

2 Postal Automation

Postal automation represents a fertile area for the application of image processing and pattern recognition techniques. Mail handling is a labor intensive process and the prohibitive cost of manual labor has made automatic sorting of mail an attractive economic proposition. The cost of processing per 1000 mail-pieces drops from $47.78 for manual processing to $27.46 for mechanized processing to $5.30 for automated processing. The savings multiply rapidly given the volume of mail processed by the United States Postal Service (About 400 million pieces of letter mail per day). In addition to the cost factor, the knowledge level required for the sorting process is considerable and must be acquired.

The postal automation program focuses on three strategies: generating bar-coded mail, processing bar-coded mail in automated operations and adjusting the work force resulting in a reduction of work hours.

Bar-codes are generated by customers or by the Postal Service. The letter mail is first processed through the MLOCRs or the MultiLine Optical Character Readers which attempt to recognize addresses on mail-pieces (machine-printed) in real-time. The MLOCRs are augmented by a second recognition system called the Recognition Co-processor. The mail-

pieces that cannot be read by the MLOCRs are captured and processed through the Remote Bar Coding System or RBCS.

The RBCS processing (Fig 1) is divided into two general categories: RCR (Remote Computer Reader) processing, which involves automatic reading and interpretation of the address information, and REC (Remote Encoding Center) keying, which requires human interpretation.

Handwritten as well as machine printed mail-pieces are processed through the RCR. CEDAR's Handwritten Address Interpretation Software (HWAI) is used for processing handwritten mail in the RCR.[1]

When all electronic means of resolving address information have been exhausted, the mail-piece image is sent to REC sites where operators use video display terminals and keyboards to manually enter the address information. This keyed information is fed back to the RBCS to allow bar-codes to be applied on to the mail-piece. Providing partial RCR results to the REC sites with the image can allow the operator to process the address with minimal number of keystrokes.

The bar-code encapsulates the address that is present on the mail-piece. The objective of reading the address on the mail-piece is to generate a DPC or a delivery point code which can be applied as a bar-code on the mail-piece and bar code sorters can then use this bar-code to sort the mail according to the carrier walk sequence which is the order in which the mail-carriers do their rounds.

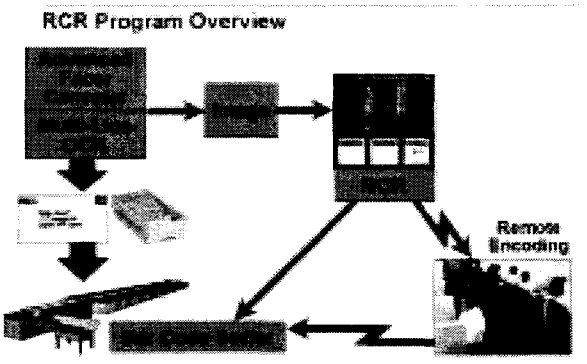


Figure 1: USPS Mail processing flow diagram

Automation is revolutionizing mail processing operations as the Postal Service continues to invest in technology infrastructure to reduce costs, improve service and increase operating efficiencies. Upgrades to the Remote Computer Readers (RCRs) have resulted in significant savings to the Postal Service, in-

cluding a reduction of 12 million work hours annually and cumulative labor savings of more than $208 million. [2]

3 USPS Databases & Addressing Standards

This section provides an insight into the types of postal data available, the wide variety of addresses possible and the kind of rules required to resolve the address on the mail-piece.

The Postal Service maintains different databases of postal addresses, two of which are the ZIP+4 file and the DPF (Delivery Point File). The ZIP+4 database has records representing a range of primary numbers (street numbers or PO box numbers as the case may be) whereas the DPF or the Delivery Point File has records representing every individual delivery point in the United States. Supplemental files provide information about ZIPtype, City-State-ZIP correspondence and ZIP translation.

The Postal Service and the bulk mailing industry have jointly developed postal addressing standards that are geared towards enhancing the processing and delivery of mail and reducing undeliverable mail thereby providing mutual cost reduction opportunities through improved efficiency. The standards are fairly well adhered to especially in the machine printed mail-stream. Handwritten mail finds greater deviations from the standards.

The last line or the last two lines of a mail-piece with a US domestic delivery address contain City, State and ZIP code information. The line above contains delivery information such as a street number, street name with a street suffix and secondary information such as apartments, suites, etc. The mail-piece could also have a firm or organization name or a personal name or PO box information.

The Postal Service has developed an encoding scheme that allows for the sorting of mail to the block level, building level or delivery point level using different record types.

3.1 Encoding a mail-piece

Using the databases and the supplemental files and a complex set of encoding rules the destination address

is encoded.

e.g. in the address :

CEDAR, UB Commons

520 Lee Entrance Ste 202

Amherst NY 14228

the ZIP 14228 is an automated ZIP and is not UNIQUE and hence the desired level of encoding is 11 digits.

The records in the DPF postal database corresponding to the ZIP 14228 and street number 520 are shown in Table 1. It can be seen from the table that the addon corresponding to the address 520 Lee Entrance Ste 202 is 2583. A set of rules helps determine the DPC corresponding to this high-rise specific addon and for Ste 202 the corresponding DPC code is 52. The finest level of encode for the given address is 14228-2583-52 and this 11 digit code determines a unique delivery point in the United States. The coarser encodes, viz. the street encode of 14228-2500-20 and the high-rise default encode of 14228-2567-99 are also valid but do not represent the finest depth encode for the mail-piece. The finest depth encode allows sortation to the finest depth possible which results in the maximum savings to the Postal Service in terms of processing costs. The coarser depths do not allow sortation to the finest extent possible and hence require some manual sorting which adds to processing costs.

4 Evaluation - Guidelines

The evaluation model for an address interpretation system has to also take into account some problems that are unique to the postal mail-stream scenario.

1. Postal addresses are not *static* for the following reasons:
   - New addresses are added to the database.
   - ZIP codes get translated into new ZIP codes and addresses move between ZIP codes.
   - PO Box numbers get cancelled or reassigned.
   - New firms are added to the database.

   The postal databases are a key component of the system encoding as well as the truthing processes. If the truthing is done using a database

Table 1: Sample records from the DPF corresponding to the ZIP 14228 and Street number 520

Primary No	Rec Type	St Name	St Suffix	Sec Des	Sec No	Sec/Firm Name	Addon	Carrier Route
520	10	N ELLICOTT CREEK	RD				2323	C059
520	10	LEE ENTRANCE					2500	C050
520	10	LEE ENTRANCE		STE	200	UB COMMONS	2500	C050
520	20	LEE ENTRANCE					2567	C050
520	20	LEE ENTRANCE		STE	101	UB COMMONS	2577	C050
520	20	LEE ENTRANCE		STE	103	UB COMMONS	2577	C050
520	20	LEE ENTRANCE		STE	202	UB COMMONS	2583	C050
520	20	LEE ENTRANCE		STE	210	UB COMMONS	2583	C050
520	20	LEE ENTRANCE		STE	106	UB COMMONS	2584	C050

from a time different from the database that is used by the system being tested, the testing process would be flawed since the truth for the same mail-piece given one temporal instantiation of the database might not be the truth given a different temporal instantiation of the database. Hence, it is important to ensure that the database used by the system during the test is the same temporally as the database used for truthing.

2. Patrons sometimes write ambiguous, incomplete or erroneous addresses on mail-pieces. Comprehensive encoding rules are required to ensure that the resultant encode for a given address can be uniquely determined and the test deck has to be truthed in accordance with these rules so that the system can incorporate the same rules in its encoding scheme and the evaluation will not suffer as a result of the inherent ambiguities in patron addressing.

3. Handwritten addresses present another source of confusion. Subjective interpretation of ambiguous digits and characters present problems in determining the truth unambiguously.

A complex set of scoring and evaluation rules has been designed by the USPS for scoring and evaluating mail pieces processed by any letter mail system. Each rule is designed to achieve the best potential possible for a mail piece. The finest depth of code is determined by using the available mail piece information to get the best and most cost effective ZIP Code resolution. This is performed by using the Expert Zip+4 Finder (EZF) or other directory query

programs.

5 Testing Scenario

Address recognition systems have been installed at over 250 mail processing centers across the country. Turnaround mail or mail destined to the sender's geographical area forms a substantial percentage of mail at each of these processing centers. Especially with handwritten mail, addressing characteristics could differ widely based on geographical regions and ethnic make-up of the populace.

Given the wide variety of handwritten mail, it would be difficult to model the variations and create synthetic data that could be used to effectively evaluate these recognition systems. Therefore, there is no alternative but to generate a representative test deck using real data.

It is vital for the test deck to reflect the national mail-stream in order to ensure that the performance of the address interpretation systems on the test deck can be expected to be reflected over most of the post offices where the software is deployed.

Figure 2 describes the testing and evaluation system. Images from the test deck are transferred to the truthing stations. The truthers create an ASCII representation of the address on the mail-piece and also record some characteristics of the mail-piece and the address block. The ASCII truth is then processed through an ASCII address matching engine that uses the rules of encoding determined by the Postal Service to interpret the ASCII address and generates the appropriate encodes at the various levels of sortation.

Patron errors on mail-pieces, incomplete, incorrect, and ambiguous addresses can cause the address matching engine to fail. If the address matching engine is unable to encode the mail-piece beyond 5 digits (for non-Unique ZIPs), the image is forwarded to a second truthing station where the truther uses additional queries to various databases to try to resolve the mail-piece. The truth data is stored in a database for use during the evaluation phase.

The images from the test deck are also processed through the address interpretation engines provided by the vendors. The results from these runs are stored in a results database.

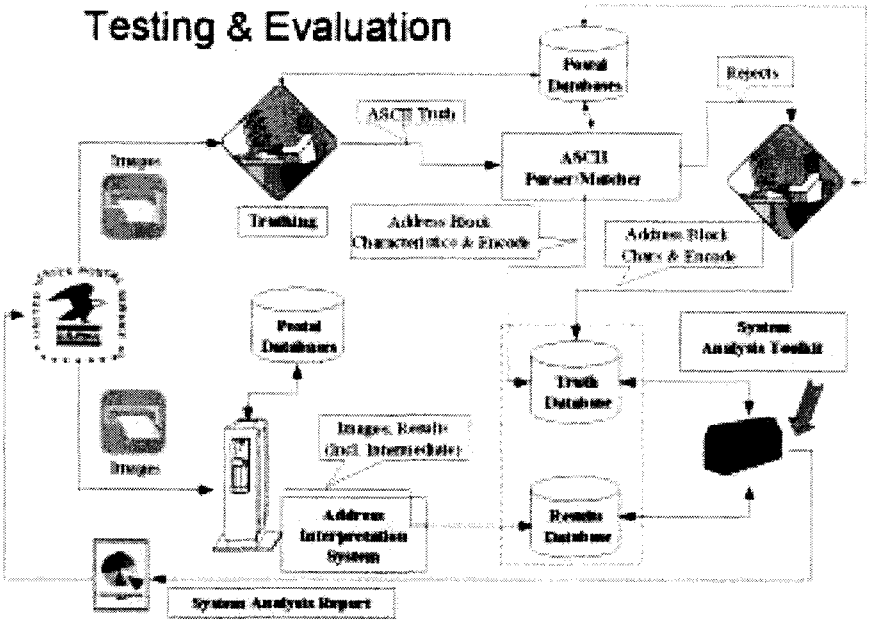


Figure 2: Testing and Evaluation System

Grading utilities which are part of the System Analysis Toolkit are then used to match the vendor results against the truth and a detailed performance analysis report is generated.

6 Test Deck Generation

A fundamental goal of the test deck generation process is to ensure that the images collected for the test deck are representative of the image population observed in actual field operation.

The evaluation process measures the performance of an address interpretation system on this test deck of images.

Two key metrics used in the performance evaluation of an address interpretation system are the encode rate of the system and its error rate.

To evaluate the error rate, it is necessary to have a deck which is "truthed" so that the accuracy of the encode can be checked. However, truthing is an extremely expensive process. So, a further down-sampling of the test deck is done to generate a smaller

deck that is truthed.

Keeping in mind the twin objectives of measuring the encode rate and the error rate, a test deck of a million handwritten images was generated, consisting of contemporary images from a minimum of 120 of the 251 Remote Computer Reader sites distributed all over the United States. Since the mail-stream consists of a mix of handwritten and machine-print images and one of the goals of the system is to separate the handwritten images from the machine-print images and process them appropriately, a machine print test deck of at least 250,000 pieces from the same 120 sites was also dispersed through the test deck prior to testing.

The handwritten image test deck consists of a collection of two sets of 500,000 images each. The first set of 500,000 images is made up of 25,000 samples collected from each of 20 selected sites. 50,000 of these images selected randomly were truthed. The second image set consists of 500,000 images from 100 different field locations. 50,000 of these images selected randomly were also truthed.

A single machine print test deck with at least 250,000 images collected from the same 120 sites as the handwritten test deck including 100,000 truthed images dispersed throughout the deck will be used to evaluate machine print encoding performance.

7 Truthing

The error rate of an address interpretation system is a very crucial metric in the performance evaluation of the system. Incorrect encodes would not only increase the cost of the processing necessary to deliver the mail-piece but would also tarnish the reputation of the postal service. Hence, it is essential that the error rate of the system be measured accurately and a very low threshold of error rate imposed on the system.

The most effective way of measuring the error rate is to compare the encodes returned by the system against the encode determined by a human looking at the mail-piece with access to the relevant postal databases. This process of recording the correct encode corresponding to a mail-piece image by humans is termed *truthing.*

Due to the complex nature of the rules that govern the encoding of an address it would be difficult for the truthers to memorize all the rules for encoding. Therefore, it is important that sufficient care be taken in the design of the tools to eliminate the need for memorization of these rules as far as possible and also to provide visual cues and design ergonomic interfaces to minimize the possibility of error. CEDAR has developed an integrated Image Evaluation System that allows entire sets of images to be truthed in a distributed environment. A good training program and an efficient monitoring system have helped maintain truthing quality.

Truthing is by nature a labor-intensive and expensive process. Hence, it is important to come up with as small a sub-set of images as possible that is representative of the population of the entire mailstream. The address interpretation system has to process an average of a 1,000,000 pieces of mail per day at over 250 mail processing centers around the country. Hence, the smallest number of images that must be sampled for truthing is still quite large.

8 Summary

A truthing and evaluation scenario for a large scale image processing system has been presented. It is clear that the nature of data being processed and how well the test set can mirror the characteristics of the real data population determine whether one should use a synthetic data set or representative samples of real data. Manual truthing of real data is by nature a monotonous and error prone activity. Design of ergonomic truthing tools with visual cues and redundant data integrity checks should help reduce truthing errors.

References

[1] V. Govindaraju, A. Shekhawat and S.N. Srihari, "Interpretation of Handwritten Addresses in US Mail Stream",*Proceedings of the Third IWFHR*, pp 197-206, 1993.

[2] *USPS Annual Report* - United States Postal Service, 1999.

[3] *USPS RCR2000 Scoring Rules* - United States Postal Service, 2000.

What System Developers Need to Select OCR for Authentic Tasks: Evaluating End-to-End Systems

V. Melissa Holland **Chris Schlesiger**
Luis Hernandez
U.S. Army Research Lab

Abstract

Evaluation research has focused on methods of assessing individual processes such as OCR and machine translation (MT), which may become components of end-to-end systems. Little attention has been given to evaluating these processes working together in end-to-end fashion. Moreover, little is known about how performance measures – whether of components or of systems – relate to effectiveness of a system for human tasks. Yet, knowledge about these two issues is needed to inform system developers about selecting components and testing systems with users. This paper suggests how current studies might be expanded to address these issues and establish more authenticity in evaluation research. We illustrate with examples from systems for multilingual processing that include machine translation and name finding.

1 Introduction: End-to-End Systems for Language Processing

The Army Research Lab (ARL) develops applications to demonstrate and test the utility of tools to process natural language. One application, Falcon (Forward Area Language Converter), is an end-to-end system that links OCR of paper input in various languages with machine translation (MT) of those languages into English [6, 8]. In integrating software for applications like Falcon, we are concerned not just with having the pieces function efficiently together but also with predicting how accurately the integrated system will perform. Indeed, we would like to have reliable predictions about system performance before we place a system with users to collect data on utility and effectiveness.

2 The Need: Performance Predictions for End-to-End Systems

Predictions about performance of an end-to-end system based on its components will help us:

- select among different competing components that are candidates for integration in the system, without having to evaluate all combinations of available OCR and MT products;
- decide whether a system performs well enough to enter subsequent tests of utility and effectiveness.

Note that we differentiate measures of performance and measures of effectiveness. A traditional distinction in system evaluation, performance relates to system-specific measures while effectiveness relates to how well the system serves a task.

2.1 Prototypes for Bosnia: What we Expected

What happens when we do not have access to predictions about performance? We sent five Falcon prototypes to Bosnia in 1997 as a quick reaction to needs voiced urgently by the field. The aim was to enable soldiers who did not speak Serbian and Croatian to screen documents in those languages. According to Taylor and White's [14] task hierarchy for MT, screening means filtering out documents that are not relevant to a broadly defined topic (military, computers, etc.) so that linguists can deal in detail with the remaining documents. Screening is, by this definition, the least demanding of tasks that MT might serve [3]. Soldiers who screen documents told us they rely on the presence of domain-specific words and phrases, known as *key words*, to decide about relevance (a document with words like *commander*, *troops*, *artillery*, and *flamethrower* would be judged as army-relevant).

Before we sent Falcon to Bosnia, we tried its components in the lab. We judged that these components, which consisted of then pre-commercial software for OCR and MT in Serbian and Croatian, each functioned well enough in isolation – although neither was yet mature – to serve the purpose of document screening. Here is how we made that judgment.

Lacking time to conduct a formal OCR evaluation, we counted character error rates per page on a small sample of Serbian and Croatian documents. We felt that these error rates, which were <5% per page, would yield enough output text to provide a sufficiency of key words for translation and subsequent screening. For the

MT component, we relied on the developer's measure of MT maturity, estimated at 40%, or "pre-prototype quality" – an internal company metric based on comparing words in machine-translated test sentences with words in a human translation. Because the Serbian and Croatian MT were built from military and technical dictionaries rather than from common vocabularies, and included terms we provided from army-relevant domains, we felt that a 40% MT system could translate enough key words to support screening.

2.2 What the Bosnia Users Said

A small group of soldiers tried the prototype in Bosnia. They rejected the end-to-end system as "not permitting a good analysis of the document...or even whether it may be of value for translation" (memo on Falcon trial results, 5/97). At the same time, soldiers confirmed our expectation about the MT component alone, noting that when they corrected OCR errors by hand, "...Falcon translated about 80% of the text," or sufficient for screening, in their judgment. Users supplied samples of real text to illustrate their judgments.

Here is Falcon's English translation *with OCR errors* of a 1997 Serbian newspaper article, judged *not good enough* for screening (words preceded by >> are source-language words not found in the MT lexicon):

> 3 >>Njn >>chlrstog the representation Serbian >>to nl~~io~~al~~ih of interest, .>>mr Momchilo Krajis^iik, president from >>To re~~u6like >>Srpsks in The presidency >>To biH, from >>to med~u~~arod~~ih of negotiator, >>pa and from the authority of Belgrade, got >>nadimlk >>@Mmsg >>No@;
>
> *[Human translation: Mr. Moncilo Kravicinik, President of the Serb Republic, in the joint Bosnia Herzogovina (BH) Presidency, has received the nickname "Mr. No" from international negotiators and even from Belgrade authorities because of his strong advocacy of Serb national interests.]*

Here is Falcon's English translation *with OCR correction* of a 1997 Croatian magazine article, judged *good enough* for screening:

> The Armed Forces Of the republic Serbian in >>ponedjeljak celebrated The day army in Prijedor, where is kept military parade with 2,500 soldiers >>VRS, for that >>Sfor issued >>odobrenje. >>@lako all soldiers will carry weapon, ammunition is not permitted, as not heavy armament.
>
> *[Human translation: SFOR approved an Army parade by the Serb Republic forces, on Monday in Prijedor. 2,500 soldiers will take part and will carry weapons, even though they won't be allowed to carry ammunition and no heavy weapons will be in the parade.]*

Users also said that manual OCR correction was too time-consuming to be an option for day-to-day operations.

2.3 Summary: The Need for End-to-End Models

It is clear that the unexpectedly high proportion of poor-quality documents encountered in Bosnia [8] contributed to the observed high OCR error and to the failure of our prediction about end-to-end Falcon. Yet it is also true that we lacked a good understanding of the problem of evaluating end-to-end systems. We needed a model of end-to-end performance that takes into account the impact of OCR errors on MT (or on other text processes).

A further need for end-to-end performance models appears in selecting among competing components. All things equal, we choose the component with superior performance results. As system developers, however, we must consider other factors pertinent to development and fielding:

- ease of integration (Is there an SDK?)
- simplicity of implementation (Are there hardware encumbrances such as dongles or tablets?)
- robustness and stability (Does the software crash?)
- compatibility with system software and platform
- speed of processing
- cost

Big differences between components on one of these criteria may, depending on the application, outweigh small differences in component performance. Without predictions about end-to-end performance, we lack a solid basis for balancing these factors.

3 The Problem: Evaluating End-to-End Systems with OCR

We can consider an end-to-end system like Falcon to have *core processes* and *feeder processes*:

Feeder process (OCR) → Core process (MT).

Feeder processes are defined in terms of the modality of the data that must be transformed for input to core processes, such as OCR and speech recognition. Core processes are defined in terms of the task the user wants to do with text, such as machine translation (MT), information retrieval (IR), information extraction (IE), or name finding (NF, which means finding the proper nouns). Feeder processes and core processes are independently subject to error; in addition, errors in feeder processes ramify into core processes.

Applicable to Falcon, Voss [16, 17] has called MT within an end-to-end system "embedded MT" and has identified a research gap around evaluation of embedded MT, where each component independently produces error and where errors also interact between

components.

3.1 State of Evaluation Research

Much recent effort has gone into developing methods for evaluating various core processes and feeder processes. Measures are well established for assessing OCR performance, in terms of character and word error rates [1], and speech recognition performance, in terms of word error rates [11]. Measures of precision and recall have become standard for assessing IR, IE, and NF [2, 7]. MT evaluation is less mature, and its methods still evolving, although a range of measures have been piloted and applied [13, 14, 15, 17, 18].

3.2 Limitations of Evaluation Research

The research on evaluating core and feeder processes has two major limitations for decision-making about end-to-end systems. First, clean (error-free) symbolic text is assumed as input to core processes. Indeed, even clean symbolic text may be preprocessed to standardize it prior to an evaluation. For example, an evaluation of Arabic MT [5] normalized the Arabic texts for spelling and format before submitting them to MT. A rationale for this practice is seen in a study of MT on non-preprocessed electronic text [12], which found that 45% of tokens whose translation failed were due to factors other than a word's not being in the lexicon, such as code page mismatches. The sponsors of core process evaluations are often concerned with huge amounts of text that do not require OCR. They want pure measures of the end process unencumbered by extraneous error. By contrast, an army context confronts system developers with the "forward area" – where troops are on the ground, the input is most often paper, and the images are usually noisy. This context motivates our attention to the effects of errorful input data.

A second, related limitation, lab evaluations tend to focus on the performance of independent (feeder or core) rather than linked processes (feeder + core), thus neglecting the effects of an input process like OCR. These two limitations, however well reasoned, serve to abstract evaluation away from authentic tasks and tend to blur implications for selecting OCR and judging whether the total system is ready for users to test.

4 Overcoming the Limitations: Toward End-to-End Evaluation Studies

Three studies epitomize research on end-to-end system performance that can inform the decisions of system developers. These three studies look at the effects of OCR on core process performance for, respectively, IR [4], NF [9], and MT [17]. The IR study helps establish levels of OCR performance that predict IR performance. The NF and MT studies are more recent and incipient. Both studies deal with foreign language documents so are of particular relevance to a system like Falcon. Consider the contributions of each study and how each might be further developed.

4.1 Evaluating OCR → NF

Kanungo and Bulbul's NF study [9] employed a set of Arabic documents and used established recall and precision rates to measure the effect of OCR error on the performance of an automatic Arabic name tagger (for names of people, places, organizations, etc.). An Arabic OCR engine selected for its relative effectiveness [10] yielded a character accuracy rate of 92% and a word accuracy rate of 65% on the target document set. Compared to ground-truth text, the OCR text led to reductions in Arabic NF performance from 31.0% to 18.2% for recall and from 40.6% to 24.6 % for precision (on one of three recall/precision measures from this study).

4.2 Evaluating OCR → MT

Voss and Van Ess-Dykema's MT study [17] piloted a set of performance evaluation measures for MT and used these measures to trace the effects of OCR errors on MT. The measures were piloted using the Falcon OCR + MT process for a single document in each of three languages (Arabic, Spanish, Haitian-Creole), with English as target language. Four MT performance measures were proposed, based on ratios that considered the following counts: total English words generated by MT, English words that carried content, English words that were "semantically adequate" in their translation (an index of translation accuracy), and English words that were "domain-defining" (an index of relevance for a screening task).

As in the NF study, OCR errors were found to affect each measure, reducing MT performance compared with ground-truth (non-OCR) text. Because content-carrying words have linguistic status as "open class," and may be identified by their exclusion from the small, well-defined set of "closed class" words in a language, the content-carrying measure can potentially be automated. Counts of semantically adequate and of domain-defining words require human judgments at this point.

4.3 Expanding End-to-End Studies

These two studies stimulate further questions that are necessary to our decisions in developing systems:

(1) What function relates level of OCR performance to level of core process performance (NF, MT)?

(2) How does this function vary with language (e.g., Arabic vs. French)?

(3) How does system performance relate to task effectiveness for the job the user needs to do?

4.3.1 Relationship Functions

Question (1) calls for studies that go beyond

examining a single level of feeder process performance. Such studies might manipulate document degradation to achieve varying levels of OCR performance, then map the influence of OCR level on core process performance.

4.3.2 Language Variation

Question (2) calls for additional cross-language studies, bolstered by a theoretical understanding of these aspects:

(a) linguistic features and their consequences for image and text processing;

(b) the nature of the engine that performs the core process.

For aspect (a) we ask what ways of classifying languages matter in predictions about end-to-end processing. It is plausible, for example, that degree of affixation and associated word length influence the extent to which OCR noise affects subsequent processing. In *agglutinative languages* (e.g., Swahili), grammatical information is conveyed by attaching affixes to roots and stems (such as prepositions to nouns, or conjunctions to verbs). Similarly, in *inflectional languages* (e.g., Korean, Arabic), many grammatical functions are conveyed by inflection and affixes to verbs and nouns. These two classes of language tend to have longer words, on average, than *isolating languages* (e.g., English, Chinese), where grammatical functions are conveyed by word order and by particles and prepositions that appear as separate words. We might hypothesize that a character or word error in OCR is more damaging to translation in agglutinative and inflectional languages than in isolating languages, since that error is likely to affect more meaning components.

As another example, it is plausible that the manner of signaling proper names in a language mediates the influence of OCR noise. Because scriptal languages (Arabic, Farsi) have no capitalization conventions, which mark proper nouns in latinic fonts, automatic name finders in Arabic must consider context, such as person titles and verbs that require human subjects. Thus, an OCR error damaging for NF could occur in surrounding words and not merely in (or even necessarily in) the naming word. In general, we might hypothesize that NF is more vulnerable to OCR error in scriptal than in latinic languages.

For aspect (b), we ask what classes of MT or other core processes make a difference in end-to-end predictions. Although we want predictions to reflect MT in general and not specific products, we expect the impact of OCR error to vary with method of MT, as follows. Syntax-based MT parses sentences and works on interactions between words. Word-replacement MT does word-by-word substitution from bilingual dictionaries and disregards word interactions. Word-replacement MT is often found in government-off-the-shelf (GOTS) software for languages that lack commercial processors, such as Indonesian-Behasa. We might hypothesize that OCR error will degrade MT performance more severely in syntax-based than in word-based MT. (See Appendix A.)

4.3.3 Task Effectiveness

Question (3) calls for assessments of language-processing tools in the context of human tasks. Can we predict task effectiveness from measures of system performance? What does a recall rate of 18.2%, or a drop in recall rate of 10 points, mean in terms of what can be done with a system? Here, the linguistic understanding called for by Question (2) can inform hypotheses about how text features contribute to task effectiveness.

For example, while it seems obvious that open class words that are domain-relevant are critical for screening text [16], we need studies of how people actually screen to see what density of domain-relevant words (or some other task-critical feature) affects that ability. We could then calculate thresholds for screenability of text based on the density measure, and determine whether an MT system in general delivers screenable text. For tasks beyond screening, other features of MT output would apply. (The work of mapping linguistic features to tasks remains to be done [14].)

5 Conclusions

The challenge to researchers engaged in document image understanding is to initiate more studies on end-to-end systems. The results will benefit our theoretical understanding of how image and text processes work together and our practical need to make decisions: Which of competing components is the best choice in an end-to-end system? When is that system ready for trials with people? How does performance predict the effectiveness of a system for a given human task?

References

1. Bunke, H., Wang, P. (eds.): *Handbook of Character Recognition and Document Image Analysis*. Word Scientific Publishing (1997).
2. Chinchor, N.: MUC-7 Named Entity Task Definition Version 3.5. *Proceedings of the Seventh Message Understanding Conference*, Washington, D.C. (1997).
3. Church, K., Hovy, E.: Good Applications for Crummy Machine Translation. In: Neal, J., Walter, S. (eds.) *Proceedings of the Natural Language Processing Systems Evaluation Workshop*. Calspan-UB Research Center (1991).
4. Croft, W., Harding, S., Taghva, K., Borsack, J.: An evaluation of Information Retrieval

Accuracy with Simulated OCR Output. *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval.* 115-126, Las Vegas (1994).

5. Doyon, J., Taylor, K., White, J.: The DARPA Machine Translation Evaluation Methodology: Past and Present. *Proceedings of the Workshop on Embedded Machine Translation: Design, Construction, and Evaluation of Systems with an MT Component.* In conjunction with the Association for Machine Translation in the Americas Annual Meeting. Langhorne, PA (1998).
6. Fisher, F., Voss, C.: Falcon, an MT System Support Tool for Non-linguists. *Proceedings of the Advanced Information Processing and Analysis Conference*, McLean VA (1997)
7. Harmon, D. NIST Special Publication No. 500-215 on the Second Text Retrieval Conference (TREC-2), Washington, D.C. (1994).
8. Holland, M., Schlesiger, C.: High-Mobility Machine Translation for a Battlefield Environment. *Proceedings of NATO/RTO Systems Concepts and Integration Symposium*, Monterey, CA. Hull, Canada: CCG, Inc. (ISBN 92-837-1006-1) (1998) 15/1-3
9. Kanungo, T., Bulbul, O.: Baseline Experiments for OCR-based Arabic Named Entity Extraction. Manuscript, Language & Media Processing Lab, Center for Automation Research, University of Maryland, College Park, MD (2001).
10. Kanungo, T., Marton, G.A., Bulbul, O.: OmniPage vs. Sakhr: Paired Model Evaluation of Two Arabic OCR Products, *Proceedings of SPIE Conference on Document Recognition and Retrieval (VI),* San Jose, CA (1999) 3651.
11. Pallett, D., et al.: 1994 Benchmark Tests for the ARPA Spoken Language Program. *Proceedings of the Spoken Language Systems Technology Workshop,* Austin, TX (1994).
12. Reeder, F., Loehr, D.: Finding the Right Words: An Analysis of Not-Translated Words in Machine Translation. In: Farwell, D. et al. (eds.), *Machine Translation and the Information Soup: Proceedings of the Association for Machine Translation in the Americas Annual Meeting.* Springer-Verlag (1998) 356-363.
13. Resnik, P.: Evaluating Multilingual Gisting of Web Pages. UMIACS Technical Report. University of Maryland Institute for Advanced Computer Studies, College Park, MD. (1997).
14. Taylor, K., White, J.: Predicting What MT is Good for: User Judgments and Task Performance. In: Farwell, D. et al. (eds.), *Machine Translation and the Information Soup: Proceedings of the Association for Machine Translation in the Americas Annual Meeting.* Springer-Verlag (1998) 364-373.
15. Vanni, M.: Evaluating MT Systems: Testing and Researching the Feasibility of a Task-Diagnostic Approach. *Proceedings of Conference of the Association for Information Management (ASLIB): Translating and the Computer 20* (1998).
16. Voss, C., Reeder, F. (Eds.): *Proceedings of the Workshop on Embedded Machine Translation: Design, Construction, and Evaluation of Systems with an MT Component.* (In conjunction with the Association for Machine Translation in the Americas Annual Meeting, Langhorne, PA). Adelphi, MD: Army Research Lab. (1998).
17. Voss, C., Van Ess-Dykema, C.: When is an Embedded MT System "Good Enough" for Filtering? In *Proceedings of the Embedded Machine Translation Workshop II.* (In conjunction with the Applied Natural Language Processing Conference), Seattle, WA (2000).
18. White, J., O'Connell, T.A.: The ARPA MT Evaluation Methodologies: Evolution, Lessons, and Future Approaches. *Proceedings of the Association for Machine Translation in the Americas Annual Meeting* (1994).

Appendix A

Mapping OCR to MT Errors in Syntax-Based MT: An Illustration.

It is obvious that an OCR error will result in mistranslation – producing either a non-word (which cannot be translated) or a word other than ground truth (which leads to inaccurate translation, if it translates at all). However, it is not obvious that a single OCR error can damage the translations of many words. Here is an example for Spanish from the Falcon end-to-end system. The OCR for an earlier version of Falcon was a single multilingual package that for Spanish made a consistent error of recognizing upper case "E" as upper case "K." While this error may seem a rare occurrence, a 1998 article from Spain's *El Pais* newspaper began as follows:

Original:
En este artículo de Javier Pradera...

English translation (MT) with OCR errors:
East Kn article of Javier Prairie...

English translation (MT) with corrected OCR:
In this article of Javier Prairie...

Because "en" is a preposition and not a content-carrying word, its misrecognition as "kn" might not be predicted to impede understanding. Indeed, since "kn" is not an English word, we tend to ignore it in scanning a translation for key words. However, the failure to translate "en" leads to a second error, for Spanish "este," which can mean "east" or (especially in prepositional phrases) "this." Without assignment to a prepositional phrase construction, as marked by initial "en," the "este" is translated as "east." This mistranslation might prove disruptive to scanning: It provides a false alarm that distracts the human who is scanning for domain-defining key words. A one-to-many mapping such as this, from OCR error to translation error, is likely to arise in syntax-based MT but not in word-based MT.

Advanced Labeling Techniques for Scanned Document Images

Daniel X. Le and George R. Thoma

National Library of Medicine
8600 Rockville Pike, Bethesda, MD 20894

SUMMARY

In order to support automated document conversion mission at the National Library of Medicine (NLM), we are developing an automated data entry system, the Medical Article Record System (MARS), to identify and convert bibliographic information from paper-based biomedical journals to electronic format for inclusion in the MEDLINE® database used by biomedical researchers and clinicians worldwide. We have implemented several advanced techniques for automatically labeling zones from scanned document images with meaningful labels such as article title, author, affiliation, and abstract using a rule-based algorithm, neural network technology, and a page normalization and string patterns template matching algorithm.

These labeling techniques use a combination of geometry-based and content-based zone features calculated from optical character recognition (OCR) output. Geometry-based zone features derived from geometric zone information include zone dimensions, zone locations, zone order, number of columns, column dimensions, and column locations. Content-based zone features derived from zone contents and font characteristics include total characters, total capital characters, total punctuation marks, number of text lines, average font size, average character height, and font attributes (normal, bold, underlined, italics, superscript, subscript, and fixed pitch).

Specially, for the page normalization and string patterns template matching algorithm, a new feature called "single and multiple column zone vertical area string pattern" is proposed to normalize document image pages. A single column zone vertical area of a binary image is defined as a vertical area in which only one text zone exists. A multiple column zone vertical area of a binary image is a vertical area in which more than one zone exists, and where the zones are "vertically overlapped". Two zones are vertically overlapped if the top and/or the bottom coordinates of one zone are within the top and the bottom coordinates of another zone. Generally, the number of text lines in a labeled zone such as title, author, affiliation, or abstract is different from one article to another in a journal issue and therefore the labeled zone coordinates of one article may not be the same as those of another article. As a result, using the same document style guide, the geometric page layout of one article may not be the same as that of another article in the same journal issue. In order to overcome this problem of irregularity, the "single and multiple column zone vertical area string pattern" feature is used to handle pages having the same document style guide but different geometric page layouts.

The automated labeling process consists of three steps: (1) scan journal images, (2) perform OCR and detect zones around contiguous text, and (3) apply automated labeling to associate a label, such as "Title", with each zone of interest. In general, the first page of each article of a journal issue is scanned and saved as a binary document image. Next, each scanned binary document image is segmented into text and graphics zones. Each text zone is then processed to deliver an OCR output (including zone coordinates, characters and their bounding boxes, confidence levels, font sizes

and style attributes). Using the OCR output generated for each zone of a page, zone features are calculated. Finally, zone features are input into each labeling system for label classification.

For the rule-based labeling algorithm, a set of 120 rules are generated for label classification and they are derived from an analysis of the page layout for each journal, from generic typesetting knowledge for English text, and from features extracted from OCR output.

For the neural network based algorithm, a two-layer back-propagation neural network is implemented with an input layer of sixteen text zone features, a five output layer (title, author, affiliation, abstract, and others), and a single hidden layer of which the number of nodes is 8.

For the page normalization and string patterns template matching algorithm, after normalizing document pages, a template matching algorithm calculates similarity classification features by matching vertical area string patterns of document pages to those of predefined layout document structures. Similarity classification features and both geometry-based and content-based zone features are then input into a rule-based learning system for the final decision on the label classification.

Experiments carried out on several hundred images of biomedical journals pages show that our labeling techniques are capable of labeling text zones at an accuracy of 96.7 % for the rule-based algorithm, 97.0 % for the back-propagation neural network technology, and 96.0 % for the page normalization and string patterns template matching algorithm.

Foreign Language Information Retrieval

Document Image Retrieval Techniques for Chinese

Yuen-Hsien Tseng
Dept. of Library and Information Science
Fu Jen Catholic University
510 Chung Cheng Road, HsinChuang,
Taipei, Taiwan, R.O.C. 242

Douglas W. Oard
College of Information Studies and
Institute for Advanced Computer Studies
University of Maryland, College Park, MD
20742, USA

Abstract:

In this paper we present experiment results for retrieval from a collection of scanned article clippings from Chinese newspapers. The test collection consists of 8,438 articles from China, Taiwan and Hong Kong in a mix of traditional and simplified Chinese. A commercial OCR system was used to produce errorful text. Exhaustive relevance assessment was performed over the entire collection for 30 Chinese queries by multiple judges. Indexing a combination of unigrams and overlapping bigrams was found to outperform overlapping bigram indexing alone, and byte length normalization was found to outperform cosine normalization. No improvement resulted from the addition of query expansion using blind relevance feedback on the same collection.

1. Introduction

The advent of the World Wide Web has made access to digital information easier than ever before. Many information providers have therefore been inspired to digitize existing paper materials to enable access through networked information services. A number of approaches for this purpose are possible, including: (1) manual re-keying of the text; (2) creation of metadata; (3) creation of document images through scanning; and (4) layout analysis and optical character recognition (OCR) of document images [1, 2]. Many current systems have combined approaches (2) and (3), using metadata to support search and document images to support electronic document delivery. Although manual creation of metadata can be much more economical than manually re-keying the full text for each document, it still involves considerable cost in time and human effort. Furthermore, manually produced metadata can only support searches based on information needs that could be anticipated when the metadata was created. Combining approaches (3) and (4) offers complementary strengths, using OCR to produce searchable (although sometime erroneous) full-text representations, and document images as a basis for electronic document delivery. When sufficiently accurate OCR is possible, this can provide relatively inexpensive support for searches based on the full vocabulary used by document authors. Combining approaches (2), (3) and (4) can provide even richer support for information access. For example, full-text search can be used to find documents that address previously unforeseen topics, and metadata can be used to limit the search to a range of creation dates that is appropriate to the user's task. In this paper, we focus on the use of approach (4) to support the process of searching a collection of printed Chinese documents.

Recently there has been considerable interest in the application of approach (4) to historical newspaper materials. The Pathfinder Library System in Grand Junction, Colorado, has started a Library Services and Technology Act (LSTA) project that will explore digitization, indexing, copyright, and other issues relating to providing access to historical Colorado newspapers over the Internet to students, researchers, and other potential users [3]. This "proof-of-concept" pilot project intends to scan the daily Aspen Times for the year 1887, perform OCR on the resulting document images to generate text files for full-text indexing, create a searchable database of content indexes with links to the newspaper images and text files, and develop a prototype Web site for the project.

Research in languages other than English clearly indicates that effective support for information access using approach (4) requires some degree of language-specific processing. For example, researchers working with historical Greek newspapers in a project at the Lambrakis Press have developed techniques that account for changes in the Greek languages over time [4]. The experiments reported in this paper are motivated by a similar project at the Socio-Cultural Research Center (SCRC) at Fu Jen Catholic University in Taiwan, which has scanned 600,000 of the 800,000 newspaper clippings that they have collected from Mainland China, Hong Kong, and Taiwan over the past 50 years with the ultimate goal of providing access to the collection over the Internet. In this case, each clipping has been separately scanned to create 300 dot per inch (dpi) TIFF images, so segmentation is less of an

issue. But the inventory of Chinese characters is far larger than for Western languages, and the lack of explicit word boundaries makes both OCR and retrieval more challenging.

Information retrieval is an experimental science in which test collections provide the basis for tuning systems for optimum retrieval effectiveness. For the Text Retrieval Conference, OCR results were simulated by applying a confusion model trained on actual OCR output to an existing information retrieval test collection. While this approach can provide some degree of insight into the sensitivity of a technique to OCR errors, evaluations based on actual scanned document images are generally preferred by OCR researchers because OCR accuracy depends on a wide array of situations (e.g., bleedthrough in two-sided printing or spurious marks on historical materials) that might not be modeled with sufficient fidelity. For this reason, the Fritz Kutter-Fonds Foundation in Zurich sponsored an evaluation of automatic cataloguing and free text searching in 1999 that was based on 500 books in four European languages that were published between 1770 and 1970 [5, 6, 7]. Book pages on which the cataloguing was to be based were scanned to produce document images that were then converted to text and Xerox Xdoc layout description files using layout analysis and OCR. This paper complements that work describing what we believe is the first information retrieval test collection for an Asian language based on scanned documents images. Experiment results obtained using the collection are presented for a variety of retrieval techniques

The remainder of the paper is organized as follows. The next section introduces the test collection, and Section 3 then briefly surveys previous work on document image retrieval. Our techniques and experiment results are then presented in Sections 4 and 5, respectively. Finally, Section 6 concludes the paper.

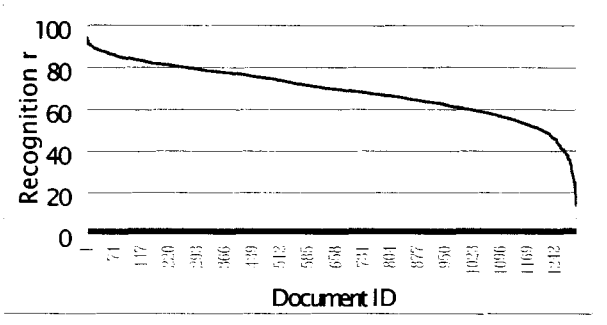
2. The SCRC Chinese Document Image Retrieval Test Collection

An information retrieval test collection contains a set of documents, a set of topic descriptions from which queries can be constructed, and a set of relevance judgments that identifies the relevant documents for each topic. From the SCRC news clippings, we selected a 11,108 document images to create the test collection. The selected stories focus mostly on diplomatic and military developments in Mainland China between 1950 and 1976. Stories from 30 news agencies are represented in the collection, from Mainland China (where simplified Chinese is used), Hong Kong, and Taiwan (where traditional Chinese is used). Most documents are in simplified Chinese, but some are traditional Chinese. A sample image is shown in Figure 1.

Figure 1. A sample newspaper clipping image.

The 11,108 images were converted to text by a commercial OCR system, yielding 8,438 valid text documents. Others are rejected by the OCR system due to low image quality or other limitations to the recognition capability of the system. To get an idea of the OCR quality, we tabulated the system-reported character recognition rates for a 1,300 document sample. We found that the average recognition success rate was only 0.69 (with a standard deviation is 0.124). The distribution of the recognition rates is shown in Figure 2. Compared to the figures claimed by the OCR vendor, where a recognition rate of over 0.9 or even 0.95can be expected for ordinary printed materials, the low OCR rate in this case might be due to low print quality of the aging clippings. Although these statistics represent system-generated estimates rather than character accuracy based on ground truth data, they do provide an initial basis for characterizing the difficulty of the recognition challenge for these materials.

Figure 2. The recognition rate distribution, sorted by



descending order.

As to the topic set, it would be best to assemble it from real searchers' information needs. However, SCRC's research library does not record the nature of users' requests for reference assistance. We therefore

gathered possible query topics from various journal articles published at about the same time as the news stories. This was based on our belief that if some issue was being written about in a journal article, there may have some information needs related to that issue. From 100 paper titles, 30 were selected and rewritten as formalized information need statements (topics) in Chinese using a format similar to that of the Text Retrieval Conference (TREC) topic descriptions. These topics have also been translated into English by SCRC's social science researchers to support possible cross-lingual (English-Chinese) retrieval experiments in the future. Figure 3 shows an example of an English translation of a Chinese topic.

<top>
<num> 12
<title> Anti-Chinese Movements
<description>
Activities related to the anti-Chinese movements in Indonesia
<narrative>
Articles must deal with activities related to the anti-Chinese movement in Indonesia; case reports or articles dealing with PRC's criticism of the Anti-Chinese movement will be considered partly relevant.
</top>

Figure 3. A sample topic in English.

The degree of relevance for each document with respect to each topic was judged by three assessors (two of whom majored in history, with the other having majored in library science). Three levels of relevance could be specified. Complete (i.e., exhaustive) relevance assessments were performed, with each document image (not the possible erroneous OCR text) being examined for relevance to any topic. We used exhaustive assessment because the alternative, a sampling strategy known as "pooled relevance assessment" would have required the participation of multiple teams using different techniques in a coordinated evaluation. A level of 0 was assigned to irrelevant documents, 1 for partially relevant documents, and 2 for fully relevant documents. The ability to specify the degree of relevance may allow assessors to express relationships in a more natural way than binary relevance judgments would. Each assessor required an average of 4 minutes to judge the relevance of one document to 30 queries, so a total of (4 x 8438 x 3) = 101,256 minutes was invested over two months to perform the (8438 x 30 x 3) = 759,420 relevance judgments. The relevance levels for each topic-document pair were then summed over the 3 assessors to produce a value between 0 and 6 that could serve as the ground truth degree of relevance for a document. Retrieval performance is often expressed in terms of precision and recall, where precision is the ratio of relevant documents in the retrieved set to the total size of the retrieved set, and recall is the ratio of relevant documents retrieved over all relevant documents in the collection. Such measures require binary-valued relevance judgments, which can be produced by applying whatever threshold to this value that the experimenter believes would best represent the retrieval task that they seek to model. In the experiments reported below, we treat a document as relevant for purposes of evaluation if it has a non-zero value, and irrelevant otherwise.

The 8,438 images were converted to text in BIG-5 code by the OCR software mentioned above. For the convenience of researchers with tools optimized for the GB code that is in common use in Mainland China, standardized GB versions of the recognized text were produced using the "ConvertZ" freeware utility (http://www.speednet.net/~shing/). In the experiments reported below, we used only the GB representation.

3. Previous Work

A number of researchers have done studies of automatic retrieval using degraded text produced by (or modeling) OCR. In this section we summarize the results with respect to the faceted classification of approaches summarized in Table 1. A fuller description of each study can be found in [1]. Although no study that we are aware of has yet explored retrieval based on Chinese OCR results, we found that the experience of others in working with degraded text shed considerable light on the directions that we could take in our work.

Taghva et al. did a series of studies to identify the effects of OCR errors on text retrieval using different models. In [8], they used a Boolean logic retrieval system, finding that the effect of OCR errors was insignificant for a small collection or relatively long documents (38 pages per document). The same group did two other studies [9, 10], one using the InQuery system [11], which uses a probabilistic retrieval model, the other using the SMART system [12], which uses a vector space retrieval model. Unlike the Boolean model, both of these retrieval produce a ranked list in which the documents most likely to be relevant to a query are listed first. Results obtained using both the probabilistic and the vector space retrieval models showed that although no statistically significant differences were found between the mean average precision of the OCR and the manually corrected collection the results for individual queries can be greatly affected. They

attributed this effect to unreliable term frequency statistics derived from the noisy OCR text. The term frequency statistics greatly affect the term weighting measure, on which the probabilistic retrieval model is based to calculate the query-document relationship. Additional findings with the vector space retrieval model was that *cosine normalization* had a negative effect when compared to the unnormalized inner product, and that *relevance feedback* could not be used to compensate for OCR errors caused by badly degraded documents. Relevance feedback is an automatic process that uses information derived from known relevant and non-relevant documents to reformulate queries. It has been consistently shown by various experiments that relevance feedback is an effective approach to improve performance for ordinary clean text [13, 14], so this was a surprising result.

Indexing method	Word-based indexing. N-gram indexing (fixed or variable length).
Retrieval model	Boolean logic positional model. Vector space model. Probabilistic model. Approximate string matching.
Test collection	Direct OCR output. Simulated OCR output.
Evaluation measure	Percentage of documents returned from the OCR set Mean Average precision (over recall levels and topics). Document ranking fluctuation (mean, variance, or correlation).
Performance comparison	Compared with original clean text. Compared with manually corrected text. Compared among different levels of simulated OCR error.
Specific strategy	No strategy: rely on information redundancy. Long query: sort of document relevance feedback. Query expansion: term expansion based on the original query. Preprocessing: automatic correction of OCR errors. Interaction with users.

Table 1. Faceted classification of OCR text retrieval approaches.

The experiments done by Taghva et al. showed that some widely used weighting schemes that are known to be effective for ordinary text might lead to more unstable results for OCR degraded text. Singhal et al. [15] analyzed this phenomenon closely using the SMART system and a simulation of the expected degradation from OCR in the large (742,202 documents) TREC collection. In their research, Singhal et al., found that an erroneous term like "systom" that might be produced by a recognition error could have a large inverse document frequency (*idf*) value, thus incorrectly affecting the weights of the index terms if a mutually dependent normalization like the cosine is used. They found that instead using a byte size normalization scheme could mitigate this source of error. For a document, their *byte size normalization factor* is computed as:

$$(byte\ size)^{0.375}$$

Singhal et al. found that bite size normalization produced a higher mean average precision and was more robust across topics than cosine normalization for both OCR output and ordinary text.

Another attempt to seek robust weighting methods was made by Mittendorf et al. [16]. They used expected term frequency (*tf*) and expected *idf* for term weighting under a probabilistic model instead of the direct term frequency statistics from the OCR collection. Eight hundred library catalog cards were scanned, OCR was performed with 67% word accuracy, and the result was split into training and test sets of equal size. Manual re-entry of the same data was done to derive the actual *tf* and *idf*, and the training set was used to estimate this actual *tf* and *idf* based on observed values. This resulted in a 23% relative improvement in the average number of relevant documents found in the first position of the ranked list in a known-item retrieval evaluation. In this case the documents were quite short (averaging 23 terms). Although this research suggests that parameter estimation can be helpful, the required training documents and their associated ground truth may not be available for other cases.

Lopresti and Zhou [17] examined the effects of varying the degree of degradation on the effectiveness of Boolean, fuzzy Boolean, vector space, extended Boolean, fuzzy extended Boolean, proximity Boolean, and fuzzy proximity Boolean retrieval models. One thousand news articles were collected from the Internet and corrupted to varying degrees using a model of OCR effects. An analysis of rank correlation coefficients within a technique for varying degrees of degradation showed that fuzzy retrieval models based on approximate string matching appear to be generally more robust than their traditional counterparts. The approximate string matching techniques used in the study were quite inefficient, however, raising questions about the practicality of the technique in large-scale applications.

Character n-grams offer a more efficient way of achieving some degree of approximate string matching. Pearce and Nicholas [18] applied fixed-length (overlapping) n-gram indexing to index both OCR results and ordinary text, exploring alternative normalization functions. Their most robust normalization function, which they called *similarity link*, was unfortunately computationally intractable. Harding et al. experimented with simultaneous use of multiple n-gram lengths for OCR-based retrieval [19], finding that this improved retrieval performance over

word-based indexing at 10% or greater OCR degradation. N-gram indices are, however, larger than word indices, and their efforts to find more efficient n-gram indexing produced adverse effects on retrieval effectiveness

From this survey of prior work, we can observe fairly clear agreement on the following factors:

- OCR errors have relatively little negative effect on retrieval effectiveness for long documents. Redundancy is often beneficial in information retrieval applications, and longer documents naturally offer more scope for redundancy.
- Byte length normalization results in better retrieval effectiveness than cosine normalization when using the vector space model.
- N-gram indexing can result in better retrieval effectiveness than word-based indexing if OCR errors are relatively common.
- There can be a tradeoff between retrieval effectiveness and retrieval efficiency when developing techniques to search in the presence of OCR errors.

In the next section we apply these observations to the design of techniques for OCR-based retrieval of Chinese document images.

4. Experiment Design

There are thousands of Chinese characters, about 2,000 to 3,000 of which are in common use. Chinese words vary in length from a single character (almost every Chinese character has meaning as a word on its own) to nine or more characters, with an average of about two characters (for contrast, the average length of an English word is about 5 characters). In many cases, longer "words" are actually better thought of as compound terms, since some speakers of the language could segment them into shorter words and recover the same meaning. From the perspective of information retrieval, Chinese differs from English in two important ways. First, Chinese retrieval is more challenging because written Chinese includes no delimiters between words and available automatic segmentation techniques are imperfect. Second, Chinese retrieval is made somewhat easier by the relative absence of morphological variants, thus obviating the need for stemming.

Comparative studies have established that n-gram indexing (usually with n=2, for bigrams) works about as well for Chinese retrieval as word-based indexing, both for ordinary text [20] and for text produced by automatic speech recognition [21]. As we have seen above, n-gram indexing is also known to be relatively robust in the presence of OCR errors. We therefore chose to use n-grams as the basis for our experiments. This raises the question of how to choose the optimum value for n. Longer n-grams might match character sequences that are recognized without error fairly well, but shorter n-grams might help mitigate the effect of OCR errors. We therefore tried n=1, n=2, and a combination of the two n-gram lengths. The large inventory of Chinese characters results in excessively large indices for n>2, so we did not try larger values of n.

Previous studies have shown an interaction between the retrieval model and the term weighting and normalization techniques. We therefore ran experiments with InQuery, a widely used system based on a probabilistic retrieval model, and a locally developed vector space retrieval system called Crystal. InQuery provides a standard reference implementation which has been extensively debugged, while Crystal provides complete access to the internal features of the system. We used the default term weighting and "weighted sum" normalization techniques of the InQuery system. In contrast, weighting schemes in vector space model may vary quite differently. With Cyrstal, we experimented with both byte length normalization and cosine normalization. For query n-gram weights, we emphasized longer n-grams over shorter n-grams as follows:

$$q_k = \frac{tf_k(3w_k - 1)}{\sqrt{\sum_{i=1}^{t} tf_i(3w_i - 1)}}$$

where q_k is the weight of n-gram k from a query, tf_k is its term frequency, w_k is the number of characters in n-gram k, and t is the total number of n-grams in the query. With this formula, single bigram match (with weight 5) is given more emphasis than two unigram matches (each with weight 2), but less than three unigram matches.

To explore whether retrieval effectiveness could be further improved, we also ran a small set of blind relevance feedback experiments using the same collection and Rocchio's method:

$$W_{new} = \alpha W_{old} + \beta \frac{1}{|R|} \sum_{X \in R} X - \gamma \frac{1}{|T - R|} \sum_{X \in T - R} X$$

where W_{old} is the initial query weight vector, R is the set of relevant document vectors, T is the set of all document vectors, and α, β, and γ are coefficients controlling the contribution of each factor. For blind relevance feedback (i.e., without manual relevance judgments), the top N documents in the initial result set are assumed to be relevant.

Run ID	System	Field	Indexing Method	Weighting scheme	Ave. P
I1s	Inquery	title only	1-gram		0.3612
I2s	Inquery	title only	2-gram		0.4083
Ins	Inquery	title only	1-gram and 2-gram		0.4397
I1	Inquery	all fields	1-gram		0.3472
I2	Inquery	all fields	2-gram		0.4621
In	Inquery	all fields	1-gram and 2-gram		0.4692
Gb1s	Crystal	title only	1-gram	log(tf)*log(IDF)*ByteSize, tf(3w-1)*Cosine	0.3509
Gc1s	Crystal	title only	1-gram	log(tf)*log(IDF)*Cosine, tf(3w-1)*Cosine	0.3059
Gb2s	Crystal	title only	2-gram	log(tf)*log(IDF)*ByteSize, tf(3w-1)*Cosine	0.4044
Gc2s	Crystal	title only	2-gram	log(tf)*log(IDF)*Cosine, tf(3w-1)*Cosine	0.4000
Gbs	Crystal	title only	1-gram and 2-gram	log(tf)*log(IDF)*ByteSize, tf(3w-1)*Cosine	0.4164
Gcs	Crystal	title only	1-gram and 2-gram	log(tf)*log(IDF)*Cosine, tf(3w-1)*Cosine	0.4098
Gb1	Crystal	all fields	1-gram	log(tf)*log(IDF)*ByteSize, tf(3w-1)*Cosine	0.3963
Gc1	Crystal	all fields	1-gram	log(tf)*log(IDF)*Cosine, tf(3w-1)*Cosine	0.3157
Gb2	Crystal	all fields	2-gram	log(tf)*log(IDF)*ByteSize, tf(3w-1)*Cosine	0.4582
Gc2	Crystal	all fields	2-gram	log(tf)*log(IDF)*Cosine, tf(3w-1)*Cosine	0.4344
Gb	Crystal	all fields	1-gram and 2-gram	log(tf)*log(IDF)*ByteSize, tf(3w-1)*Cosine	0.4757
Gc	Crystal	all fields	1-gram and 2-gram	log(tf)*log(IDF)*Cosine, tf(3w-1)*Cosine	0.4459

Table 2. Experiment results using different retrieval models, query sets, indexing methods, and weighting schemes.

Run ID	System	Field	Indexing Method	Weighting scheme	Feedback parameters	Ave. P
Gb	Crystal	all fields	1-gram and 2-gram	Best in basic strategies	(No expansion)	0.4757
Gbrf	Crystal	all fields	1-gram and 2-gram	Same as above	Top N=1, $\alpha = \beta = 1$	0.4117
Gbrf5	Crystal	all fields	1-gram and 2-gram	Same as above	Top N=5, $\alpha = 5$, $\beta = 1$	0.4255

Table 3. Experiment results using Rocchio's relevance feedback formula.

5. Results

The results obtained are listed in Table 2 and 3. In those tables, the retrieval effectiveness is characterized by the mean (over topics) of the uninterpolated average precision, a commonly used evaluation measure computed by the trec_eval program (available at ftp://ftp.cs.cornell.edu/pub/smart/). As can be seen, the combination of unigrams and overlapping bigrams consistently performs better than that overlapping bigrams alone, which in turns consistently outperforms unigrams alone. This is true for the probabilistic and vector space retrieval models, and for both long (all topic fields) and short (title field only) queries. Another consistent result is that byte length normalization performs better than cosine normalization for different indexing methods and for different query lengths.

Several studies of Web searching behavior have shown that searchers typically use only a few query terms, perhaps because it is easier to understand the behavior of the system when only a few terms are used. The title field in each topic description typically contains between two and five Chinese characters that are chosen as terms that a Web searcher might issue as a query. The results show that long queries (using all fields) generally perform better than short queries (using only the title field), achieving a 6.7% relative improvement with InQuery and a 14.2% relative improvement with Crystal. Viewed another way, the two retrieval models (probabilistic and vector space) perform about equally well with long queries, but InQuery's probabilistic model achieved a 5.6% relative improvement over Crystal's vector space model with short queries.

Table 3 shows the results from our blind relevance feedback experiments. With the best retrieval strategy, no performance improvement was observed using the Rocchio parameter values that we tried. Although these results should be interpreted as extremely preliminary because we have yet to systematically explore the space of possible parameter values, they do suggest that correctly tuning blind relevance feedback parameters without a test collection of the type we have developed would be impractical. In fact, it is not yet clear that blind

relevance feedback using documents that contain OCR errors will be helpful. Results obtained by Singhal et al. in a spoken document retrieval application [22] suggest that effective blind relevance feedback may actually require comparable (i.e., topically similar) text that is free of OCR errors.

6. Conclusions and Future Work

OCR text provides the cheapest and fastest way to make full-text images searchable, but optimizing retrieval effectiveness under these conditions requires that the retrieval technique be adapted to mitigate the effect of OCR errors. Previous work has shown that OCR errors have little effect on retrieval at low OCR error rates, but that relatively short documents with poor image quality can produce severe adverse effects. Unfortunately, this is often the case in real-world applications such as retrieval from the SCRC collection of Chinese newspaper clippings. We have described a new Chinese document image retrieval test collection and a set of retrieval experiments that we performed with that collection to explore the effect of different retrieval models, query lengths, n-gram lengths, and normalization schemes. The results show that n-gram length has the greatest effect on retrieval effectiveness, with a combination of unigrams and overlapping bigrams producing the best results. Query length was also found to have a substantial effect, as has been seen in other retrieval evaluations.

The test collection that we have developed opens the door to a number of interesting questions that we are interested in exploring. Because we have already translated the topic descriptions into English, cross-language document image retrieval is a logical next step. We already have some experience with cross-language spoken document retrieval between English and Chinese, so much of the required infrastructure for such an experiment is already in place. A second interesting direction for exploration is differential handling for documents with many recognition errors. For each document in the test collection, we know the estimated error rate reported by the OCR system. It might prove useful to develop corpus-trained correction algorithms even if such algorithms are computationally expensive, because we might productively apply those algorithms to only the most severely degraded documents. As a first step in this direction, we have begun to create manually corrected text for a portion of the test collection. Finally, we believe that it would be interesting to explore additional normalization techniques, particularly those that are optimized for short queries. InQuery's excellent performance with short queries suggests that there is likely some room for improvement in this regard. We expect that others will find additional uses for the same test collection. With multi-level judgments from multiple judges, topic descriptions in two languages, and manually corrected text for some of the documents, we believe that it is a rich resource that can support important research on document image retrieval in the years to come. Researchers interested in using the collection should contact the first author.

Acknowledgements

This work is supported in part by National Science Council, Taiwan, under the grant numbers: NSC 88-2418-H-001-011-B8908 and NSC 88-2418-H-001-011-B9003 and by the U.S. Defense Advanced Research Projects Agency under contract N6600197C8540 and under cooperative agreement N660010028910. The authors will like to thank researchers in the Socio-Cultural Research Center, Fu Jen Catholic University, for helping create the OCR test collection, the University of Massachusetts for the use of InQuery, and Tapas Kanungo for his helpful suggestions.

References

[1] Yuen-Hsien Tseng, "Automatic Cataloguing and Searching for Retrospective Data by Use of OCR Text", Journal of American Society for Information Science and Technology (JASIST), Vol. 52, No. 5, March, 2001, pp. 378-390.

[2] Yuen-Hsien Tseng, "An Approach to Retrieval of OCR Degraded Text", Journal of Library Science, National Taiwan University, No. 13, Dec. 1998, pp.153-168.

[3] http://www.colosys.net/pathfinder/AboutPathfinder/HistoricNewspapers.htm

[4] S.L. Mantzaris, B. Gatos, N. Gouraros and S.J. Perantonis, "Linking Article Parts for the Creation of a Newspaper Digital Library", *Content-Based Multimedia Information Access International Conference (RIAO2000)*, Paris.

[5] "Contest 1999 - Automatic Cataloguing and Searching Contest", Fritz Kutter-Fonds, http://www.kutter-fonds.ethz.ch/contest99.html

[6] Andreas Myka, "QDOC'99 - Querying Document Bases," Final report submitted to the Fritz Kutter-Contest 1999, ETH Zurich.

[7] Yuen-Hsien Tseng, "Report for Automatic Cataloguing and Searching Contest," Final report submitted to the Fritz Kutter-Contest 1999, ETH Zurich.

[8] K. Taghva, J. Borsack, A. Condit, S. Erva, "The Effects of Noisy Data on Text Retrieval," Journal of the American Society for Information Science, Vo.45. No. 1, 1994, pp.50-58.

[9] Kazem Taghva, Julie Borsack and Allen Condit, "Results of applying probabilistic IR to OCR text," Proceedings of the seventeenth annual international ACM-SIGIR conference on Research and development in information retrieval July 3 - 6, 1994, Dublin Ireland, pp. 202-211.

[10] K. Taghva, J. Borsack, and A. Condit, "Effects of OCR Errors on Ranking and Feedback Using the Vector Space Model," Information Processing and Management, Vol. 32, No.3, 1996, pp. 317-327.

[11] J. P. Callan, W. B. Croft, and S. M. Harding, "The INQUERY retrieval system," Proceedings of the 3rd International Conference on Database and Expert Systems, Springer-Verlag, New York, 1992, pp.78-83.

[12] Gerard Salton, editor, *The SMART Retrieval System - Experiments in Automatic Document Retrieval*, Englewood Cliffs, NJ, 1971, Prentice Hall Inc.

[13] William B. Frakes and Ricardo Baeza-Yates, *Infomation Retrieval: Data Structure and Algorithms*, Prentice Hall, 1992.

[14] Harman, D. "Overview of the third Text REtrieval Conference (TREC-3)" Proceedings of the Third Text Retrieval Conference, 1994, pp.1-19.

[15] Amit Singhal, Gerard Salton, and Chris Buckley, "Length Normalization in Degraded Text Collections," Proceedings of Fifth Annual Symposium on Document Analysis and Information Retrieval, April 15-17, 1996, pp. 149-162.

[16] Elke Mittendorf, Peter Schäuble and Páraic Sheridan, "Applying probabilistic term weighting to OCR text in the case of a large alphabetic library catalogue", Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval July 9 - 13, 1995, Seattle, WA USA, pp. 328-335.

[17] Daniel Lopresti and Jiangying Zhou, "Retrieval Strategies for Noisy Text," Proceedings of Fifth Annual Symposium on Document Analysis and Information Retrieval, April 15-17, 1996, pp. 255-269.

[18] Claudia Pearce and Charles Nicholas, "TELLTALE: Experiments in a Dynamic Hypertext Environment for Degraded and Multilingual Data," Journal of the American Society for Information Science, 47(4), 1996, pp.263-275.

[19] Harding, W. B. Croft, and C. Weir, "Probabilistic Retrieval of OCR Degraded Text Using N-Grams," in Research and Advanced Technology for Digital Libraries, Carol Peters and Costantino Thanos, Editors, 1997. pp. 345-359. http://ciir.cs.umass.edu/info/psfiles/irpubs/ir-115.ps.gz

[20] Ross Wilkinson, "Chinese Document Retrieval at TREC-6," in The Sixth Text REtrieval Conference (TREC-6), edited by D. K. Harman, Nov., http://trec.nist.gov/, 1997.

[21] Helen Meng, Berlin Chen, Erika Grams, Wai-Kit Lo, Gina-Anne Levow, Douglas Oard, Patrick Schone, Karen Tang and Jian Qiang Wang, "Mandarin-English Information (MEI): Investigating Translingual Speech Retrieval," Technical Report, Johns Hopkins University, Oct. 2000, http://www.clsp.jhu.edu/ws2000/groups/mei/.

[22] Amit Singhal and Fernando Pereira, "Document expansion for speech retrieval", Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval August 15 - 19, 1999, Berkeley, CA USA, pp. 34 – 41.

Advances In Arabic Text Recognition

John Trenkle, Andrew Gillies, Erik Erlandson, Steve Schlosser, Stan Cavin
NovoDynamics Incorporated
123 N. Ashley Street, Suite 120
Ann Arbor, MI 48104

Abstract

This paper describes improvements to a system that recognizes Arabic and Farsi text in low-quality, low-resolution, binary document images. Performance advances reflected in the current system largely result from the introduction of ensembles of decision trees as the base recognizer and the development of a new methodology for training tree ensembles that relies on boosting in the sample space, bagging in the feature space and randomly selected splits at the tree nodes. These ensembles have many advantages over neural nets. Among the most important are: 1) increased recognition rates, 2) faster evaluation, 3) reliable confidence factors, and 4) run-time tradeoffs between speed and accuracy made possible by controlling the number of trees included in an ensemble. Additional performance improvement is gained from an adaptive image-noise filter that uses image metrics to decide whether filtering is necessary and, when it is, to select filter algorithms appropriate to the severity of the noise. On 722 independent test images digitized at 200x200 with various degrees of noise, the current system attains a character recognition rate of 91%. On the same images digitized at 100x200, the system attains an 89% recognition rate. A typical page, which has an average of 45 lines in a single-column layout, can be processed in 17 seconds by the current recognition system on a 750 MHZ Pentium III running Linux.

1. Background

Language Characteristics

The printed forms of Modern Standard Arabic, the predominant language of North Africa and the Middle East, and Farsi, the official language of Iran, present many challenges for OCR algorithms. The text is written right-to-left and uses a script alphabet in which consecutive letters within a word are typically joined together by a baseline stroke. In order to accommodate the baseline, characters may assume one of four forms: *isolated*, *initial*, *medial*, and *final*. Six common letters in the alphabet are exceptions to this convention and lack the medial and final forms. When one of these non-joining characters is encountered within a word, the preceding letter assumes its final (or isolated) form, and the non-joiner assumes its initial (or isolated) form. The Farsi character set differs from the Arabic set in the addition of four letter forms.

Arabic text contains a large number of special forms, called ligatures, which replace particular character pairs or even triples. For example, when the LAM character is followed by the ALEF character they will almost always be combined into a single ligature character called the LAM-ALEF. While use of the LAM-ALEF ligature is almost universal, most ligatures are optional, at the discretion of the typographer. We have encountered over 200 ligatures in our development effort, although many of these are extremely rare, occurring mainly in older typeset books. The Farsi language, by virtue of having a distinctly different set of character bigram probabilities, adds more ligatures to the mix. The non-standard use of ligatures in Arabic and Farsi publications means, in essence, that these languages present a variable number of glyphs, on the order of 200 or more, and that the frequency of occurrence of these glyphs is dependent on many factors, such as: 1) the content of a document -- poetry tends to have more ligatures for the sake of aesthetics, while business documents are more practical and use fewer; 2) the typography and font used -- various fonts support different subsets of ligatures; 3) the timeframe within which the document was produced -- newer, computer generated documents exhibit a narrower range of ligatures; 4) the preferences of the document's creator -- it is often a conscious choice to use or not use a particular ligature; and, 5) the country in which the document originated -- there are national inclinations if not overt preferences that citizens of various countries follow as a matter of habit.

Arabic text is often justified so that the right and left edges of the text column are aligned. In a Roman alphabetic setting this would be accomplished by stretching the spaces between words to fill out the desired length. In Arabic, portions of the baseline are stretched. These stretched baselines, called kashidas, occur in different words throughout the line. These extended baselines may cause recognition problems because they can resemble actual characters, particularly the medial SEEN character. Finally, although classic Arabic texts use a relatively limited number of font faces, new typographic systems have led to the proliferation of Arabic font faces which are almost as varied as those for Roman alphabets.

Expected Input Characteristics

The recognition system is specifically designed to process both images of high and low quality, and images of high and low resolution. Moreover, in the interest of providing the highest level of automation possible, the system's design assumes no human intervention occurs prior to OCR processing.

To date, system development efforts have emphasized the most challenging classes of document images. Thus, document images are selected for the following characteristics: 1) Image resolution between 200x200 dpi and 100x100dpi, including low-resolution FAX at 200x100 dpi. Higher and lower resolutions are also handled, but the system has been trained using the aforementioned resolutions. 2) Various levels of noise, and a significant fraction of all images with high levels of speckle. Artifacts from copying processes such as darkened corners and from FAX processes such as linear dropouts are also included. 3) Variable page layout complexity, ranging from simple (single-column text) to complex (newspaper or magazine). 4) Variable text position, including arbitrary gross orientations and skews. An example image is shown in Figure 1. The recognition rate for this image is 71%.

For the purposes of OCR, Arabic text is far more sensitive to salt-and-pepper noise and speckle noise than are Latin-character-based languages because most characters share a common body and are differentiated solely by the location and the presence or absence of one to three dots. Thus, in a noisy image of Arabic text, dots may be mistaken for noise and accidentally removed or, conversely, noise may be mistaken for dots and related to characters. Consequently, an intelligent filtering mechanism is essential for successful recognition of document images containing moderate to high levels of noise.

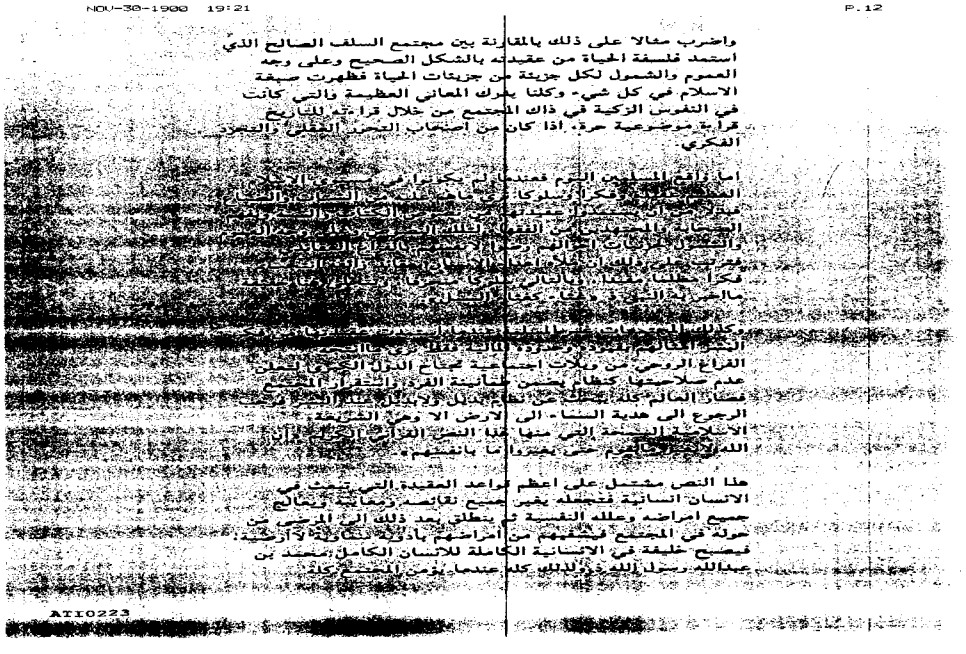


Figure 1. An Example of a Low Quality, Low Resolution Image of Arabic Text

From the above, it is clear that OCR of Arabic and Farsi text is more difficult than for Roman text due to the attributes of printed Arabic/Farsi text in combination with the features of the data input stream that have been assumed. As will be seen in the following sections, great headway has been made in achieving high accuracy, usable results on very difficult data sets. The remainder of this paper will detail the recent breakthroughs made while enhancing our system, including improved noise removal strategies and the substitution of decision tree ensembles for neural nets with subsequent speed and accuracy increases.

2. System Overview

The system described in this paper is a complete Arabic/Farsi page recognizer implemented in a UNIX environment. The system takes in document pages as binary TIFF images, and produces Unicode text files as output. A block diagram of the system is shown in Figure 2. This diagram will serve as a reference point for more specific discussions of the enhancements added recently. This will be particularly useful in understanding the implications of the enhancements made.

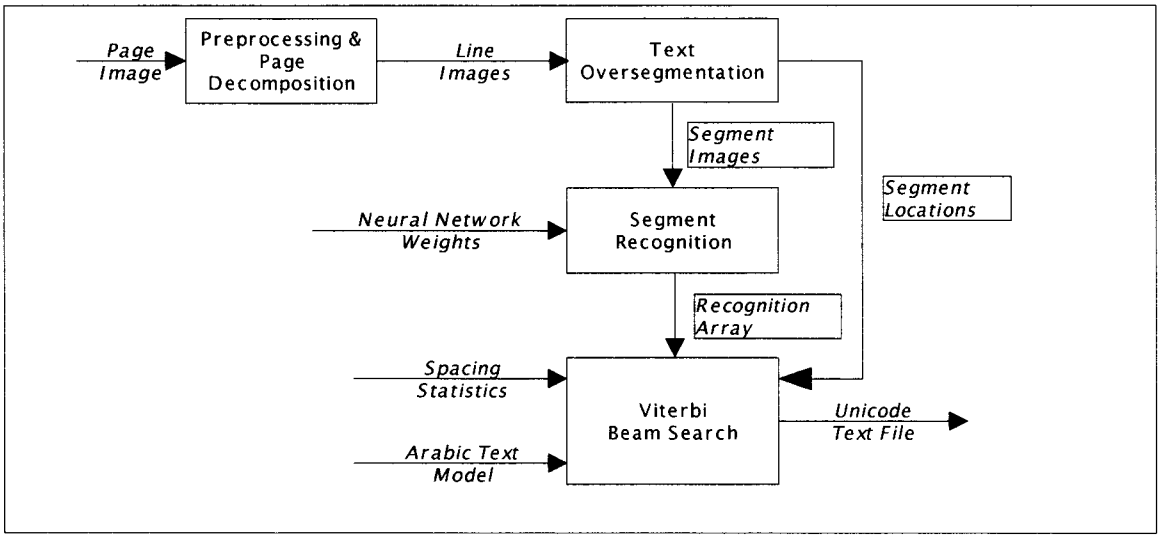


Figure 2. Arabic/Farsi Text Recognition System

Preprocessing and Page Decomposition

The raw page image arrives at the preprocessing and page decomposition module. The preprocessing stage performs the following actions: 1) Noise analysis and removal; 2) Handling of FAX banners if present; 3) Validating that the current image contains Arabic or Farsi text, rejecting otherwise; 4) Validating that it contains machine–printed, not handwritten text; 5) Determining and correcting the gross page orientation (i.e., upside–down, sideways); 6) Detecting and removing arbitrary skew; 7) Detecting low resolution FAX (i.e., 2:1 aspect ratio) and upsampling in the Y dimension. And, 8) Detecting and removing graphic elements. The page decomposition module then decomposes the page into text blocks. Each text block is segmented into individual lines of Arabic text. The text line images are normalized to a height of 40 pixels and passed to the text segmentation module.

Text Segmentation

The text segmentation module is illustrated in Figure 3. This module is an oversegmenter, designed to produce atomic image segments which are no larger than a single character. In other words, each *atomic segment* should come from only a single character of the ideal segmentation. If this goal is met, then a preferred segmentation can be produced by combining the atomic segments in the appropriate groups. Of course, the combination must be done in such a way as to maintain the spatial relationships between the atomic segments in the group. A Viterbi algorithm, discussed later in this section, produces appropriate groups as a by–product of the recognition process.

The resulting atomic segments are ordered in a left–to–right fashion, based on the center of the minimum bounding box enclosing the segment. They are combined in groups of from two to five consecutive segments. The full set of segments includes both the atomic segments and the combined segments. This results in $K < 5N$ segment images, where N is the number of atomic segments. Among the K segments are the ideal characters of the text, indicated by dashed boxes in Figure 3. The K segment images are then passed to the segment recognition module.

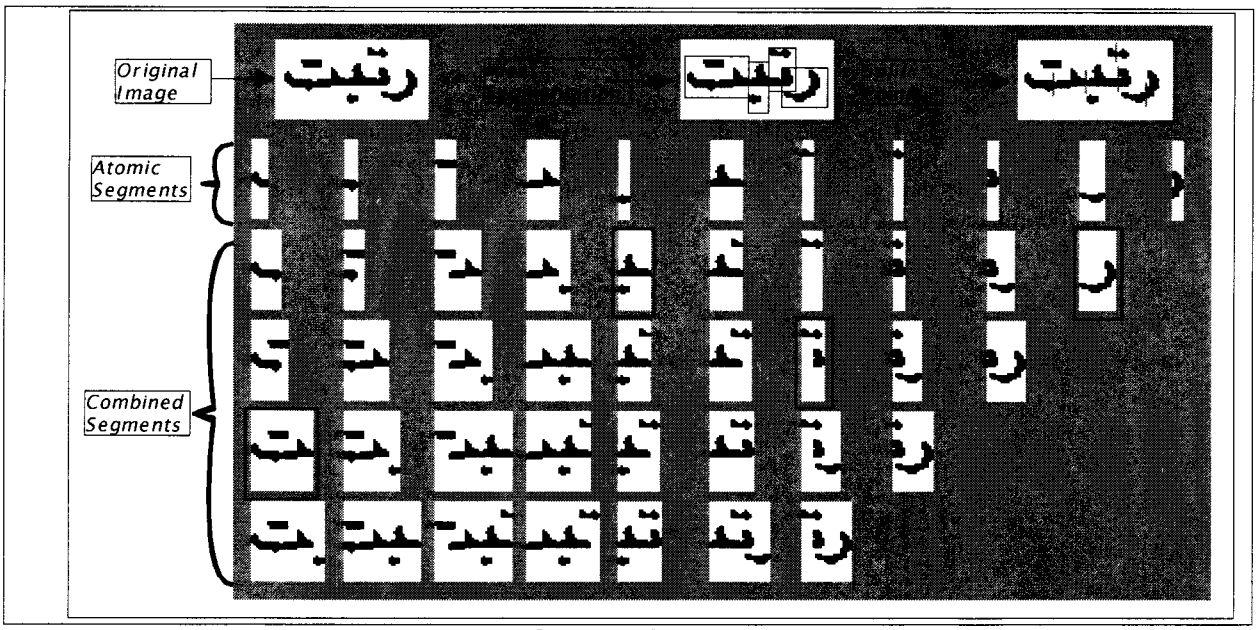


Figure 3. - Text Segmentation

Segment Recognition

This module has been the subject of intense work which will be discussed in the next section. Generally speaking, this module runs a classifier on each of the *K* input segments. The Arabic version of this classifier has 424 inputs and 229 outputs. The 424–element input–layer feature vector includes 40 projection features and 384 chain code features. The projection features contain a 20–element horizontal projection and a 20–element vertical projection of the segment image. The projections are taken with respect to a 40 x 40 pixel box in which the segment image is left–justified, and hence each count represents two rows or columns of the image. Also, if the segment is less than 40 pixels wide, the vertical projection is padded with zeros on the right.

The chain code features are based on a chain–like representation of the edges (or borders, or contours) of the image components before splitting. Each point on the chain can be uniquely associated with one of the atomic segments, and hence with each of the combined segments. At each point along the chain, the direction of travel (clockwise around the component) and the contour curvature are computed. These measurements use a window of 4 chain points on each side of the point in question. The 384 chain code features correspond to quantizing the x–coordinate, y–coordinate, direction, and curvature values into 4, 4, 8, and 3 bins, respectively. For each dimension, quantization is gaussian–weighted between neighboring bins so that a point in the center of a bin contributes a weight of 1.0 to that bin, and a point on the border between two bins contributes 0.5 to each of the two bins. Thus, each chain code point contributes to up to 2x2x2x2 = 16 feature values. The chain code features are sparse, in that many of the values are zero for a typical image.

The output of the classifier contains 229 signals corresponding to 117 regular Arabic character forms, 80 ligature forms, 10 Arabic digits, 20 punctuation characters, and two reject classes. The classifier outputs (confidences or pseudo–probabilities) for each of the segment images are combined into a *N* x 5 x 229–element array, called the recognition array. This array is passed to a Viterbi algorithm for decoding.

Viterbi Beam Search

The Viterbi beam search module transforms the information in the recognition array into a sequence of characters output by the program. The module uses a dynamic programming algorithm to match the array of segments against a model of Arabic text. The model encodes the rules of Arabic typography, for example the constraints between the forms of neighboring characters.

The Arabic text model comprises lexicon–based word recognition, lexicon–free word recognition, and recognition of Arabic punctuation and digits. The basic element of the model is a state, which ultimately associates a given image segment with a given character (or ligature). The complete Arabic text model contains over 100,000 states, most of which occur inside the lexicon–based word recognition component. The lexicon contains 50,000 Arabic words that are in common use.

A more detailed discussion of this algorithm may be found in [1], but for the purposes of this paper, it is primarily important to realize that the accuracy of the confidences or pseudo–probabilities coming from the segment recognition module is extremely important to the success of the Viterbi algorithm.

3. System Enhancements

In this section, we discuss several specific enhancements to the recognition system. The first topic is noise detection and removal, which has a very significant role in our system both because we expect a large fraction of the incoming images to have high levels of noise and because of the sensitivity of Arabic script to speckle due to the frequency of dotted characters. The second topic is the switch in base classifiers in our system from neural nets to ensembles of decision trees. This necessitates some background on trees and ensemble construction, a comparison of trees to neural nets, and a discussion of tree ensemble training methodology.

Noise Analysis

Because high levels of noise are included in our data input stream, specific methods to detect and filter this noise have been devised. It is important to differentiate between *structured* and *unstructured* noise. *Structured* noise refers to artifacts such as line dropouts in FAX, dark corners due to copying, halftone backgrounds, etc. These types of noise are handled with specific algorithms. Other noise of concern is *unstructured* in the sense that it is random and can appear in any part of the image. Examples are speckle noise and salt–and–pepper noise. If the unstructured noise were simply a matter of a isolated specks occurring discretely in the image, then simple algorithms would suffice. Unfortunately, in our data the unstructured noise is agglomerative, meaning that it consists of complex, often large, clumps which overlap the text.

When developing algorithms within the context of our system we must constantly make tradeoffs between speed and accuracy. In this situation we needed a method of rapidly determining which images actually needed filtering and then applying the moderately expensive filtering operation only to those images. The underlying principles of the filtering revolve around the fact that the system must attempt to differentiate dots from noise so as to minimize the impact of filtering on the ultimate accuracy.

Detection

The initial attempt to detect noise efficiently uses a set of computationally cheap features which are extracted from the incoming binary image and are then processed by a small decision tree ensemble to determine whether the image should be filtered. The features outlined below attempt to capture various abstract attributes of noise such as dottiness.

- Foreground Density – the fraction of foreground pixels in the binary image. Clean binary text images tend to have a fairly low density.
- Dot Factor – a mathematical morphological measure designed to gauge the presence of dots in an image as the ratio of foreground pixels before and after a closing by a cross structuring element. If the image has a lot of dense speckle, then the closing operation would tend to join them together thus reducing this ratio.
- Perimeter Factor – the ratio of all foreground pixels to those that occur on the boundaries of components. A significant fraction of pixels in small speckle and agglomerative noise tends to be on the border of the particle.
- Median Factor – ratio of the foreground pixels remaining after a median filter operation to the number of foreground pixels in the original. Speckle tends to be reduced by a median filter, thus if noise is present this ratio will tend to be lower.
- Components per Area – the number of connected components per image area. Noisy images will tend to have many more components.
- Mean Component Density – the mean foreground density of each component over the minimum bounding box oriented with the coordinate axes. Smaller noise tends to pack the bounding box tightly while character shapes tend to be less dense.
- Interior to Exterior Chains Ratio – the ratio of interior chains (*holes* in components) to exterior chains. This metric aims to measure small *holes* occurring in the text. (One might think of this as inverted speckle)

- Transition Density – the relative number of transitions from background to foreground down the columns of an image. Noisy images should have more transitions.

- Skeleton Area Fraction – the ratio of the number of foreground pixels in the skeleton of an image to the number of foreground pixels in the original. The skeletons of noise components are usually not too different in mass from the original component.

To establish the decision procedure, the above features were collected on an independent set of about 800 images which were grouped into two classes, *needs–filtering* and *doesn't–need–filtering*. This was done automatically by processing the images with our recognition system both with and without filtering, and then determining which images benefited from the treatment and which did not. The goal was to have a very fast metric for deciding whether or not to filter while simultaneously maximizing accuracy and minimizing computation. Finally, an ensemble of decision trees was trained to discriminate between the two categories of images. This resulted in an effective decision procedure because the trees are extremely fast to evaluate and are comparable in accuracy, if not better, than neural networks trained to perform the same task.

Removal

The filtering process for Arabic / Farsi text was specifically designed to attempt to preserve the structure of the Arabic script, most importantly the dots, while removing as much noise as possible. This aggressive filtering consists of two stages, *spatial filtering* and *connected–components filtering*. The spatial filter removes heavy, densely–connected noise, while the connected–component filter removes isolated noise components. Figure 4 illustrates spatial filtering, while Figure 5 shows connected components filtering. The algorithm is capable of removing very dense speckle–noise created by dark photocopier and fax settings, while preserving most genuine text. The algorithm can also be tuned for flexible tradeoffs between noise removal and text preservation.

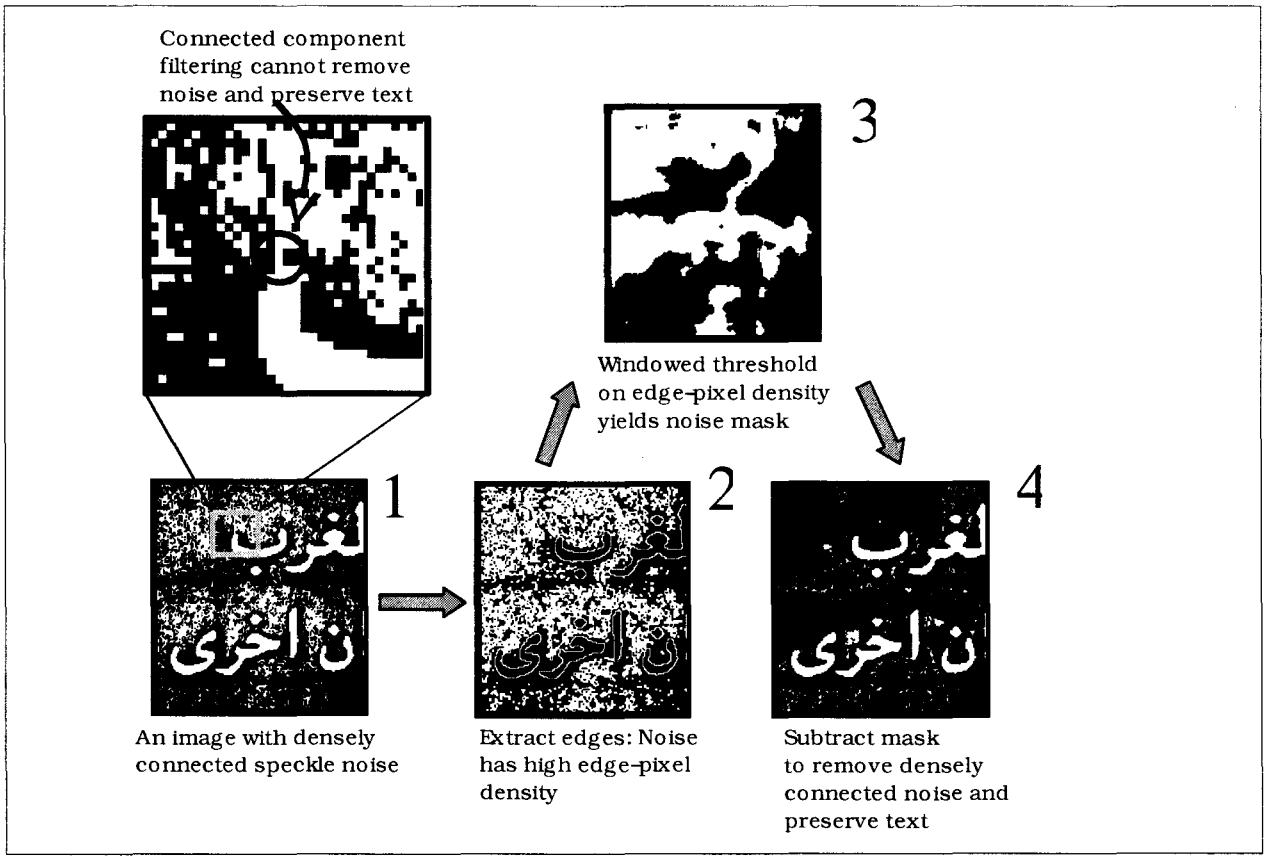


Figure 4. Spatial Noise Filtering

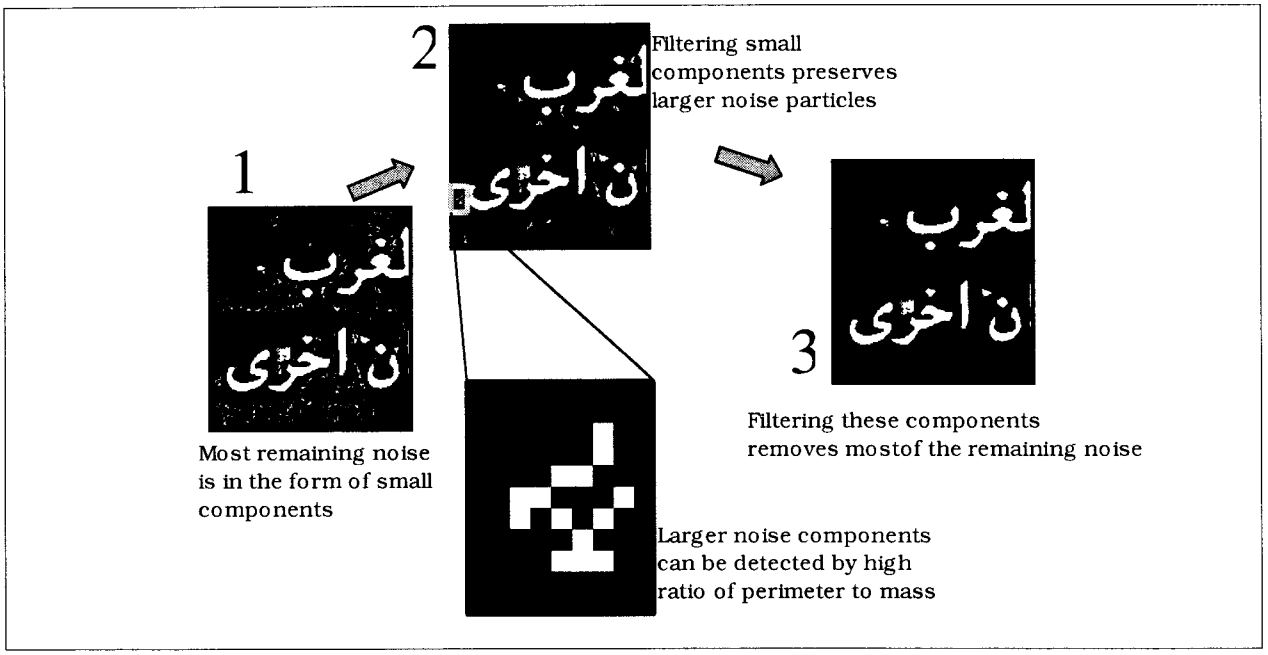


Figure 5. Connected-Components Noise Filtering

Decision Trees and Ensembles

The most significant breakthrough that we've had in the course of our recent work was to switch from our tried–and–true neural networks to ensembles of decision trees. The key reason that we had always relied heavily on neural nets was because of the high accuracy that they provided, even with the on– and off–line computational cost. Past experience proved that single decision trees could never offer the accuracy that we need because of the inherently brittle nature of trees; however, with the advent of methods for aggregating forests of trees in the past several years such as bagging [2], boosting [3,4], and feature boosting [5], the means of achieving high accuracy with *ensembles* of trees was a possible opportunity. In this section we will summarize decision tree construction, ensemble construction, and compare and contrast neural nets and ensembles of decision trees.

Decision Tree Construction

Decision trees have been a popular machine–learning paradigm for many years, primarily because a single tree offers a level of explainability not afforded by many other approaches. A decision tree (also called recursive partitioning) divides a training dataset into disjoint subsets in a greedy, deterministic way. This partitioning is achieved by using the attribute–values of instances to find the *optimal* split at any given node. This optimal choice can be formulated as a maximization of the separability of the two new subsets with respect to the sample labels. We use a variant of the CART algorithm [6] that utilizes the χ^2 statistic and picks the split with the maximum significance. A nice feature of our algorithm is that no pruning stage is necessary –– the significance of the split is also a useful terminating criterion for splitting. Decision trees classify instances by sorting them down the tree from the root node to some leaf node, which provides the classification of the instance, along with a probability estimate, based on the distribution of samples from the training set that fell at that leaf.

Ensemble Methods

An ensemble is a collection of weak, biased classifiers. Each member of an ensemble may be viewed as being a specialist on understanding some subset of the original training data. A decision is computed by averaging over all members of the ensemble. In the case of classification, this amounts to averaging over the probabilities that each class would occur given the input features. Ensembles can be homogenous or heterogeneous, that is, they can be composed of classifiers constructed using different paradigms or the same paradigm. In our case, we utilize a large set of decision trees only.

Ensembles are typically constructed by subsampling the original data in some way or by altering some parameter of the learning algorithm to induce a different classifier. In our work, we use a combination of boosting, feature selection, and random splitting. In short, boosting creates an ensemble by dynamically altering the probability of using a sample to construct the current classifier based on the previous classifier's ability to correctly identify that

sample. At the offset, all samples have an equal chance of being selected, but as subsequent classifiers are built the probability of using more difficult samples is increased, while easy–to–classify samples are de–emphasized. Thus, later trees focus on discriminating more difficult cases than earlier trees. Another weakness of trees is that due to their greedy nature, they tend to ignore many input features even though they might be useful in gauging fine distinctions between some subsets. In order to combat this effect, we randomly select different subsets of features to be used for constructing each tree in a bagging fashion, i.e., if we have F features, we create a bootstrap aggregate by picking F times, with replacement, from the set of possible features, remove the duplicates and are then left with a subset of about 63% of the original features. Finally, we use a popular trick in tree construction designed to increase variation –– rather than selecting the feature with the maximum significance, we uniformly randomly pick a winner from amongst the top N significant features. The combination of these approaches results in an ensemble that provides good coverage over the space of all possible classifiers.

Ensembles of Decision Trees vs. Neural Networks

This section presents the crux of the argument for switching from a neural net as the base recognizer to an ensemble of decision trees. When comparing these paradigms we were primarily looking at the accuracy and speed of the final classifier; however, there were other aspects of the ensemble that weren't evident at first, but came to light in our experimentation. Table 1 shows a one–to–one comparison of neural nets to tree ensembles.

	Neural Networks	Ensemble of Decision Trees
Training	Difficult to gauge parameters (number of nodes per layer) Computationally intense Lab techniques to attain high accuracy [1]	Simple parameter choices (number of trees to construct, stopping criteria) Taxing on memory but fairly fast Aggregation better than lab techniques
Speed of Evaluation	Constant time per segment Related to the dimensions of the net On the order of 50k floating point multiplies and adds Average of 60 seconds per image	Variable time per segment Related to depth of member trees On the order of 1k integer compares and branches Average of 17 seconds per image
Accuracy	Trained to 92.6% character accuracy Over all independent images: 89.06% character accuracy	Trained to 92.5% character accuracy Over all independent images: 90.68% character accuracy
Outputs	A vector of nonzero confidence values between 0.0 and 1.0	A vector of pseudo–probabilities, most of which are zero

Table 1. Comparison of Neural Net to Tree Ensemble

Table 1 demonstrates that an ensemble of trees can attain an equivalent accuracy to a neural net (in this case slightly higher) and requires a factor of three less time to evaluate. Something not obvious from this table is the fact that the form of the outputs has significant repercussions in the context of the Viterbi beam search. That method uses the outputs from the classifier to maintain a ranked list of all of the possible outcomes as the segments are considered by the dynamic programming. Since the neural networks provide confidences that are rarely zero and do not have the nice properties of probabilities, many possible results of the search are considered as viable alternatives. The outputs of the ensemble, however, *do* have all of the properties of probabilities and thus drastically reduce the breadth of the search significantly –– zero entries can be immediately dismissed.

Another nice property of ensembles is the fact that we can see their inner structure and vary the way we apply the member classifiers. To apply a neural net, we must always use the full net with all of its 100k floating point operations, but for ensembles, we have flexibility to apply and aggregate the member trees in different ways. We

can play games with the order and number of trees used for any given classification task. This kind of manipulation is an exercise in the tradeoff between speed and accuracy and involves an attempt to minimize the number of trees used for any given segment classification by utilizing prior statistics. We can do runs of independent training sets and gather statistics related to the top choice probability -- ultimately we can formulate lookup tables that will allow us to predict the probability that a given result from T trees is correct given the top choice hypothesis. In this way we can limit the number of trees used for any given classification sharply which will save significant amounts of time while still providing robust aggregated results. In preliminary experiments we were able to reduce processing time from 17 seconds per image down to 13.5 seconds while going from an overall accuracy of 90.68% down to 89.05%. In other words, for shaving 20% off the time, we only lose 1.5% accuracy. Work in this area is ongoing.

4. Results

The original page images used for training came from a data collection performed by SAIC. These consisted of 344 pages imaged at 600dpi. To obtain the test set used for the development of our Arabic system, we printed out all images and then created variants by copying each image at different levels of contrast from very dark to very light and then faxed these variants using 9 different fax machines at both high (200 x 200 dpi) and low (200x100 dpi) resolution. The resulting dataset consists of 3380 binary TIFF images. To obtain training data, we used our autotruthing approach detailed in [1]. We present the results using the old neural net approach and the most recent system which uses the tree ensembles. All told, about 1.5 million segments were used to train the neural net/ensemble.

In Table 2, LO and HI stand for low and high resolution fax, respectively. DEP and IND stand for *dependent* and *independent* images. The *dependent images* are those for which some portion of the characters have been used for training the net/ensemble, while the *independent images* are those reserved solely for the evaluation of the end–to–end system from which no data was ever used to train or tune the system. Obviously it is expected that results would be higher for the dependent images.

	NN Accuracy (%)	Ensemble Accuracy (%)	Images
OVERALL MEAN:	88.4	90.6	3380
LO MEAN:	86.6	89.1	1703
HI MEAN:	90.2	92.2	1677
IND MEAN:	89.1	90.7	722
DEP MEAN:	88.2	90.6	2658
IND LO MEAN:	87.5	89.3	360
IND HI MEAN:	90.6	92.1	362
DEP LO MEAN:	86.3	89	1343
DEP HI MEAN:	90.1	92.3	1315

Table 2. Comparison of Results

Some interesting points that are apparent from this table:

1. On the whole the system is more than 1.5% more accurate using the tree ensemble.
2. On the dependent images the ensemble is much better than the neural net due to the fact that the independent nature of the members of the ensemble can lead to more explicit memorization than a single neural net.

3. It *is* very interesting to observe that the difference between the dependent and independent results is much smaller for the ensemble than for the net indicating better generalization and less overtraining in the ensemble.

With respect to processing speed, we have seen the average time required to process each image on a 750 MHz Dell decrease from about 60 seconds per image to an average of 17 seconds.

5. Conclusion

This paper addressed enhancements made to our Arabic/Farsi recognition system that dramatically increased the speed of the system as well as increasing overall accuracy by about 1.5%. There is room for more improvement particularly in image filtering, and there are several experiments we will conduct to obtain better accuracy and speed from the decision tree ensemble approach.

6. References

[1] Gillies, A.M., Erlandson, E.J., Trenkle, J.M, Schlosser, S.G., "Arabic Text Recognition System", Proceedings of the Symposium on Document Image Understanding Technology, Annapolis, Maryland, 1999.

[2] Breiman, L., "Bagging Predictors", *Machine Learning*, 24(2):123–140, 1996.

[3] Freund, Y., "Boosting a Weak Learning Algorithm by Majority", *Information and Computation*, 121(2):256–285, 1995.

[4] Freund, Y., Schapire, R.E., "A Short Introduction to Boosting", *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, September, 1999.

[5] Sullivan, J., Langford, J., Caruana, R., Blum, A., "FeatureBoost: A Meta–Learning Algorithm that Improves Model Robustness", Proceedings of the Seventeenth International Conference on Machine Learning, Stanford University, June 29 – July 2, 2000.

[6] Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C. J, "Classification and Regression Trees", Wadsworth and Brooks/Cole Advanced Books and Software, Pacific Grove, California, 1984.

Experiments in Trilingual Cross-Language Information Retrieval

Giovanni B. Marchisio **Jisheng Liang**
Insightful Corporation, Suite 500, 1700 Westlake Ave N
Seattle, WA 98109-3044, USA

Abstract

Researchers have demonstrated that Latent Semantic Indexing (LSI) can emulate the process of semantic acquisition in humans. LSI is a computationally intensive algorithm for text retrieval that is based on matrix factorization (SVD). LSI is theoretically appealing, but has serious limitations. Typically, it takes several hours to index tens of thousands of documents. Lack of scalability limits the amount of information that is available for semantic learning, and places a limitation on the precision of the search. Lack of scalability has also prevented the extension of the technique to cross language retrieval. This is a field in which it holds much promise, as shown by early experiments at Bellcore.

A computational breakthrough allows us to perform a latent semantic analysis on hundreds of thousands of documents in a few seconds. Our implementation allows for the inclusion of a large training sample, and supports novel search operators and noun phrases. We can achieve large-scale cross-language information retrieval (CLIR) without translation or pivot concepts. The advantages of the approach are many. It does not require machine translation of entire document corpora across multiple languages. It therefore reduces indexing space and computational overhead while achieving better precision. In this scenario, the user searches information in the untranslated vernacular first, and then translates a small subset of relevant documents on demand. The method is easily customizable to vertical domains and handles multilingual concepts and acronyms effectively. In addition, we base our morphological analysis on an n-gram stemmer. This provides the advantage of portability across many languages and relative insensitivity to OCR errors.

1 Alternatives for Cross Language Information Retrieval

Using Machine Translation (MT) systems to translate the entire textual database from every single language to the user language is clearly unrealistic for large multilingual databases and for the Internet. Practical implementations of CLIR fall into two categories: knowledge-based and corpus-based approaches. A variation on the MT approach is multilingual interrogation. The idea is to translate the query from a source language to multiple target languages. One approach relies on interlingual dictionaries and knowledge bases. Translation into different languages must account for the fact that concepts expressed by a single term in one language sometime are expressed by distinct terms in another. For example, the term *tempo* in Italian corresponds to two different concepts in English: *time* and *weather*. The creation of interlingual pivot concepts requires the introduction of keyword tags that can discriminate between word meanings in different languages. This controlled vocabulary approach cannot account for all semantic variations in all languages, and often prohibits precise queries that are not expressed with the authorized keywords. A more data driven approach consists in deducing, during indexing, the keywords that would be supplied for a document from the terms contained in the full-text or summary. The creation of these directories is time consuming. It can be done either manually by a team of experts or by an automatic learning process from previously indexed documents. Again, linking different languages requires the introduction of a pivot language. Still another approach consists of combining machine translation methods with information retrieval methods. Such a hybrid approach has been adopted for instance by the European ESPRIT consortium in the project EMIR (European Multilingual Information Retrieval) [8]. This system uses three main tools: a linguistic processor that performs morphological and syntactic analysis; a statistical model that weights the query-document intersection; and a monolingual/multilingual query reformulation system. Tests with a trilingual (English, French and German) version of the Cranfield corpus show that multilingual interrogation is 8% better than using MT followed by monolingual interrogation.

The most interesting approach to corpus-based CLIR is an extension of LSI given by Dumais et al. [6, 7]. It is known as CL-LSI (Cross-Language LSI). In a vector space model, documents for which there exist a translation into multiple languages can be observed in language subspaces. CL-LSI approximates these

language subspaces by an eigenvector decomposition. By identifying and aligning principal axes for the various languages, the LSI algorithm correlates clusters of documents across the various language subspaces. The alignment is made possible by 1) cross-language homonyms and 2) the general statistics of term distributions in a reasonably large training collection. Testing on a sample of 2,500 paragraphs from the Canadian Parliament bilingual corpus (the Hansard collection), has demonstrated that cross-language retrieval with LSI yields precision recall curves that are equivalent to monolingual interrogation of a fully translated database.

2 LSI and Our Innovation

Latent semantic analysis is a promising departure from traditional Information Retrieval (IR) models. The method attempts to provide intelligent agents with a process of semantic acquisition. Researchers at Bellcore (Deerwester et al [4]; Berry et al [1]; Dumais et al [5]) have patented a computationally intensive algorithm known as Latent Semantic Indexing (LSI). This is an unsupervised classification technique based on a matrix factorization method. Cognitive scientists have show that the performance of LSI on multiple-choice vocabulary and domain knowledge tests emulates expert essay evaluations (Foltz et al [9]; Kintsch [11]; Landauer and Dumais [13]; Landauer et al. [14, 15]; Wolfe et al [18]). However, while theoretically appealing, this approach has serious limitations because it is based on a matrix factorization technique known as Singular Value Decomposition (SVD). With the conventional approach, it takes several hours, if not days, to index tens of thousands of documents. Lack of scalability limits the amount of information that is available for semantic learning. This in turn places a serious limitation on the precision of the search. Lack of scalability has also prevented the extension of the LSI technique to cross language semantic analysis, a field in which it holds much promise.

Our breakthrough comes from the realization that the linear decomposition underlying LSI is a special solution to the overdetermined decomposition problem

$$\boldsymbol{D} = \Psi \boldsymbol{A}$$
$$\boldsymbol{q} = \Psi \alpha$$

where $\boldsymbol{D}$ is a $m \times n$ term-document matrix, $\boldsymbol{q}$ is a query vector with m elements; Ψ is $m \times k$ and its columns are a dictionary of basis functions $\{\Psi_j, j=1,2,...,k<n\}$; $\boldsymbol{A}$ and α are a $k \times n$ matrix and k-length vector of transform coefficients, respectively. The columns of $\boldsymbol{A}$ are document transforms, whereas α is the query transform. Clustering documents around a query is a matter of comparing α and the corresponding column of $\boldsymbol{A}$ in a reduced transform space spanned by Ψ. Decomposition of an overdetermined system is not unique. The nonuniqueness leaves some open statistical issues. In the case of LSI, these are: (i) determining how many eigenvectors one should retain in the truncated expansion for the indices; (ii) determining subspaces in which latent semantic information can be linked with query keywords; and (iii) efficiently comparing queries to documents (i.e., finding near neighbors in high-dimension spaces).

LSI transforms the matrix $\boldsymbol{D}$ as $\boldsymbol{D'} = \boldsymbol{U_k \Lambda_k V_k^T}$ where $\Lambda = \text{diag}(\lambda_1,...,\lambda_k)$, and $\{\lambda_i, i=1,k\}$ are the first k ordered singular values of $\boldsymbol{D}$, and the columns of $\boldsymbol{U_k}$ and $\boldsymbol{V_k}$ are the first k orthonormal eigenvectors associated with $\boldsymbol{DD^T}$ and $\boldsymbol{D^TD}$ respectively. From this we see that $\Psi=(\boldsymbol{U\Lambda})_k$ and $\boldsymbol{A}=\boldsymbol{V_k^T}\{A_j, j=1,2,...,n\}$. QR bases (Booker et al [2]), wavelet decomposition and atomic decomposition by basis pursuit (Chen et al [3]) and wavelet packets approximations to SVD (Wickerhauser [17]) provide a number of computationally efficient alternatives for decomposing an overdetermined system into an optimal superposition of dictionary elements. None of these decompositions will ever maximize the variance of the data cloud, but they can come close.

Insightful Corp. has three patents pending on an algorithmic breakthrough in Latent Semantic Analysis (LSA). Seen from the viewpoint of numerical analysis, LSI is a special case of regression analysis. Our technique generalizes the idea to sets of basis functions other than eigenvectors. Therefore we call our technique LSR for Latent Semantic Regression. Monolingual precision recall experiments on 742,000 documents from the TREC corpus confirm the validity of our approach. We do well on all counts: precision, order of relevance, quality of results, speed (down from 18hrs to 30 sec) and scalability (up from thousands to millions of documents). LSR outperforms previous LSI scores by 25% precision at lower levels of recall. Previous experiments with LSI were seriously limited by LSI's lack of scalability. In fact, matrix factorization by singular value decomposition could only be performed on 10% or abbot 70,000 of the available documents; the remaining 90% were folded in. Historically, lack of scalability has severely limited the precision of LSI. The model can amplify subtle positive and negative relationships between concepts in a term-document matrix. The greater the size of the matrix that it can fit in memory and decompose, the more precise are the logical links that it can establish. Figures 1 shows CPU times for LSR on a rather modest computational platform (a SUN Ultra 60).

Figure 1: Query times vs. number of documents (for single processor Sun Ultra 60).

3 CL-LSR

To recapitulate, our implementation is similar to CL-LSI, but is uses a much faster matrix decomposition algorithm, instead of SVD, to perform matrix reduction. In addition to performing cross language document retrieval, our search technology also provides automatic tools for accelerating the construction of a multilingual lexicon, and for extracting terminology from multilingual corpora of texts.

Similar to CL-LSI, in CL-LSR the information matrix is replaced by:

$$D = \left[\begin{array}{c|ccc} \boldsymbol{R}^E & \boldsymbol{T}^E & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{R}^F & \boldsymbol{0} & \boldsymbol{T}^F & \boldsymbol{0} \\ \boldsymbol{R}^I & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{T}^I \end{array}\right]$$

where the superscripts identify foreign language document partitions in the term document matrix. In this example E stands for English, F for French, and I for Italian. The partition on the left represents blocks of reference documents ($\boldsymbol{R}$). These are documents for which there is a translation in every language. The diagonal blocks on the right are all other target ($\boldsymbol{T}$) multilingual documents to be searched.

In addition to retrieving multilingual documents with a monolingual query, we propose a method to align multilingual text along lexicographical axes. To accomplish this, we use a CS decomposition. This is an SVD-like decomposition that is handy when comparing subspaces. Let IR^n denote the vector space of real n-vectors, then the SVD of an $m \times n$ multilingual document matrix $\boldsymbol{D} \in \mathrm{IR}^n$ is

$$D = \left[\begin{array}{c|ccc} \boldsymbol{R}^E & \boldsymbol{T}^E & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{R}^F & \boldsymbol{0} & \boldsymbol{T}^F & \boldsymbol{0} \\ \boldsymbol{R}^I & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{T}^I \end{array}\right] \approx \begin{bmatrix} \boldsymbol{U}_k^E \\ \boldsymbol{U}_k^F \\ \boldsymbol{U}_k^I \end{bmatrix} \Lambda_k \boldsymbol{V}_k^T$$

Note that in this formulation, the left orthogonal matrix of term projections $\boldsymbol{U}$ is split into three subspaces corresponding to the three languages. On the other hand, the right orthogonal matrix of document projections $\boldsymbol{V}$, is shared by all three languages, enabling cross language retrieval. There are several important orthogonal projections associated with the SVD, and there is a one-to-one correspondence between orthogonal projections and subspaces in matrix algebra. This principle leads to the notion of measuring distances between subspaces in terms of the orthogonal projections that span them. Suppose that T_1 and T_2 are

subspaces of IR^n and that $\dim(T_1) = \dim(T_2)$, then we can measure the distance between these two spaces as

$$Dist(T_1,T_2) = ||P_1-P_2||_2$$

where P_i is the orthogonal projection onto T_i. Consider for simplicity the case of two unit 2-norm vectors (or one-dimensional subspaces) x and y of T_1 and T_2. The vectors can be expressed as

$$\boldsymbol{x} = \begin{bmatrix} \cos(\theta_1) \\ \sin(\theta_1) \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} \cos(\theta_2) \\ \sin(\theta_2) \end{bmatrix}$$

with projection operators given by the orthonormal matrices

$$\boldsymbol{U} = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) \\ \sin(\theta_1) & \cos(\theta_1) \end{bmatrix}$$
$$\boldsymbol{V} = \begin{bmatrix} \cos(\theta_2) & -\sin(\theta_2) \\ \sin(\theta_2) & \cos(\theta_2) \end{bmatrix}$$

A little algebra shows that the distance between projections, $||P_1-P_2||_2$ is

$$\boldsymbol{U}^T (xx^T - yy^T)\boldsymbol{V} = \begin{bmatrix} 0 & \sin(\theta_1 - \theta_2) \\ \sin(\theta_1 - \theta_2) & 0 \end{bmatrix}$$

Therefore, since $\boldsymbol{U}$ and $\boldsymbol{V}$ are norm preserving orthonormal matrices, $||\boldsymbol{U}^T (xx^T-yy^T)\, \boldsymbol{V}||_2 = ||xx^T-yy^T||_2 = |\sin(\theta_1-\theta_2)| = Dist(T_1,T_2)$. We can extend this geometrical interpretation to higher dimensions. The CS decomposition theorem states (Golub and Van Loan [10]) that if

$$\boldsymbol{Q} = \begin{matrix} \begin{bmatrix} \boldsymbol{Q_{11}} & \boldsymbol{Q_{12}} \\ \boldsymbol{Q_{21}} & \boldsymbol{Q_{22}} \end{bmatrix} & \begin{matrix} k \\ j \end{matrix} \\ \begin{matrix} k & j \end{matrix} & \end{matrix}$$

is orthogonal, then there exist orthogonal matrices $\boldsymbol{U_1}$, $\boldsymbol{V_1} \in IR^{k\times k}$ and $\boldsymbol{U_2}$, $\boldsymbol{V_2} \in IR^{j\times j}$ such that

$$\begin{bmatrix} \boldsymbol{U}_1 & 0 \\ 0 & \boldsymbol{U}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{Q}_{11} & \boldsymbol{Q}_{12} \\ \boldsymbol{Q}_{21} & \boldsymbol{Q}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{V}_1 & 0 \\ 0 & \boldsymbol{V}_2 \end{bmatrix}$$
$$= \begin{bmatrix} \boldsymbol{I}_{k-j} & 0 & 0 \\ 0 & \boldsymbol{C} & \boldsymbol{S} \\ 0 & -\boldsymbol{S} & \boldsymbol{C} \end{bmatrix}$$

where

$\boldsymbol{C} = diag(c_1,\ldots,c_j) \in IR^{j\times j}$ $\quad c_i = cos(\theta_i)$

$\boldsymbol{S} = diag(s_1,\ldots,s_j) \in IR^{j\times j}$ $\quad s_i = sin(\theta_i)$

A corollary to this theorem states that

$$Dist(T_1,T_2) = \sqrt{1-\lambda^2_{\min}(\boldsymbol{U}_1^T \boldsymbol{V}_1)}$$

where $\boldsymbol{U}=[\boldsymbol{U_1},\boldsymbol{U_2}]$ *and* $\boldsymbol{V}=[\boldsymbol{V_1},\boldsymbol{V_2}]$ are orthogonal projection matrices for the subspaces $T_1=Q_{11}$ and $T_2=Q_{22}$, and λ_{min} is the smallest eigenvalue for a decomposition of Q_{11}. We can use CS decomposition to measure distances between the trilingual orthogonal transform spaces $\boldsymbol{U^E_k}$, $\boldsymbol{U^F_k}$, $\boldsymbol{U^I_k}$.

4 Generalized Knowledge Based Training

This subsection is a corollary to the previous one. We generalize the idea of using a training set for cross language retrieval to the problem of searching databases where information is diluted or not reliable enough to allow the creation of robust semantic links. Our solution is to load the left partition of the term document matrix with a large amount of concurrent documents from reliable sources. The information matrix for this problem looks like:

$$\boldsymbol{D} = \left[\boldsymbol{D^R} \mid \boldsymbol{D^S}\right] \approx \boldsymbol{U}_k \Lambda_k \left[\boldsymbol{V^R} \mid \boldsymbol{V^S}\right]^T_k$$

where the superscripts R and S stand respectively for reference and search document sets. In the cross language retrieval scenario, the right orthogonal matrix $\boldsymbol{V}$ of document projections is shared by multiple languages. In the knowledge-based model, it is the left orthogonal matrix $\boldsymbol{U}$ of term projections that is shared by all documents. Retrieval is on the S document set only. The R set is invisible to the user, but it is where most of the reliable semantic links for the search in S are established. This type of knowledge based training is inexpensive, since it requires no expert intervention and can be quickly tailored to many different domains. In vertical search applications, you can improve the performance of latent semantic search by loading the left partition of the term document matrix with domain specific content. Our benchmarks show that, on a modest computational platform, the increase in query response time is only 3 seconds per 70,000 additional new entries of knowledge training documents.

5 Morphological Model

We have developed a conflation technique based on the statistics of n-grams. We use a weighted similarity measure to produce a similarity matrix. Depending upon the language, we cluster terms using a single or

complete link clustering method. The first is more suitable for modeling elongated ellipsoidal clusters, that we observe in the some of the Romantic language; the second is more appropriate for modeling small, tightly bound clusters of terms, as observed in English. The advantage of our morphological model is that it is relatively language independent and robust to OCR errors.

6 Examples

We have prototyped cross language retrieval by augmenting the simple term-document matrix in Table 1 with French and Italian documents. The new term document matrix (not shown in detail because of its size) consists of a trilingual translation of the keywords (rows) in the fifteen pseudo-documents (columns) of Table 1, plus three sets of new documents for each of the three languages: English, French, and Italian. The translation in our test example is noisy. It allows for semantic ambiguities and translator preferences that may result when translating across languages. For instance, *Tempest* in English splits into *Tempête* and *orage* in French; *playwright* in English may split into *tragediografo* and *drammaturgo* in Italian. On the other hand, the keyword *theatre* has the same spelling in English and French. In our experiment, we apply LSR to the multilingual term document matrix, and search only the target documents. Table 2 shows the list of documents returned in response to the English language query *theatre, comedy*. As before we keep two separate ranked lists: a list of direct hits, and a list of latent hits. Foreign language documents are found prevalently in the latent list. Some French documents appear in the direct hit list because they contain one of the keywords in the query, *theatre*. A by-product of our approach to cross language retrieval is the alignment of semantic axes for the English, French and Italian subspaces (seen under the heading Keyword Suggestions and Relative Weights in Table 2). Here we render distances between keywords in the three languages as the absolute weights that each keyword should have in a fully multilingual query. That is, in response to the monolingual query *theatre, comedy* the engine retrieves multilingual documents, and also suggests to the user the foreign language keywords and relative weights that a fully multilingual query should have. Note that the keyword *theatre* is weighted twice as much as the Italian *teatro*, since it applies to twice as many languages (English and French). The keyword *Shakespeare* dominates the latent semantic space since it is the same in all languages.

Our main demo employs a French/Spanish/English parallel corpus consisting of 11,000 cases from the International Labor Organization "Official Bulletin, B Series": "Reports of the Committee on Freedom of Association of the Governing Body of the ILO and related material 1984-1989". We use 6,000 documents for training and we search the remaining 5,000. We assume that the 5,000 documents in the search partitions are lexically independent. Preliminary results with a trilingual corpus of 11,000 documents are extremely encouraging. Figures 2 through 4 show an example of cross-lingual retrieval on the input noun phrase "sick-leave". Notice in particular how in response to the English input, we can successfully retrieve multiple renderings in Spanish: "incapacidad laboral derivada de enfermedad cómun", "licencia por enfermedad" and "riesgo de enfermedad". We will discuss this and other examples involving acronyms and more complex queries in our presentation.

7 Conclusion

We are developing a practical tool for CLIR that does not require MT and minimizes storage overhead for multilingual databases. It is easily customizable to vertical domains and, like CL-LSI, employs an unsupervised approach to language learning. We have proven that it can handle acronyms and idiomatic expressions across languages. Our statistical morphological model can handle OCR errors. We plan to participate in future CLEF competitions. We anticipate that our results for cross lingual retrieval will be greatly superior to the already encouraging results reported by Dumais et al. [7]. Lack of scalability limited the training set to about 1,000 paragraphs from the Hansard collection (the bilingual Canadian Parliament Proceedings). The scalability of our approach allows us to use hundreds of thousands of documents from the same bilingual collection as the training set. Also, the speed of our LSR algorithm depends more critically on the number of documents than on the number of terms. In computational terms, this means that we can add many languages at little extra price. With many languages, the fact that concepts expressed by a single term in one language sometime are expressed by distinct terms in another language plays to our advantage. For instance, multiple occurrences of the Italian keyword *tempo*, which translates as both *time* and *weather* in English, will be semantically linked to either concept in English and many other languages, enabling unambiguous retrieval of relevant documents.

References

[1] Berry, M., S. Dumais, G. O'Brien, Using linear algebra for intelligent information retrieval, *SIAM Review*, Vol. 37, No. 4, pp. 553-595, December 1995.

[2] Booker, A., Condliff, M., Greaves, M., Holt, B., Kao, A., Pierce, D., Poteet, S. and Wu, J., Visualizing text data sets, Computing in Science and Engineering, 26-35, July/August 1999.

[3] Chen, S., D. Donoho, M. Saunders, Atomic decomposition by basis pursuit, *Stanford*

University, Department of Statistics Technical Report, February 1996.

[4] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing By Latent Semantic Analysis. Journal of the American Society For Information Science, 41, 391-407.

[5] Dumais, S.T., Platt, J., Heckerman, D., and Sahami, M., Inductive Learning Algorithms and Representations for Text Categorization, *Proceedings of ACM-CIKM98*, Nov. 1998.

[6] Dumais, S. T., Landauer, T. K. and Littman, M. L. (1996) "Automatic cross-linguistic information retrieval using Latent Semantic Indexing." In SIGIR'96.

[7] Dumais, S. T., Letsche, T. A., Littman, M. L. and Landauer, T. K. (1997) "Automatic cross-language retrieval using Latent Semantic Indexing." In AAAI Spring Symposuim on Cross-Language Text and Speech Retrieval, March 1997.

[8] EMIR. Final report of the EMIR project number 5312. Technical report, European Multilingual Information Retrieval Consortium For the Commission of the European Union, Brussels, October 1994.

[9] Foltz, P. W., Kintsch, W.,& Landauer, T. K. (1998). The measurement of textual Coherence with Latent Semantic Analysis. Discourse Processes, 25, 285-307.

[10] Golub, G. and Van Loan, C. (1996), *Matrix computations*, third edition, The Johns Hopkins University Press Ltd., London

[11] Kintsch, W. Metaphor comprehension: A computational theory. Psyhonomic Bulletin and Review, (in press)

[12] Landauer, T. K., Foltz, P. W., & Laham, D. (1998). Introduction to Latent Semantic Analysis. Discourse Processes, 25, 259-284.

[13] Landauer, T. K. and Dumais, S. T. (1997). A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. Psychological Review, 104, 211-240.

[14] Landauer, T. K., Laham, D., & Foltz, P. W., (1998). Learning human-like knowledge by Singular Value Decomposition: A progress report. In M. I. Jordan, M. J. Kearns & S. A. Solla (Eds.), Advances in Neural Information Processing Systems 10,(pp. 45-51). Cambridge: MIT Press.

[15] Landauer, T. K., Laham, D., Rehder, B., & Schreiner, M. E., (1997). How well can passage meaning be derived without using word order? A comparison of Latent Semantic Analysis and humans. In M. G. Shafto & P. Langley (Eds.), Proceedings of the 19th annual meeting of the Cognitive Science Society (pp. 412-417). Mawhwah, NJ: Erlbaum.

[16] Waltz, D.L., and Pollack, J.B., massively parallel parsing: a strong interactive model of natural language interpretation, Cognitive Science, 9, pp. 51-74, 1985.

[17] Wickerhauser, M.V, Adapted Wavelet Analysis from theory to software, 1994.

[18] Wolfe, M. B., Schreiner, M. E., Rehder, B., Laham, D., Foltz, P. W., Kintsch, W., & Landauer, T. K. (1998). Learning from text: Matching readers and text by Latent Semantic Analysis. Discourse Processes, 25, 309-336.

Table 1: Example of a document index: a simple term-document matrix shows word counts for 16 keyword terms (rows) in 15 documents (columns).

	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15
comedy	3	0	0	0	3	0	3	3	3	3	3	0	0	0	3
theatre	5	0	0	0	5	0	5	5	5	5	5	0	0	0	5
Shakespeare	0	0	0	0	0	0	3	1	0	0	4	0	0	0	0
Tempest	1	0	0	0	0	0	1	1	0	1	1	0	1	0	1
playwright	1	0	0	0	0	0	1	1	1	1	1	0	0	0	0
London	1	0	1	1	0	1	1	1	1	1	1	1	0	0	0
ocean	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0
Thames	0	0	1	1	0	0	0	0	0	1	1	1	1	0	0
tea	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0
bridge	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
knight	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0
Agincourt	1	1	0	0	0	0	1	0	1	1	1	0	0	0	0
sword	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
armour	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
gate	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
pennants	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 2: prototyping cross language retrieval with the LSR algorithm.

Input Monolingual Query and Keyword Weights

theatre	comedy
1	1

Direct Document Hits

rank	ID	Keyword content
0.329	E3	comedy playwright Thames theatre
0.329	E4	comedy playwright Thames theatre
0.271	E1	comedy Tempest playwright gate Shakespeare theatre
0.189	E2	comedy Tempest London Shakespeare theatre
0.126	E10	comedy Tempest playwright London Agincourt theatre
0.05	F9	dramaturge Londres mer theatre
0.023	F10	Tempete dramaturge Londres Tamise theatre
0.006	E5	London Thames tea bridge theatre

Latent Document Hits

rank	ID	Keyword content
0.201	I7	commedia teatro Tempesta drammaturgo Tamigi tragediografo rappresentazione
0.168	I1	commedia teatro Tempesta drammaturgo Shakespeare Agincourt
0.161	I10	commedia teatro drammaturgo Londra Tamigi Shakespeare
0.16	I3	commedia teatro Tempesta drammaturgo Londra Tamigi rappresentazione Shakespeare
0.151	I4	commedia teatro drammaturgo Londra rappresentazione
0.099	F1	comedie Tempete dramaturge Tamise auteur Shakespeare theatre
0.098	F2	comedie Tempete dramaturge Londres Shakespeare Agincourt theatre
0.086	E8	Tempest playwright London
0.084	I2	commedia teatro Tempesta drammaturgo Londra Tamigi tragediografo Shakespeare
0.08	I5	commedia teatro drammaturgo Londra tragediografo Shakespeare
0.065	F7	comedie Tempete dramaturge Londres Shakespeare Agincourt theatre
0.043	I6	commedia teatro Tempesta drammaturgo Tamigi tragediografo

Direct Keyword Suggestion and Relative Weights

theatre	teatro	comedy	commedia	comedie
1	0.486	0.384	0.326	0.266

Latent Keyword Suggestion and Relative Weights

Shakespeare	playwright	drammaturgo	Tamigi	tragediografo
0.251	0.127	0.121	0.045	0.044

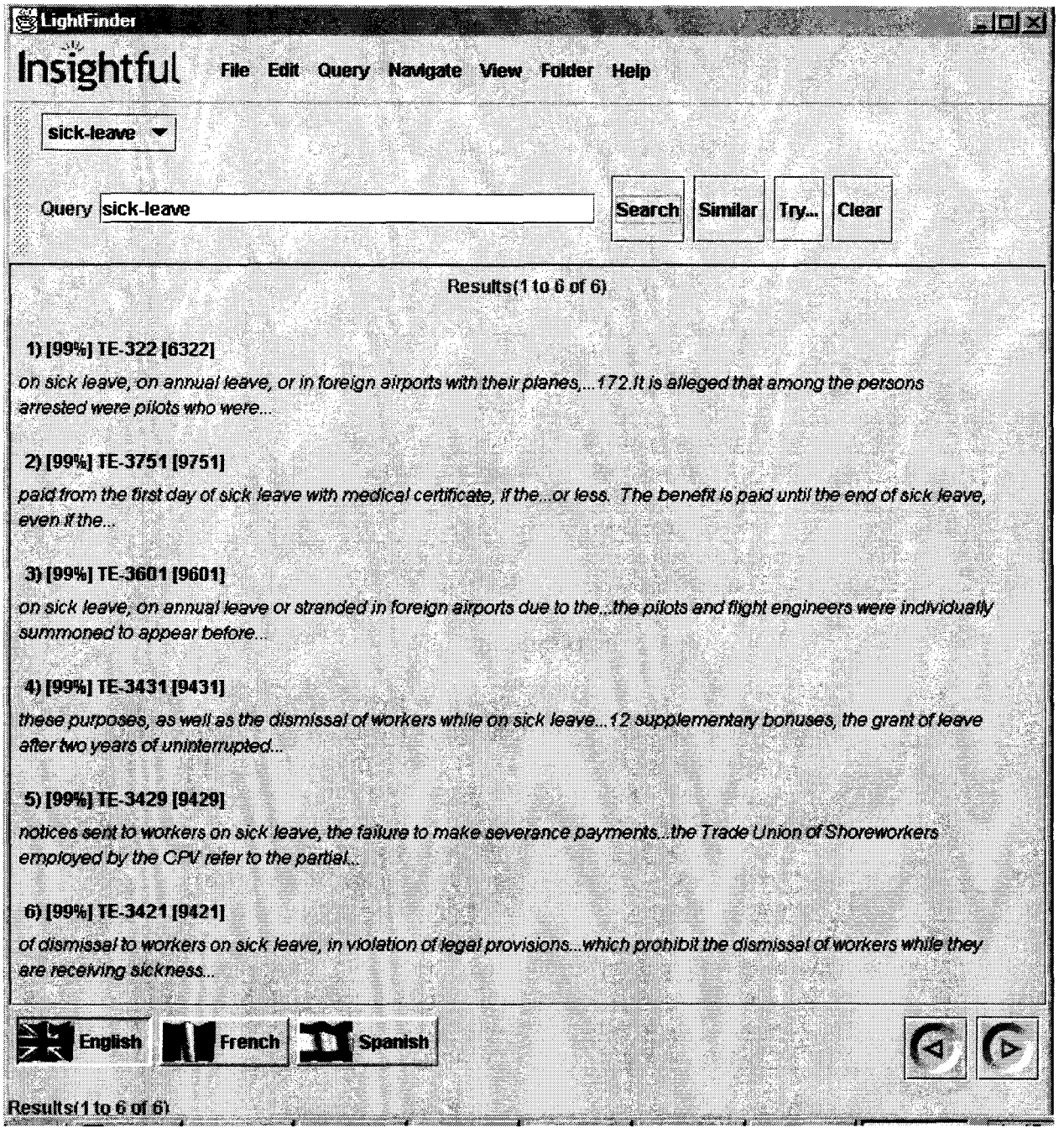


Figure 2: input English query and retrieved English documents.

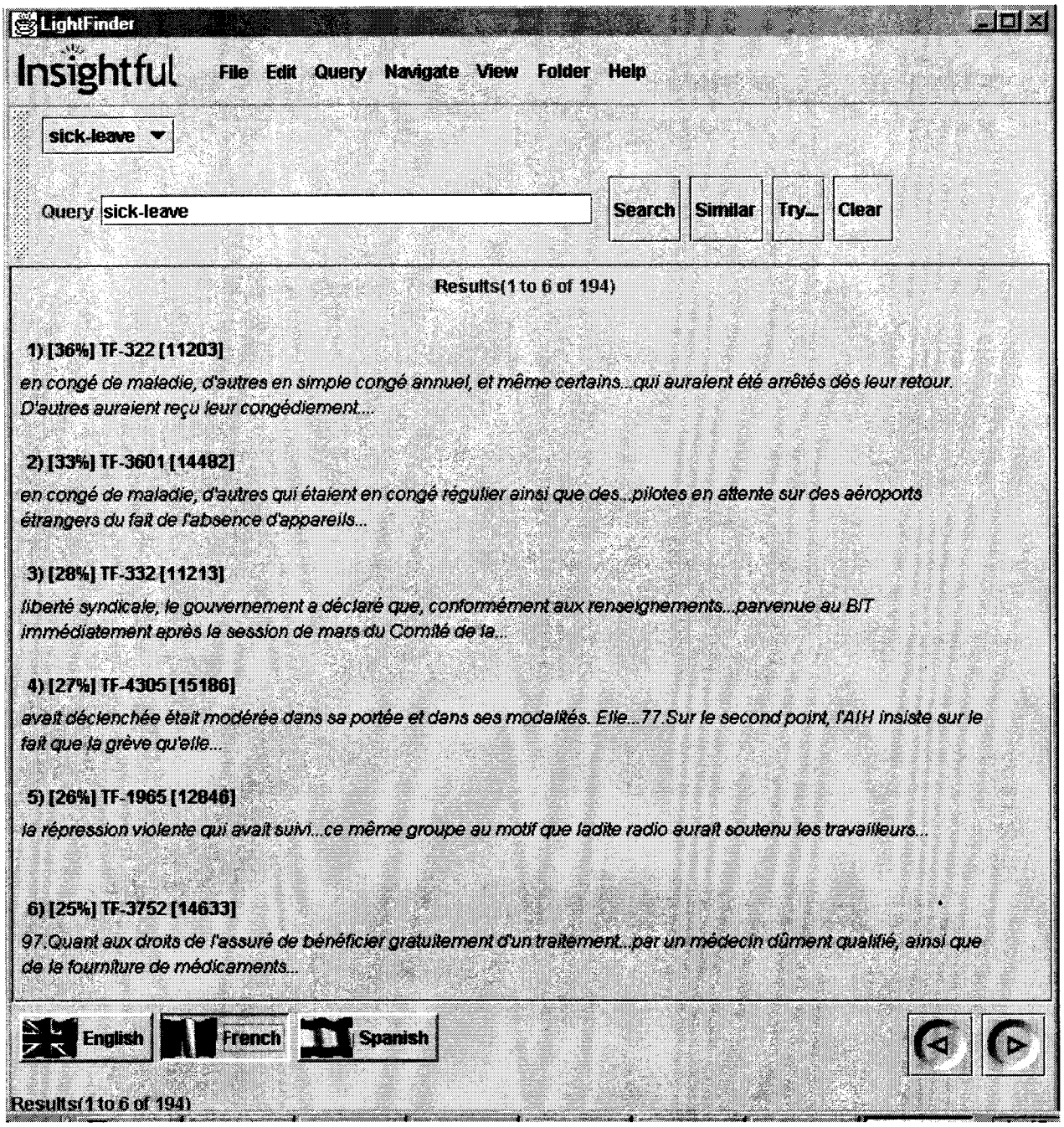


Figure 3: retrieved French documents.

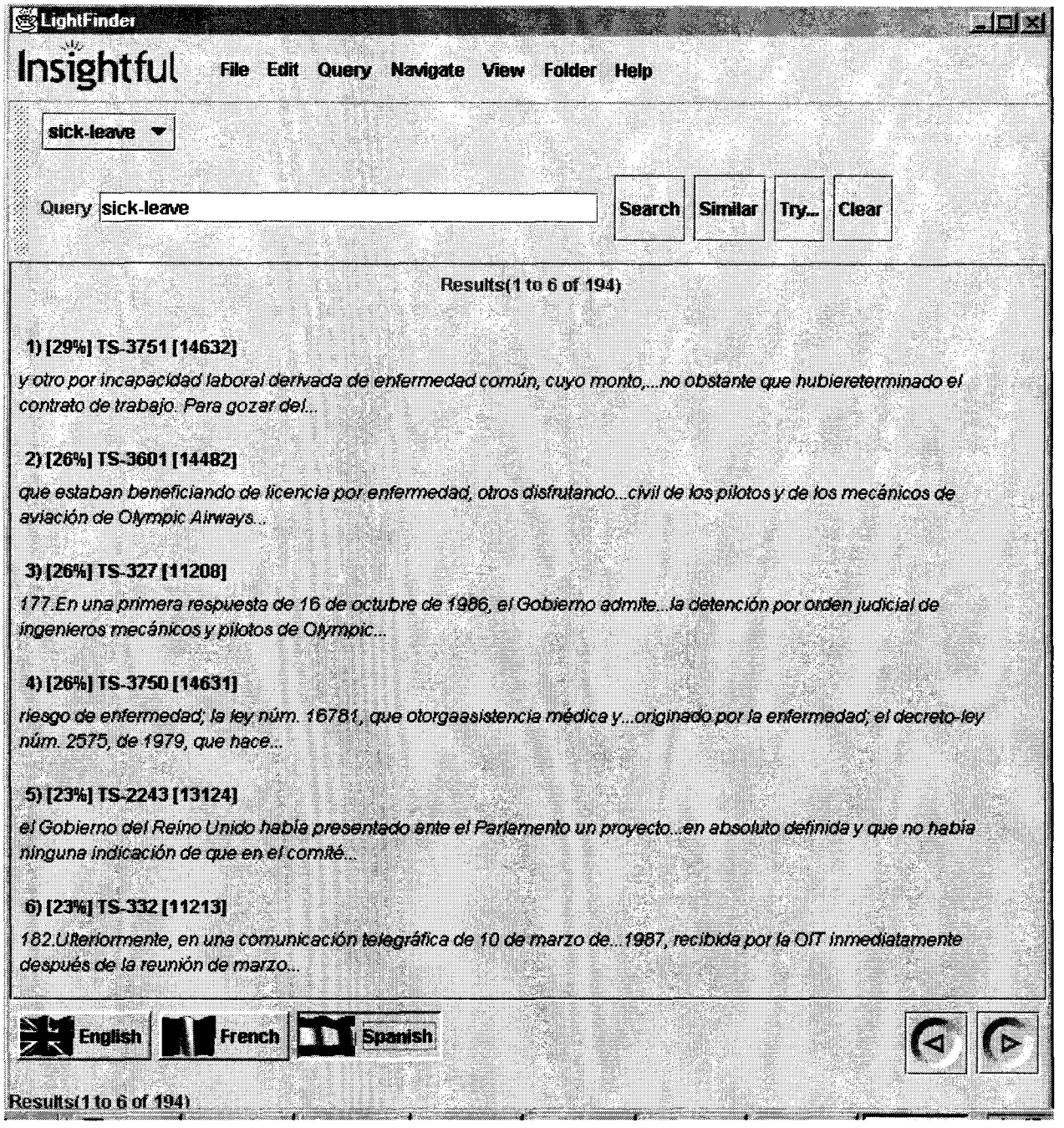


Figure 4: retrieved Spanish documents.

Page Analysis and Classification

Binary Document Image Using Similarity of Multiple Texture Features

David Doermann and Jian Liang
Laboratory for Language and Media Processing
Institute for Advanced Computer Studies
University of Maryland, College Park, MD 20742

Abstract

The processes of page segmentation and zone classification are essential elements in a complete document analysis system. They provide a physical segmentation of page layout into regions that are classified by type and ultimately passed to document analysis processes such as OCR, graphics recognition, compression, or logical analysis, for example. The challenges of page segmentation deal primarily with the processing of mixed content pages where regions are either spatially close but have different physical attributes such as text regions with different sizes, styles, or orientations, or are spatially close but are in fact different classes such as text vs. noise or clutter. In this paper, we present preliminary results that will be used to address the problem of combining and refining results from either multiple page segmenters or the parameterization of a single segmenter. The work aims to define measures of quality of page segmentation results and to provide ways of determining if two samples should ultimately belong to the same zone or be kept separate. In this paper, we explore the use of texture-like features to judge similarity between samples as a baseline for combining results for different classifiers.

1 Introduction

Page segmentation has been addressed extensively over the past 20 years of document analysis research and many algorithms have evolved which address the fundamental aspects of segmentation. Both bottom-up [1,2,3,4,5,6] and top-down [7,8,9,10,11] methods have been proposed and many work well for classic layouts. Nevertheless, there remain clear challenges where the current state-of-the-art systems fail to produce adequate results. A case can be made that we should not expect a single optimal algorithm that can handle all cases, and to some extent we agree. There are many situations were ambiguities exist and depending on the domain, different interpretations of "correctness" can be made. We have found, however, that a great deal of ambiguity can be resolved from context. If we can provide tools that can bootstrap themselves into acquiring this context, and if we have implicit measures of what it means to be correct in the absence of ground truth, we have a greater chance of providing generic tools for page segmentation.

The problem we are ultimately considering centers on the ability to provide an improved segmentation of noisy and degraded documents. When regions are uniform and spatially disjoint and there is clearly background between them, algorithms exist that will work reasonably well. There are a number of situations, however, that degrade the performance of almost all segmentation schemes. For example:

- **Noise:** The introduction of various noise and clutter elements in a document image can interfere both with our ability to accurately delineate uniform regions and with our ability to classify them consistently.

- **Spatial Overlap**: Humans expect regions with different content to be segmented separately, independent of proximity. Most automated segmentation algorithms, however, are based at least in part on some spatial separation. When regions touch or overlap, we can no longer rely on proximity. Previous work using texture-based segmentation has addressed these problems [4, 18, 19].

- **Different perceptual attributes:** Correct segmentation often requires that regions with different attributes be segmented separately. For example if we have a line of text that is bold, we may expect it to be segmented separately even if it is in close proximity to a non-bold region. Similar constraints apply to handwritten text, text with different fonts or sizes and text at different orientations.

- **Different scales**: For systems that are parameterized, variations in character, word and line spacing cause problems at decision boundaries. Humans make local decisions based on perceptual grouping and logical interpretation. Providing such capabilities for a segmentation scheme is difficult.

- **Desire for logical segmentation:** One of the broadest challenges is to draw the line between physical and logical characteristics of a page. We cannot expect to segment regions whose logical differences are not reflected in their physical attributes. It is therefore extremely important that segmentation routines consider even the most minor differences in physical attributes that may indicate that they come from different regions.

Many solutions have been proposed for dealing with these problems. Some researchers advocate training the system to provide a model optimization and use the recovered parameters to get better performance. This may work in some domains where the variation is fairly limited, but it often requires extensive ground truth. Other approaches advocate over- or under-segmentation, and allowing for higher-level processes to resolve ambiguity. In these cases, consistency is the key.

1.1 The Problem

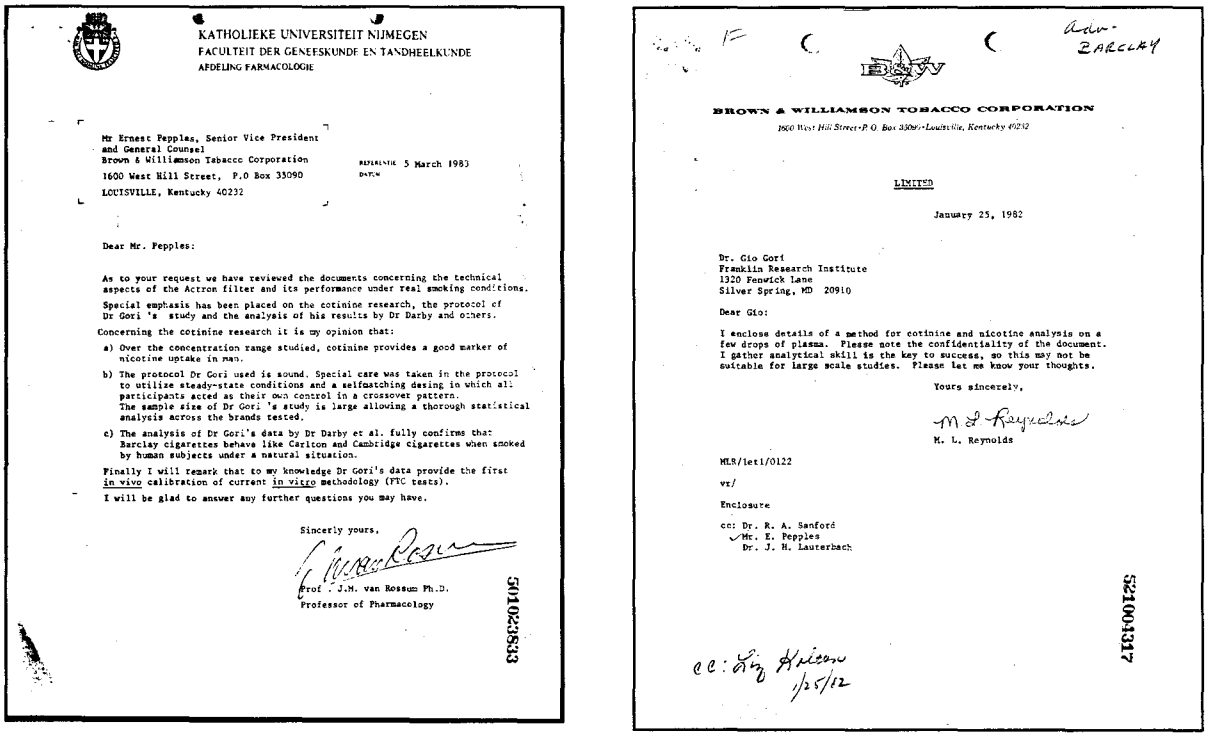
As previously mentioned, the traditional approach to the problem of page segmentation is either to provide decomposition of the document image into spatially disjoint regions, or to use both spatial measures and classification, to jointly segment and classify regions as

text, graphics and image. Some approaches have also tried to further segment the document regions based on text properties or to segment overlapping regions. Our work does not try to reinvent these processes, but rather attempts to build on them.

The basic problem we are trying to address is: Given a segmentation or set of segmentations from either a collection of algorithms or a re-parameterization of the same algorithm, can we:

1) *provide a (possibly relative) measure of the quality of each segmentation and locally choose the best one, and*
2) *provide a way of telling whether a given region is uniform and/or whether two regions belong to the same population.*

If we can adequately address these issues, we will be able to consider how to combine the results of multiple classifiers and how to refine the results of a given classifier to obtain more consistent segmentation results.



KATHOLIEKE UNIVERSITEIT NIJMEGEN
FACULTEIT DER GENEESKUNDE EN TANDHEELKUNDE
AFDELING FARMACOLOGIE

Mr Ernest Pepples, Senior Vice President
and General Counsel
Brown & Williamson Tobacco Corporation
1600 West Hill Street, P.O Box 35090
LOUISVILLE, Kentucky 40232

5 March 1983

Dear Mr. Pepples:

As to your request we have reviewed the documents concerning the technical aspects of the Actron filter and its performance under real smoking conditions.

Special emphasis has been placed on the cotinine research, the protocol of Dr Gori 's study and the analysis of his results by Dr Darby and others.

Concerning the cotinine research it is my opinion that:

a) Over the concentration range studied, cotinine provides a good marker of nicotine uptake in man.

b) The protocol Dr Gori used is sound. Special care was taken in the protocol to utilize steady-state conditions and a selfmatching desing in which all participants acted as their own control in a crossover pattern. The sample size of Dr Gori 's study is large allowing a thorough statistical analysis across the brands tested.

c) The analysis of Dr Gori's data by Dr Darby et al. fully confirms that Barclay cigarettes behave like Carlton and Cambridge cigarettes when smoked by human subjects under a natural situation.

Finally I will remark that to my knowledge Dr Gori's data provide the first in vivo calibration of current in vitro methodology (FTC tests).

I will be glad to answer any further questions you may have.

Sincerly yours,

Prof. J.H. van Rossum Ph.D.
Professor of Pharmacology

501023533

Adv-
BARCLAY

BROWN & WILLIAMSON TOBACCO CORPORATION
1600 West Hill Street•P.O. Box 35090•Louisville, Kentucky 40232

LIMITED

January 25, 1982

Dr. Gio Gori
Franklin Research Institute
1320 Fenwick Lane
Silver Spring, MD 20910

Dear Gio:

I enclose details of a method for cotinine and nicotine analysis on a few drops of plasma. Please note the confidentiality of the document. I gather analytical skill is the key to success, so this may not be suitable for large scale studies. Please let me know your thoughts.

Yours sincerely,

M. L. Reynolds

MLR/let1/0122

vr/

Enclosure

cc: Dr. R. A. Sanford
Mr. E. Pepples
Dr. J. H. Lauterbach

cc: Liz Hixson
1/25/82

521004317

Figure 1: Examples of Mixed Content Documents

Figure 1 shows two pages that contain regions of different orientations, classes, sizes and styles. We as humans naturally segment these regions with respect to perceptual

properties, grouping content that is visually similar even if it is spatially disjoint (such as the noise regions) and splitting regions which have different perceptual properties.

Figure 2 shows examples of segmentation results from three different published algorithms. Note that in all three segmentations some portions of the segmentation are valid, but no single algorithm has all segments correct. Ultimately our goal is to combine the results of these segmentations to provide an "improved result". For this, we need to be able to tell which local segmentation is correct, if a given region should be split or multiple regions should be merged.

In this paper, we will focus on our preliminary results on the second of the two problems stated above: how we can measure the similarity of two arbitrary regions, and determine if they are in fact from the same population. Note this is vastly different from attempting to use features to classify the regions as text, graphics or image, as has been previously done with texture features. Here we expect a much finer level of discrimination so, for example, we can distinguish between different text fonts, styles and sizes, for example.

1.2 Approach

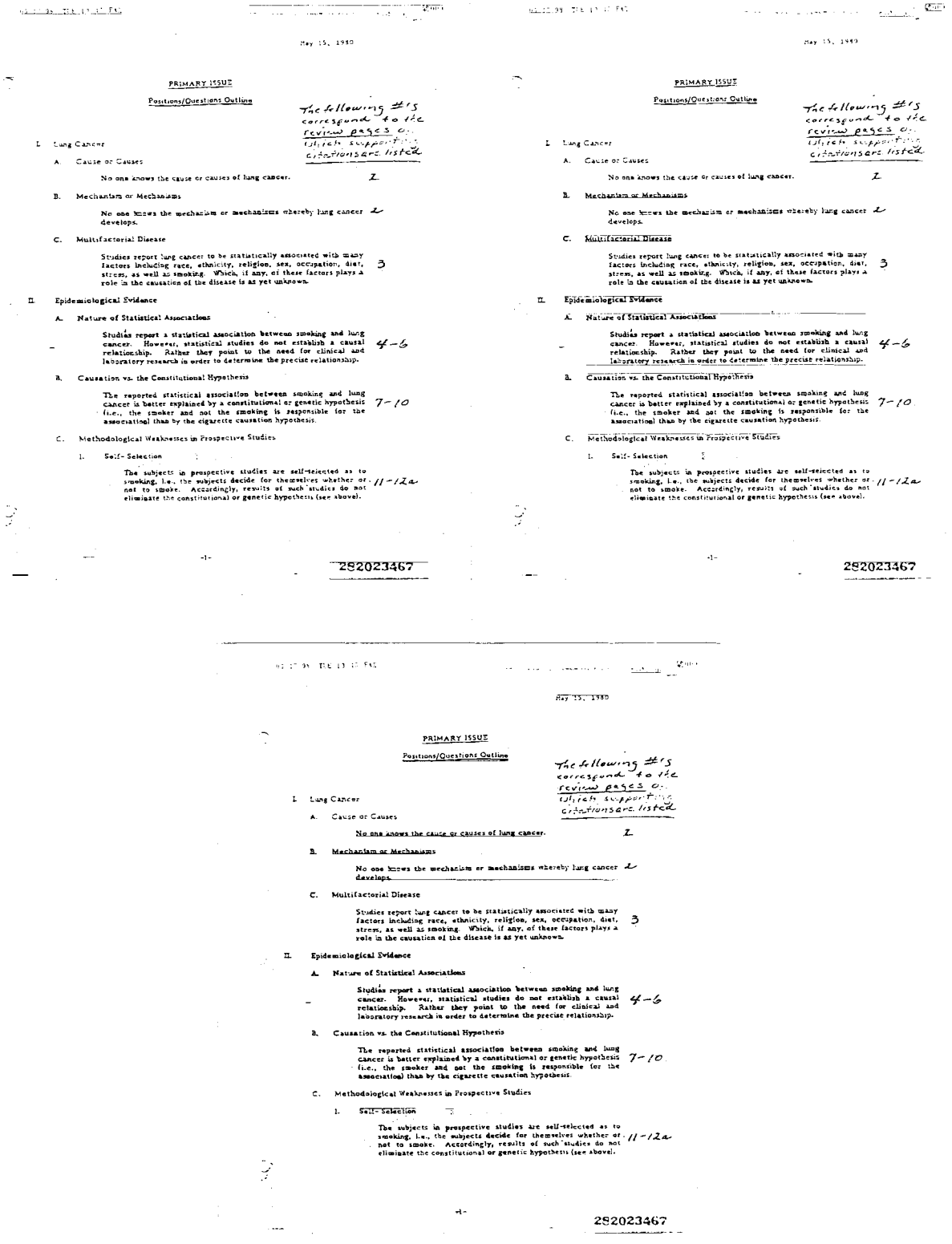
Our initial experiments take a very straightforward approach. We start with an arbitrary region of the document image and compute a series of texture properties as a feature vector. We then compute the similarity between these vectors and use them to determine if two regions should be merged or grouped. This is preliminary work, so only a sketch of the results is presented here.

2 Feature Extraction

In this section we describe a set of basic texture-like features that are computed locally from each region. These regions can ultimately be the result of a set of page segmentation processes or even an arbitrary decomposition of the page.

2.1 Bi-level co-occurrence

Gray level co-occurrence represents the two-dimensional spatial dependency of pixel gray levels at fixed distances and/or directions [13, 14]. Bi-level co-occurrence is a specialization of gray-level co-occurrences applied to binary images. A co-occurrence histogram is simply the number of times a given pair of pixels occurs at a fixed distance and orientation. In the case of binary images, the possible occurrences are white-white, black-white, white-black and black-black at each distance and orientation.



May 15, 1980

PRIMARY ISSUE

Positions/Questions Outline

The following #'s correspond to the review pages on which supporting citations are listed

I. Lung Cancer

A. Cause or Causes

No one knows the cause or causes of lung cancer. 1

B. Mechanism or Mechanisms

No one knows the mechanism or mechanisms whereby lung cancer develops. 2

C. Multifactorial Disease

Studies report lung cancer to be statistically associated with many factors including race, ethnicity, religion, sex, occupation, diet, stress, as well as smoking. Which, if any, of these factors plays a role in the causation of the disease is as yet unknown. 3

II. Epidemiological Evidence

A. Nature of Statistical Associations

Studies report a statistical association between smoking and lung cancer. However, statistical studies do not establish a causal relationship. Rather they point to the need for clinical and laboratory research in order to determine the precise relationship. 4-6

B. Causation vs. the Constitutional Hypothesis

The reported statistical association between smoking and lung cancer is better explained by a constitutional or genetic hypothesis (i.e., the smoker and not the smoking is responsible for the association) than by the cigarette causation hypothesis. 7-10

C. Methodological Weaknesses in Prospective Studies

1. Self-Selection

The subjects in prospective studies are self-selected as to smoking, i.e., the subjects decide for themselves whether or not to smoke. Accordingly, results of such studies do not eliminate the constitutional or genetic hypothesis (see above). 11-12a

-1-

282023467

Figure 2: Examples of segmentation results from three different segmenters.

In our case, we are concerned primarily with the foreground. Since the white background region often accounts for up to 80% of a document page, the occurrence frequency of white-white or white-black pixel pairs would always be much higher than that of black-black pairs. The statistics of black-black pairs carry most of the information. For example, in a block of text, at a distance that is a multiple of the line spacing we can expect a high occurrence of black-black pairs in the vertical direction. To eliminate the redundancy and reduce the effects of over-emphasizing the background, we simply exclude the occurrence of all white-white, white-black and black-white pairs from consideration. At large enough scales we can claim that we capture the background information by measuring repetitive patterns in the foreground.

0	0	1	1
0	1	1	0
1	1	0	0
1	0	0	0

Figure 3: Example of Image Pixel Values

Figure 3 represents a small image block, where 1 is black (foreground) and 0 is white (background). At distance 1, there are 3 (1-1) occurrences in the horizontal direction, 3 in the vertical direction, 0 in the primary diagonal direction (from left-top to right-bottom), and 5 in the minor diagonal direction (from right-top to left-bottom). At distance 2, the counts are 0, 0, 0, and 3 respectively. At distance 4, they are all 0's. In general, if the distance is larger than either the width or height of the image block, four zeros are assigned. The reason is that it is the relative value of the occurrence in one direction with respect to the other three that is useful, so the absence of statistics in one direction voids the other three. For a fixed distance, we normalize the occurrence by dividing by the sum of the occurrences in all four directions. This produces a vector that represents the region.

2.2 Bi-level 2x2-grams

The NxM-gram was introduced by Aya Soffer [12] in the context of image classification and retrieval. Bi-level 3x3-grams were used in her experiments. We are using simpler bi-level 2x2-grams at a hierarchy of distances from the origin. As above, we first remove the dominant background, all white background grams. We then scale each entry by multiplying the number of occurrences by a coefficient proportional to the number of black pixels in the 2x2-gram. The more black pixels the larger the coefficient. In this work, we used $p^b(1-p)^{4-b}$, where p is the density of the image block (number of black pixels divided by the area), and b is the number of 1's in the 2x2-gram. We then normalize the entire vector of occurrences by dividing them by the sum of all occurrences, and arrange the $15n$ values (for n distances) in a vector. As before, if the distance is larger than the width or height of the image block, all 15 values are set to zero.

Consider the example in Figure 3 again. The occurrence of 2x2-grams at distances 1 and 2 are listed in Table 1. The density of black pixels is p=7/16=0.4375. The final feature values are shown in a column to the right of the occurrence numbers in Table 1.

2x2-gram label	Occurrence number at distance 1	Feature value	Occurrence number at distance 2	Feature value
0	1	N/A		N/A
1	1	0.0887		
2				
3				
4				
5				
6			3	0.7942
7	2	0.2935		
8	2	0.1775	1	0.2058
9				
10				
11				
12				
13				
14	3	0.4403		
15				

Table 1: 2x2-gram vector example: express the label (0-15) as a four-digit binary sequence; the digits stand for the pixel values in the left-top, right-top, left-bottom, and right-bottom positions.

2.3 Pseudo run lengths

Run length statistics, in both the horizontal and vertical directions, are important features when comparing document images, especially when trying to distinguish text areas with different fonts. However, true run length counts are inefficient to compute because this involves examining pixels one by one. We propose a much faster method for computing pseudo-runlength statistics as our feature values.

The basic idea is that we first down-sample the signal. We are effectively preserving the low frequency components; the larger the down-sample rate, the lower the frequency of the preserved components. By comparing the original signal with the down-sampled one, we can estimate the high frequency components that are present in the original signal.

For binary images, the signal, for example, a horizontal scan line, consists only of 1's and 0's. We pass it through a non-linear down-sampling low-pass filter, such that every pair of pixels at an even position 2n and a subsequent odd position 2n+1 combine to produce one pixel in the output. For black pixel runlengths, we use an AND operation as the combination so that the output is black (1) only if two neighboring pixel are black.

Presumably, when there is a single black pixel, it will be eliminated, and if there are two pixels each starting at an even position, they will be combined into one. For white runlengths, we use the OR operation.

After the down-sampling, we apply repetitive up-sampling, i.e., each pixel is repeated as the next pixel. The number of black pixels in the up-sampled image is less than that of the original if AND is used in down-sampling, and the same for white if OR is. This number tells us how many of the low-frequency components are left by the low-pass filters.

If we apply AND down-sampling and up-sampling once, we get the number, *l1*, of all black pixels. If we then apply AND down-sampling twice followed by up-sampling twice we get *l2*, which is smaller than *l1* since more high frequency components are excluded. The difference between them, $b1 = l2 - l1$, is like the result of a band-pass filter, telling how many components are between the two frequencies associated with *l1* and *l2*.

These band-pass-like statistics are an approximation of the statistics of run lengths. They follow the same trend, and we call them pseudo-runlength statistics. Our numbers are much easier to compute since down-sampling and up-sampling can be done using lookup tables on eight pixels at a time, and computation can be done in the horizontal and vertical directions together. Actually the up-sampling isn't needed in the implementation. We only need to multiply the number of black pixels in the down-sampled image by 2. Using the image block in Figure 3 again, we get the pseudo-runlength statistics listed in Table 2. As before we arrange these pseudo-runlength numbers, i.e., *b*'s, in a vector, and normalize them by dividing by their sum.

Direction	Horizontal	Vertical	Primary diagonal	Minor diagonal
Black pseudo-runlength (AND down-sampling)				
l1	4	4	0	6
l2	0	0	0	4
b1=l1-l2	4	4	0	2
White pseudo-runlength (OR down-sampling)				
l1	6	6	2	8
l2	0	0	0	8
b1=l1-l2	6	6	2	0

Table 2: The pseudo-runlength example: for black pseudo-runlength, *l1* and *l2* represent numbers of black pixels in the up-sampled image, while for white pseudo-runlength, they represent the numbers of white pixels.

3 Similarity measures

To compare the feature vectors, we adopted a common correlation measurement. First we exclude the ranges from both vectors where either of them contains all zeros, when the distance for that range is larger than the size of one of the image blocks. Then the similarity score is given by

$$S_x(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|}$$

where the subscript x will be either c for co-occurrence, g for 2x2-gram, or r for pseudo-runlength. We have found that different feature vectors are sensitive to different types of content. Nevertheless, we found each of them to be quite consistent with respect to image blocks of the same content. Our final similarity measurement is therefore a combination of the different feature vectors:

$$S(A,B) = S_c(A,B) \times S_g(A,B) \times S_r(A,B)$$

4 Experimental results

We carried out several experiments on images containing mixed content. Figure 4 shows a sampling of image blocks containing about 15 different types of contents taken from several scanned documents. The scanned images are all at 300dpi resolution. The contents vary from high-quality text in Times Roman font, to low-quality typed text, bold text, small text, handwritten comments, signatures, halftone in images, and noise. 22 blocks are defined on the image, with the first three being handwritten, the next five being image, large shadowed text, and noise. The remaining 14 blocks are all pairs of text for each of seven different printed text blocks.

We computed bi-level co-occurrence features for distances 8, 16, 32, 64, and 128. Those are also the distances we used for bi-level 2x2-gram features. For pseudo-runlength statistics we did down-sampling from once through seven times. The directions we used in computation of co-occurrence and pseudo-runlength features were horizontal, vertical, primary and minor diagonal. The co-occurrence feature vectors were 20 values long, 75 values for 2x2-gram, and 28 for pseudo-runlength per image block. Their mutual similarity scores are shown in Table 3, and the image is shown in Figure 4.

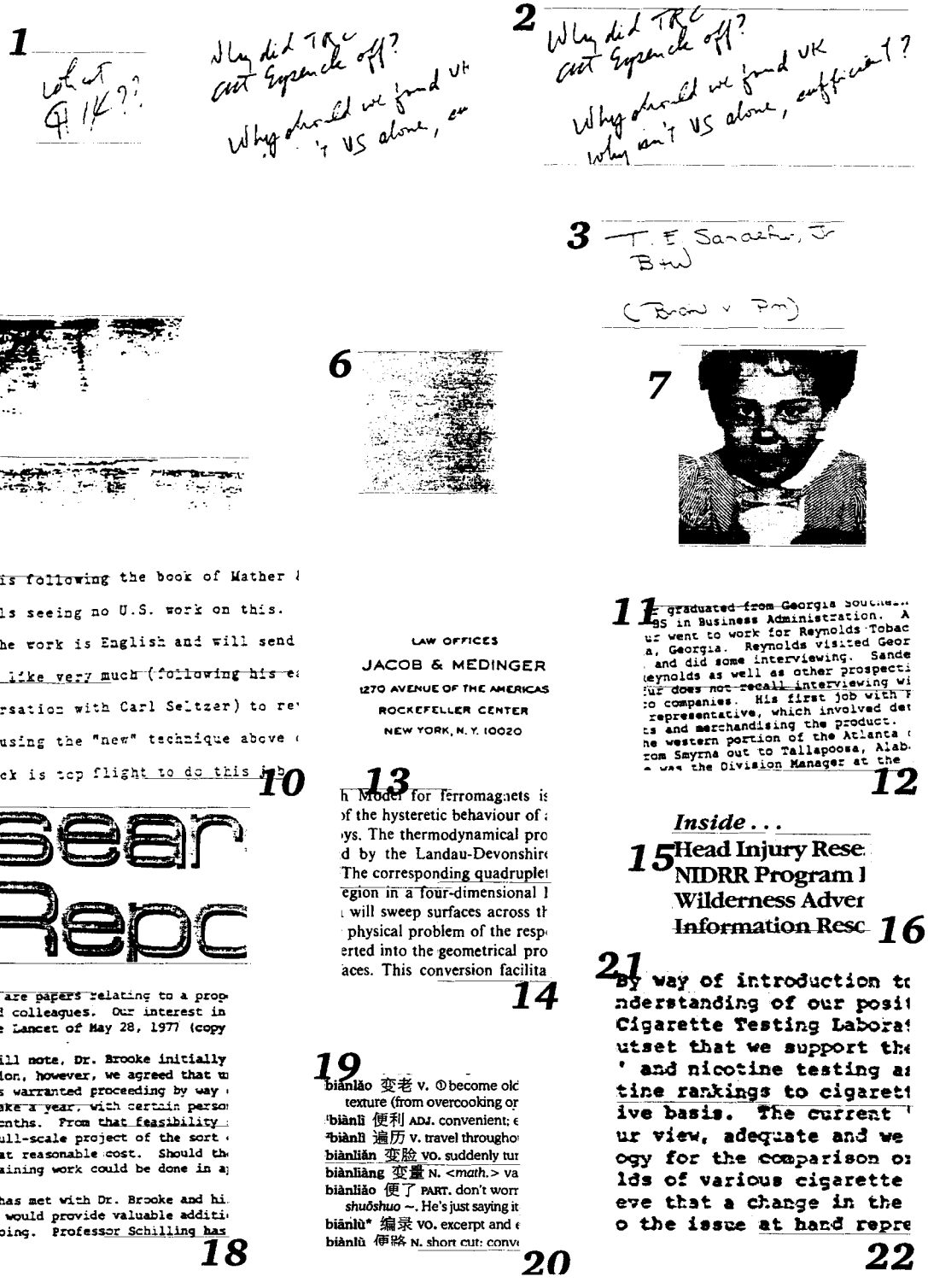


Figure 4: Sample Regions used in Experiments

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	
1	-	77	37	34	39	41	36	51	38	33	55	53	46	39	47	48	37	40	54	38	52	48	1
2	77	-	50	47	58	45	42	72	49	42	61	56	45	39	61	60	52	54	64	52	66	64	2
3	37	50	-	44	67	40	33	61	41	42	33	27	41	39	36	39	44	49	48	52	46	49	3
4	34	47	44	-	50	37	51	67	26	21	35	32	40	37	46	40	37	35	43	39	45	45	4
5	39	58	67	50	-	38	39	65	42	38	40	37	34	34	39	35	59	58	55	56	45	46	5
6	41	45	40	37	38	-	69	49	17	16	53	56	51	48	45	43	40	41	69	58	41	39	6
7	36	42	33	51	39	69	-	60	19	17	43	44	38	35	45	42	37	37	60	48	42	38	7
8	51	72	61	67	65	49	60	-	49	43	55	52	47	45	72	69	59	60	66	59	78	78	8
9	38	49	41	26	42	17	19	49	-	95	31	26	39	36	43	47	42	43	36	34	54	60	9
10	33	42	42	21	38	16	17	43	95	-	26	21	36	33	38	41	38	41	34	33	47	52	10
11	55	61	33	35	40	53	43	55	31	26	-	96	56	51	60	53	65	59	71	61	58	53	11
12	53	56	27	32	37	56	44	52	26	21	96	-	51	46	63	56	60	55	70	59	57	52	12
13	46	45	41	40	34	51	38	47	39	36	56	51	-	95	55	56	44	45	62	61	62	60	13
14	39	39	39	37	34	48	35	45	36	33	51	46	95	-	50	50	42	44	57	60	57	55	14
15	47	61	36	46	39	45	45	72	43	38	60	63	55	50	-	90	55	53	71	59	86	82	15
16	48	60	39	40	35	43	42	69	47	41	53	56	56	50	90	-	48	47	65	54	89	88	16
17	37	52	44	37	59	40	37	59	42	38	65	60	44	42	55	48	-	95	76	81	57	53	17
18	40	54	49	35	58	41	37	60	43	41	59	55	45	44	53	47	95	-	76	84	58	53	18
19	54	64	48	43	55	69	60	66	36	34	71	70	62	57	71	65	76	76	-	87	68	62	19
20	38	52	52	39	56	58	48	59	34	33	61	59	61	60	59	54	81	84	87	-	61	57	20
21	52	66	46	45	45	41	42	78	54	47	58	57	62	57	86	89	57	58	68	61	-	96	21
22	48	64	49	45	46	39	38	78	60	52	53	52	60	55	82	88	53	53	62	57	96	-	22
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	

Table 3: Similarity scores for 22 image blocks: the maximum possible score is 100, which is shown as "-", and the minimum is 0.

Let us consider several examples. For columns 9-22, notice that they are pairs of the same text and that the scores between pairs from the same region are all high (in the 90's). All other scores are rather low, most of them falling below 70, with a few 80's. One exception is blocks 19 and 20, which have a score of 87. This is because block 19 actually contains Chinese characters, thus is in fact different from the pure English block 20. Another exception is that blocks 15 and 16 both have fairly high scores with blocks 21 and 22. This can be accepted since all of them have similar font sizes and line spacing, which means they look similar to some extent.

Now consider blocks 1, 2, and 3. All are handwritten comments. Blocks 1 and 2 come from same document page, and they are written by the same person. The score between them is 77, much higher than all their scores with any other block. On the other hand, block 3 has low scores with all other 21 blocks.

Blocks 4 through 8 are noise or noise-like. The scores show that they are all dissimilar to any other blocks. The only exception is block 8, which has a not so low similarity score with blocks 21 and 22.

5 Summary and Conclusions

In conclusion, our texture feature and similarity measurements work very well with printed text, disregarding the font type, size, and image quality. They work well with handwritten characters. As for noise and images, they tend to be dissimilar to anything according to our measurements.

We feel that these results are encouraging and warrant further consideration toward the goal of refining segmentations based on texture discrimination.

6 References

[1] L. O'Gorman, R. Kasturi. "The document spectrum for page layout analysis", IEEE Trans. on Pattern Analysis and Machine Intelligence, 15:1162-1173, 1993.

[2] K. Kise, A. Sato, M. Iwata. "Segmentation of page images using the area Voronoi diagram", Computer Vision and Image Understanding, 70:370-382, 1998.

[3] F. Wahl, K. Wong, R. Casey, "Block segmentation and text extraction in mixed text/image documents", Graphical Models and Image Processing, 20:375-390, 1982.

[4] A. K. Jain, B. Yu, "Document representation and its application to page decomposition", IEEE Trans. on Pattern Analysis and Machine Intelligence, 20:294-308, 1998.

[5] L. A. Fletcher, R. Kasturi, "A robust algorithm for text string separation from mixed text/graphics images", IEEE Trans. on Pattern Analysis and Machine Intelligence, 10:910-918, 1988.

[6] A. Antonacopoulos, R. T. Ritchings, "Representation and classification of complex-shaped printed regions using white tiles", 3rd ICDAR'95, 1132-1135, Montreal, Canada, August 1995.

[7] H. Baird, "Background structure in document images", Inter. Journal of Pattern Recognition and Artificial Intelligence, 8:1013-1030, 1994.

[8] H. S. Baird, S. E. Jones, S. J. Fortune, "Image segmentation by shape-directed covers", in Proceedings of Inter. Conf. on Pattern Recognition, 820-825, Atlantic City, NJ, June 1990.

[9] G. Nagy, S. Seth, "Hierarchical representation of optically scanned documents", in Proceedings of Inter. Conf. On Pattern Recognition, v1, 347-349, Montreal, Canada, July 1984.

[10] G. Nagy, S. Seth, M. Viswanathan, "A prototype document image analysis system for technical journals", Computer, 25:10-22, 1992.

[11] J. Ha, R. M. Haralick, I. T. Phillips, "Recursive X-Y cut using bounding boxes of connected components", Proc. of the 3rd ICDAR, 952-955, Montreal, Canada, August 1995.

[12] A. Soffer, "Image categorization using texture features", Proc. of the ICDAR, 1997.

[13] E. S. Deutsch, N. J. Belknap, "Texture descriptors using neighborhood information", Computer Graphics and Image Processing, 1:145-168, 1972.

[14] R. M. Haralick, K. Shanmugam, I. Dinstein, "Textural features for image classification", IEEE Trans. on Systems, Man and Cybernetics, 3:610-621, 1973.

[15] K. I. Laws, "Rapid texture identification", Proc. of the SPIE, v238, 376-380, 1980.

[16] B. S. Manjunath, W. Y. Ma, "Texture features for browsing and retrieval of image data", IEEE Trans. on Pattern Analysis and Machine Intelligence, 18:837-842, 1996.

[17] D. S. Bloomberg, "Textured reductions for document image analysis", Proc. of the SPIE v2660, 160-174.

[18] P. Gupta, N. Vohra, S. Chaudhury, S. D. Joshi, "Wavelet based page segmentation", Proceedings of ICVGIP, 2000.

[19] K. Etemad, D. Doermann, R. Chellappa, "Multiscale document page segmentation using soft decision integration", IEEE Trans. on Pattern Analysis and Machine Intelligence, 19:92-96, 1997.

Style-Directed Document Segmentation

A. Lawrence Spitz

Document Recognition Technologies, Inc.
616 Ramona Street, Suite 20, Palo Alto, CA 94301 USA
email: spitz@docrec.com phone: +1-650-688-0842 fax: +1-650-688-0841

Abstract

Many, perhaps most, applications of document recognition would be improved if the logical structure of the document were encoded along with the content. Though some progress has been made in the development of logical structure from layout information alone, or from the combination of layout and content information, application of document style models provides a considerable advantage in many of those applications.

We have developed a technique for document segmentation that is tunably optimized for performance on documents reflecting specific stylistic models rather than a broad class of documents. The first step is an interactive program that allows an operator to define the logical structure and layout relationships of a prototypical document and thereby generate an XML encoded style sheet. The second step is a non-interactive layout segmenter that takes these style sheets as input ancillary to the document images. The style information serves as hints to the layout segmenter. Logical tags are applied to its output.

1 Introduction

Knowledge and encoding of the layout structure of a document is interesting in its own right. Additionally, comparison of detected layout against a stylistic model assists in other aspects of document recognition.

Others have made some progress in the development of logical structure from layout information alone, or from the combination of layout and content information [1][2][3]. We apply document style models to provide guidance in the layout segmentation and a basis for the logical tagging of the recognized content.

Most document recognition systems compromise between accuracy and generality. Documents are basically instances of artistic expression; there are few unbreakable rules of document logical or layout structure that apply across the universe of documents. Development of completely general purpose systems is, therefore, difficult, if not impossible. Some developers have been willing to sacrifice accuracy in order to deal with large ranges of image quality and document complexity. Other systems handle only high quality digitizations of a narrow class of documents.

Traditional document recognition systems start with little or no prior knowledge of the particular documents that are to be recognized. They seek to provide recognition services for a broad range of documents and in failing to constrain the input document set or input image quality, such systems are burdened by the need for generality. The range of documents that can be processed by such systems is limited to those that comply with the rules invisibly embedded in the recognition algorithms. This information is developed for an individual document and usually is not retained for as yet unseen documents.

In contrast, our system is designed for application to a relatively small set of documents which are readily represented by a style sheet.

In Section 2 we will describe the process of interactively generating style information on a prototypical document, the types of information and architecture and effects of considering stylistic limitations during the recognition of page images. Even if the document's logical structure is not of particular interest, application of the model results in enhanced speed and accuracy of content recognition.

Section 3 describes how the payout segmentation proceeds to take the style information as hints in the layout segmentation process and the application of the logical tags to the appropriate parts of the interest.

2 Style Directed Recognition Processing

A previously described system relied on an encoding of style (logical structure, layout structure and optional examples of required content) to direct the course of recognition[4], In the current system the style information is generated interactively.

The traditional role of style information is used in controlling the synthesis of documents. We have inverted the function of style to provide cues useful in the recognition process.

By reducing the ambiguity at several stages of the process, both speed and accuracy are enhanced. Because this system is designed to provide progressive recognition, it advantageously incorporates knowledge of the style of the document throughout the process. The style encoding represents a standard set for the document. Should the document not comply with that

standard, recognition will fail. When these failures occur, they occur early enough in the processing to leave the possibility of backtracking rather than proceeding with incorrect basic assumptions.

Our explanation below will relate to traditional scientific journal layout, but application of the system is not restricted to this narrow range of styles.

2.1 Generation of Style Information

Our current system uses a tool to interactively define the areas of the page and to provide the logical labeling. The user is presented with a representative page image from the publication in question and can use either of two methods of defining rectangular page segments: drawing circumscribing boxes with the mouse that are then shrunk to fit, and selecting a point within a segment from which the region is "grown".

Figure 1 shows a prototypical page image with each of its segments defined by a colored box.

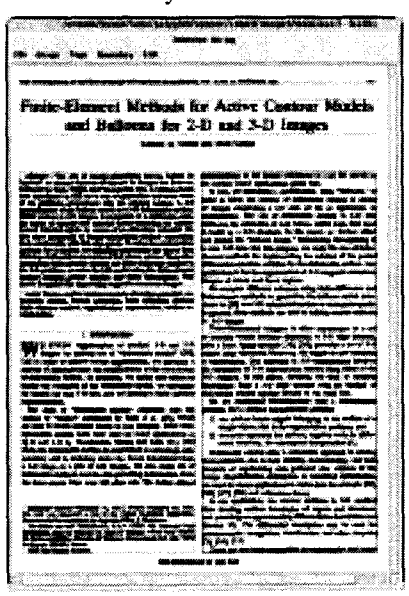


Figure 1: Shrink-wrapped boxes around layout segments

2.2 Layout Relationships

The coordinates of each side of each rectangle can have one of three types: absolute, relative, and variable. Page elements such as headers are likely to have absolute coordinates on the page. The top edge of a title segment can be represented as having a position relative to the bottom of the header. The bottom edge of the title will be variable since the title may include more than one text line

Figure 2 shows the process of defining the top of the `Title` segment as being relative to the bottom of the `Header` segment.

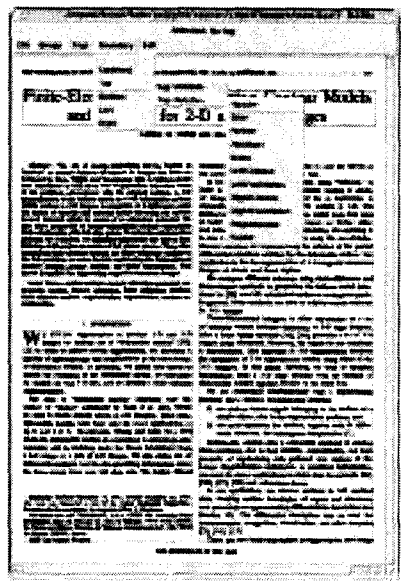


Figure 2: Defining relative layout relationships between segments

2.3 Logical tagging

At the time the segment boundaries are defined, the user labels the segment with its logical tag. The logical tag can not only take on traditionally expected values such as author, title, abstract etc., but can also indicate that a particular segment is optional, meaning that it might not be present in every instance where this style sheet is to be used. An example of such an optional segment is author affiliation which some journals have only on some articles

Figure 3 shows the process of defining the logical tag `Title`.

2.4 Style Representation

Our earlier system [4]. used the Standard Generalized Markup Language (SGML) because of its relative simplicity, its extensibility in terms of processing instructions, its compatibility with existing systems and the availability of tools for its creation, verification and manipulation. Two important compatible extensions of basic SGML were implemented. Flexible style encoding for recognition required inclusion of computed variable values based on mathematical expressions that are functions of measured variables, parametric values, and constants. Since that system was developed, XML has been specified and standardized, and we have decided to adopt it for style representation.

Figure 4 shows the document style information in XML. this representation.

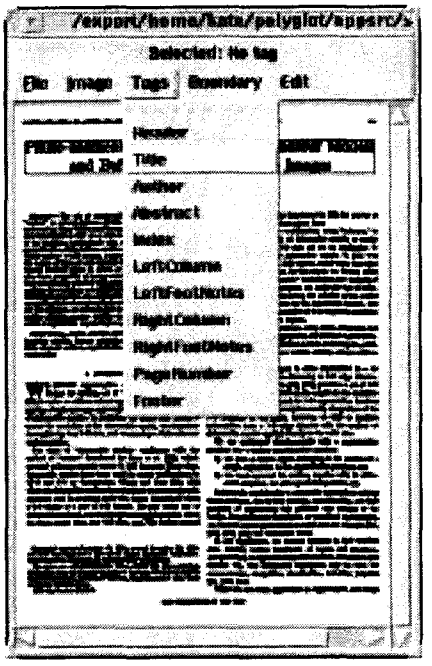


Figure 3: Defining logical contents of segments

3 Layout Segmentation

The style sheet, though developed interactively, can be used in batch mode to guide the layout segmentation of the page. Our layout segmenter is an extension of those developed by Dias [5] and Kise, *et al.* [6] for the segmentation of text lines. It also incorporates important concepts described in Baird for the segmentation of text blocks [7].

The process of applying style to image segments is one of graph isomorphism, albeit with a very simple graph structure. The graph is simple because there is no accommodation for segments which are nested or partially overlap each other. Also segments are, at this time, restricted to being rectangular.

In determining the internal layout structure of the document image, coordinates expressed in the style encoding are used as starting points in a search for the coordinates in pixel space that indicate the position and size of page segments. Scale factors between specified and measured values are developed at the time of processing and therefore are independent of scan resolution and permit automatic compensation for magnification errors and other affine distortions resulting from photocopying and digitization.

For more detail about the methods of registration of the model to the image see Spitz [4].

Early in the recognition process our system develops knowledge of intractability of a document, whether that intractability is due to lack of document style compliance, or to image quality characteristics. Documents are continuously checked against the style encoding for compliance. At any step in the process

```
<!ELEMENT first-page
        (Header, Title, Author, Abstract, Index,
LeftColumn, LeftFootNotes,
    RightColumn, RightFootNotes, Footer)>
<!ELEMENT Header (top, bottom, left, right)>
<!ATTLIST Header
    top CDATA #FIXED "210"
    bottom CDATA #FIXED "231"
    left CDATA #FIXED "238"
    right CDATA #FIXED "2276"
    >
<!ELEMENT Title (top, bottom, left, right)>
<!ATTLIST Title
    top CDATA #FIXED "329"
    bottom (variable)
    left (centered)
    right (centered)
    >
<!ELEMENT Author (top, bottom, left, right)>
<!ATTLIST Author
    top (below:Title)
    bottom (variable)
    left (centered)
    right (centered)
    >
<!ELEMENT Abstract (top, bottom, left, right)>
<!ATTLIST Abstract
    top (below:Author)
    bottom (variable)
    left CDATA #FIXED "235"
    right CDATA #FIXED "1232"
    >
<!ELEMENT Index (top, bottom, left, right)>
<!ATTLIST Index
    top (below:Abstract)
    bottom (variable)
    left CDATA #FIXED "235"
    right CDATA #FIXED "1232"
    >
<!ELEMENT LeftColumn (top, bottom, left, right)>
<!ATTLIST LeftColumn
    top (below:Abstract)
    bottom (variable)
    left CDATA #FIXED "200"
    right CDATA #FIXED "1232"
    >
<!ELEMENT LeftFootNotes (top, bottom, left, right)>
<!ATTLIST LeftFootNotes
    top (below:LeftColumn)
    bottom (above:Footer)
    left CDATA #FIXED "200"
    right CDATA #FIXED "1232"
    >
<!ELEMENT RightColumn (top, bottom, left, right)>
<!ATTLIST RightColumn
    top (below:Author)
    bottom (variable)
    left (rightof:LeftColumn)
```

Figure 4: Part of the style description passed from interactive tool to layout segmenter

when it is impossible to reconcile the instance of the document image against the allowable structure as represented in the style encoding, it is possible to backtrack or to abort processing.

Layout structure is checked against the style encoding only to the depth of that encoding. In other words, structure below the level of that recorded in the style is permitted, but the process of page segmentation is terminated by satisfaction of the style requirements.

3.1 Segmentation Output

The segmenter output described in our earlier system is enhanced in the current system by the addition of logical structure information derived directly from the style encoding. The information includes a set of rectangles, each potentially with a logical tag, coordinates on the page and a pointer to a TIFF file containing the relevant part of the image.

Figure 5 shows a page image with a style similar to the style of the prototype on which the logical tags and layout relationships were defined.

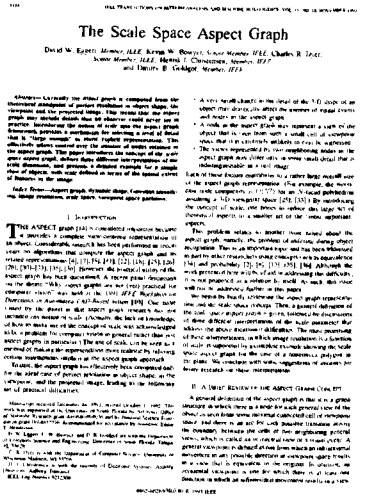

The Scale Space Aspect Graph

Figure 5: Page image from a document of the same style as the prototype but with different

Table 1 shows a log of the positions and dimensions of the layout elements that correspond to the logical components of the image.

Table 1: Locations (in pixels) and file names of logical structure elements

Logical Element	x_1,y_1	x_2,y_2	Image file
Header	(111,87)	(2297,108)	Header.tif
Title	(544,213)	(1864,306)	Title.tif
Author	(326,354)	(2081,497)	Author.tif
Abstract	(108,654)	(1178,1194)	Abstract.tif
Index	(110,1227)	(1176,1304)	Index.tif
LeftColumn	(106,1461)	(1180,2510)	LeftColumn.tif
RightColumn	(1229,652)	(2301,3043)	RightColumn.tif
LeftFootNotes	(109,2564)	(1179,3050)	LeftFootNotes.tif
Footer	(952,3093)	(1455,3120)	Footer.tif

And Figure 6 shows some of the segments cropped from the source image which are to be passed to the OCR engine.

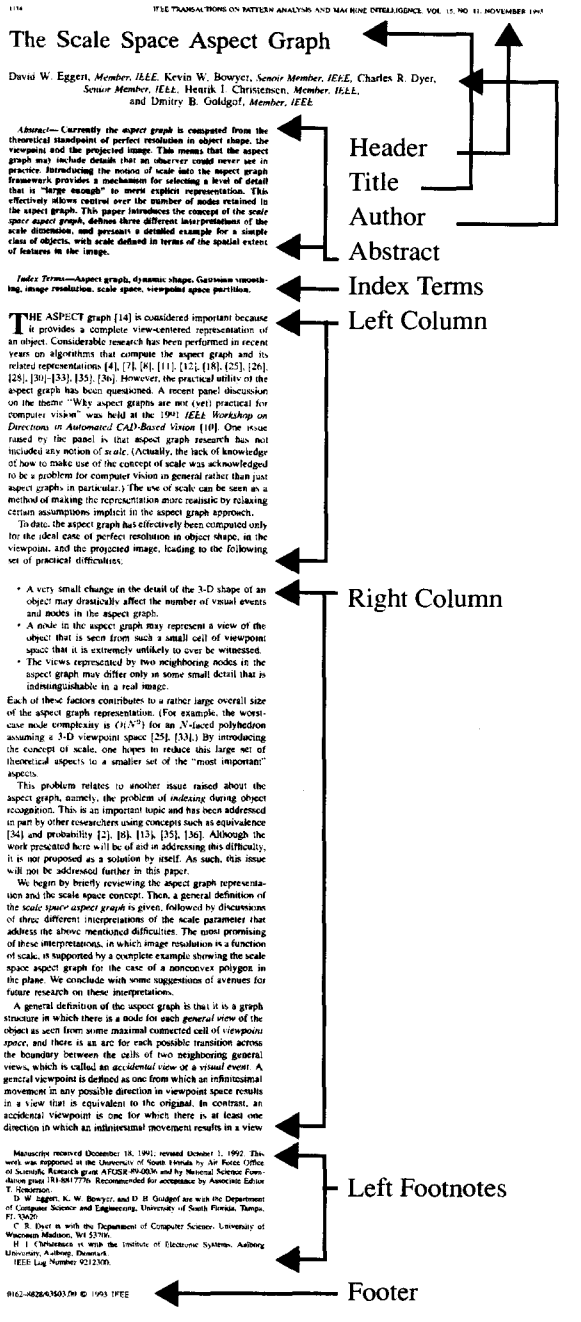


Figure 6: Images of page segments

4 Conclusion

The output of our recognition process includes information about the logical structure of the document derived from layout information, content and stylistic models.

This work has been directed at documents with knowable style and therefore is inappropriate for "omni-document" recognition. However, particularly in digital library or document database applications where logical structure information is an extremely valuable asset for information retrieval, recognition based on style-directed segmentation provides a much richer representation of the documents on which to search.

References

[1] A. Dengel. "Document Image Analysis - Expectation Driven Text Recognition", *Syntactic & Structural Pattern Recognition*, Murray Hill, New Jersey, pp 78-87, 1990.

[2] Y. Tsuji "Document Image Analysis for Generating Syntactic Structure Description", *International Conference on Pattern Recognition*, Rome, pp 744-747, 1988.

[3] S. Tsujimoto, "Understanding Multi-articled Documents", *International Conference on Pattern Recognition*, Atlantic City, New Jersey, pp 551-556, 1990.

[4] A.L. Spitz, Style Directed Document Recognition. *International Conference on Document Analysis and Recognition*, St. Malo, France, pp 611-619, 1991.

[5] A.P. Dias, Minimum Spanning Trees for Text Segmentation, *Symposium on Document Analysis and Recognition*, Las Vegas, pp 51-65, 1995.

[6] K. Kise, M. Iwata, K. Matsumoto and A. Dengel, A Computational Geometric Approach to Text-line Extraction from Binary Document Images, *Document Analysis Systems*, Nagano, Japan, pp 346-355, 1998.

[7] H. Baird, S. Jones and S. Fortune., "Image Segmentation by Shape Directed Covers", *International Conference on Pattern Recognition*, Atlantic City, New Jersey, pp 820-825, 1990

Evaluating Document Analysis Results via Graph Probing

Daniel Lopresti **Gordon Wilfong**
Bell Laboratories
Lucent Technologies, Inc.
600 Mountain Avenue
Murray Hill, NJ 07974
USA
{dpl,gtw}@research.bell-labs.com

Abstract

While techniques for evaluating the performance of lower-level document analysis tasks such as optical character recognition have gained acceptance in the field, attempts to formalize the problem for higher-level algorithms that incorporate more complex structure have been less successful. In this paper, we describe an intuitive, easy-to-implement scheme for the problem of performance evaluation when document recognition results are represented in the form of a directed acyclic graph.

The paradigm, which we call "graph probing," has a sound basis in past work on heuristics for solving the graph isomorphism problem. However, our goal extends beyond simply testing for equivalence; we also wish to be able to quantify the similarity between two graphs. The technique described in this paper provides such a measure. We present results from three simulation studies based on different graph models and one experiment using real OCR data to demonstrate the applicability of the approach.

1 Introduction

As document analysis systems grow more and more sophisticated, it becomes increasingly important to be able to evaluate and compare their performance. With a few notable exceptions, however, little has been achieved along these lines beyond the informal assertions that often accompany work published in the field. A thoughtful overview of the subject of automated performance evaluation can be found in [19].

While the directed acyclic graph, or *DAG*, is a nearly universal representation across recognition algorithms, the graph structure is typically discarded when it comes time for evaluation. In the case of page segmentation, for example, several practical approaches have been proposed based on distance-type measures, but these make little or no use of the whole graph, and instead focus on pixel-level comparisons [13, 26] or matching the text characters output from OCR [2, 13].

There is already a substantial amount of theory for the problem of evaluating logical structure recognition (see, e.g., [12, 18, 22–24]). Nevertheless, the empirical literature has largely ignored this work, perhaps owing to its complexity, and usually resorts to a simple, manual approach to evaluation: counting by hand the number of components that have been missed or added (e.g., [21]).

In this paper, we examine in detail a paradigm we first put forth in the context of our work on table recognition [10, 11]. This methodology, known as "graph probing," offers an intuitive, easy-to-implement scheme for the general problem of evaluating document recognition when the results are represented in the form of a DAG, and may be extensible to other applications as well. Our approach uses a probing process to assess the agreement between the DAG returned by a recognition system and the DAG created during ground-truthing. Since different classes of probes are possible, ranging from very low- to very high-level, from concrete to abstract, this paradigm can be viewed as subsuming existing techniques that try to measure the structural similarity of the graph representations on the one hand, and the effectiveness of recognition results when incorporated in a particular application on the other.

We begin by describing the concept of graph probing in Section 2. After this overview, we present three different graph models in Section 3 that will be used in examining how well graph probing might work in practice. In Section 4, we discuss the results from three simulation studies and one experiment using real OCR data to demonstrate the applicability of the approach. Finally, we offer our conclusions and topics for future research in Section 5.

2 Graph Probing

Given the DAG for a recognition result and the DAG for its corresponding ground-truth, it is natural to

consider comparing the two as a way of determining how well an algorithm has done. Attempting this directly, however, gives rise to two dilemmas. The first is that any reasonable notion of graph matching subsumes the graph isomorphism problem, the complexity of which is open, as well as possibly the subgraph isomorphism problem, which is known to be NP-complete [9]. Hence, it seems unlikely that there exists an efficient, guaranteed-optimal algorithm for comparing two DAG's in the general case. While heuristics have been developed that are sometimes fast, their worst-case behavior is still exponential (see, e.g., [18]).

The other obstacle is that there may be several different ways to represent the same logical structure as a graph, all equally applicable. Minor discrepancies could create the appearance that two graphs are dissimilar when in fact they are functionally equivalent from the standpoint of the intended application. Forcing one graph to correspond to the other through a rigidly defined matching procedure obscures this important point.

At the other end of the spectrum, we could embed the recognition algorithm in a complete, end-to-end system and measure the system's performance on a specific task from the user's perspective: Does it provide the desired information? (this is "goal-directed evaluation" as discussed in [19]). This approach has its own shortcomings, however, as it limits the generality of the results and makes it difficult to identify the precise source of errors that arise when complex processes interact.

We have developed a third methodology that lies midway between these two. We work directly with the graph representation. However, instead of trying to match the graphs under a formal model, we probe their structure and content by asking relatively simple queries that mimic, perhaps, the sorts of operations that might arise in a real application.

Conceptually, the idea is to place each of the two graphs under study inside a "black box" capable of evaluating a set of graph-oriented operations (e.g., returning a list of all the leaf nodes, or all nodes labeled in a certain way). We then pose a series of probes and correlate the responses of the two systems. A measure of their similarity is the number of times their outputs agree. This process is depicted in Fig. 1. Note that it is essential the probes themselves have simple answers that are easily compared. They might return, for example, a count of the number of nodes satisfying a certain property (e.g., possessing a particular label), or the content of a designated leaf node. The probing becomes recursive if the target of a probe is a graph itself. The intention is that this probing process abstracts the access of content away from the specific details of the graph's structural representation.

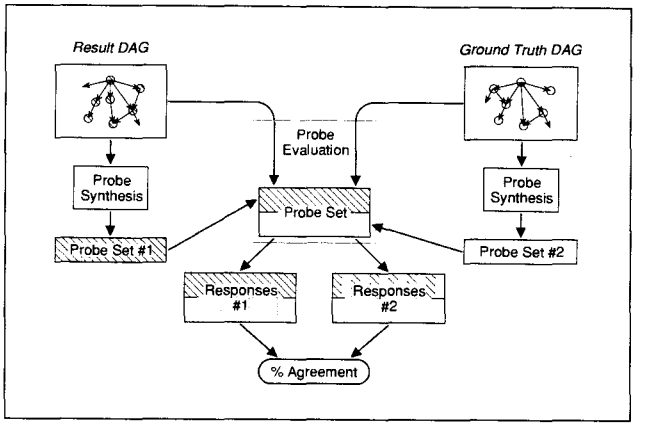


Figure 1: Overview of graph probing.

As noted earlier, the problem studied here is clearly related to the problem of graph isomorphism. The complexity of graph isomorphism remains open and, unfortunately, all known deterministic algorithms have worst-case exponential running times [8]. Many heuristics for determining isomorphism have relied on using *vertex invariants*, where a vertex invariant consists of a value $f(v)$ assigned to each vertex v, so that under any isomorphism I, if $I(v) = v'$ then $f(v) = f(v')$. One such vertex invariant is the degree of the vertex (or the in- and out-degrees, if the graph is directed). There are numerous applications where graph isomorphism arises, such as mathematical chemistry [25], knowledge retrieval [6], robotics [5] and object recognition [1], and in these cases vertex invariants are often used to try to determine isomorphism. In fact, **nauty**, a successful software package for determining graph isomorphism (see [16, 17]), relies on vertex invariants.

There has been some analysis showing that such heuristics for determining graph isomorphism can fail in a catastrophic manner [4]. On the other hand, it has been proven that for random graphs, there is a very simple linear time test for checking if two graphs are isomorphic that is based on the degree of the nodes of the graphs, and this test succeeds with high probability [3]. This type of result motivates the idea of performing local probes to try to determine if there exist differences in a pair of graphs. In fact, these local probes will likely provide us with sufficient evidence to determine whether or not the two graphs are isomorphic. However, we wish to solve more than this simple "yes/no" problem; we are interested in *quantifying* the similarity between two different graphs.

Three specific directed acyclic graph models will be described in the next section. As is common in the literature, all incorporate nodes that are labeled

as to type. In addition, some nodes also contain content. The set of types is relatively small and fixed in advanced, while content is unconstrained and open-ended. Edges, possibly labeled, express relationships between nodes.

While the probing paradigm is open-ended, currently we have defined three categories of probes for the graphs in our studies:

Class 0 These probes count the number of occurrences of a given type of node in the graph. A typical Class 0 probe might be paraphrased as: *How many nodes labeled "Line" does the graph have?*

Class 1 These probes combine content and label specifications. A representative Class 1 probe might be: *How many nodes labeled "Word" with content "pentagon" does the graph have?*

Class 2 These probes examine the node and edge structure of the graph by counting in- and out-degrees. An example of a Class 2 probe is: *How many nodes have in-degree 2 and out-degree 2?*

The generation of a probe set is based on one or the other of the graphs in question (recall Fig. 1). That graph will obviously return the definitive responses for all of the probes in the set, while the other graph will do more or less well depending on how closely it matches the first. We then repeat the process from the other direction, generating the probe set from the second graph and tallying the responses for both. The probes are synthesized automatically, working from the DAG's that are output by the recognition and ground-truthing processes. For specifying probes, we have implemented a graph-oriented query language embedded in a general-purpose programming language, Tcl/Tk [20]; this offers a great deal of flexibility.

We define a *discriminating probe* to be a probe that demonstrates a difference between two graphs. Two fundamental questions are of interest: (1) For two graphs that are different, does there exist at least one discriminating probe?, and (2) Over the entire set of probes, how many are discriminating? The first of these reflects the graph isomorphism problem. The second can serve as a measure of how similar the two graphs are. To make this more explicit, we define the *agreement* between two probe sets to be:

$$agreement \equiv 1.0 - \frac{\#\ of\ discriminating\ probes}{total\ \#\ of\ probes} \quad (1)$$

If the agreement is 1.0, then the two graphs are indistinguishable with respect to the probe set in question. Values less than 1.0 indicate some degree of similarity falling short of a perfect match. Our aim is to equate "agreement" with the traditional concept of "accuracy."

3 Graph Models

In this section, we describe the three graph models we use in our experiments. As will be discussed later, we have written programs to randomly generate instances of graphs of a given type, as well as to edit graphs in ways reminiscent of recognition errors.

3.1 Entity Graph Model

The *entity graph* model reflects a standard document hierarchy: nodes labeled as *Page*, *Zone*, *Line*, or *Word* [14].[1] The edge structure represents two relationships: *contains* and *next*. An example of one such entity graph is shown in Fig. 2, corresponding to the nonsense document fragment given below:

```
satisfactory extrinsic inexpert frankfurter

abutting tarantula
grillwork pentagon attribution bilharziasis
```

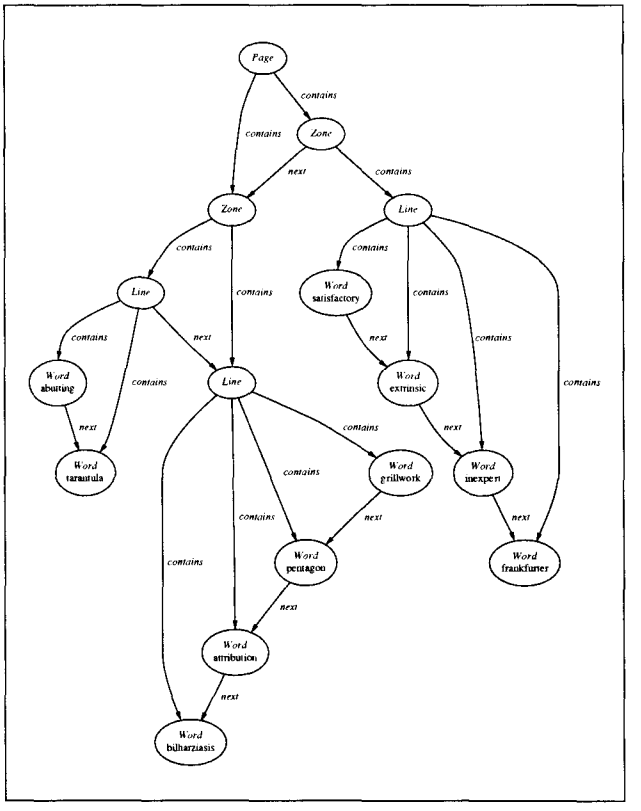


Figure 2: An instance of an entity graph.

3.2 Table Graph Model

Entity graphs encode document page structure in a very general way. A more restricted type of graph is

[1] The entity model as it appears in [14] also includes *Char* as another level below *Word*. We ignore that refinement here for efficiency reasons.

the *table graph*, as defined in our past work on table recognition [10, 11]. Tables consist of lower-level cells, grouped in terms of logical rows and columns. Hence, nodes in table graphs can be labeled *Cell*, *Row*, and *Column*. Edges encode the *contains* relationship. An example of a table graph as derived from the following randomly generated table is shown in Fig. 3:

regression radiant	gusset prick	sima Nostrand
clubroom incubi	593134723	ant Sussex
ascribe gam	1813217419	opulent
shovel registrable	615003753	astride Peru

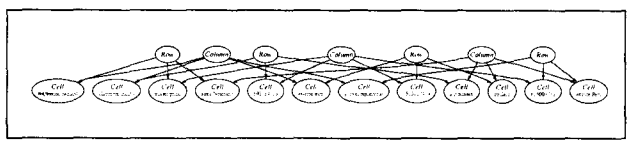


Figure 3: An instance of a table graph.

For the table graph model, we add a fourth, more sophisticated class of probes:

Class 3 These probes mimic simple database-style queries, although phrased entirely in terms of graph manipulations. For a given target node, keys that uniquely determine its row and column are identified. These are used to index into the graph, retrieving the content of the node (if any) that lies at their intersection. An example of a Class 3 probe for the graph in Fig. 3 is: *What is the content of the cell that lies at the intersection of the row indexed by "ascribe gam" and the column indexed by "astride Peru"?* The response would be: *opulent*.

Class 3 probes are particularly interesting in that they lie at a higher level of abstraction than the other, simpler kinds of probes. Indeed, if the application in question was to build an interactive table look-up system, it could be argued that the results of Class 3 probing are more important than, say, counting the number of nodes labeled a certain way. Two graphs could be structurally quite different, but still respond similarly to Class 3 probes; to the user, they would be functionally the same.

3.3 Random Graph Model

The final model we consider in our experiments, the *random graph* model, is a completely random directed acyclic graph. As with the previous two models, nodes are labeled with a type and optional content. There is only one kind of edge, so these are not labeled. However, unlike the entity and table graphs, there are no restrictions on what constitutes a "legal" instance of the model; any node can be connected to any other node irrespective of its label, which is assigned randomly. Fig. 4 shows an example of a random graph generated by our procedure.

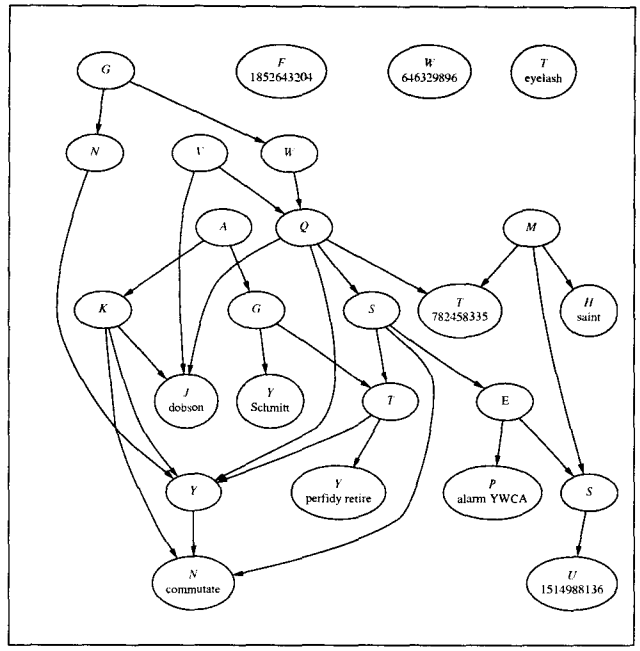


Figure 4: An instance of a random graph.

4 Experimental Results

To test the concept of graph probing, we designed a series of simulation studies as well as an experiment using real results from a commercial OCR system. As indicated, we would like to be able to equate probing agreement (i.e., Eq. (1)) with some general notion of accuracy. Unfortunately, there is no measure that is both universal and easy-to-compute which we can use for comparison purposes (indeed, this point is a primary motivation of our research). Hence, we have chosen to work "backwards" by randomly generating a ground-truth graph, and then simulating recognition "errors" by editing the graph in various ways: adding and deleting nodes, altering labels and content, etc. The number of edits we perform is an approximation (an upper bound, in fact) of the true distance between two DAG's. In the case of the OCR experiment where there does already exist a practical, accepted methodology for computing accuracy, string edit distance, we correlate probing agreement with normalized edit distance.

In the studies that follow, it is important to keep in mind that the probes are always generated automatically, working directly from the recognition result and the ground-truth. Once the probe classes have been defined (which need only be done once), graph probing is a completely autonomous evaluation paradigm.

4.1 Simulation Results for the Entity Graph Model

Our procedure begins by creating a graph for a page with a random number of zones (all random quan-

Table 1: Statistics for the entity graph experiment (500 random graphs).

Attribute	*Min*	*Max*	*Ave*
Zones	1	8	4.9
Lines	1	54	21.7
Words	2	278	108.4
Nodes	5	341	136.0
Edits	1	57	13.3
Class 0 Probes	8	8	8.0
Class 0 Agreement	0.250	1.000	0.530
Class 1 Probes	9	522	216.9
Class 1 Agreement	0.000	0.998	0.912
Class 2 Probes	10	39	29.4
Class 2 Agreement	0.000	1.000	0.576
Overall Probes	28	561	254.3
Overall Agreement	0.177	0.996	0.853
Probes/Node	0.866	1.750	0.985
Probe Time (secs)	0.780	145.010	32.791
Secs/Probe	0.025	0.258	0.107

tities in our simulations are chosen uniformly from within a specified range). For each zone, we then generate a random number of lines, and for each line a random number of words. Content for *Word* nodes is chosen to be either: (1) a word randomly selected from the Unix **spell** dictionary, or (2) a random integer. The editing operations used to simulate recognition errors are guaranteed to yield another legal entity graph. These include altering the content of a *Word* node, deleting an existing *Word*, *Line*, or *Zone* node (and its associated edges), or inserting a new *Word*, *Line*, or *Zone* node.

The entire simulation involved generating 500 "ground-truth" entity graphs, performing a randomly selected number of edits on each, synthesizing and evaluating Class 0, Class 1 and Class 2 probes, and gathering relevant statistics. The study required about 4.5 hours to run on an SGI O2 workstation.

The results for the entity graph experiment are presented in Tables 1 and 2 and in Fig. 5. As can be seen from the first table, there was a wide range in the size of the graphs under consideration. In terms of probes, those from Class 1 were by far the most prevalent (the other two classes sum the results for all nodes in certain broad categories: having the same label or the same in-/out-degree). On average, approximately one probe was generated for each node, and each pair of graphs required about half a minute to compare via probing.[2] Overall, the average probing agreement was 0.853, and the maximum was 0.996 (i.e., the probes always captured the fact that one of the graphs contained errors).

The ability of the three probe classes to differentiate the two graphs is shown in Table 2. Class 1 probes never failed in this experiment. Note that,

[2] As noted earlier, our probes are written in an extension of Tcl/Tk, an interpreted scripting language. In a "production" environment, a more efficient implementation could be achieved using a compiled language.

Table 2: Performance by probe class for the entity graph experiment (500 random graphs).

Probes	*Detected*	*Missed*	*% Detected*	*Unique*
Class 0	450	50	90.0%	0
Class 1	500	0	100.0%	34
Class 2	466	34	93.2%	0
Overall	500	0	100.0%	n/a

by definition, Class 0 and Class 2 probes will always miss differences that involve only content, but various offsetting combinations of edits have the potential to confuse any of the classes. The last column in Table 2 indicates that there were 34 graph-pairs that were distinguished only by using Class 1 probes.

The number of discriminating probes as a function of the number of graph editing operations is shown in the chart in Fig. 5. The datapoints show the average at each step along the x-axis, while the vertical bars give the min/max range. Turning this around, it can be seen that the size of the discriminating probe set provides a reasonably dependable measure of the difference between two graphs. It seems likely that refining and/or weighting the probe sets appropriately could lead to an improvement in the "outliers;" this is a topic for future research.

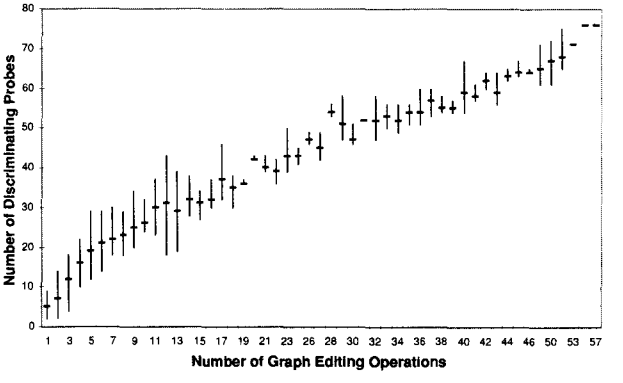


Figure 5: Discriminating probes as a function of edits for the entity graph experiment (500 random graphs).

4.2 Simulation Results for the Table Graph Model

Like the previous simulation, we begin by generating a ground-truth graph containing a random number of rows and columns. Each column is randomly designated as being either alphabetic or numeric. For the former, table cells are selected to be a string of one or more words chosen from the **spell** dictionary, while for the latter the contents of cells are assigned to be random integers. Cells in the first row and column are always set to be alphabetic (to represent table headers). Editing operations include changing the contents of a *Cell* node, deleting a *Row* or

Table 3: Statistics for the table graph experiment (500 random graphs).

Attribute	*Min*	*Max*	*Ave*
Rows	2	15	8.5
Cols	2	6	4.0
Nodes	8	111	46.7
Edits	1	24	9.8
Class 0 Probes	6	6	6.0
Class 0 Agreement	0.000	1.000	0.362
Class 1 Probes	8	174	68.3
Class 1 Agreement	0.154	0.986	0.811
Class 2 Probes	4	6	5.9
Class 2 Agreement	0.000	1.000	0.122
Class 3 Probes	8	174	67.5
Class 3 Agreement	0.000	0.943	0.641
Overall Probes	26	360	147.7
Overall Agreement	0.056	0.966	0.686
Probes/Node	1.231	1.674	1.561
Probe Time (secs)	0.870	178.610	32.793
Secs/Probe	0.032	0.496	0.164

Table 4: Performance by probe class for the table graph experiment (500 random graphs).

Probes	*Detected*	*Missed*	*% Detected*	*Unique*
Class 0	447	53	0.894	0
Class 1	500	0	1.000	0
Class 2	440	60	0.880	0
Class 3	500	0	1.000	0
Overall	500	0	1.000	n/a

Column node (along with all of its associated *Cell* nodes), or inserting a new *Row* or *Column*. In addition to the Class 0, 1, and 2 probes of the first study, we also include the Class 3 probes described in subsection 3.2.

Tables 3 and 4 and Fig. 6 present the results for running this simulation for 500 random tables. While these graphs were smaller than those for the entity graph experiment, the compute-time was nearly identical owing to the new probe class. As Table 3 indicates, the Class 1 and 3 probes never failed. Overall, the average agreement was found to be 0.686.

The detection capabilities of the various probe classes are listed in Table 4. That the Class 0 and 1 probes exhibit roughly the same number of misses as they did in Table 2 is coincidental (this depends on the distributions of the random edits used). Note that there was no instance where a single class of probes found a difference that escaped all of the other classes.

The range in discriminating probes as a function of editing operations is charted in Fig. 6. As before, the number of such probes appears to be a good predictor of the number of edits used to simulate recognition errors, although the behavior of graph-pairs near the extremes of the ranges merits closer examination.

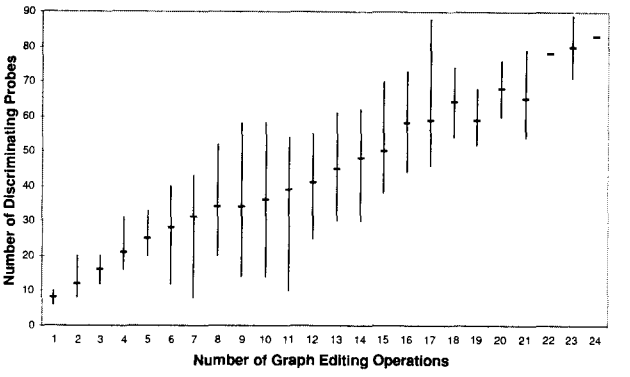


Figure 6: Discriminating probes as a function of edits for the table graph experiment (500 random graphs).

4.3 Simulation Results for the Random Graph Model

The previous two studies were restricted in terms of the initial graphs and the permissible edits that could be performed on them; the graphs always had to be legal instances of an entity or table graph. In this final simulation, the generated graphs are completely unconstrained random DAG's. Node labels are chosen from among the 26 upper case letters A-Z. Certain nodes are designated as leaf nodes; these are assigned random content (words or integers). The remainder of the nodes are attached to varying numbers of children. As was depicted in Fig. 4, the graphs need not be fully connected. The set of possible editing operations consists of changing node labels or content, deleting or inserting nodes, and deleting or inserting edges.

Results for 500 random graphs are given in Tables 5 and 6 and Fig. 7. There were, on average, 251 nodes and 328 edges in the graphs in this study. As Table 5 shows, the average overall agreement was 0.872, while the maximum agreement was 0.966 (i.e., the fact that two graphs were different was detected without fail when all probe classes were taken into account).

The ability of each class to detect the differences is shown in Table 6. The best-performing class (Class 2) contained at least one discriminating probe 97% of the time, while the worst (Class 1) was successful 89% of the time. Perhaps the most important conclusion to be drawn from this table is that none of the classes was redundant; each of them detected at least one case that the other two classes missed.

The plot of discriminating probes versus graph editing operations shown in Fig. 7 bears a strong resemblance to those for the previous two simulations (Figs. 5 and 6). This provides support for our belief that graph probing is a general evaluation paradigm that can be applied across a range of applications that employ graph representations.

Table 5: Statistics for the random graph experiment (500 random graphs).

Attribute	*Min*	*Max*	*Ave*
Nodes	73	251	164.4
Edges	58	328	180.9
Edits	1	25	11.5
Class 0 Probes	46	52	51.6
Class 0 Agreement	0.520	1.000	0.841
Class 1 Probes	50	322	165.2
Class 1 Agreement	0.783	1.000	0.962
Class 2 Probes	28	56	42.3
Class 2 Agreement	0.171	1.000	0.570
Overall Probes	140	413	259.1
Overall Agreement	0.620	0.993	0.872
Probes/Node	0.577	1.142	0.816
Probe Time (secs)	3.070	249.940	17.926
Secs/Probe	0.019	1.032	0.069

Table 6: Performance by probe class for the random graph experiment (500 random graphs).

Probes	*Detected*	*Missed*	*% Detected*	*Unique*
Class 0	474	26	0.948	1
Class 1	447	53	0.894	8
Class 2	486	14	0.972	6
Overall	500	0	1.000	n/a

4.4 Experimental Results for Output from an OCR System

Our past experience using graph probing for performance evaluation in small-scale experiments involving real (as opposed to simulated) document analysis results has been quite favorable (see [10, 11]). As is often the case, however, the considerable effort required to create the necessary ground-truth presents a barrier to performing larger studies featuring our table understanding work at the present time. Instead, we designed an experiment making use of the output from a commercial OCR system, post-processed in the obvious way to yield a graph employing the lower two levels of the entity graph model (i.e., *Line* and *Word* nodes). Since string edit distance is an accepted methodology for evaluating OCR results [7], we have access to a standard to which to compare graph probing.

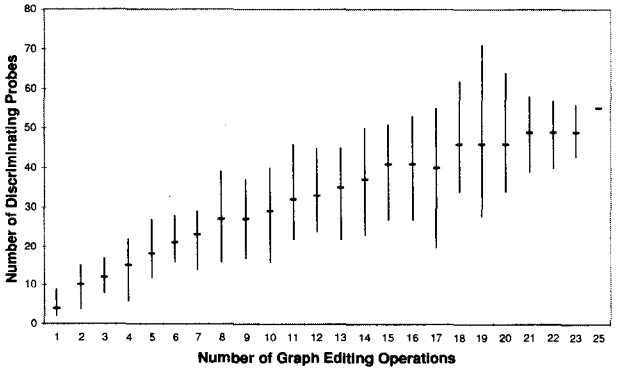


Figure 7: Discriminating probes as a function of edits for the random graph experiment (500 random graphs).

Table 7: Accuracies for the OCR experiment (60 document pages).

Document Type	*OCR Accuracy*		
	Min	*Max*	*Ave*
Printed	93.9%	96.7%	95.8%
Faxed	60.7%	88.4%	73.7%
3rd Generation	63.4%	93.2%	80.7%
Light	78.8%	89.8%	84.6%
Dark	92.6%	96.6%	95.5%
Annotated	56.3%	83.3%	74.9%

The test collection consisted of 10 professionally written news articles gathered from Usenet, ranging in length from 91 to 379 words. For each document, six different versions were created, each formatted in 11-point Times font with a 13-point line spacing under Microsoft Word. One copy of each page was printed and then scanned at 300 dpi using a UMAX Astra 1200S scanner. The remaining five versions of the page were subjected to one of five different degradations before scanning: faxing, noticeably light or dark or third generation photocopying, or handwritten annotation ("redacting") that obscured a randomly chosen 20% of the lines on the page. All of the page images were then OCR'ed using Caere OmniPage Limited Edition.

Normalized string edit distance was used to compute the OCR accuracies [7]. The minimum, maximum, and average accuracies for the 10 pages of a given type are listed in Table 7. As can be seen, some of the documents experienced severe damage, yielding a wide range of accuracies (dropping from 96.7% down to 56.3%). In addition to the many expected character misrecognitions (which induce word-level errors in the entity graph representation), this particular OCR system attempts to concatenate text lines that it believes fall logically within the same paragraph. This policy leads to the potential for disagreements at the *Line* level in the entity graph model as well.

Basic statistics for the probing evaluation of the 60 test pages are presented in Table 8. There were errors in every one of the recognized pages, and this is reflected in the maximum overall agreement which is 0.963. As in the simulations, approximately one probe was generated for each node in the graphs (under the current definitions of the probe sets, this quantity is tied strongly to the number of words in the two documents in question). The probing time averaged 90 seconds per document page. While this is significantly longer than the time needed to compute simple string edit distance, one must remember that graph probing is a more general measure, capable of detecting errors in logical structure as well as in content.

Table 8: Statistics for the OCR experiment (60 document pages).

Attribute	*Min*	*Max*	*Ave*
Zones	1	1	1.0
Lines	2	43	16.1
Words	42	379	195.4
Nodes	48	424	213.5
Class 0 Probes	8	8	8.0
Class 0 Agreement	0.500	1.000	0.604
Class 1 Probes	107	758	390.8
Class 1 Agreement	0.371	0.998	0.754
Class 2 Probes	17	40	27.5
Class 2 Agreement	0.051	1.000	0.173
Overall Probes	134	804	426.3
Overall Agreement	0.359	0.963	0.709
Probes/Node	0.965	1.107	1.011
Probe Time (secs)	7.210	269.600	90.585
Secs/Probe	0.054	0.337	0.177

Table 9: Performance by probe class for the OCR experiment (60 document pages).

Probes	*Detected*	*Missed*	*% Detected*	*Unique*
Class 0	59	1	0.983	0
Class 1	60	0	1.000	1
Class 2	59	1	0.983	0
Overall	60	0	1.000	n/a

Table 9 shows that all of the probe classes were capable of detecting that there were differences between the ground-truth and recognized documents in nearly every case (recall that the OCR accuracies ranged as high as 96.7%). Only in one instance did the Class 1 probes outperform the other two.

The remaining issue, then, is seeing how well graph probing correlates with traditional string edit distance. These results are plotted in Fig. 8. Here we show a distinct style of datapoint for each of the six kinds of copies in the test set. This sort of evaluation is not a particularly fair test for graph probing as the entity graph model we are currently using is word- and not character-based (the model can only distinguish between "zero" and "one or more" errors in a word – it cannot count errors). Even so, while the correspondence between the two measures is somewhat hazier than in the simulations, an overall-monotonic behavior is still visible.

5 Conclusions

This paper has described an intuitive, easy-to-implement scheme for the problem of performance evaluation when document recognition results are represented in the form of a directed acyclic graph. Graph probing can be seen as having its roots in past work on heuristics for solving the graph isomorphism problem, however its utility extends beyond simply testing graphs for equivalence; it also allows us to quantify the similarity between two graphs. We presented results from three simulation studies using

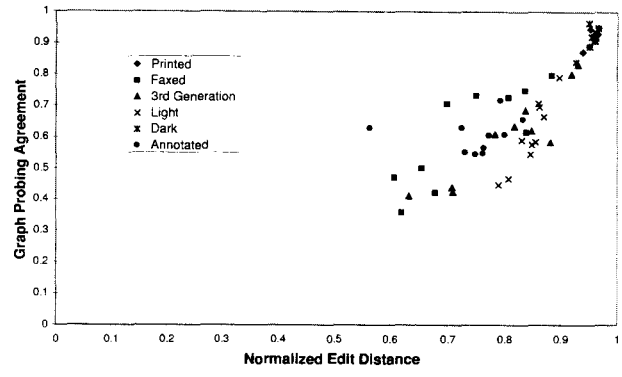


Figure 8: Graph probing agreement as a function of edit distance for the OCR experiment (60 document pages).

different graph models and an experiment employing real OCR data to demonstrate the applicability of the approach.

There are a number of ways in which this work could be extended. The design of optimal probe sets and/or weighting schemes is an open question. Beyond experimental studies, it should be possible to develop formal assertions about various classes of probes and their abilities to detect certain kinds of errors with high probability. The probing paradigm as we defined it in Section 2 is an off-line procedure (i.e., all of the probes are computed in advance, before the first probe is evaluated). Allowing the probing to take place on-line, making it adaptive, might add significant power.

Lastly, other applications could make use of this technique for graph comparison. In information retrieval, for example, queries and target documents can sometimes be represented in terms of graphs (e.g., HTML parse trees).

6 Acknowledgements

Jianying Hu and Ramanujan Kashi played important roles in the table recognition research which lead to the development of the graph probing paradigm. The graph drawings shown in this paper were generated using the **dot** tool built by Eleftherios Koutsofios and Stephen C. North [15]. The trademarks mentioned in this paper are the properties of their respective companies.

References

[1] M. Abdulrahim and M. Misra. A graph isomorphism algorithm for object recognition. *Pattern Analysis and Applications*, 1(3):189–201, 1998.

[2] S. Agne, M. Rogger, and J. Rohrschneider. Benchmarking of document page segmentation. In *Proceedings of Document Recognition and*

Retrieval VII (IS&T/SPIE Electronic Imaging), volume 3967, pages 165–171, San Jose, CA, January 2000.

[3] L. Babai, P. Erdös, and S. M. Selkow. Random graph isomorphism. *SIAM Journal on Computing*, 9(3):628–635, August 1980.

[4] D. G. Corneil and D. G. Kirkpatrick. A theoretical analysis of various heuristics for the graph isomorphism problem. *SIAM Journal on Computing*, 9(2):281–297, May 1980.

[5] G. Dudek, P. Freedman, and S. Hadjres. Using local information in a non-local way for mapping graph-like worlds. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1639–1645. Morgan Kaufmann, August 1993.

[6] G. Ellis and F. Lehmann. Exploiting the induced order on type-labeled graphs for fast knowledge retrieval. In *Proceedings of the 2nd International Conference on Conceptual Structures, Lecture Notes in Artificial Intelligence, Number 835*, pages 293–310. Springer-Verlag, August 1994.

[7] J. Esakov, D. P. Lopresti, J. S. Sandberg, and J. Zhou. Issues in automatic OCR error classification. In *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval*, pages 401–412, Las Vegas, NV, April 1994.

[8] S. Fortin. The graph isomorphism problem. Department of Computer Science Technical Report TR 96-20, The University of Alberta, July 1996.

[9] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, San Francisco, CA, 1979.

[10] J. Hu, R. Kashi, D. Lopresti, and G. Wilfong. A system for understanding and reformulating tables. In *Proceedings of the Fourth IAPR International Workshop on Document Analysis Systems*, pages 361–372, Rio de Janeiro, Brazil, December 2000.

[11] J. Hu, R. Kashi, D. Lopresti, and G. Wilfong. Table structure recognition and its evaluation. In *Proceedings of Document Recognition and Retrieval VIII (IS&T/SPIE Electronic Imaging)*, volume 4307, pages 44–55, San Jose, CA, January 2001.

[12] Y. Ishitani. Model matching based on association graph for form image understanding. In *Proceedings of the Third International Conference on Document Analysis and Recognition*, pages 287–292, Montréal, Canada, August 1995.

[13] J. Kanai. Automated performance evaluation of document image analysis systems: Issues and practice. *International Journal of Imaging Science and Technology*, 7:363–369, 1996.

[14] T. Kanungo, C. H. Lee, J. Czorapinski, and I. Bella. TRUEVIZ: a groundtruth / metadata editing and visualizing toolkit for OCR. In *Proceedings of Document Recognition and Retrieval VIII (IS&T/SPIE Electronic Imaging)*, volume 4307, pages 1–12, San Jose, CA, January 2001.

[15] E. Koutsofios and S. C. North. Drawing graphs with dot. Technical Report 59113-910904-08TM, AT&T Bell Laboratories, September 1991.

[16] B. McKay. *Nauty User's Guide (Version 1.5)*. Computer Science Department, Australian National University.

[17] B. McKay. Practical graph isomorphism. *Congressus Numerantium*, 30:45–87, 1981.

[18] B. T. Messmer and H. Bunke. Efficient error-tolerant subgraph isomorphism detection. In D. Dori and A. Bruckstein, editors, *Shape, Structure and Pattern Recognition*, pages 231–240. World Scientific, Singapore, 1995.

[19] G. Nagy. Document image analysis: Automated performance evaluation. In A. L. Spitz and A. Dengel, editors, *Document Analysis Systems*, pages 137–156. World Scientific, Singapore, 1995.

[20] J. K. Ousterhout. *Tcl and the Tk Toolkit*. Addison-Wesley, Reading, MA, 1994.

[21] C. Peterman, C. H. Chang, and H. Alam. A system for table understanding. In *Proceedings of the Symposium on Document Image Understanding Technology*, pages 55–62, Annapolis, MD, 1997.

[22] A. Sanfeliu and K.-S. Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(3):353–362, May/June 1983.

[23] R. J. Schalkoff. *Pattern Recognition: Statistical, Structural and Neural Approaches*, chapter 8. John Wiley & Sons, New York, NY, 1992.

[24] L. Shapiro and R. M. Haralick. A metric for comparing relational descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7(1):90–94, January 1985.

[25] G. Tinhofer and M. Klin. Algebraic combinatorics in mathematical chemistry. methods and algorithms. iii. graph invariants and stabilization methods. Technical Report TUM M9902, Techn. Univ. München, March 1999.

[26] B. A. Yanikoglu and L. Vincent. Ground-truthing and benchmarking document page segmentation. In *Proceedings of the Third International Conference on Document Analysis and Recognition*, pages 601–604, Montréal, Canada, August 1995.

Applications of the turbo recognition approach to layout analysis

Taku A. Tokuyasu
Computer Science Division
University of California, Berkeley, CA

Abstract

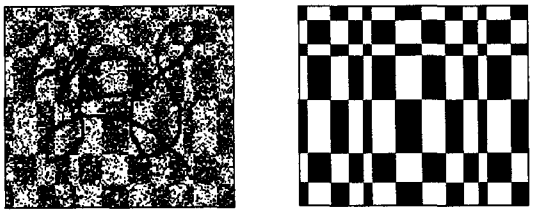
Turbo recognition (TR) is a promising approach to the nearly optimal recognition of document layout structure. We exhibit some of the features of TR and consider the space of structures that TR is able to recognize. Some preliminary experiments on segmentation and semantic labeling of scanned document images are described.

1 Introduction

Scanned document images continue to play an important role for archival purposes and the dissemination of content online, for example within new digital document formats [1]. As such, they represent intriguing targets for tasks such as document classification and information retrieval[2]. Methods for analyzing such images have been under continuous development since the advent of digital image processing. With large amounts of processing power and memory now readily available, document image analysis (DIA) has begun to explore a variety of new methods and applications.

Within DIA research, optical character recognition (OCR), in the sense of isolated character classification or recognition along a given textline, is the most highly developed. While it is arguable whether OCR is a solved problem or not, there are many other open issues within DIA, reflecting the difficulty inherent in this seemingly simple problem domain. The challenge arises from a number of sources: 1) noisy images; 2) a huge number of layouts, fonts, etc.; 3) a large variety of conceivable applications, each with different performance metrics, 4) tradeoffs between efficiency, generality, the availability and use of prior knowledge, and the desired quality of the extracted information.

Page layout analysis is one level abstracted away from OCR. Here the tasks include segmenting the page image into zones (physical segmentation) and identifying zones by semantic role (logical labeling) such as title, author, header and footer. The results



(a) (b)

Figure 2: a) Checkerboard image with 30% bitflip noise b) TR result with a checkerboard grammar with no constraint on size or number of blocks.

can be viewed as a prelude to OCR, or used directly to classify documents by type (journal article, business letter), subtype (publisher, corporate name), etc. Other applications include the detection and parsing of tables and forms, and the extraction of halftones and graphics. See e.g. [3, 4] for reviews of layout analysis methods. Layout analysis has been attracting an increasing amount of interest, as evidenced by the development of its own workshop [5].

Turbo recognition (TR) [6] views page layout analysis as statistical parsing of the structure of an image. Following in the tradition of Document Image Decoding (DID) [7], it characterizes recognition as the recovery of a (two-dimensional) message from noisy observations. The communications theory framework that TR employs for this purpose is shown in Figure 1. This figure (except for the decoder module) represents a statistical model for the generation of images. A single (two-dimensional) source message U is encoded into two ideal images X_h and X_v (where the subscripts h and v stand for "horizontal" and "vertical," respectively). These are transmitted through two noise channels, resulting in two corrupted images Y_h and Y_v. The reconciler passes these images to the decoder as a single image Y if an only if Y_h and Y_v are identical. Otherwise the reconciler passes a null image to the decoder.

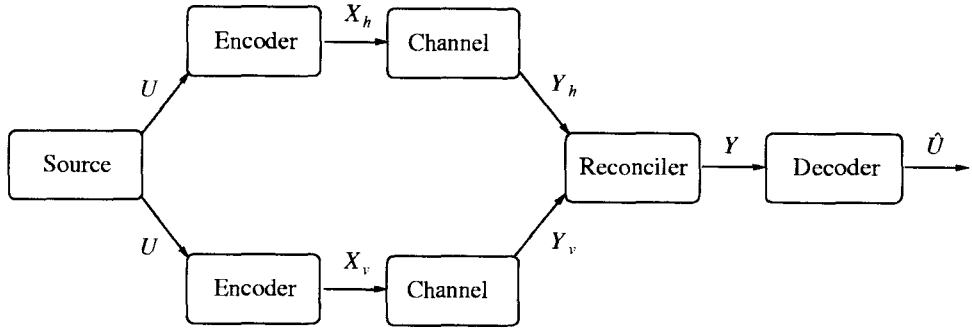


Figure 1: Communication system view of turbo recognition.

This of course is not intended to be a faithful representation of actual image generation. Rather, this framework is implicit in the decoding process which, in essence, attempts to invert the above generation scheme. Similar reasoning applies in the use of, e.g., HMMs for speech recognition.

The objective of the decoder then is to recover the message field U, given the observed data Y and prior knowledge (the parameters) embodied in the generative model. The minimum probability of error is achieved when the decoder returns the field $\hat{U}$ that maximizes the posterior, $\hat{U} = \arg\max_U P(U|Y)$.

The decoding algorithms used by TR are derived from graphical models [8], as applied in particular to turbo codes [9] (from which TR gets its name). Decoding consists of applications of the forward-backward algorithm, once per row and once per column, iterated to convergence. As reported previously [14], TR produces results that are nearly statistical optimal. [1]

Within DID, a general layout analysis framework was proposed in [10], which employed probabilistic 2D context-free grammars. The TR generative model on the other hand is based on two sets of finite state grammars in parallel. TR can thus be considered as a specialization of the work in [10] to a subclass of layouts (see below). The basis in finite state grammars allows TR to be considerably more efficient, and also means TR is quickly retargetable.

The problem of statistical interpretation of two dimensional images is of course a longstanding one. In the present context, related techniques include Markov random fields (MRFs) [11], pseudo-2D hidden Markov models [12], and 2D HMMs (a kind of causal MRF) [13]. TR differs from such local methods in being able to embody non-local constraints, such as the notion that a rectangle has straight edges.

Figure 1 shows a checkerboard pattern, akin to a kind of table, corrupted with 30 percent bit-flip noise and an extended pencil mark. Using a grammar for such patterns, where the number and size of the individual blocks is not fixed, TR easily recovers[2] the original clean image. This shows the ability of TR to integrate information over an entire image before returning an estimate. Note that local estimates of the individual checkerboard blocks are unlikely to result in a pattern which is globally optimal (or even consistent).

2 A menagerie of grammars

An ongoing enterprise is to explore the space of structures that can be parsed by TR. In this section we go through a variety of different grammars in order to illustrate some of the possibilities. Our main focus here is thus on abstract (ideal) shapes.

2.1 One rectangle

The standard one-rectangle grammar that we use has the following form.

$$\begin{array}{rl} \textit{horizontal}: & a^+ \mid b^+c^+b^+ \\ \textit{vertical}: & a^+b^+a^+ \mid a^+c^+a^+ \end{array}$$

Here a and b transduce to 0 (white), and c transduces to 1 (black). Given such a grammar, TR parses the image shown in Figure 3a in the manner shown in Figure 3b. A rectangle can be described in many other ways. For example, a grammar such as the following

$$\begin{array}{rl} \textit{horizontal}: & a^+b^+a^+ \mid c^+d^+c^+ \\ \textit{vertical}: & a^+c^+a^+ \mid b^+d^+b^+ \end{array}$$

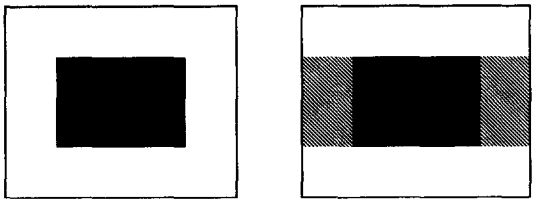
is aesthetically pleasing because it treats horizontal and vertical directions symmetrically. A drawback of such grammars is that the edges between regions extend to the boundary of the canvas. The following "one-layer" grammar solves this problem:

$$\begin{array}{rl} \textit{horizontal}: & a^+ \mid (\ a^+(b^+ \mid\ c\ d^+c\ ))^+a^+ \\ \textit{vertical}: & a^+ \mid (\ a^+b\ (c^+ \mid d^+)\ b\ )^+a^+ \end{array}$$

[1] As in the case of turbo codes, TR is not guaranteed to recover the optimal result, though empirically its performance is nearly so. See [15] for theoretical progress on this point.

[2] Run time in Java with no attempt at optimization is about 10 seconds on a 300MHz Pentium PC.

where a represents the background, and b and c describe a rectangular shell one pixel wide which surrounds the black pixels described by d. This localizes the symbols which enforce rectangularity to the immediate vicinity of the rectangle, a feature which will be useful below.



(a) (b)

Figure 3: (a) Original one-rectangle image; (b) Partition induced by the standard TR grammar

2.2 Generalizations

The standard one-rectangle grammar can be generalized to a grid structure with an arbitrary number of columns and/ or rows with the addition of a few Kleene + symbols:

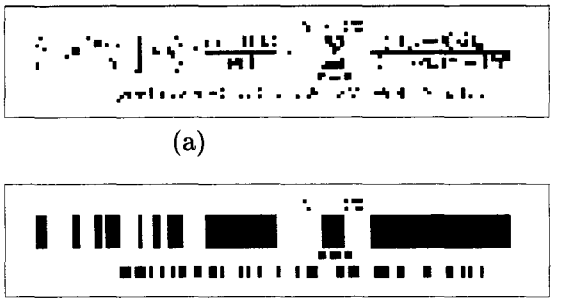
$$\begin{array}{rl} \textit{horizontal}: & a^+ \mid (\, b^+c^+)^+b^+ \\ \textit{vertical}: & (a^+b^+)^+a^+ \mid (a^+c^+)^+a^+ \end{array}$$

This describes rectangular blocks whose edges are perfectly aligned, both horizontally and vertically. Variations on this theme, where e.g. some cells in the grid are absent, are easily produced.

The alignment between blocks across rows can be relaxed using a generic "multirow" grammar:

$$\begin{array}{rl} \textit{horizontal}: & a^+ \mid (\, b^+c^+)^+b^+ \\ \textit{vertical}: & (a^+(b^+ \mid c^+))^+a^+ \end{array}$$

A somewhat unusual application of this grammar is shown in Figure 4. This shows a mathematical expression at very low resolution and its TR decoding (where only the regions associated with the c symbol are displayed in black). TR in this case is focussed on "clumping" black pixels into rectangles in a manner consistent with the grammar. This is largely in accord with typesetting conventions, allowing TR to reduce the chaotic collection of dots into readily interpretable structure. For instance, the fact that this expression consists of two linear sequences, a large one on top and a smaller one below, can be extracted with minor postprocessing. The centerline, a hypothetical reference line passing horizontally through the center of the formula, can then be determined, which is crucial for any subsequent parsing [16]. In addition, the two fractions appear as prominent blocks, and the summation symbol with limits above and below is easily identified. While this does not demonstrate that TR is suitable for an actual mathematical parsing task, it is gratifying that such a summarization of image structure comes about with very little effort, with no need to build in application-specific knowledge via thresholds, etc.[3]



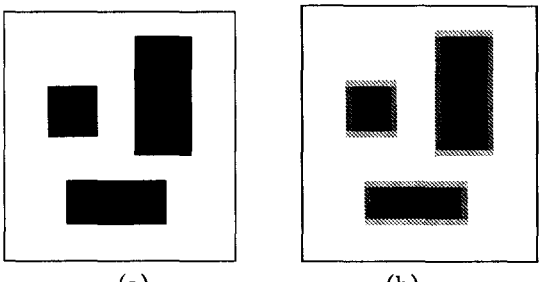
(a)

(b)

Figure 4: (a) Mathematical expression at low resolution; (b) TR decoding (c-symbol only).

When alignment between blocks is not pertinent at all, it is convenient to switch to the above-mentioned one-layer description of rectangles. This allows the rectangles to be freely aligned relative to each other, as shown in Figure 5.

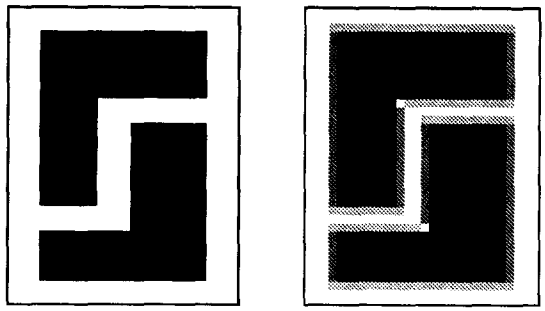
A general Manhattan layout can be described with the inclusion of an additional input symbol, which denotes interior (concave) corners. An application of such a grammar is shown in Figure 6. The input alphabet in this case has five symbols, and the horizontal (vertical) transducer has 7 (6) states and 13 (12) transitions, respectively.



(a) (b)

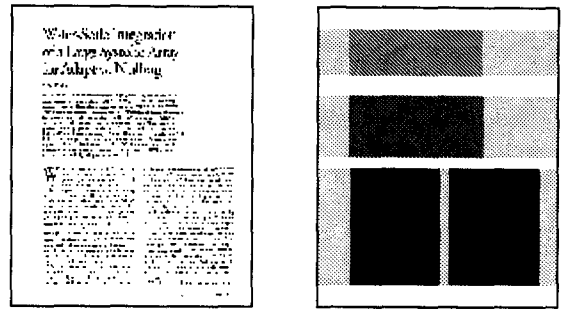
Figure 5: (a) Freely aligned rectangles; (b) TR decoding using a one-layer grammar.

[3]This result is reminiscent of run-length smoothing (RLS) [17] in either the horizontal or vertical direction (or both). It is unlikely that an RLS approach could exactly reproduce the structure in Figure 4, since the amount of smoothing required would vary across the image.



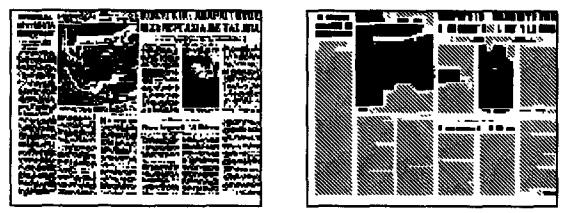
(a) (b)

Figure 6: (a) Original image with Manhattan layout; (b) TR decoding.



(a) (b)

Figure 7: a) Journal article title page; b) TR decoding.



(a) (b)

Figure 8: (a) Newspaper clipping; (b) TR decoding (not converged).

3 Work in progress

We illustrate our future research directions with some preliminary results. Metadata extraction from journal article title pages is an important application of layout analysis [18]. Figure 7 shows a scanned article title page and the TR result, using a custom grammar which describes two one-column blocks on top of a two-column block. The different shades of gray in the top two blocks denotes the fact that they have been decoded by different input symbols. The TR insistence on grammatical interpretations suggests that it can boost overall zoning performance, even on such clean images.

Figure 8 shows a newspaper clipping and an intermediate TR result. The grammar allows freely aligned rectangles of two types, one with a larger probability of producing black pixels than the other. We see that TR is performing a rudimentary texture classification of the image. The parse in fact fails on the next iteration, for reasons which are not yet fully understood. We are nevertheless encouraged by this initial result.[4]

We note that many extensions of the present implementation of TR are possible. This includes integrating layout analysis and OCR, and applying TR to grayscale images. We are also investigating further applications where the (logical) decoding capabilities of TR can be put to good use.

4 Acknowledgments

I thank Richard Fateman and Phil Chou for conversations. This work was supported by the Berkeley Digital Library Project under NSF grant number CA98-17353.

[4]The problem arises when one pass assigns a high probability to a configuration that is in fact ungrammatical in the orthogonal direction. This suggests an underflow problem of some kind, which we are presently investigating.

References

[1] R. Phelps, T.A.and Wilensky. Multivalent documents. *Communications of the ACM*, 43(6):82–90, June 2000.

[2] J. Hu, R. Kashi, and G. Wilfong. Comparison and classification of documents based on layout similarity. *Information Retrieval*, 2(2-3):227–43, 2000.

[3] R.M. Haralick. Document image understanding: geometric and logical layout. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 385–90. IEEE Comput. Soc. Press, 1994.

[4] R. Cattoni, T. Coianiz, S. Messelodi, and C. M. Modena. Geometric layout analysis techniques for document image understanding: a review. January 1998. ITC-irst Technical Report #9703-09.

[5] Document layout interpretation and its applications. http://www.science.uva.nl/events/dlia2001/.

[6] Taku A. Tokuyasu and Philip A. Chou. An iterative approach to document image analysis. In *DLIA99 workshop, Bangalore, India*, 1999. http://www.science.uva.nl/ events/dlia99/.

[7] G. E. Kopec. Document image decoding in the UC Berkeley digital library. In *Document Recognition III, Proc. of the SPIE*, volume 2660, pages 2–13, 1996.

[8] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference.* Morgan Kaufmann, San Francisco, 1988.

[9] Brendan J. Frey. *Graphical models for machine learning and digital communication.* The MIT Press, Cambridge, MA, 1998.

[10] P.A. Chou and G.E. Kopec. A stochastic attribute grammar model of document production and its use in document image decoding. In *Document Recognition II, Proc. of the SPIE*, volume 2422, pages 66–73, 1995.

[11] Ross Kindermann and J. Laurie Snell. *Markov random fields and their applications.* American Mathematical Society, Providence, R.I., 1980.

[12] O.E. Agazzi and S.-S. Kuo. Pseudo two-dimensional hidden markov models for document recognition. *AT&T Technical Journal*, 72(5):60–72, Sept-Oct 1993.

[13] A. Najmi Jia Li and R.M. Gray. Image classification by a two-dimensional hidden markov model. *IEEE Transactions on Signal Processing*, 48(2):517–33, Feb 2000.

[14] Taku A. Tokuyasu and Philip A. Chou. Turbo recognition: a statistical approach to layout analysis. In *Document Recognition VIII, Proceedings of the SPIE*, volume 4307, pages 123–129, 2001.

[15] Y. Weiss and W.T. Freeman. On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs. 1999. MERL TR99-39. to appear in IEEE Transactions on Information Theory.

[16] R.J. Fateman, T. Tokuyasu, B.P. Berman, and N. Mitchell. Optical character recognition and parsing of typeset mathematics. *J. of Visual Communication and Image Representation*, 7(1):2–15, March 1996.

[17] K.Y. Wong, R.G. Casey, and F.M. Wahl. Document analysis system. *IBM Journal of Research and Development.*

[18] Jongwoo Kim, Daniel X. Le, and George R. Thoma. Automated labeling in document images. In *Document Recognition VIII, Proceedings of the SPIE*, volume 4307, pages 111–122, 2001.

Indexing and Retrieval

A Conceptual Model of Image Similarity

Nigel Dewdney

U.S. Department of Defense
njdewdn@afterlife.ncsc.mil

Abstract

There is, in general, no consensus for the definition of similarity in images. Duplicate detection and image classification algorithms display different capabilities on different test corpora. This paper argues that many types of variation exist across images. Since variation types will differ in importance with application, image similarity is modeled here as a set of these variation types defined to be orthogonal to one another. Metrics can then be designed to measure different variations independently, and may be combined to give an overall measure of image similarity. The potential for this model is illustrated by applying a partial solution to an example image set.

1. Rationale

Image similarity definition often depends on the image processing application being considered, if it is addressed at all. For the purposes of this paper similarity is considered to be multi-dimensional with a concept of overall similarity being that of a vector in that space. The fact that different people may be interested in different aspects of images is recognized. For example in the Blobworld [1] image retrieval system a user may specify an identified region from an image for a query and weight aspects of that region. Results are mixed though with many images semantically different from the query image being returned [2].

The problem of measuring image similarity is not restricted to photo repositories. In the more restricted problem domain of duplicate, and near duplicate detection in scanned document corpora many algorithms exist with variable performance. Rogers et. al. describe an algorithm within Mathsoft's DocBrowse [3] which achieves over 90% precision at 100% recall when used against their own test set, but shows marked deterioration in precision beyond 50% recall when used against the University of Washington English/Japanese document image database.

Scanned documents form a sub-set of the image domain in which duplicate detection can be viewed as a constricted image retrieval task. Prueitt has presented a formalism for document duplicate detection that goes beyond the exact duplicate detection problem [4]. In so doing he analyses similarity within the realm of near duplication which is defined as an intentional alteration from one image to another or from a common unseen parent image. It is recognized that images may be similar but not duplicate at all. The 4-tuple metric used in the formalism presented is designed for human judgments on how similar documents are. This paper does not address how similarity should be measured beyond classification though. The boundaries between exact, near, non-near, and different must be fixed in order for the classification to be made.

Prueitt takes his analysis further and considers what he calls the Mosaic effect [5], that is where parts of one image become disjoint, and may even overlap, when observed in a paired image. This work begins to analyze one way in which two images may be similar, considering an image to be a set of objects. It is recognized though that the boundaries of objects, if not their very definition, depends on perspective, i.e. no uniform segmentation exists.

Image retrieval and scanned document duplicate detection require images to be compared to a target image, whereas image classification applications require that an image be compared to one or more models of an image class. In image classification applications, the question of whether an image is a near duplicate or not does not arise. Either the image is of the class or it is not. But to classify an image it is required that the image is similar in some way to a model built for that class. Similarity metrics need to be selected appropriate to the features extracted (which also should be selected with the application in mind). Vasconcelos and Lippman observe that many similarity metrics have been proposed for content based image retrieval but with little consideration for their underlying assumptions about the data [6]. They

demonstrate that these distance metrics are sub–classes of minimizing the probability of a retrieval error.

This paper argues that image similarity should be viewed as a multi–dimensional space; each dimension representing an independent type of image variation. Considering ways in which semantically similar images may vary can improve retrieval performance as shown by Natsev et. al. [7] In this work Natsev describes an image retrieval system, WALRUS, which breaks down a query image into regions and computes wavelet signatures for each region. During retrieval all signatures are compared with all signatures of candidate images. In the system this allows for different scales and locations of objects within images. On a test set of 10,000 images a marked improvement is shown by example over WBIIS [8] in perceived image retrieval results.

2. Constructing metrics for image classification and duplicate detection

In designing a general model upon which image comparison applications can be built several factors should be taken into consideration. This section gives an overview of the factors considered in the design of the model presented here.

Different image corpora tend to each have some commonality, e.g. all outdoor scenes, or all images of people, usually because they originate from a limited source. This may be by design, for particular applications, or due to availability. The common factors in one corpus may not be present in another corpus. These factors may not be relied upon, either explicitly or implicitly, when comparing images to one another in general. This suggests that particular types of difference between two images should be measured independently of one another, therefore, as assumptions of invariance can not be made. (Such assumptions typically made are page orientation, image is de–skewed, and resolution is 600dpi.)

Image classification and duplicate detection are related: An image pair satisfying the requirement for the latter would meet the requirement for the former. The constraints for a duplicate, or near duplicate, classification are generally tighter than in more general image classification tasks. This has the consequence that variance as a proportion of the class space is much higher though. In some applications the proportion may be greater than 1.

It ultimately makes little sense to think of classifying two images as duplicate beyond those that are bit–for–bit identical. The presence of noise in images leads to the desire to find underlying duplicate images. However, only the images of the original image are available. In classifying duplicates in the presence of noise it is necessary to allow an arbitrary amount of noise. This allowance consequently allows for image differences which are intentional. It is then impossible, in general, to determine whether differences between two images are due to noise or to an author's alteration without being able to consistently classify the nature of the difference. Furthermore, if it can be detected that the difference is in fact a change in image content rather than noise, the nature of that alteration requires that interpretation be carried out in order to distinguish between "near duplicate" and "different document". This is best illustrated by an example. Consider the four sentences below:

1. The President will not visit France next week.
2. The President will nod visit France next week.
3. The President will now visit France next week.
4. The President shall not visit France next week.

In terms of information sentences 1 and 4 are the same but have the highest distance. It could be supposed that sentence 2 has an error and is duplicate of sentence 1, but sentence 3 has the same distance and has completely different information. True, the example is contrived for illustration, but as higher degrees of difference are allowed in order to find underlying duplication the more likely it is to suffer from this effect.

Duplicate detection should not, then, be approached as a tightly bound classification problem if reliable results are required. This is not to say that information should not be obtained from image comparisons though. A distance between two images can be calculated *irrespective* of whether a human judge would consider the pair duplicate or not. The way in which the distance is measured should reflect what is important in the application. This suggests a collection of different metrics or a parameterized metric. In bulk processing applications results can be rank ordered or thresholded. Thresholds can be determined from desired levels of accuracy and training sets. The question then arises of how images may vary and how to measure those differences.

Different image pairs may vary by different types of transform processes. This has the consequence that the distance between a pair depends on how those

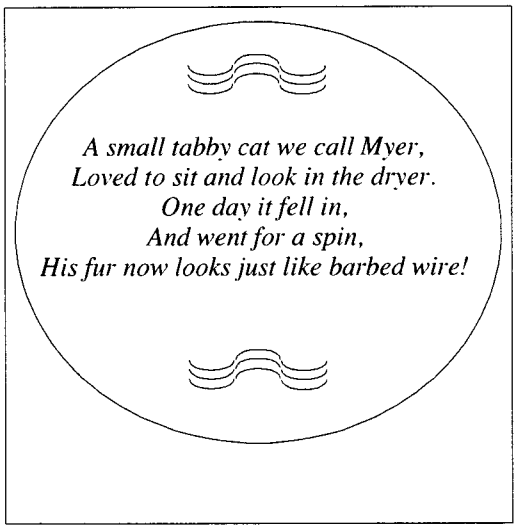


Figure 1

Figure 2

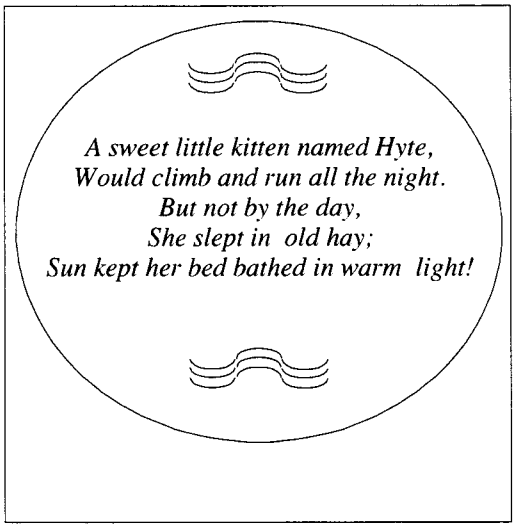


Figure 3

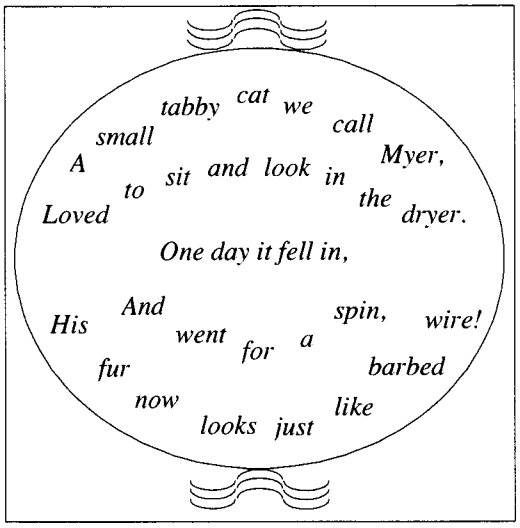


Figure 4

difference types affect the metric being used to measure the distance. Consider the image examples shown in Figures 1 through 4. Figure 1 shows an image of a limerick. The same limerick is shown in Figure 2. Pixel distance between the two is high since a different font is used, but the information is the same. One could argue that the layout of the pixels relative to one another is broadly the same between the two, but compare the images in Figure 1 and Figure 4: Here the font is the same but the layout is very different. Finally compare Figure 1 and Figure 3: At first glance these appear to be duplicate, as the pixel distance and the layout differences are small, but the information content is completely different. Duplicate detection algorithms which do not analyze content tend to fail on such examples because of the low variance in image "texture". (Think of comparing different pages from the same book at arms length.) Again one could argue that comparing pixel structure on the scale of the text would reveal the difference, but this requires that the font size be known and would not generalize.

3. Defining dimensions of image similarity

Since images, in general, may display similarity or differences in many different ways, we need to define these types of differences such that we can design metrics for them. Ideally these definitions should be dimensions, through which comparison metrics can be taken, which are orthogonal to one another. This work has defined six dimensions.

Layout: An image is comprised of the objects within it. In a scanned document such objects may be address blocks, signature blocks, titles, paragraphs, etc. The "layout" in an image is its set of comprising objects and their positions relative to one another.

Color: Traditionally scanned documents are binary bit–maps. Increasingly scanned documents may be gray–scaled or even color. The "color" of an image is the set of color values of the comprising pixels.

Representation: In scanned documents "representation" is most commonly the font of the text making up the document. However, representation goes beyond font and includes any form of symbolic representation. For example, a dashed line or solid thick line, to represent a footpath in a map.

Orientation: Angle of skew from upright. Note this includes page orientations of 90, 180, and 270 degrees. (Note that orientation is global to the image whereas skew local to objects is accounted for in the definition of Location.)

Scale: For scanned images this is the scanning resolution. For non scanned images scale becomes arbitrary and requires relative measurement of common objects between images.

Information. The "semantic" content of the image. For text documents this is what the text states. The definition for non–text is not so easily arrived at however. This will depend on the information that can be about object(s) portrayed without world knowledge.

A system which compared images for similarity, based on the model described here, would require methods for measuring similarity along each of the dimensions. Ideally a real valued distance would be computed for each. The distances could be weighted, if desired, and geometrically combined to give an overall distance. Thresholds could be assigned to each dimension, for duplicate detection applications for example, a combination of which would form a complex hull decision boundary on the overall distance. Alternatively, a simpler approach would be to normalize the dimensions and form a spherical decision boundary as illustrated for three dimensions in Figure 5. The Euclidian distance between the two subject images is geometric combination of the similarity function values, the decision boundary set to an arbitrary value, and, if each dimension metric is in the range [0,1], can be normalized:

$$\text{Image distance} = \left(\Sigma_{d=1..D} [\Delta_d(A,B)] / D \right)^{1/2} \qquad \text{eqn.(1)}$$

where Δ_d is the similarity distance function for dimension d, and A and B are the two subject images. D is the number of dimensions being considered in the similarity space.

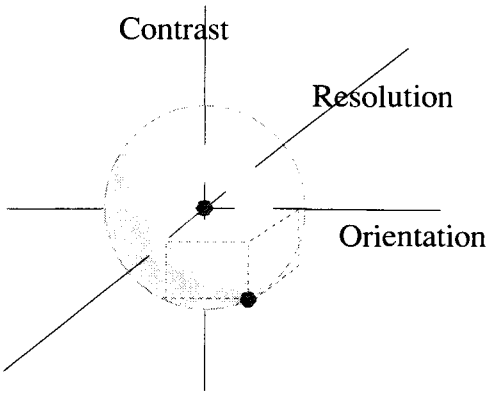


Figure 5: Spherical decision boundary in similarity space.

It should be noted that this concept does not preclude analysis of partial duplication where one image contains part of, or includes, another image. The "Layout" dimension allow for identification of regions which may then be compared on an individual basis. Set operators and logic and then be used, or individual distances combined, to further analyze the nature to the relationship between the two images.

Image classification systems would also require similarity in one or more dimensions. Measurements in each dimension could be used as high level features. These features could be weighted, according to application, and used, perhaps with further low level features, as input to a statistical classifier.

4. Application of model

Metrics for all of the dimensions proposed here for similarity space have yet to be defined. Use of a sub–space can be made to demonstrate how the model can add value. An example application of the model using just the orientation and resolution dimensions follows.

Skew detection algorithms typically measure angle of skew based on such features as detected edges. Define a metric *orientation = angle of skew from upright vertical / 180* where angle is measured in degrees. Orientation similarity, or distance, between two images is then the difference in orientation.

$$O(A,B) = | o(A) - o(B) |$$

eqn. (2)

Image resolution is sometimes given in an image header block, such as in the TIFF standard, but resolution could be more generally defined for wider applicability. Define *resolution = mean number of pixels / square root of the number of connected components.* Note that connected components are readily defined for binary images but are open to interpretation for color images because non–identical pixels may be included in what is considered as an object. For simplicity, here, consider connected components to be made up of identical pixels. Define resolution similarity, or distance, as the difference in resolution of two images normalized by mean of the resolutions.

$$R(A,B) = 2 * (| r(A) - r(B) |) / (r(A) + r(B))$$

eqn. (3)

Consider a corpus comprising images scanned from an original at multiple resolutions. Call this corpus 1. Consider a second corpus comprising images scanned from an original at the same resolution but at variable angles. Call this corpus 2. Call the combination of the two corpora corpus 3.

In corpus 1 the resolution detection algorithm will give the result that the images are different (and give a measure of how different each pair is). The orientation detection algorithm will not show any differences, however, leading the user to believe that the images are the same. The reverse situation is true in corpus 2. The resolution detection algorithm will yield similar, if not exact, distances for each image pair. The orientation detection algorithm will give the difference between each image pair.

Mixed results will be observed from applying either algorithm to corpus 3. The resolution detection algorithm will only detect differences in resolution so that images with different angles will be considered duplicate, and the orientation detection algorithm will only detect differences between images with different skew so that images scanned at different resolutions (at the same angle) will be considered duplicate. (Note that all the images are different. The desirability to ignore skew or resolution in duplicate detection is application specific. One can zero–weight an individual metric under this model.) Use of both the similarity metrics yields differences between image pairs irrespective of the corpus they originally came from.

Figure 7 shows four images of Abraham Lincoln. Image 1 is a 64–dpi image with no skew. Image 2 is a 64–dpi image with 15° skew to normal. Image 3 is a 64–dpi image with no skew but a different pixel color to that in Image 1. Image 4 is a 128–dpi image with no skew. Applying the similarity metrics of resolution and orientation to each image pair yields the results shown in Table 1. A skew correction algorithm and a connected component algorithm were used to calculate these values. (The ideal figures are given in Table 2.) Note that the image pair with the greatest amount of variation, Image 2 and Image 4, yields the largest image similarity. Image 1 and Image 3, whose only difference is pixel color, has a zero image similarity distance because image color is not considered.

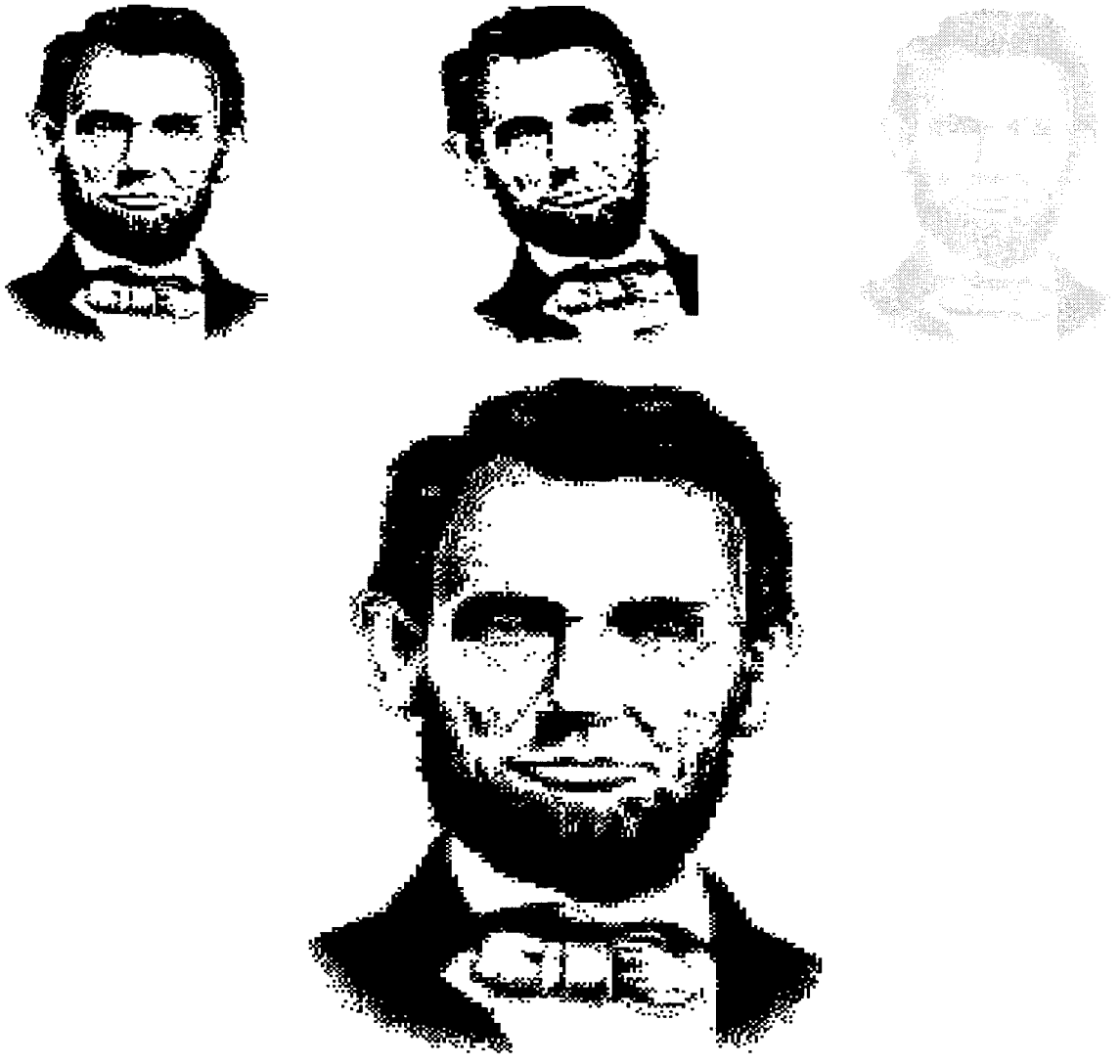


Figure 7: Different Abraham Lincoln images

Table 1: Results of applying 2 similarity metrics and the combination thereof to Lincoln images in Figure 7

Pair	*Resolution function.*	*Orientation function.*	*Combined Similarity*
(1,2)	0.0326	0.0597	0.0481
(1,3)	0.0000	0.0000	0.0000
(1,4)	0.5571	0.0000	0.3939
(2,3)	0.0326	0.0597	0.0481
(2,4)	0.5871	0.0597	0.4173
(3,4)	0.5571	0.0000	0.3939

Table 2: Ideal values applying 2 similarity metrics and the combination thereof to Lincoln images in Figure 7

Pair	*Resolution function.*	*Orientation function.*	*Combined Similarity*
(1,2)	0.0000	0.0833	0.0589
(1,3)	0.0000	0.0000	0.0000
(1,4)	0.6666	0.0000	0.4714
(2,3)	0.0000	0.0833	0.0589
(2,4)	0.6666	0.0833	0.4750
(3,4)	0.6666	0.0000	0.4714

5. Conclusions

A pair of images may indeed be exact duplicates. It is, in general, impossible to tell whether two images are near duplicates or two images considered as different but have significantly similar values for measured features. The presence of noise only serves to confound the problem. If a pair is not exactly duplicate then they can only be said to be similar. But image pairs may be similar in different ways. Methods of comparing images tend to conflate these dimensions of similarity, or make assumptions about invariance within some of them.

This paper has argued that different dimensions of image similarity should be treated independently so that they can be weighted according to application. Having independent metrics also allows analysis of why two images are deemed similar and to what extent. Six such dimensions have been identified here. An overall metric of image similarity can be defined as a geometric combination of each of the individual dimension metrics.

Use of multi–dimensional similarity metrics has been demonstrated by the use of two dimensions on a small set of test images, with encouraging results. Future work will concentrate on developing algorithms to measure distance in each of the dimensions identified and to evaluate the model on broad ranging image corpora.

6. References

[1] Carson, C., Thomas, M., Belongie, S., Hellerstein, J. M., Malik, J., *Bolbworld: A System for Region–Based Image Indexing and Retrieval*, Proceedings of International Conference on Visual Information Systems, Amsterdam, Netherlands, 1999.

[2] http://elib.berkeley.cs.edu:80/photos/blobworld

[3] Rogers, R., Chalana, V., Marchisio, G., Nguyen, T. and Bruce, A., *Duplicate Document Detection is DocBrowse*, Proceedings of 1999 Symposium on Document Image Understanding Technology, University of Maryland, April 1999.

[4] Prueitt, P.S., *The 4 by 4 Duplicate Document Detection (D3) Formalism*, Proceedings of 1999 Symposium on Document Image Understanding Technology, University of Maryland, April 1999.

[5] Prueitt, P.S., *Similarity Analysis and the Mosaic Effect*, Proceedings of 1999 Symposium on Document Image Understanding Technology, pp315–320, University of Maryland, April 1999.

[6] Vasconcelos, N. and Lippman, A., *A Unified View of Image Similarity*, in Proceedings of International Conference on Pattern Recognition", Barcelona, Spain 2000.

[7] Natsev, A., Rastogi, R., Shim, K., *WALRUS: A Similarity Retrieval Algorithm for Image Databases*, Proc. ACM SIGMOD conference on management of data, Philadelphia, 1999.

[8] Wang, J., Wiederhold, G., Firschein, O. and Sha Xin Wei, *Content Based Image Indexing and Searching Using Daubechies' Wavelets*, International Journal of Digital Libraries, Vol. 1, No. 4, pp311–328, 1998.

Recognize, Categorize, and Retrieve

Kazem Taghva, Thomas A. Nartker, and Julie Borsack
Information Science Research Institute
University of Nevada, Las Vegas

Abstract

A successful text categorization experiment divides a textual collection into pre-defined classes. A true representative for each class is generally obtained during training of the categorizer.

In this paper, we report on our experiments on training and categorization of optically recognized documents. In particular, we will address the issues regarding the effects OCR errors may have on training, dimensionality reduction, and categorization. We further report on ways that categorization may help error correction and retrieval effectiveness.

1 Introduction

Retrieving relevant information from a large textual corpus is a challenging task and entails many manual and automated efforts. Query construction and training of categorizers are two prominent examples of manual efforts while document indexing and clustering would be considered automated. Another expensive but widely used manual effort is the assignment of a controlled vocabulary to predefined categories for documents in the corpus. Expert searchers can then use these categories and controlled vocabularies to formulate more compact and effective queries [4].

Although these manual efforts will continue for the foreseeable future, it is of great interest to establish automated techniques to assign controlled vocabularies and categories to new documents based on similar documents in the corpus. One of the proven approaches is to use an already categorized set of documents to build categorizers for future document classification.

In a large textual repository such as the Licensing Support Network (LSN) being built by the Department of Energy (DOE), many of the documents are recognized using commercial OCR engines. The OCR errors typically result in an index which is considerably larger than keyed text. Also, for some queries, one may not be able to find certain relevant documents due to poor quality OCR that generates a high number of errors[13, 14]. Although these errors do not affect average precision and recall, they may produce variations in ranking[12]. But in general, studies show that one can work with OCR text in an information retrieval (IR) environment with few adverse consequences.

In a similar situation, one must know what effects these errors may have on automated text classification. In particular, we want to know what effects (if any) these errors may have on training the classifier to build the categorizers. We would also like to discover if errors cause improper categorization for new documents.

In section 2 of this paper, we give a brief introduction to the work done in the area of OCR and IR. In Section 3, we describe both the Bernoulli and multinomial Bayes categorizers. Sections 4 and 5 report on our categorization experiments in the presence of OCR errors. Finally, in section 6 we give our conclusion and future work.

2 OCR and Information Retrieval

In the early 1990's the use of OCR devices became more widespread for the conversion of printed material to electronic form. At this time, ISRI was heavily involved in OCR system comparison, testing, and research. What became clear was that eventually, this data was to be loaded into an IR system for subsequent retrieval. Initially, it was thought that manual correction was required to bring this OCR generated text to a level of "retrievability." In the case of the LSN, text accuracy was to be no less than 99.8% accurate.

ISRI questioned the necessity of this level of accuracy and the "Noisy Data" experimentation began. Since our first publication on this topic in *JASIS* in 1994[15], ISRI has performed hundreds of experiments involving optically recognized data to discover its effect on related technologies. Nearly all of our studies have pointed to the same conclusion: In

general, using OCR text has little effect on average precision and recall when compared to re-keyed or manually corrected text. This was a consequential result, in particular for collections such as the LSN, because re-keying millions of pages would have been a tedious and expensive task.

With this result in mind, some of our other studies did a little more investigation into the effects of OCR on IR. For example, we found that the index of an OCR collection can be as much as five times the size of a clean data set and that most of this overhead was of no value for retrieval. Further, formulas used to calculate term weights can be affected in several ways by erratic term frequencies in OCR generated text[12, 13]. This in turn affects document ranking. We also found that short documents in particular are affected because of the lack of redundancy in the text. So although our results on average precision and recall holds in nearly every test we performed, there are some considerations when a collection is made up of OCR data.

The categorization experiments we report on here are similar to our IR experiments we have done in the past. First, the data set is derived from documents pertaining to the LSN; the documents tend to be long journal documents with a scientific flavor. Classification software such as *BOW*[8] constructs an index and then uses its categorizers (based on the training sets) as vectors to determine the classification of incoming documents. In this sense, a categorizer is similar to a vector query in the IR domain.

But once a document has been classified properly, even if the OCR is poor, more information about that document is now known. In this case, it may be feasible to apply more specific and exhaustive automatic error correction to documents within a category. We see ample potential in the continued research of OCR, categorization, and other related technologies.

3 Probabilistic Classifiers

There are two distinct approaches to automatic text classification. The first approach is based on machine learning techniques. In this method, the system is given a set of training documents for each category. These documents are used to typically generate a set of propositional Horn clauses that will be used to classify future documents [3, 10, 1]. The second approach is based on traditional IR techniques. In this method, the training documents are used to form an ideal document which represents each category. These ideal documents are known as *categorizers*[8, 7, 6, 4]. The system uses similarity measures between incoming documents and the categorizers to classify these new documents properly. Our experiments only pertain to the latter approach.

Let $V = \{v_1, v_2, \ldots, v_{|V|}\}$ be the set of words in a lexicon. Each document, and each categorizer, can be represented as a vector of the form $(w_1, w_2, \ldots, w_{|V|})$, where each component w_t of this vector represents the weight of the term v_t in the document and categorizers. In its simplest form, w_t can be either 0 or 1. In this case, the weight represents the presence or absence of the term v_t in the document. This weight though can carry more information such as the frequency of the term in the document.

Now, let $C = \{c_1, c_2, \ldots, c_{|C|}\}$ and $D = \{d_1, d_2, \ldots, d_{|D|}\}$ be sets of categories and training documents, respectively. Each category c_j, is represented by a vector of the above form, where the weight, w_t, is calculated from using term frequencies based on the training set of the documents. Using the naive Bayes assumption that the probability of each word occurring in a document is independent of the occurrence of other words in a document [9], then these weights can be easily calculated. In our experiments, we focus on both the Bernoulli and multinomial methods.

In the Bernoulli method, the frequency of the words do not play any role. Hence, each document is represented by a vector of the form $d_i = (B_{i1}, B_{i2}, \ldots, B_{i|V|})$, where each B_{it} is either 1 or 0. In this case, the weight of each component of the categorizer c_j is calculated using the following formula:

$$P(w_t|c_j) = \frac{1 + \sum_{i=1}^{|D|} B_{it} P(c_j|d_i)}{2 + \sum_{i=1}^{|D|} P(c_j|d_i)} \quad (1)$$

In other words, the weight of the term w_t given category c_j is obtained by dividing the number of documents containing the term v_t and in category c_j by the total number of documents in the category c_j.

Now, the probability of a new document d_i belonging to category c_j is calculated by the following formula:

$$P(d_i|c_j) = \prod_{t=1}^{|V|} (B_{it} P(w_t|c_j) + (1 - B_{it})(1 - P(w_t|c_j))) \quad (2)$$

In the multinomial model, the frequency and the length of the document (i.e. the number of words in the document) play a role. In this setting, a document d_i is represented with a vector of the form $d_i = (N_{i1}, N_{i2}, \ldots, N_{i|V|})$, where N_{it} is the frequency of the term v_t in the document d_i. If we use the notation $|d_i|$ for the length of a document, then the following formulas represent the corresponding calculations for the multinomial model.

$$P(w_t|c_j) = \frac{1 + \sum_{i=1}^{|D|} N_{it} P(c_j|d_i)}{|V| + \sum_{s=1}^{|V|} \sum_{i=1}^{|D|} N_{is} P(c_j|d_i)} \quad (3)$$

$$P(d_i|c_j) = P(|d_i|)|d_i|! \prod_{t=1}^{|V|} \frac{P(w_t|c_j)^{N_{it}}}{N_{it}!} \quad (4)$$

In practice, there are various methods to decrease the dimension of $|V|$. These methods are known as *dimensionality reduction* techniques which tend to improve the performance of a categorization system. These are further discussed in Section 4.

4 Experimental Environment

Unlike most classification experiments, we do not use a standard categorized collection like *Reuters*[5]. Instead, our testing is dictated by the needs of the Department of Energy and the LSN. They have millions of optically recognized documents and terabytes of email that need to be classified into *Regulatory Guideline 3.69*[2]. This guideline determines which documents and email messages are required for the licensing proceedings of the High-Level Radioactive Waste Repository. The documents we use in these experiments are a subset of this collection. Our experimental environment consists of:

BOW text classifier from CMU: a statistically-based text categorization system applying the probabilistic naive Bayes model [8, 9]. BOW offers several ways to reduce dimensionality of classes. They include:

- **Default:** removes no words from the vocabulary.
- **Document Count:** removes words that occur in N or fewer documents. In our experiments, $N = 3$.
- **Occurrence Count:** removes words that occur less than N times. In our experiments, $N = 10$.
- **Information Gain:** removes all but the top N words by selecting words with the highest information gain. We use $N = 10,000$ in our experiments.

DOE documents: studies, reports, plans, correspondence, etc. that may be potentially relevant to the licensing of the High-Level Radioactive Waste Repository. All of these documents are optically recognized.

3.69 Topical Guideline categories: a hierarchical guide of topics that encompass potential licensing issues. Following is the selected categories we use for our experiments:

- 02.1 The Natural Systems of the Geologic Setting: Geologic Systems
- 02.2 The Natural Systems of the Geologic Setting: Hydrologic Systems
- 02.4 The Natural Systems of the Geologic Setting: Climatological and Meteorological Systems
- 04.1 Engineered Barrier Systems: Waste Package
- 12.1 Geologic Repository Environmental Impact Statement: Environmental
- 12.2 Geologic Repository Environmental Impact Statement: Socioeconomic
- 12.3 Geologic Repository Environmental Impact Statement: Transportation

Collection Statistics	
Document count	138
Number of pages	9015
Average document length (pages)	65
Median document length (pages)	37

Table 1: Experimental collection statistics

In Table 1 we give some statistics on our collection.

5 Effects of OCR on Document Classification

Our goal was to formulate experiments that would give us the most insight into what effect OCR errors may have on document classification. Broadly, there are two ways in which errors can influence categorization. First, by introducing errors into the training set, and second, by reducing the ability of incoming documents to get categorized correctly. We report on four experiments that help explain both these possibilities.

Good Training/Bad Test Set: In this experiment, the training set, although uncorrected OCR, was selected for its good quality. The test set was just the opposite; it was selected for its poor OCR quality. These experimental runs are labeled `E1`.

Mixed Training/Mixed Test Set: This experiment used the same set of documents used in `E1`, but documents were selected randomly from the complete set for both training and testing. This group of runs we label `E2`.

Good Training/Auto-Corrected: This experiment is labeled `E3`. `E3` is exactly the same as `E1`, except that two difficult-to-categorize documents were first run through *MANICURE*, a system we built to improve recognized documents prior to classification or retrieval[16].

Good Training/Manually-Corrected: This set of experimental runs, labeled `E4`, is the same

Bernoulli	E1	E2	E3	E4
Default	32.35	42.03	32.35	32.35
Document Count	64.71	56.52	64.71	64.71
Information Gain	70.59	59.42	70.59	70.59
Occurrence Count	64.71	57.97	64.71	64.71

Table 2: Average accuracy rates for each dimensionality reduction

Multinomial	E1	E2	E3	E4
Default	94.12	84.06	94.12	94.12
Document Count	97.06	85.51	97.06	97.06
Information Gain	94.12	86.96	97.06	97.06
Occurrence Count	97.06	85.51	97.06	97.06

Table 3: Average accuracy rates for each dimensionality reduction

test as E3 except that the two documents in E3 are *manually corrected.*

In each experiment described above, we perform several runs based on both the Bernoulli and multinomial probability models. In addition, each experiment includes a limited vocabulary run that applies the multinomial probability technique. The limited vocabulary "list" consists of several merged dictionaries that include domain specific terms that a general dictionary may have missed in the indexing process. It is comprised of several general dictionaries, geologic and radiologic specific dictionaries, and LSN specific thesauri, and contains 413,216 words. Limiting the vocabulary to pre-defined control terms is a common method of indexing in both retrieval and categorization.

For each run, we apply all the dimensionality reductions described in Section 4. Tables 2, 3, and 4 report the average *accuracy rates* for the various runs. The accuracy rate of a class is the ratio of the *number of correct decisions* made by the system over the *total number of documents in the class.*

5.1 Bernoulli vs. Multinomial

Note first that the Bernoulli results do not compare to either of the multinomial runs. We know from previous research [9] that with longer documents, like the ones we use here, Multinomial typically produces better results than Bernoulli. These

Limited Vocabulary	E1	E2	E3	E4
Default	94.12	85.51	94.12	94.12
Document Count	97.06	84.06	97.06	97.06
Information Gain	94.12	85.51	97.06	94.12
Occurrence Count	94.12	82.61	97.06	94.12

Table 4: Average accuracy rates for each dimensionality reduction

Term	Bernoulli Weight	Multinomial Weight
southern	0.750000	0.000138

Table 5: Comparison of weight for OCR error

Dimensionality Reduction	% of Misspellings
Default	48%
Document Count	8%
Information Gain	11%
Occurrence Count	8%

Table 6: Percentage of misspellings for each dimensionality reduction

results mirror this research. We do believe however, based on these results here and results from other experiments we have done [17], that the accuracy rate is particularly poor due to the use of OCR text. Table 5 shows an obvious OCR error in both the Bernoulli run and the Multinomial run and its respective probability weights. For Bernoulli, this term is given the highest weight in the category while for Multinomial, this error's probability is only 1% of the highest ranked term in the category. Examples like this can be found throughout the Bernoulli categories.

5.2 Default vs. Dimensionality Reductions

As with other classification experiments[11], our results show that dimensionality reduction improves categorization. Dimensionality reduction eliminates terms that contribute the least amount of information for the categories. With respect to OCR text, this includes terms that are misrecognized by the device and contribute no value to the category. Examples of obvious OCR errors that were removed due to dimensionality reduction include: `aluminurn`, `tomograpby`, `sufface`, `therinal`, `reqyirements`.

We believe that with OCR text, reduction is not an option, it is a requirement. Table 6 reveals the drop in the percentage of misspellings included in the categories when dimensionality reduction is applied.[1] Removal of OCR errors through dimensionality reduction clearly improves the accuracy of categorization.

5.3 Good Training vs. Mixed Training

Recall that Experiment E1 uses all "good training" documents and a poorly recognized test set and E2

[1]This table excludes the limited vocabulary runs which of course had no misspellings.

Document/Category Changes	
Corrected words	195
Garbage strings removed	19,772
Net improvement to category 02.2	5%

Table 7: Improvements made by MANICURE

uses a randomly selected "mixed training" and the complement for its test set. Note that in every run except Bernoulli Default, the average accuracy results of E1 are significantly better than in E2. We believe that these improved results are a function of using good quality OCR for training vs. a randomly selected training set from mixed quality documents.

Although more analysis may be required to verify this conclusion, these consistently better accuracy rates point to the fact that although OCR-generated text may have little or no effect in general when incoming documents are being classified, the selection of good quality OCR training documents is essential.

5.4 Classifying Poor OCR

We discovered in several of our experiments with information retrieval and OCR that some poorly recognized documents were unretrievable without some corrective intervention[15, 14]. This dilemma is paralleled in categorization. In nearly all our experimental runs, there were two poorly recognized documents that just couldn't seem to get categorized properly. We wanted to see if correcting errors in these documents would help. Experiment E3 applies several algorithms within a single pre-processing system, MANICURE, to see if automated OCR cleanup and error correction could improve classification. In fact, one of the two documents did get categorized correctly after running the documents through MANICURE.

MANICURE (**M**arkup **AN**d **I**mage-based **C**orrection **U**sing **R**apid **E**diting)[16] applies several algorithms to improve OCR-generated text that not only correct misrecognized terms but also remove "garbage strings" and repetitive text (like headers and footers). The improvements to these two documents after being run through MANICURE appear in Table 7.

The fact that automatic correction helped classify this document correctly is just part of the story. We also report on the improvement to the category itself. Both of these poorly recognized documents belonged to a single category. After MANICURE and retraining, the percentage of correctly spelled category terms also improved. Of the 118 changed terms, seven more were correct when compared to the non-manicured runs in E1. Although this increase may seem slight, only two documents were run through MANICURE. Additional document processing may prove even more beneficial.

Full manual correction of these documents offered no additional categorization improvement over the automatically MANICURE'd runs.

6 Conclusion and Future Work

Document classification is not an exact science and rarely produces 100% accuracy even with clean textual documents. The results from these experiments show that high accuracy can be attained even when OCR documents are being classified. By comparing experiments using training and test sets with known characteristics, we have identified a few elements that improve categorization.

- Multinomial techniques produce significantly better results for OCR documents than does Bernoulli. We attribute this to the value of weighting based on term frequency in the collection, categories, and incoming documents.

- Good optically recognized documents is essential for training. The difference between using good OCR and randomly selected documents from the full set was pronounced. Category term selection and weighting is heavily influenced by statistics in the collection and training documents. This influence manifests itself in the accuracy of incoming document classification.

- Dimensionality reduction is highly recommended. Reduction rids the categories of insignificant terms, which in this case, includes hundreds of misspellings and garbage strings produced by the OCR. As with IR, these terms have no value. But for categorization, these words are a detriment to proper document placement. Of course, in general, reduction techniques will improve categorization. Several should be tried so that the accuracy is maximized.

- Unless a controlled vocabulary shows marked improvement over free text categorization, we do not view it as highly beneficial. Even though our dictionary was quite extensive and specific to our collection's domain, none of our experiments showed improved results for these runs. This can undoubtedly be attributed to important terms and proper names that are not included in the dictionary.

- If a document is poorly recognized because of OCR errors, it may never get classified properly. This was an issue for IR as well. In some cases, if enough of the errors are corrected, proper classification is the result. Some pre-processing

may be required for certain poorly recognized documents.

Most of what we learned through our experimentation is that by applying good classification techniques, improvement in results should be expected. But more than that, for OCR text, if these techniques are not applied, results will be inferior.

References

[1] William W. Cohen and Haym Hirsh. Joins that generalize: text classification using WHIRL. In Rakesh Agrawal, Paul E. Stolorz, and Gregory Piatetsky-Shapiro, editors, *Proceedings of KDD-98, 4th International Conference on Knowledge Discovery and Data Mining*, pages 169–173, New York, 1998. AAAI Press, Menlo Park.

[2] Nuclear Regulatory Commission. Regulatory guide 3.69. http://www.nrc.gov/NRC/RG/03/03-069.html, 1996.

[3] P. J. Hayes, P. M. Andersen, I. B. Nirenburg, and L. M. Schmandt. TCS: a shell for content-based text categorization. In *Proc. of CAIA-90, 6th IEEE Conf. on Artificial Intelligence Applications*, pages 320–326, Santa Barbara, CA, 1990.

[4] D. D. Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. In *Proc. of ECML-98, 10th European Conf. on Machine Learning*, pages 4–15, Chemnitz, Germany, 1998.

[5] David D. Lewis. Reuters–21578 text categorization test collection, distribution 1.0. September 1997.

[6] M. E. Maron. Automatic indexing: An experimental inquiry. *Journal of the ACM*, 8:404–417, 1961.

[7] M. E. Maron and J. L. Kuhns. On relevance, probabilistic indexing and information retrieval. *Journal of the ACM*, 7(3):216–244, 1960.

[8] Andrew McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. `http://www.cs.cmu.edu/~mccallum/bow`, 1996.

[9] Andrew McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *AAAI-98 Workshop on Learning for Text Categorization*, 1998.

[10] Isabelle Moulinier and Jean-Gabriel Ganascia. Applying an existing machine learning algorithm to text categorization. In Stefan Wermter, Ellen Riloff, and Gabriele Scheler, editors, *Connectionist, statistical, and symbolic approaches to learning for natural language processing*, pages 343–354, Heidelberg, DE, 1996. Springer Verlag.

[11] Fabrizio Sebastiani. Machine learning in automated text categorisation. *ACM Computing Surveys*, 2001. to appear.

[12] Kazem Taghva, Julie Borsack, and Allen Condit. Results of applying probabilistic IR to OCR text. In *Proc. 17th Intl. ACM/SIGIR Conf. on Research and Development in Information Retrieval*, pages 202–211, Dublin, Ireland, July 1994.

[13] Kazem Taghva, Julie Borsack, and Allen Condit. Effects of OCR errors on ranking and feedback using the vector space model. *Inf. Proc. and Management*, 32(3):317–327, 1996.

[14] Kazem Taghva, Julie Borsack, and Allen Condit. Evaluation of model-based retrieval effectiveness with OCR text. *ACM Transactions on Information Systems*, 14(1):64–93, January 1996.

[15] Kazem Taghva, Julie Borsack, Allen Condit, and Srinivas Erva. The effects of noisy data on text retrieval. *J. American Soc. for Inf. Sci.*, 45(1):50–58, January 1994.

[16] Kazem Taghva, Allen Condit, Julie Borsack, John Kilburg, Changshi Wu, and Jeff Gilbreth. The MANICURE document processing system. In *Proc. IS&T/SPIE 1998 Intl. Symp. on Electronic Imaging Science and Technology*, San Jose, CA, January 1998.

[17] Kazem Taghva, Tom Nartker, Julie Borsack, Steve Lumos, Allen Condit, and Ron Young. Evaluating text categorization in the presence of ocr errors. In *Proc. IS&T/SPIE 2001 Intl. Symp. on Electronic Imaging Science and Technology*, pages 68–74, San Jose, CA, January 2001.

Large-Scale Duplicate Document Detection in Operation

Mark Turner **Yuliya Katsnelson**
Highland Technologies, Inc.
4831 Walden Lane
Lanham, MD 20706
{mturner, ykatsnelson}@htech.com

Jim Smith
Department of Defense

Abstract

We present experimental results for duplicate document detection using a combination of metadata (date) and image feature comparison, and experience implementing the method in large-scale in declassification of government documents.

1 The Duplicate Document Detection Problem

We undertook an application of large-scale duplicate document detection for a government agency that expects to review millions of documents over the next several years as part of the document declassification process mandated by executive order. This agency and others need to detect duplicate documents as part of the review and redaction process to ensure that documents are treated consistently prior to release. Ideally, a single "source" version of each document will be reviewed and released.

The scope of the duplicate document detection problem is significant. Some 20,000 documents per week enter the system for review and are processed in a workflow that begins with scanning, indexing, and quality control steps. An analyst reviews each document to ensure that sensitive material meeting security criteria is redacted with an appropriate code. Before redaction, the application searches the existing collection for potential duplicates for the current document. These are found and presented to a human operator, who may resolve them as exact duplicates, near duplicates, or non-duplicates. If there is more than one potential duplicate, they are presented in decreasing order of similarity, using the distance measurement described below.

The current criterion for duplicate detection is page-level. If any page from a document resembles any page in another, both documents are candidate duplicates to be reviewed, regardless of difference in their sizes. This is to handle the case when one document is included in another. This policy is conservative in that it will tend to capture all instances of a duplicated page, and adds significantly to the number of potential duplicate pairs needing review.

To meet the operational requirements, each document needs to be represented in memory in a compact form for fast comparison against all others in the collection. Specifically, each document page is represented as a set of image features. These, as well as the document date, are the basis of comparison. For redundancy, a current copy of the document features is kept in a database, which mirrors the memory contents.

2 Current Approaches to Duplicate Document Detection

Duplicate document detection has been approached in the past mainly by text approaches, including line and string-based methods, word-centered approaches, and sentence-based techniques.

Line-oriented techniques include the *diff* program, used mainly to detect variations in source code, data files, and similar content. The COPS system [1], as well the system developed by Campbell *et al.* [2], used sentence-based methods of comparing documents which involved applying a hash code to each sentence in the documents being compared, then counting the number of hash code collisions. Campbell *et al.* applied their method to news stories, giving a measurement of duplicated content.

String-oriented techniques include the SIF [3] system, intended for file management, and KOALA [4], which targeted plagiarism explicitly. These used a string signature for comparison of documents. Work based on edit distance was conducted by Lopresti [5], and an unpublished approach based suffix-tree approach has been developed by Vassilvitskii.

Buckley *et al.* [6] have applied vector-based information retrieval techniques to look for duplicate documents using word-level evidence. In this approach, each document is represented as a vector in a high-dimensional space, in which each dimension represents a word, and the length in that dimension

is the number of times the word occurs. SCAM [7] also used vector space methods, using a modified cosine measure. Grossman [8] has used a bag-of-words approach. This method discards the most frequent third of words as well as the least frequent third. The lists of remaining words are then compared without regard to frequency information.

All of these approaches require plain text representation of documents and a significant amount of memory for each document. In the large-scale application we consider here, two constraints prohibit these approaches. Documents are stored as images, without high-accuracy text representation. The large number of documents to be compared requires that we represent each document as a small set of features that can be stored in RAM for rapid comparison.

3 The Image and Metadata Approach

The approach to duplicate document detection described here limits comparison first by year, then by image similarity.

For comparison, we extract image features from each document. This feature extraction method does not take into account any semantic characteristics of an image document. It is purely image-based. The advantage of such an approach is that if a document is 25-50 years old and is a third or fourth copy, the OCR engine may not be able to produce a useful rendition of the content.

Image features are generated by first forming histograms of pixel density on the X- and Y-axes and then extracting Fourier coefficients of these histograms. The coefficients are stored as a vector of 102 floating-point values per document image. The first two numbers are the same for every feature vector. This representation takes about 0.5K per vector and is space-efficient.

4 Feature Comparison

The distance $d(a_i, a_j)$ between two document vectors a_i and a_j is equal the sum of the absolute values of the differences of the corresponding elements of the vectors:

$$d(a_i, a_j) = \sum_{k=1}^{102} |a_{ik} - a_{jk}| \qquad (1)$$

Such a distance measure ranges from zero to approximately 1000.

In operation, two documents are considered potential duplicates if the smallest distance between any two of their pages falls below a certain threshold (see Experimental Results section for examples). The page pairs thus found by the system for a particular pair of documents are presented to the users in decreasing order of similarity (greater distance). One document may have a number of candidate duplicate documents. The decision of actually marking documents pairs as duplicates, near-duplicates or non-duplicates is made by the users and is determined by government policy.

5 System Setup

In operation, the Duplicate Detection system described here has the following components:

1. A queue of incoming documents
2. A pool of existing document vectors

As the document is fetched from the queue of incoming documents, the image feature vector is extracted from it. After that, the feature vector is compared with all the documents in the pool that satisfy the metadata constraints, if any. Document pairs whose distance is under the specified threshold are stored for further decision making, sorted in decreasing order of similarity (increasing order of distance). After its processing is completed, the incoming document is added to the pool, both in memory and in the database.

The pool of documents is stored in memory throughout the operation of the Duplicate Detection system. This approach is necessary because of the huge volume of documents. Storing and retrieving vectors from the database creates an overhead of IO that makes the system non-scalable and ultimately un-usable. Keeping the vector space in memory is a hardware-intensive, but effective solution.

6 Experimental Results

This method proved to be effective in finding duplicate pages. A special case of was official forms that had similar but not necessarily identical information and identical grid formatting. The distance values for such pairs were usually borderline to the established duplicate threshold. They frequently are low enough to fall into the duplicate category. However, since the type of information contained in them is similar, the processing they require is likely to be similar as well. Figure 1 shows the case of identical forms and Figure 2 shows the two forms with the same grid format but slightly different data.

Notice that despite the fact that the differences in Figure 2 are minimal, the distance value is still significantly higher that in Figure 1.

The experimentally determined threshold for the data examined that provided high precision proved to be about 200. The recall could not be measured, since no ground truth was available.

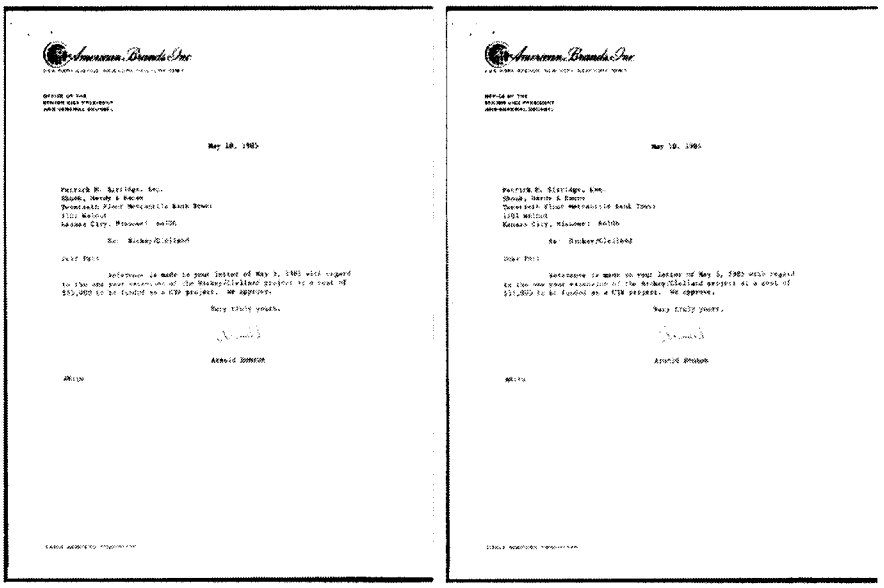


Figure 1: Duplicate forms. Distance = 0.

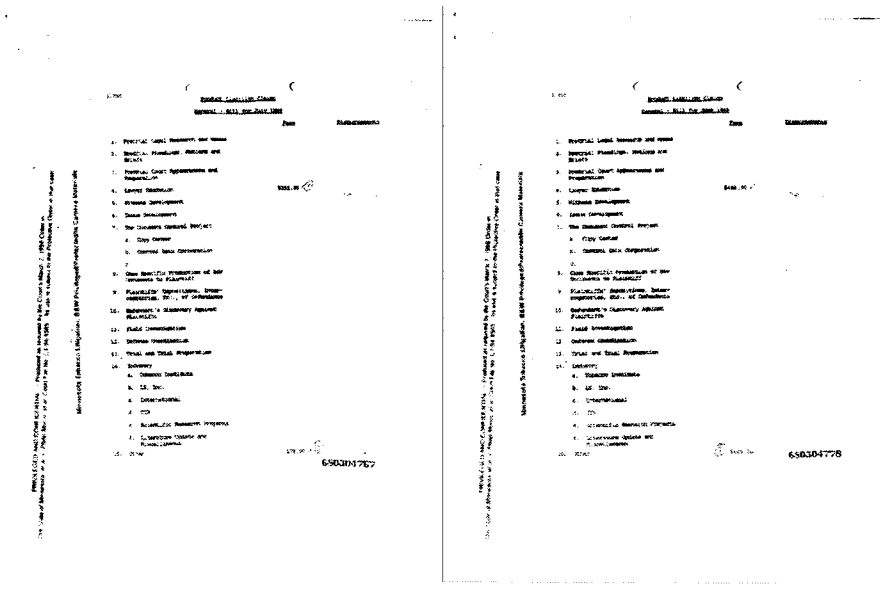


Figure 2: Identical forms with slightly different data. Distance = 145.54.

The method is sensitive to certain amounts of skew and sharpness and brightness variations. Figure 3 shows an example of the effects (e.g. variations in sharpness, shifting) that are detrimental to the quality of duplicate detection. Figure 4 shows an example of a non-duplicate pair of pages.

There are policy-related decisions that have to be made by the user in determining what documents can be considered exact or near duplicates (stamps, signatures, marginalia, etc.).

7 Determining the Threshold of Document Similarity in Use

We have conducted one experiment to determine the threshold of document similarity to be used in operation, and will be conducting another. In the first, a different distance threshold was used each day, recording the number of resulting near duplicates. The ratio of near duplicates to the number of candidate duplicates gives a value for precision at each threshold value. Recall cannot be measured with certainty due to the lack of a ground-truth for duplicates.

Initial results from this experiment showed a precision of 34% at image similarity threshold of 150, when no restrictions were placed on the dates of the candidate duplicates.

Size of candidate duplicates was also affected by the threshold setting. The criterion for two documents being potential duplicates is that any two pages fall beneath the image similarity threshold. For two documents with m and n pages, the number of page comparisons is $m \times n$. As m and n increase, the larger the odds that some pair of pages between the two will show a high image similarity, i.e. will fall below the similarity threshold. (In operation, this is likely to mean that all or part of one document was included in another.) Thus at high thresholds, many large documents were shown as candidate duplicates, despite being relatively rare in the collection, in which the mean document size is 3.7 pages and the median between 1 and 2 pages.

In the second experiment, we will set the threshold at a high value, and users will be allowed to abandon the remainder of the potential duplicates when their judgment of similarity indicated that the quality of document matches is poor. At this point, the actual distance ("cutoff" distance) will measured. In this way, the distribution of distance measurements used as cutoffs can be noted, and an operating value selected.

8 User Reaction

User reaction to the system has generally been favorable. Changes to the image similarity threshold have made a large difference in the usability. At thresholds of 150 or less, the precision has been high enough to make users feel that the system is effective. Above this level, the numbers of non-duplicates increased greatly, and the drop in precision made users feel that the system was less useful.

9 Future Work

Future research goals include determining the distribution of duplicates by closeness in time. We are also investigating the use of evidence from logical document structure [9–12] in duplicate detection. Logical document structure is the layout of document components - titles, page numbers, paragraphs, etc. - and is potentially an important way of finding duplicate documents, which are likely to share the same structure.

Longer term, it is likely that combining evidence from multiple sources of evidence – image properties, metadata, text, and logical document structure – has the most potential for a high-accuracy system. Each one of these sources of evidence has strengths and weaknesses, e.g. text most fully expresses the exact meaning of a document, but may not always be available due to low accuracy of OCR. The goal of accuracy when faced with widely varying document properties will require flexibility in weighing all available evidence

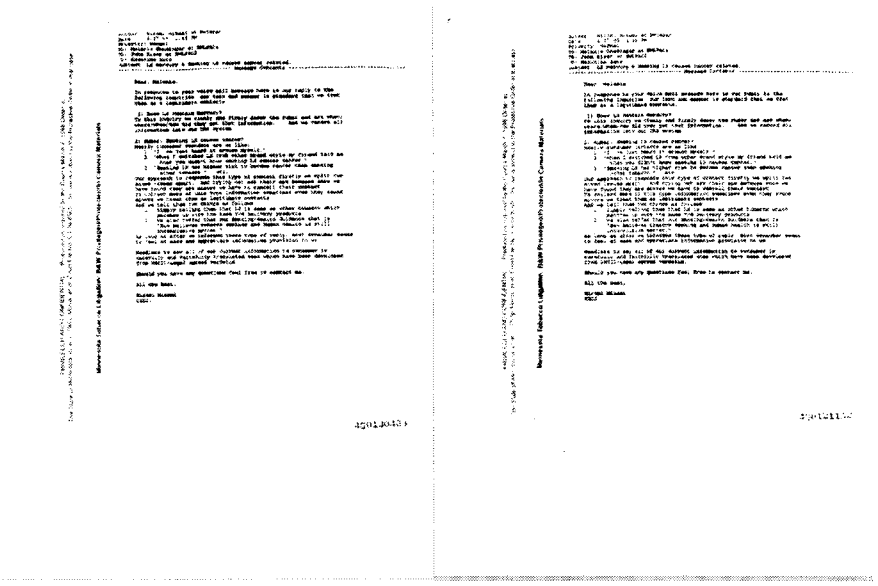


Figure 3: Duplicate documents with some variation in sharpness and brightness and some shifting. Distance = 225.98

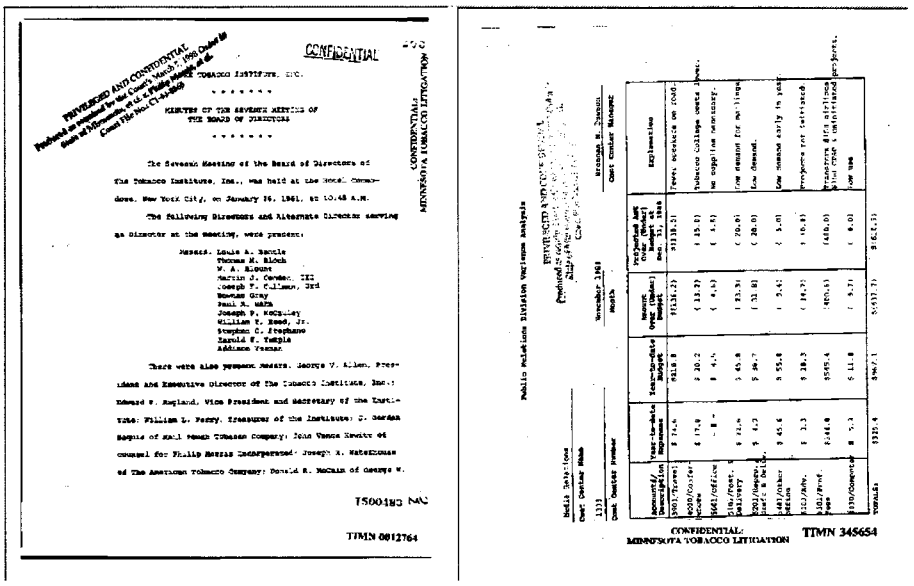


Figure 4: Non-duplicate documents. Distance = 885.52

References

[1] S. Brin, J. Davis, and H. Garcia-Molina. Copy detection mechanisms for digital documents. In *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, San Jose, California, May 1995.

[2] D. M. Campbell, W. R. Chen, and R. D. Smith. Copy detection systems for digital documents. In *Proceedings of the IEEE Advances in Digital Libraries 2000*, Bethesda, Maryland, May 2000.

[3] U. Manber. Finding similar files in a large file system. In *USENIX Winter 1994 Technical Conferences*, San Francisco, California, January 1994.

[4] N. Heintze. Scalable document fingerprinting. In *Proceedings of the Second USENIX Workshop on Electronic Commerce*, Oakland, California, November 1996.

[5] D. Lopresti. A comparison of text-based methods for detecting duplication in document image databases. In *Proceedings of Document Recognition and Retrieval VII (IS&T/SPIE Electronic Imaging)*, San José, California, January 2000.

[6] C. Buckley, C. Cardie, S. Mardis, M. Mitra, D. Pierce, K. Wagstaff, and J. Walz. The Smart/Empire TIPSTER IR system. In *TIPSTER Phase III Proceedings*, pages 107–121. Morgan Kaufmann, 1999.

[7] N. Shivakumar and H. Garcia-Molina. Scam: A copy detection mechanism for digital documents. In *Proceedings of the Second International Conference in Theory and Practice of Digital Libraries*, Austin, Texas, June 1995.

[8] A. Chowdhury, O. Frieder, D. Grossman, and M. C. McCabe. Collection statistics for fast duplicate document detection. Submitted for publication to ACM Transactions on Information Systems.

[9] T. Hu and R. Ingold. A mixed approach toward an efficient logical structure recognition from document images. *Electronic Publishing: Origination, Dissemination, and Design*, 6(4), 1993.

[10] O. Altamura, F. Esposito, and D. Malerba. Wisdom++: An interactive and adaptive document analysis system. In *Proceedings of the Fifth International Conference on Document Analysis and Recognition*, pages 366–369, September 1999.

[11] G. Kopek. Document image decoding in the UC berkeley digital library. In L. M. Vincent and J. J. Hull, editors, *Proceedings: Document Recognition III*, volume 2660 of *SPIE Proceedings Series*, pages 2–13, 1996.

[12] K. Summers. *Automatic Discovery of Logical Document Structure*. PhD thesis, Cornell University, August 1998.

Shape Extraction from Digital Document Images

Glenn Becker
CTO, Magnify Research, Inc.
510 McCormick Drive, Suite A
Glen Burnie, Maryland 21061
gcbecke@magnifyresearch.com

Peter Bock
Professor
The George Washington University
Washington, DC 20037
pbock@seas.gwu.edu

Abstract

This paper presents an overview of features for classifying shapes in digital document images. A new process for classifying shapes in digital images using a radial feature token (RFT) is also presented. This shape classification process has been implemented as the ALISA® Shape Module, the third module in the Adaptive Learning Image and Signal Analysis (ALISA) system hierarchy, which also includes Geometry and Texture classification modules.

Shape classification in images is a challenging problem because the basic shapes in an image can occur in any position, at any orientation, and at any scale. For this reason, translation, rotation, and scale invariance is a critical property of this or any general-purpose shape recognition system.

The ALISA Shape Module learns to recognize shapes from a supervised set of training images. These learned shapes are stored as a set of vectors that are then used to classify shapes in test images.

Results indicate that this process can learn to classify shapes from small training sets and then classify similar shapes despite extraneous edges or partially overlapping shapes. The radial feature token also enables the ALISA Shape Module to classify some shapes that are only partially visible or that have gaps in their edges.

1 Shape Feature Extraction

Several simple features have been applied to shape classification. Some of these features include: the length of the shape's boundary, the orientation and size of the shape's major axis, and the number of convex deficiencies (concavities). These simple features have been successfully applied to some very specific applications, but they do not generalize well because too much information about the original shape is lost.

Techniques that consider a shape's entire boundary, not just gross structural characteristics, include parameter transforms, chain codes, signatures, and region skeletons. Each of these techniques is presented here with their strengths and weaknesses. In general, all of these techniques are computationally very expensive.

1.1 Parameter Transforms

Parameter transforms, like the Hough and RANSAC Transforms, are general algorithms that can be customized to classify particular shapes or edge patterns. Parameter transforms are so named because the selected transform has bound random variables, called the parameters, that are varied through their range to calculate a dependent variable. Curve-fitting algorithms, in contrast, try to find edge points that best fit a function with fixed parameters. [10]

Parameter transforms are a type of voting algorithm because the resulting value of the dependent variable from each set of parameter values is tallied in an *N* dimensional histogram, where *N-1* is the number of parameters being changed. The ranges for these changing parameters are quantized, divided into discrete subranges. These subranges correspond to the individual rows and columns in the N-dimensional histogram. The Hough and RANSAC transforms discussed below are specific examples of parameter transforms characterized by the parameters that are changed.

1.1.1 Hough Transform

The Hough Transform is one of the most widely used parameter transforms. The original Hough Transform uses the general function for a line, Eq. (1), with parameters *m*, for the slope, and *b*, for the offset.

$$y = mx + b \qquad (1)$$

ALISA is a registered trademark of The ALIAS Corporation

A 2-dimensional histogram is used to count the number of times each discrete value of b results from all discrete values of m.

Although Eq. (1) is very simple, problems arise with vertical and nearly-vertical lines because both m and b approach infinity. To solve this problem, Duda and Hart proposed using Eq. (2), based on the Radon Transform [10][19], which is continuous for all values of theta (θ). The resulting 2-dimensional histogram uses discrete ranges of ρ and θ for axes.

$$\rho = x\cos(\theta) + y\sin(\theta) \qquad (2)$$

In practice, the Hough Transform is used to find shapes in images through the following steps:

1. compute the gradient of an image using a Sobel operator or something similar to find edges,
2. select a quantization resolution for the ρ,θ plane,
3. calculate ρ for all values of θ when <x,y> is on an edge,
4. find the cells in the ρ,θ plane that have the greatest values, and
5. compare these dominant cells with those from known classes.

The Hough Transform is popular because it is robust for edges with gaps and noise. The quantization done by selecting discrete ranges for ρ and θ is a form of undersampling which reduces the affects of noise and gaps on the resulting ρ,θ histogram. [10]

The Hough Transform also has some disadvantages. Eqs. (1) and (2) are only two of the many possible transforms that can be used with the Hough Transform. Selecting the appropriate transform and size of the discrete ranges in the histogram is application dependent and can require a lot of experimentation. Also, the resulting histogram is not invariant to changes in scale and rotation. The histogram itself must be transformed or normalized to find matching shapes at different scales or orientations.

1.1.2 RANSAC

The RANSAC transform is similar to the Hough Transform in that it is a parameter transform and voting algorithm. Where the Hough Transform uses a function, the RANSAC transform uses a geometric shape. A series of shapes are used as templates to see how often points on the template shape correspond with edges in the image. [11]

As an example, consider a RANSAC application that uses an ellipse as a template shape to look for elliptical edge structures in an image. A set of ellipses is predefined as the template. These ellipses represent some range of radii, aspect ratios, and orientations. Each ellipse is scanned across the image to find locations where five (or some other predetermined number) preselected points on the ellipse template match with the edges in the image. The ellipse template that accumulates the most votes wins, and its class is chosen to represent that image. [11]

Like the Hough Transform, this method is robust when the edges have gaps or noise. But also like the Hough Transform, the RANSAC method can be difficult to optimize because so many possible template shapes and shape parameters make the search space large.

Shapiro [19] developed a special case of the Hough Transform, also based on the Radon Transform, called Reconstructive Matching (RM). RM uses either a rectangular token or radial token (for scale invariance) to build templates for shape recognition. The cross-correlation between the image and the template is used to match a template to a shape. RM also uses the first through fourth moments of the template and shape region as criteria for a match. As a result, RM is inherently invariant to position, rotation, and optionally scaling for shape recognition. Results reported to date are only on experiments with simple shapes. Also, no way has been found to deal with concave shapes.

1.2 Chain Codes

Chain codes are used to characterize closed shapes as a series of fixed length line segments. These fixed length line segments can either be 4-way or 8-way connected as shown in Figure 1. From an arbitrary starting point, designated by the large dot, each segment is identified by its direction, zero through seven in the case of 8-way coding. The example shown produces the code "**21100077644444**". [14]

The starting point is arbitrary because the coded string "**21100077644444**" is a circular code that can be rotated to match the same shape at any starting position. Rotational invariance can be achieved by using the differences of these code values instead of the directional codes themselves. These differences are consistent as the shape is rotated to 90°, 180°, and 270° for a 4-way code and 45°, 90°, 135° and so on for an 8-way code. Orientation changes that are smaller than the intervals shown may change the difference codes because of aliasing.

Scale invariance can be achieved by changing the length of the fixed length segments used to build the code string or scaling the code string itself. For example, a larger version of "**21100077644444**" might be "**22111100000077777664444444444**", where each code is used twice as often.

Even though implementing scale and rotation invariance are easy to demonstrate for chain coding techniques, in practice they are difficult to generalize. Small changes in rotation or scale can not be represented with the methods shown. Changes in rotation and scale in digital images also causes aliasing,

or stair-step affects, which can become false vertices and change the code string dramatically.

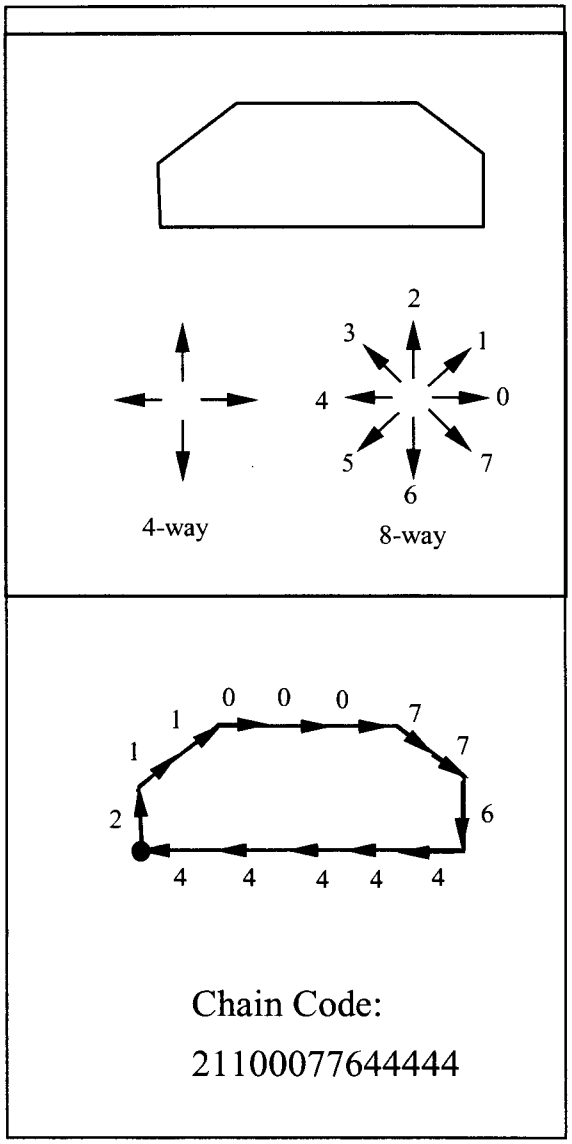


Figure 1 - Chain Code Example

1.3 Signatures

A signature is a function that represents a shape's boundary. Many functions can be used but the most common is the distance from the shape's centroid to the boundary plotted as a function of angle. Signature functions can only accurately represent convex shapes or mostly convex shapes with boundaries that can be completely "seen" from the shape's centroid. [14]

Rotational invariance can be achieved by phase shifting the signature function. Scale invariance, for the distance-from-centroid function described above, can be implemented by normalizing the function's magnitude. Other functions, like the angle between the radii from the center and the boundary's tangent, are naturally scale invariant.

The signature function is typically implemented as a histogram. For the distance-from-centroid example, the histogram's magnitude is the distance from the shape's centroid for each angle increment. The signatures of two shapes can be compared by correlation.

Signatures can only be used in convex or near convex shapes. They also require finding the shape's centroid, which is difficult for shapes with boundary gaps or shapes that are partially obscured.

1.4 Region Skeletons

Two-dimensional shapes can be reduced to form a region skeleton or graph which can then be used to characterize the shape. The skeleton of a shape can be determined by a thinning algorithm that reduces a shape to a skeleton that is only one pixel wide. This skeleton can then be stored as a tree graph. [14]

Blum [4] proposed the medial axis transformation (MAT) as an algorithm to determine the skeleton of a shape. The MAT algorithm considers every point within a shape and determines the closest boundary pixel to that point. If two or more boundary pixels are equidistant from a point in a shape then that point is part of the shape's skeleton.

Many variations of this basic MAT algorithm have been developed to improve the computational efficiency of the algorithm and also handle gray-scale and color images. The resulting skeleton, or tree graph, can be handled very much like a chain code. The tree graph is rotationally invariant and scale invariance can be achieved to some extent by scaling the graph's elements. Although a shape's skeleton represents the shape's basic structure, it does not represent the shape's boundary. For example, symmetrical widening or thinning in a shape has no affect on the shape's skeleton.

2 Constraint Satisfaction Methods

Image analysis systems that attempt to extract shape and other structural information from images must take advantage of domain specific constraints to resolve ambiguities. The three types of constraints that can be exploited are constraints imposed by templates, natural constraints, and linguistic constraints. [9]

Algorithms that implement these constraint systems are generally called **relaxation algorithms**, so named because they classify as much of the image as possible with tight tolerances on the constraints. The tolerances are then gradually relaxed to also classify objects that do not match perfectly.

2.1 Template Matching Systems

Template matching systems compare edges in images with a set of known patterns. These known patterns, or templates, are compared with the image edges using a correlation metric to determine which template is the best fit. This section discusses the following four general families of template systems: total, partial, piece, and flexible templates. [9]

2.1.1 Total Templates

Total templates are the simplest but most restrictive type of templates. Total templates are applied to an entire image and do not allow for translation, scaling, or rotation of the template pattern. Because of these limitations, total templates offer fast performance, but they are very application specific and they do not generalize well. Total templates are typically used to check parts on an assembly line where the location and position of the part is controlled.

Comparing total templates to a test image is a fast and easy operation because any deviation from the template, beyond some tolerance, is considered a mismatch. To determine the degree to which the image matches a template is generally done by correlating template points with those in the image.

As background, three methods for calculating this correlation, **cross-correlation, Chamfer matching,** and the **Hausdorff Distance**, are discussed here. These are popular methods but are not the only methods used. These methods also apply to all template types, not just total templates.

2.1.1.1 Cross-correlation

The correlation between the pixels of a gray-scale image, $f(x,y)$, and a template bitmap, $\omega(x,y)$ is generally called the **cross-correlation** if $f(x,y)$ and $\omega(x,y)$ are different functions. The image, $f(x,y)$, consists of gray-scale intensity values or gray-scale edge values, depending on the application. The template, $\omega(x,y)$, is a 2-dimensional array of either 1's and 0's, like a mask, or gray-scale values. Simple correlation, Eq. (3), works well as long as the intensity values of the gray-scale image and template are of the same scale. If they are scaled differently then the correlation coefficient, Eq. (4), must be used to correct for this difference. [19]

Simple correlation (3)

$$\underline{R}(m,n) = \sum_x \sum_y f(x,y)\omega(x-m,y-n)$$

Correlation coefficient (4)

$$\underline{r}(m,n) = \frac{\sum_x \sum_y [f(x,y)-\overline{f}(x,y)][\omega(x-m,y-n)-\overline{\omega}]}{\sqrt{\sum_x \sum_y [f(x,y)-\overline{f}(x,y)]^2 \sum_x \sum_y [\omega(x-m,y-n)-\overline{\omega}]^2}}$$

where $\overline{f}$ and $\overline{\omega}$ are the mean intensity values of $f(x,y)$ and $\omega(x,y)$ within the template area.

If the template is smaller than the image, the template can be scanned across the image by incrementing m and n. The largest value in the resulting array, $r(m,n)$, indicates which position in the image most closely matches the template. A threshold is generally used to decide whether the largest value in $r(m,n)$ is large enough to be considered a match or to decide if more than one value exceeds the threshold indicating multiple matches in the same image.

Although Eqs. (3) and (4) do allow for translation, they do not support rotational variations between the image and template. Rotation can be handled by adding the stepwise rotation of $\omega(x,y)$ about its center. The resulting array, $r(m,n,f)$, is now 3-dimensional, but it can be processed as before, because any values greater than the threshold are considered matches. Handling rotation in this way is expensive because the correlation must be calculated for all values of m, n, and f.

Templates that use polar coordinates can be used to eliminate the need for rotating the rectangular templates. Processing polar templates, however, can also be slow because the polar coordinates must be mapped into the rectangular coordinates of the image. Aliasing can be a problem, because mapping polar to rectangular coordinates involves rounding to the nearest image pixel, which may cause classification errors.

2.1.1.2 Chamfer Matching

Chamfer matching, originally introduced by Barrow [1], is a way to compare templates of edges with edges in images. The image was expected to be binary with zeros for background and ones for edge pixels. The template's pixel values represented the integer distance from the ideal edge represented by that template. These distances are known as chamfer distances, and so the name "chamfer matching". A chamfer distance of zero represents a template pixel that falls exactly on an image edge. Pixels farther away from the ideal edge had greater values. The template is scanned across the binary image looking for local minima that represent areas of the image that best match the template.

The scanning and calculations required for chamfer matching are very expensive, especially for large templates and if the templates are rotated to match edges at any orientation. Borgefors introduced a hierarchical approach using a resolution pyramid that

can quickly find potential matches at a reduced resolution and then restrict the image scanning at high resolution to only those areas with likely matches, thus speeding up the process. [7]

2.1.1.3 Hausdorff Distance

The Hausdorff Distance [18] calculates the closeness of fit between points on a template and points in an image shape. The Hausdorff Distance, $H(A,B)$, between two finite sets of points $A=\{a_1, a_2, \ldots, a_n\}$ and $B=\{b_1, b_2, \ldots, b_n\}$ is:

$$H(A,B) = \max (h(A,B), h(B,A)),$$

where $h(A,B) = \max_{a \in A} \min_{b \in B} \| a-b \|$.

The function $h(A,B)$ is called the **directed Hausdorff distance** from A to B. The point $a \in A$ is selected as the farthest point from any points in B, and the function $\| \cdot \|$ is used to calculate the distance between a and the closest point of B. The Hausdorff Distance is then the maximum distance of $h(A,B)$ and its inverse $h(B,A)$.

The Hausdorff Distance is an interesting template matching metric because it does not attempt to match pairs of reference points between the template and the image shape. Techniques have been developed to handle translation, rotation, and the matching of partial templates. No way has been found to easily handle scale differences, however. This method also has trouble separating shapes that share edges or overlap.

2.1.2 Partial Templates

Partial templates are essentially the same as total templates, but they are smaller than the image so they can be scanned across the image to find a matching pattern anywhere in the image. The advantage of this scanning is that the template can now find a matching pattern independent of its translation within the image. The methods used to compare partial templates to image patterns are the same as those described in the total template section. [9]

2.1.3 Piece Templates

Systems that use total templates or partial templates compare each template with an image to find matches. The individual templates are generally not related to each other in any way, so they can be compared in any order without affecting the results. Piece templates are similar to partial templates with the addition of a hierarchical structure. Lower level piece templates represent basic or simple shapes, and higher level piece templates represent more complex shapes. [9]

When comparing piece templates with images, the highest level templates are compared first in an attempt to discover the most complex shapes first. Subsequently, lower level templates are compared to find simpler shapes in the remaining areas of the image that have not matched higher level templates.

As an example, consider a set of piece templates for identifying printed letters. High level templates in this system are entire letters in specific fonts (*e.g.*, A, **B**, **C**, D, E, **f**, **G**). Any shapes that do not match these letter templates are then compared with a set of simpler templates (*e.g.*, |, —, /, \, ⊥).

2.1.4 Flexible Templates

Flexible templates, sometimes called "rubber masks", have a set of template shapes, just as partial and piece templates. Flexible templates extend these template prototypes by allowing for stretching, reorientation, and other deviations.

A flexible template begins as a partial or piece template, which is based on an example or prototype of a shape. Examples of recognized variations of this same shape are then used to relax the parameters that describe the shape so that it includes all of these variations. Many methods have been used to implement support for these template variations, including: Gaussian distribution, uniform range, or a set of valid values for template parameters. Some flexible templates also include methods for handling scale and orientation differences between the template and image patterns.

2.2 Exploiting Natural Constraints

Natural constraints are those imposed by the physical world. Some famous optical illusions, like M. C. Escher's "Dutch Waterfall", are intriguing because they subtly violate these natural constraints. At the shape level, typical natural constraints include validating vertices in context with connected vertices and finding continuous edges and surfaces.

In 1971, Huffman [17] and Clowes [8] published their work which showed how natural constraints could be used the label the edges of shapes in an image. They used three symbols (+, -, and →) to characterize edges as convex, concave, or occluded relative to the camera. Huffman and Clowes used these edge classifications to build a table of all legal vertices of two or three edges. This table can be used to label edges in images of polyhedra. [9]

Huffman and Clowes had limited their work to trihedral polyhedra with no shadows, cracks, or separable concave edges. Separable concave edges are edges that belong to two or more separable objects, but they appear in the image as a single edge. They also used an exhaustive search that changes edge labels incrementally until all of the vertices are legal, or failing that the polyhedron is declared impossible.

Waltz [9] [20] developed an extension of Huffman and Clowes' work that addressed the issues of shadows, cracks, and separable concave edges. Waltz's algorithm includes the following six different edge types: concave, convex, obscuring edges, cracks, shadows, and separable concave edges. With these additional edge types and allowing vertices with up to five coincident edges, the number of vertex types grew from Huffman's 15 to 2,593 legal combinations.

Although these new vertex types allow Waltz to handle more complex images than Huffman, the large number of legal vertices makes Huffman's exhaustive search impractical. The Waltz algorithm considers vertices in pairs. For any given pair of vertices, all edges connected to this pair are labeled with all possible legal labels. After all edges have been labeled with all legal labels, the pair-wise analysis continues to find labels that can not exist in each edge's context. This process is commonly called Waltz filtering.

Despite Waltz's advances on Huffman's original work, both systems focus on vertices and assume that all edges are straight. This requirement limits the use of these systems to controlled environments, like "Blocks World", making them inappropriate for many real-world applications.

3 Overview of the ALISA System

The ALISA System is an image analysis system organized as layers of modules in which the modules in each successive layer look at the output of the module (or modules) in the previous layer. The modules of each successive layer then classify more conceptually complex objects than their predecessors. [2]

The ALISA system uses an artificial intelligence paradigm called **histogram learning** to build a statistical representation of image data from a training set. ALISA uses texture and geometry features to characterize patterns in training images. These features are used to build a statistical representation of each class. ALISA then uses this class representation to determine if a set of test images are similar in part or in whole to the training set. Because ALISA learns from example, it can be quickly trained to recognize new textures or geometries without reprogramming. [5] [16]

Histogram learning is a machine learning paradigm that learns which feature vectors are expected or unexpected for a particular class of images. Histogram learning uses a histogram to accumulate the frequencies at which feature vectors occur in example images. [6]

The current ALISA system has two modules, a Texture Module [5] and a Geometry Module [16]. The Texture Module uses features that characterize sub-symbolic textures in an image. These texture features include measurements of roughness, wavy patterns, and the directionality of any intensity gradients. [6]

The ALISA Texture Module classifies pixel patterns in an image into known texture classes and generates an output image, called a **texture map**, in which each pixel's value corresponds to the texture class of the area surrounding that pixel in the original image. This means that areas of the texture map that have the same pixel value have the same texture in the corresponding area in the original image. The boundaries between these solid areas are the edges in the image. [6]

The Geometry Module can be trained to classify a wide variety of 2-dimensional patterns in the texture map. In this work, the Geometry Module looks at the edges found in the image by the Texture Module and classifies the edges based on their orientation (*i.e.*, vertical, horizontal, up-slant, and down-slant). The Geometry Module builds an output image, called the **geometry map**, in which each pixel's value represents one of these four classes. [16]

4 ALISA Shape Module

The Shape Module accepts the geometry map from the Geometry Module as input and generates a class **shape map** for each shape class as output. Each pixel's value in a class shape map indicates the confidence that that pixel is part of that shape.

The Shape Module expects the input geometry map to contain edges classified as any of the following four **canonical geometry classes**: horizontal, vertical, up-slant, and down-slant. The first processing step in the Shape Module is to **segment** the input geometry map. The segmentation process finds continuous regions of any one of the geometry classes.

A radial feature token (RFT) is then applied to each segment to generate a feature vector, $\underline{v}[n]$, that characterizes the relationship between that segment and its surrounding segments. As with the other ALISA modules, the Shape Module must be trained to recognize particular shapes before it can be used in an application. While the Shape Module is being trained to recognize a new shape class, the feature vectors generated by the RFT are accumulated in the Shape Module's vector list, $\underline{c}[m]$, for that shape. Where $\underline{c}[m]$ is an array of feature vectors, $\underline{v}[n]$, that constitute the definition of a shape m. When the Shape Module classifies unknown shapes, the feature vectors generated by the RFT are then compared with the feature vectors, in $\underline{c}[\ ]$, looking for the closest match.

4.1 Segmentation Process

The canonical geometry classes assigned by the ALISA Geometry Module classify edge pixels from the original image according to their orientation. The segmentation process uses a recursive flood-fill algorithm [12] to find all 8-way connected pixels belonging to the same geometry class. That is, all 8-way connected edge pixels

with the same orientation are grouped together into a **segment**.

Flood-fill algorithms are commonly used in computer graphics applications for finding regions of adjacent pixels with some common attribute or for finding all pixels within some arbitrary boundary. In both cases the boundary of the shape being filled can either be specified by image coordinates or pixel values of either the boundary or the shape contents. The recursive flood-fill algorithm finds all 8-way connected pixels with the same "value", where the pixel's "value" is its geometry class.

Figure 2 shows a square with four segments, one corresponding to each of the four sides. The bottom horizontal segment is shown with its RFT radiating from its center.

4.2 Segment Classification

After the geometry map has been segmented, an RFT is used to generate a feature vector for each segment. These feature vectors characterize the shapes formed by the relative positions of the image segments. The RFT is a set of consistently spaced radii emanating from a common center, as shown in Figure 2.

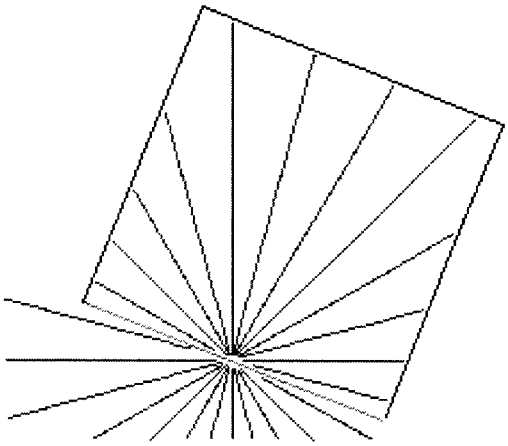


Figure 2 - Radial Feature Token Example

Token Operation. Shape classification is complicated by, among other things, variations in the scale and orientation of the shapes in an image. One method that has been used to overcome this problem is converting the rectilinear image coordinates to radial coordinates. Converting a rectilinear image to radial coordinates converts scale and rotation changes to translation in the radial image. Hough transforms and radial signatures are examples of how this approach has been applied to shape classification. The RFT used in this work is a variation of the radial signature technique. [14]

Converting to radial coordinates is complicated by the selection of an origin position. Proper placement of the origin is critical to making the resulting radial image as invariant to scale and rotation as possible, while capturing as much information about the shape as possible.

Because the goal is to group neighboring segments into shapes, the radial view is limited during training to the first segment encountered in each radial direction. This limitation is consistent with published implementations of the radial signature. Limiting the radial view in this way means that selecting an origin outside the shape is sub-optimal because part of the shape is always obscured. The number of RFT radii that intersect the shape is also reduced during both training and testing if the RFT center is outside the shape.

Selecting an origin inside the shape is problematic, because finding a consistently positioned location inside the unknown shape is difficult. Finding a point (or points) on the shape's "skeleton" can be done consistently if the shape is not partially obscured. The center of mass can be found for a region surrounded by segments, but using this point also has limitations. Partially obscured shapes have a center of mass position that has little or no relationship to the underlying shape. Also, in the case of some concave shapes, the center of mass is outside the shape.

Shape vertices can be used as origins, as shown in Winston's work [20], but his work was limited to idealized images with straight edges. In real images, vertices can be obscured or fuzzy, making them unreliable. Winston avoided using curved edges because of the difficulties involved in representing these curved edges in the context of a vertex, which is defined as the conjunction of two or more edges. With curved edges involved, the definition of vertex becomes an issue. A threshold radius of curvature must be chosen to differentiate vertices from sharp curves in an edge. For these reasons, vertices are not good candidates.

Because the goal of this research is to identify shapes among neighboring segments, using the segments themselves to position the radial origins seems most appropriate. Segments representing shape edges have been used in the Hough transform and signature methods. These two methods rely on edge information primarily because the edges and their relative position to their surrounding edges are what constitute a shape. For these reasons, all RFT origins considered in this work are positioned on the pixel closest to the segment's center of mass.

Having identified the position of an RFT's center on a specific segment, S_i, the evenly spaced radii are extended from this center. Note that the pixels of S_i are ignored by the radii. The radii follow straight lines from the center, being mapped to specific pixel positions in the image by Breshenham's raster algorithm [12]. In

Figure 2, an RFT with 24 radii separated by 15° is shown. During training each radii stops at the first segment encountered. During testing the radii record all segments encountered and only stop at the edge of the image.

The distance that each radius extends from the RFT origin to reach a segment is stored in a feature vector, $\underline{v}[n]$, where n is the number of radii. The distances are stored in counter-clockwise order with an arbitrary starting point. Those corresponding to radii that reach the image edge are set to zero.

Figure 3 shows the results of a series of experiments to determine the optimal number of radii for the RFT. Each cell in this table shows the percentage of correct classifications after the Shape Module was trained to recognize squares, 5-gons, 10-gons, 20-gons, and circles. The percentage shown is an average over the module's performance in classifying all five shapes. Performance does not improve significantly with more than 24 radii, but the computation time increases by $O(n^2)$.

Number of radii	8	16	24	32	40	48	56	64
Percent correct	0.913	0.987	0.995	0.995	0.995	0.995	0.996	0.996

Figure 3 – Percentage of Correct Classifications

Feature Vector Representation. The feature vector, $\underline{v}[n]$, generated by the RFT in Figure 2 is shown in Figure 4a). Rotational invariance is achieved by performing a circular shift on the feature vector. In the case of $\underline{v}[n]$, it could be compared with another vector which was shifted 45° from $\underline{v}[n]$. If the two vectors were similar, their closest match would occur when $\underline{v}[n]$ was shifted by three positions as shown in Figure 4b).

In addition to matching shapes that are rotated, this feature vector can also match shapes that are flipped (mirror images). This can be done by inverting the feature vector about some axis as shown in Figure 4c). In this example, radii numbers 6 and 18 are used as the axis of rotation, meaning that the shape was flipped around a vertical axis. The inverted vector can also be shifted, making it rotation invariant.

Symbolic Translation Matrix (STM) Organization. The Shape Module's training is supervised by showing the system selected example shapes from each class. Each new feature vector from a training image is added to the Shape Module's STM for the designated class. A shape class is a collection of feature vectors that belong to the same class. Classification is performed by comparing feature vectors computed from test images to those stored in the STM.

rad	distance
0	76
1	89
2	114
3	156
4	145
5	145
6	155
7	120
8	92
9	79
10	74
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	71

a) original vector

rad	distance
0	0
1	0
2	71
3	76
4	89
5	114
6	156
7	145
8	145
9	155
10	120
11	92
12	79
13	74
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0

b) rotated 45°

rad	distance
0	0
1	0
2	74
3	79
4	92
5	120
6	155
7	145
8	145
9	156
10	114
11	89
12	76
13	71
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0

c) mirror image

Figure 4 - Radial Token Feature Vector

The Shape Module classifies segments based on their position relative to neighboring segments. Each segment is characterized by an RFT that generated a feature vector. This feature vector is then compared against all feature vectors in each shape class. A single segment can match multiple classes because a segment can be part of more than one shape class.

Figure 5 shows a simple example image of a square with an ellipse overlaying it. The RFT (shown radiating from the bottom segment) measures the distances from that segment to the other edges its radii intersect. These distances are tabulated in Figure 6 a) and compared with distances learned from the training image for squares in Figure 2.

Figure 6 b) shows the STM vector that best matches this segment. These vectors are compared by dividing the distances from the test image, F[r, c], by the distances in the STM vector, S[r], to find the ratios, R[r,c], between the edges detected and those expected if the shape is a match, Eq. (5).

The resulting R[ ] values are then sorted to find the group with the closest values. The number of values required to match is a parameter of the system. It controls how tolerant the Shape Module is to gaps. The range of the values within this group is used to determine the confidence of the match. The range is compared with the intra-class distances among the training vectors.

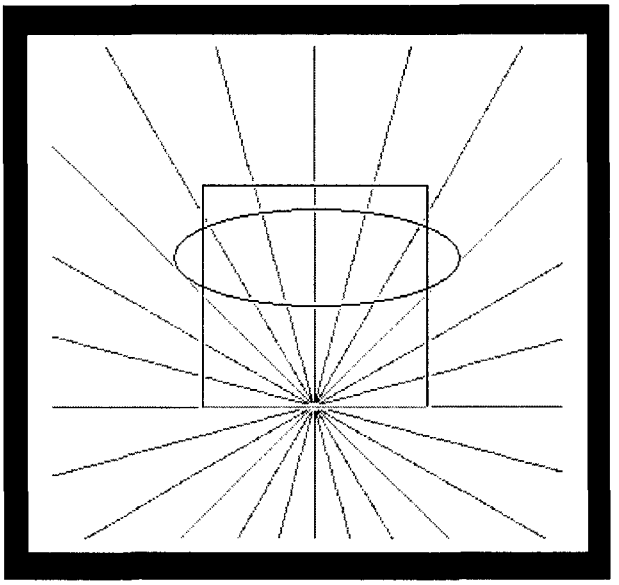


Figure 5 – Test square with RFT shown

$$\mathbf{R}[r,c] = \mathbf{F}[r,c] / \mathbf{S}[r] \qquad (5)$$

radii #	distance				radii #	distance		radii #	R[ ]		
0	-	-	-		0	-		0	-	-	-
1	67	-	-		1	75		1	**.89**	-	-
2	75	-	-		2	84		2	**.89**	-	-
3	92	104	-		3	101		3	**.91**	1.03	-
4	71	122	128		4	145		4	.49	.84	**.88**
5	62	118	135		5	149		5	.42	.79	**.91**
6	59	116	130		6	144		6	.41	.81	**.90**
7	61	118	134		7	149		7	.41	.79	**.90**
8	70	123	129		8	141		8	.50	.87	**.91**
9	92	98	-		9	100		9	**.92**	.98	-
10	75	-	-		10	82		10	**.91**	-	-
11	67	-	-		11	74		11	**.91**	-	-
12	-	-	-		12	-		12	-	-	-
13	-	-	-		13	-		13	-	-	-
14	-	-	-		14	-		14	-	-	-
15	-	-	-		15	-		15	-	-	-
16	-	-	-		16	-		16	-	-	-
17	-	-	-		17	-		17	-	-	-
18	-	-	-		18	-		18	-	-	-
19	-	-	-		19	-		19	-	-	-
20	-	-	-		20	-		20	-	-	-
21	-	-	-		21	-		21	-	-	-
22	-	-	-		22	-		22	-	-	-
23	-	-	-		23	-		23	-	-	-

a) test vector F[r,c] b) STM vector S[r] c) R[r,c] = F[r,c] / S[r]

Figure 6 – Comparison of Test Feature Vector and STM Feature Vector

4.3 Shape Module Output

The Shape Module's output is a **shape map**. The output shape map is a $2^1/_2$ dimensional image where x and y denote the spatial coordinates and $f_z(x,y)$ is an integer which corresponds to the shape class of the pixel at location (x,y). The shape map consists of multiple layers (z-axis) because each segment can belong to more than one shape. Therefore, each pixel, $f_z(x,y)$, can have more than one value.

5 Results

A great deal of research is currently being done on content-based query in image databases. [3] Experiments with the Shape Module have demonstrated that it can locate known shapes in images, despite their location, orientation, and scale. The ALISA Shape Module is being applied to a variety of shape classification problems, including trademark logo matching and industrial parts identification.

The subject selected for one trademark experiment was Waldo from the "Where's Waldo?" game developed by Martin Handford. The goal in "Where's Waldo?" is to find the Waldo character in a complex cartoon image. [15]

Cartoon images are good test subjects for the Shape Module because edge detection is easy and texture classification is not required. The edge image of the cartoon can be input directly into the Geometry Module that will classify the edge pixels into the four canonical geometry classes. Figure 7a) is an example from the 20 Waldo face outlines used to train the Shape Module.

Figure 7b) is the geometry map of a scene cropped from a much larger Waldo image. When the Shape Module analyzed this test scene it produces a shape map for the Waldo class, Figure 7c), which shows the segments that match with at least 90% confidence. The segments shown in Figure 7c) that are not part of Waldo were found as matches because they are very similar to shapes within the "Waldo shape". Most of these segments match Waldo's circular glasses.

Figure 8 shows some sample images from another set of trademark experiments. Using the registered trademark of Lone Star restaurants because it is a star inscribed in a circle inscribed in a triangle. This shape shows how the Shape Module can identify shapes despite other overlapping shapes or missing portions. It also shows how the Shape Module can differentiate between a circle with a star inside and a circle without a star.

Figure 8 a) shows the original training image on the left and its reclassified shape map on the right. All segments shown in the shape map have a confidence of at least 90%. This shape map has some small gaps because segments smaller than 10 pixels were ignored in this experiment.

Figure 8 b) shows the logo rotated 90 degrees with a rectangle covering the logo's top. The shape map on the right shows that the Shape Module can still identify most of the visible logo.

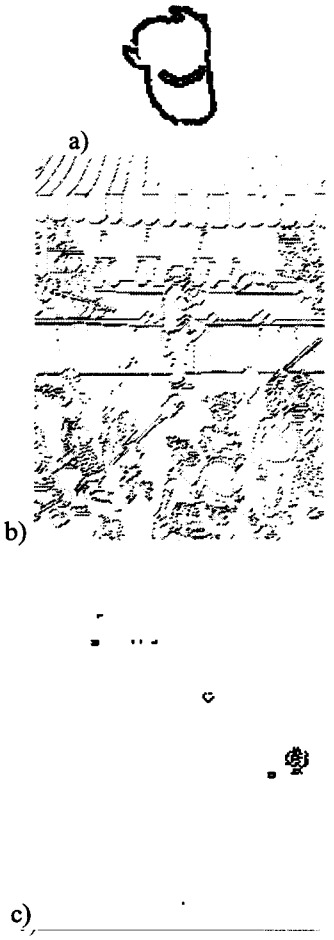


Figure 7 – Waldo examples [15]

In Figure 8 c), the logo is rotated 20 degrees from the original and scaled down to 80% of the original size. It is also partially covered by a rectangle with a circle inside it. The corresponding shape map shows that the visible part of the logo is still identified and that the circle in the rectangle is not confused for another logo.

Figure 8 d) shows the logo rotated 80 degrees with just over half of it missing. Although the Shape Module still recognizes about 60% of the visible logo, recognition begins to fail as more of the logo is erased.

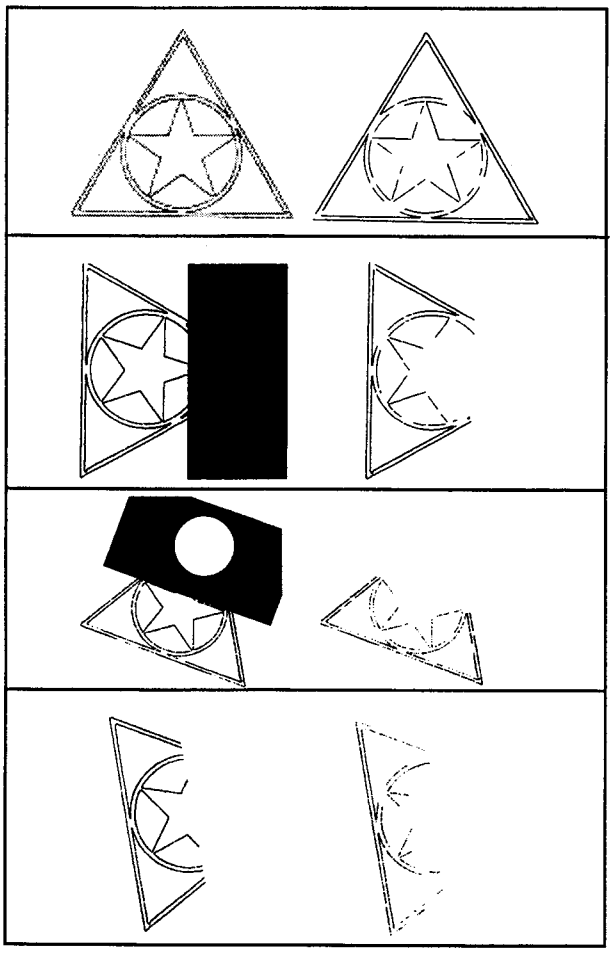


Figure 8 – Trademark matching
(test shape on left and resulting shape map on right)

®- symbols are a registered trademark of Lone Star Restaurants

6 References

[1] Barrow, H., et. al., "Parametric correspondence and chamfer matching: Two new techniques for image matching", Proceedings 5th Int'l Joint Conf. Artificial Intelligence, Cambridge, MA, 1977, pp.659-663.

[2] Becker, G. and P. Bock (1993). "Collective Learning Systems III: The ALISA Image Analysis System", PC AI Magazine.

[3] Becker, G. (September 1996). Information in Images, Thomson Technology Services Group, URL=http://www.thomtech.com/mmedia/tmr97/tmr97.htm.

[4] Blum, H., "A Transformation for Extracting New Descriptors of Shape", Models for the Perception of Speech and Visual Form, MIT Press, 1967.

[5] Bock, P., R. Klinnert, R. Kober, R. Rovner, and H. Schmidt (April 1992). "Gray-Scale ALIAS", IEEE Transactions on Knowledge and Data Engineering, Vol 4 No 2.

[6] Bock, P. and G. Becker (January 1993). "Collective Learning Systems II: The ALISA Image Analysis System", PC AI Magazine, p 42-44.

[7] Borgefors, G., "Hierarchical Chamfer Matching: A Parametric Edge Matching Algorithm", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 10, No. 6, November 1988, pp. 849.

[8] Clowes, M. B., "On seeing things", Artificial Intelligence 2, 1971.

[9] Cohen, P. and E. Feigenbaum, The Handbook of Artificial Intelligence, Vol. 3, William Kaufmann, Inc., 1982.

[10] Duda, P.E., and R.O. Hart, Pattern Classification and Scene Analysis, John Wiley & Sons, New York, 1973.

[11] Fischler, M. and R. Bolles, "Random Sample Concensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automatic Cartography", Communications of the ACM, ACM, 1981, vol 24 no 6, pp. 381-395.

[12] Foley, J., A. van Dam, S. Feiner, and J. Hughes (1990). Computer Graphics: Principles and Practices, 2nd Edition, Addison-Wesley Publishing Company.

[13] Fu, King Sun, Syntactic Pattern Recognition and Applications, Prentice/Hall, 1982.

[14] Gonzalez, Rafael and Paul Wintz (1987). Digital Image Processing, 2nd Edition, Addison-Wesley Publishing Company.

[15] Handford, Martin (1987), Where's Waldo?, Candlewick Press.

[16] Howard, C. (March 1995). "An Adaptive Learning Approach to Acquiring Geometric Concepts in Images", Dissertation Proposal, George Washington University, School of Engineering and Applied Science.

[17] Huffman, D. A., "Impossible objects in nonsense sentences", editors R. Meltzer and D. Michie, Machine Intelligence 6, Elsevier, 1971.

[18] Huttenlocher, D., Gregory Klanderman, and William Rucklidge, "Comparing Images Using the Hausdorff Distance", IEEE Journal of Pattern Analysis and Machine Intelligence, September 1993, vol. 15 #9, p 850.

[19] Shapiro, V., "On the Reconstructive Matching of Multidimensional Objects", IEEE Transactions on Image Processing, Vol. 5, No. 4, April 1996, pp. 653-661.

[20] Winston, P. H. (1977). Artificial Intelligence, Addison-Wesley Publishing Company, pp. 60-67, 206-235.

Text Recognition and Page Analysis

OCR of Low-resolution Text Images from Diverse Sources

Prem Natarajan, Richard Schwartz and John Makhoul
BBN Technologies, Verizon
Cambridge, MA 02138
{prem, schwartz, makhoul}@bbn.com

Abstract

In this paper, we discuss the effects of image resolution on OCR accuracy and propose techniques for improving recognition performance on low-resolution images. The system used is the BBN BYBLOS OCR system, which employs a script-independent approach based on Hidden Markov Models (HMM) for training and recognition. Three different low-resolution scenarios are discussed; (a) an Arabic newspaper corpus with 150 dpi grayscale images, (b) English videotext data where the resolution can vary dramatically from one segment to another, and (c) Arabic digital fax data received at 100x200 dpi resolution. We report OCR results for each of the three cases and demonstrate the effectiveness of the upsampling techniques as well as the basic robustness of the BYBLOS OCR system. We will also demonstrate how our ability to train the OCR models on each type of data allows us to deliver accurate and reliable recognition on a wide variety of data.

1 Introduction

In this paper we present techniques for dealing with low-resolution documents. A comprehensive evaluation of different OCR systems by Rice, et al. [1] has shown that OCR performance is dependent strongly upon the resolution of the text image. Furthermore, the authors of also provide plots of the character error rate (CER) versus resolution and show that the degradation for most systems accelerates once resolution drops below 200 dpi. In our work we have noticed that the CER increases by about 25% when the resolution drops from 600 dpi to 200 dpi. Based on this evidence, it is clear that if there exists an opportunity to algorithmically increase the resolution of the text image, then the concomitant reduction in CER would certainly justify such effort.

In the following sections we present our work with the three different types of low-resolution, text documents mentioned earlier. In each case we performed some pre-processing to enhance the low-resolution image into a higher resolution image with the aim of reducing the final recognition error rate.

The newspaper data consists of grayscale images of An-Nahar (a leading Arabic daily) scanned at 150 dpi grayscale. In Section 3 we show how upsampling the grayscale images before binarizing

enables us to cut the character error rate (CER) by a factor of 3 over direct binarization. Training our models on newspaper data that has been similarly upsampled and binarized reduces the CER by another factor of 3.

Our videotext data was segmented from color video images obtained from off-the-air TV broadcast news in English. This data presents three problems: (a) the text has very low resolution, (b) it is in color, and (c) it typically overlays a rich and varying background. We pre-process, upsample and binarize the videotext image before running it through our recognition system. While the baseline models yield results that are too errorful to use, training on similar data cuts the CER down to 8.3%. Details of this work are presented in Section 4.

In SDIUT'99, we dealt with the case of text images obtained from faxed and printed data, and proposed techniques to ameliorate the degradation in recognition accuracy. Here we deal with the case of electronically transmitted fax data, which is sampled nonuniformly at 100x200 dpi. Since our models are trained on data that is scanned at uniform resolution in both directions, a change in the aspect ratio of the text images causes serious recognition problems. So, we implemented a filtering scheme to upsample the data to 200x200 dpi and then ran our recognition system on the upsampled data. Section 5 discusses the details of this work. In the next section we provide a brief review of the BYBLOS OCR system.

2 Review of Basic OCR System

In this section we briefly review the working of the BBN BYBLOS OCR system which is a Hidden Markov Model based system. A significant advantage of HMM-based systems is that they provide a language-independent framework for training and recognition. At the same time, they do not require the training data to be segmented into words or characters, i.e., they automatically train themselves on non-segmented data. For a more detailed description the reader is referred to [2-5].

A pictorial representation of the system is given in Figure 1. OCR system components are identified by rectangular boxes and are independent of the particular language or script. Knowledge sources are depicted by ellipses and are dependent on the particular language or script. Thus, the same system can be configured to perform recognition on any language, provided training data in that language is available.

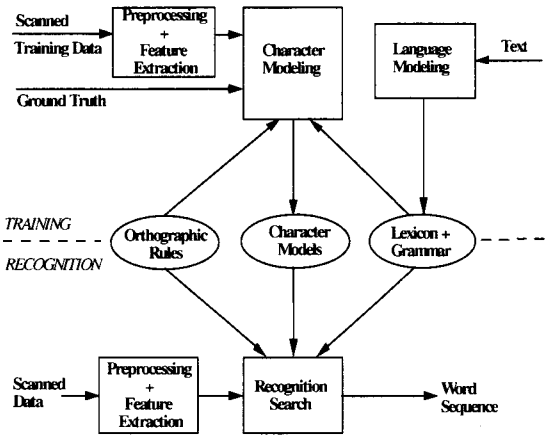


Figure 1: Block diagram of BBN BYBLOS OCR system

At the top level, the OCR system can be sub-divided into two basic functional components: training and recognition. Both, training and recognition share a common pre-processing and feature extraction stage. This stage starts off by first de-skewing the scanned image to ensure that the text boundaries are parallel to the image boundaries. After de-skewing, the line-finding

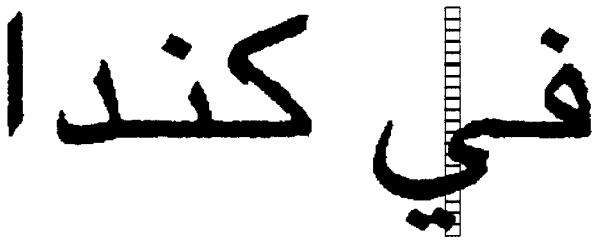


Figure 2: Feature Extraction

program locates the positions of the text lines on the de-skewed image. Finally, the feature-extraction program computes a sequence of feature vectors for each line (see Figure 2) as follows:

- Each line of text is horizontally segmented into a sequence of thin, overlapping, vertical strips called frames (one frame is shown in Figure 2).

- Corresponding to each frame, a language-independent 60-dimensional feature vector is computed to numerically represent the frame. Each feature vector is a function of the horizontal position of the corresponding frame.
- Linear Discriminant Analysis transformation is performed on the feature vector to reduce dimensionality from 60 to 15.

The OCR system models each character with a 14-state, left-to-right HMM, as shown in Figure 3. Each state has an associated output probability distribution over the features. The model parameters (the output probability distributions and the transition probabilities between states) are estimated from training data using the Baum-Welch or Forward-Backward algorithm.

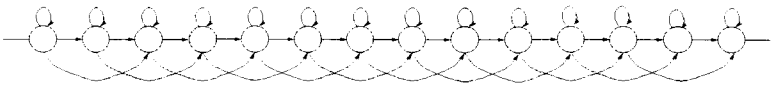


Figure 3: 14-state, left-to-right HMM topology with self-loops and skips

The Baum-Welch algorithm aligns feature vectors with the character models to obtain maximum likelihood estimates of HMM parameters. During recognition we search for the sequence of characters that is most likely given the feature-vector sequence and the trained character-models, in accordance with the constraints imposed by a lexicon and/or a statistical grammar. The use of a lexicon during recognition is optional but its use generally results in a lower CER. The lexicon is estimated from a suitably large text corpus. Typically the grammar (language model), which provides the probability of any character or word sequence, is also estimated from the same corpus.

3 OCR of Grayscale Document Images

While the BYBLOS OCR system works with binary images that contain black text on a white background, the core recognition engine is, by design, fundamentally independent of the nature of the input image. As such, all image and format specific information is handled at the pre-processing and feature extraction stage.

Given our current recognition system, there are two possible approaches to deal with grayscale data. The first approach is to train the system using features extracted directly from the grayscale images. The second approach calls for upsampling and binarizing the grayscale image. The binarized image can then be processed using the current recognition system without any changes.

Upsampling is done to preserve some of the grayness information in the original image by spatially encoding it in a higher dimensional bi-level image. In considering the two approaches it is useful to note that the grayscale representation notwithstanding, the underlying "true" text image is a bi-level image. Also, from a feature computation perspective grayscale images offer a serious normalization challenge because the dynamic range of the intensities of text and background pixels varies, both, locally within a zone and globally across zones. Keeping the above considerations in mind, we chose to implement the second approach. The upsampling and binarization procedure offers the additional advantage that it is a principled, generic methodology for completely reusing the existing system without changes.

3.1 Grayscale Corpus

The grayscale text-image corpus comes from the Lebanese newspaper An-Nahar. The images (256 level grayscale, i.e., 8 bits/pixel, scanned at 150 dpi) are taken from different issues over a

Figure 4: Example grayscale newspaper image

span of 2 years and are of one font type. At a grayscale resolution of 150 dpi, sufficiently high image quality is maintained. Figure 4 shows a sample column from the newspaper. As can be seen from the sample column, the amount of background gray varies in different regions of a single text zone.

3.2 *Pre-Processing the grayscale images*

As mentioned earlier, a comprehensive study [1] of different OCR systems performed by researchers at the University of Nevada shows that the recognition performance degrades sharply at (bi-level) resolutions lower than 200 dpi. In the case of the BYBLOS OCR system we see a more graceful degradation of performance but nevertheless, character error rates (CER) are significantly lower at a resolution of 600 dpi than at 200 dpi. As the numbers in Table 1 indicate, binarizing the

Figure 5: Raw (150 dpi) grayscale newspaper image

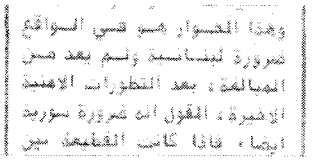


Figure 6: Upsampled (600 dpi) grayscale image

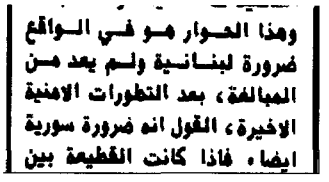


Figure 7: Binarized (600 dpi) newspaper image

grayscale image at 150 dpi results in a CER of 10%; much higher than our usual system CER of about 2% on newspaper data from the DARPA Arabic corpus. In order to better leverage the grayscale information in the raw image we decided to first upsample the grayscale image to 600 dpi before binarizing. Our upsampling procedure was as follows: insert three zero-valued pixels between the "original" pixels in every direction and apply an anti-aliasing filter with a $\pi/4$ cutoff frequency. The low-pass filter used was a 29x29 pixel finite impulse response filter whose coefficients were computed using the Remez algorithm. Figures 5, 6, and 7 show the steps in binarizing a small zone of grayscale text.

After upsampling the image we then binarize it using a simple nearest neighbor method. We chose the nearest neighbor method after trying several other, more complex, global and adaptive

binarization schemes. The nearest neighbor algorithm seemed to generate the best result as evaluated by visual inspection.

3.3 Experiments and Results

We performed three sets of experiments (see Table 1). In the first set we used models we have previously trained on 40 pages from the DARPA Arabic corpus. When we binarize the An-nahar data at 150 dpi without any upsampling, the baseline CER we obtained was 10.0%. In the second set, we upsampled and binarized the images to 600 dpi and the CER decreased to 3.4%. The third set of experiments involved training on data from the corpus. We selected a small training set, 9 zones totaling 540 lines, all single font. For testing we used another three zones of 204 lines. We used the same upsampling and binarization scheme described above both for training and test.

Training/preprocessing	Test	CER %
DARPA Arabic Corpus	Newspaper data from DARPA Corpus	2.0
DARPA Arabic Corpus	An-nahar data: *binarized at 150 dpi*	10.0
DARPA Arabic Corpus	An-nahar data: *Upsampled (600 dpi), filtered and binarized*	3.4
An-nahar data: *Upsampled (600 dpi), filtered and binarized*	An-nahar data: *Upsampled (600 dpi), filtered and binarized*	1.1

Table 1: Error-rates for different pre-processing and training conditions

The CER we obtained was 1.1%, a reduction by a factor of 3 over the system trained on the DARPA data. This result shows that our system can handle grayscale data very well, and that we can get as good performance on low-resolution grayscale data as on high-resolution binary data.

4 OCR of Video-text Images

In broad terms, the Video OCR problem may be decomposed into three somewhat independent processes (see Figure 8):

- Detecting the existence and location of text within each frame in the video stream
- Enhancing (removing background, upsampling) and binarizing the text image
- Recognizing the text in the processed images

In this paper our focus is on steps 2 and 3 above. As such, in the experimental results presented later we used annotated text blocks as the input text images to the system.

The BYBLOS OCR system works with binary images that contain black text on a white background whereas video data contains colored text on a colored, textured background. The aim

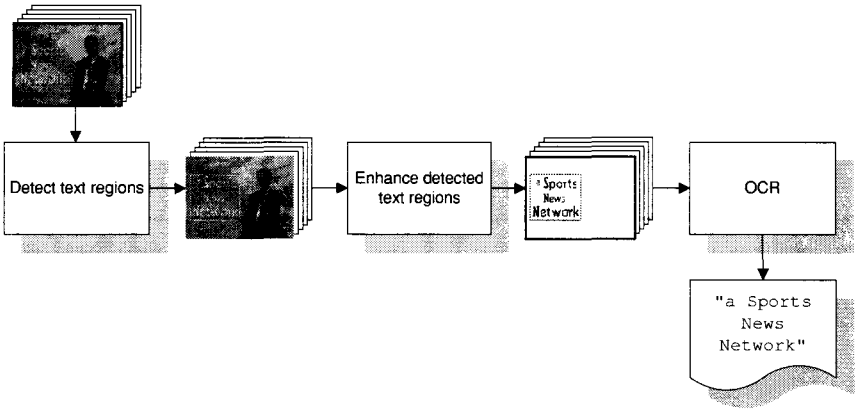


Figure 8: Pictorial illustration of OCR procedure for video documents

of our text enhancement procedure is to binarize the color text image for input to the OCR system. While there have been several approaches to binarizing color text images, most of them include a pre-processing stage that converts the color image to a grayscale image. All downstream processes are then performed on the grayscale image. Oftentimes, video data contains examples where the text region and the background have almost the same intensity; the only difference being that the two regions are of different color. In such cases conversion to grayscale blurs the text-background boundaries, making it impossible to binarize the image accurately. Our approach has been to incorporate the color information into the binarization procedure in order to improve system performance.

The first step in our binarization procedure is to enhance the text image to amplify the contrast between the text and the background; this is discussed in the following section.

4.1 Enhancing the Text Image

A typical characteristic of text in video is that a given text region persists over a few frames of video feed during which the background may or may not vary. In fact, more often than not the background varies while the text remains static. By leveraging this persistent nature of text and the dynamic nature of the background, it is possible to substantially improve performance of any binarization procedure. The *enhanced* image is computed by aligning the different instances of a particular text region across frames and, for each pixel, choosing the color that corresponds to the minimum intensity value across frames. We tried other order statistics such as the mean, median and the maximum but the minimum order statistic yielded the best image in terms of visual perception. At present our binarization technique is designed to handle data in which the text is brighter than the background. For such text, using the minimum is extremely effective in reducing the complexity of the background. Figure 9 below shows the application of the min image procedure to a text region that persists for 107 frames. As can be seen, the minimum image provides an excellent starting point for the text extraction and binarization procedure. The resulting image in Figure 9 illustrates the effectiveness of the minimum image procedure in enhancing the contrast of the videotext image.

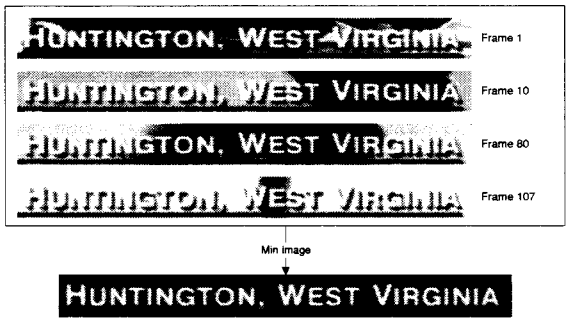


Figure 9 : Computation of minimum image

4.2 Correlation-Based Technique

Sato [6] describes a correlation technique for binarizing videotext images. The essence of Sato's technique is to model various text strokes using different matched filters. In his paper he reports on the use of four separate filters to model horizontal strokes, vertical strokes and two diagonal strokes, at 45 and 135 degrees to the horizontal. The filters are trained by marking suitable regions on sample training data. Examples of marked training regions for horizontal and vertical filters and corresponding trained filters are shown below in Figure 10.

Each videotext image is correlated separately with each of the four filters and the correlation outputs are thresholded to yield four intermediate binary images. The final binarized image is the union of the four intermediate binarized images. While the technique is simple to implement, we found that it had several shortcomings as described by Sato himself. The most prevalent problem was that background components occasionally exhibit text-stroke like characteristics and are

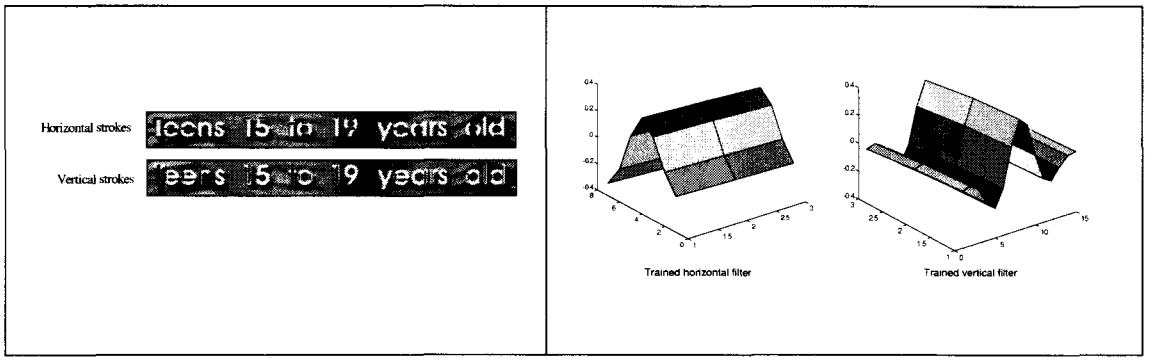


Figure 10: Marked training regions and trained correlation filters for text extraction

routinely picked up by the filters. Furthermore, such background components have high correlation scores and cannot be completely eliminated by adjusting the threshold. Another obvious drawback is that by constraining the binarization procedure to use the grayscale image alone, valuable

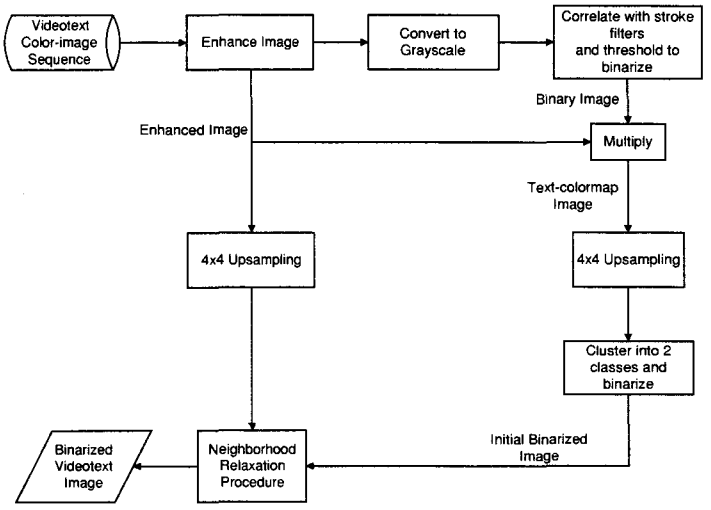


Figure 11: Block diagram of binarization procedure

information in the color image is summarily discarded. The binarized images also lack the smoothness that characterizes the curves and loops in characters such as **c**, **d**, **o**, etc.

Notwithstanding all of the problems listed above, the correlation technique does an excellent job of locating the position of text pixels. Thus, while the final binarized image may be morphologically lacking, it does contain most of the text pixels in the original image. Based on this observation we have developed a binarization scheme that uses the correlation method as the first step and then reverts back to the color image for improved performance.

Figure 11 shows a block diagram of our binarization procedure. We first use the correlation technique to binarize the text image so that text pixels have a value of 1 (white) and background pixels are assigned a value of 0 (black). The binarized image is then multiplied with the minimum image to create a *text-colormap* image. In the text-colormap image, those pixels that were classified as background remain black while those that were classified as text take their color from the minimum image.

Since the resolution of the video is typically low, we then upsample the text-colormap image and minimum image by a factor of 4 along each dimension, i.e., each pixel in the original image is

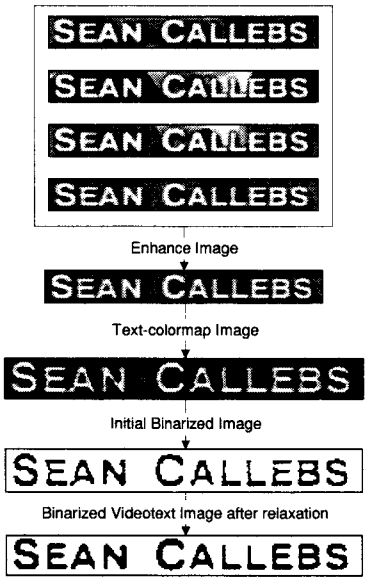


Figure 12: Videotext image enhancement and binarization example

replaced with a 4x4 block of 16 pixels. We then use binary K-means clustering on the pixels of the upsampled text-colormap image to estimate the average color value for text pixels and background pixels. For each pixel on the text-colormap image, if the pixel's color is closer to the average text-color than to the average background-color the pixel is classified as a text pixel, otherwise it is

classified as a background pixel. This clustering procedure allows us to eliminate background pixels that have the same intensity as text pixels, but are of a different color. To further ensure that the edges of the text strokes are not jagged due to aggressive thresholding, we use a relaxation procedure that searches for text-color pixels within a pre-specified neighborhood of text pixels. After relaxation we obtain a smooth, binarized version of the original text image. The binarized text image is then recognized using the BYBLOS OCR system. Figure 12 above illustrates the procedure for a sample text image.

4.3 Recognition Experiments

To test the performance of the BYBLOS English OCR system on the binarized videotext data, we used an in-house video corpus collected from CNN Headline News broadcasts. A training set of about 14,000 characters was identified from data collected on three different days. For the test set, we used about 4500 characters from two other days. The test and training were from different days.

Category	Contribution to Total CER %
Line-finding	0.9
Binarization	1.8
Underlined text	1.7
Drop shadow text	1.2
Small font size	1.1
Other	1.6
TOTAL	8.3

Table 2: Decomposition of overall CER into different categories

First, the training and test were both binarized using the binarization procedure described above. The OCR system was then trained on the binarized data using the usual training procedure. The trained models were tested against the test data and a character error rate of 8.3% was obtained using a tri-gram language model on characters. We find this result extremely encouraging and

promising, especially, given that this also includes the effect of the binarization procedure. No lexicon was used during decoding.

Table 1 below divides the errors into broad categories. About half the errors are contributed by categories that constitute a small fraction of the test data; underlined, drop-shadow, and small font-size text. A more generic type of error is that caused by occasional inconsistencies in the binarization and/or line-finding procedure. Other errors come from text that is overlaid on a highly textured background, from text that is darker than the background, etc.

5 Digital Fax Data

Digital fax documents provide a natural application for an OCR program. Raw text images already exist in electronic format and the advantages of transforming the images to ASCII text are enormous. Also, unlike videotext and newspaper text, digital fax images are bi-level text images by design. As such the issue of binarizing the fax image does not arise. Nevertheless, standard digital fax images require upsampling: the standard fax image is sampled at a non-uniform (nominally) 100x200 dpi resolution. The BYBLOS OCR system is trained on text images that are sampled at the same rate in both directions. In order to recognize digital fax images using the regular BYBLOS OCR system, it is necessary to upsample the images and make the sampling rate the same in both directions.

While upsampling grayscale and videotext images with the ultimate aim of creating a high-resolution bi-level image, there was the opportunity to take the grayscale/color information and spatially encode it in a higher dimensional bi-level image. Also, both grayscale and color images tend to be smooth, continuous signals that are suitable for filtering by standard Fourier techniques. On the other hand bi-level images are discontinuous and are clearly not suited for Fourier processing. Based on this observation we decided to implement a simple majority-rule filter with the aim of producing a smooth 200x200 dpi text image. First we inserted a column of background-valued pixels between every two columns in the original fax image as well as a row of background-valued pixels between every two rows in the original image. For each background-valued pixel that we had inserted we counted the number of text pixels in a 3 x 3 pixel window centered on that background-valued pixel. If the number of text pixels was greater than 4, we modified the pixel value to a text pixel otherwise we left it as it was.

For our experiments we used data from the DARPA Arabic corpus that contains Arabic text images scanned at a resolution of 600 dpi. Our baseline system was trained on data from the DARPA Arabic Corpus, and the average character error rate was 3.2% on a test set from that corpus. The same test data was printed and then faxed electronically at 100x200 dpi; then the faxed data was upsampled to 200x200 dpi. Using the same baseline recognition system on the fax data, the error rate increased to 7.5%. When we retrained our system on similarly faxed data and tested on the same fax test set, the error rate decreased to 5.7%. From our earlier experiments with the BYBLOS OCR system we know that a reduction in resolution from 600 dpi to 200 dpi results in about a 30% increase in error rate. Based on that evidence we expect the CER to go up from 3.2% to about 4.2%. The additional increase from 4.2% to 5.7% can, in all probability, be attributed to fax noise and the fact that the actual sampling in one direction was only 100 dpi.

6 Conclusions

In this paper we have presented a principled methodology for dealing with low-resolution documents within the framework of the BYBLOS OCR system. We have also demonstrated that with proper upsampling (and binarization) procedures, it is possible to obtain performance levels similar to that with high-resolution bi-level images. In each case it was seen that training the BYBLOS OCR system on relevant data provided a significant improvement in the performance as measured by character error rate. Given the modular nature of the training and recognition steps, it is very easy to reconfigure them to use an image processing front-end that can process various types of images into the standard bi-level text image format that the BYBLOS OCR system expects. Also, since the image processing techniques presented are independent of the language/script, the language and script-independence of the BYBLOS OCR system is preserved in the new configuration.

7 References

[1] S.V. Rice, J. Kanai, and T.A. Nartker, "An Evaluation of OCR Accuracy," *Information Science Research Institute, 1993 Annual Research Report*, University of Nevada, Las Vegas, pp. 9-20, 1993.

[2] P. Natarajan, Z. Lu, R. Schwartz, I. Bazzi, and J. Makhoul, "Multi-lingual Machine Printed OCR," *Intl. Journal of Pattern Recognition and Artificial Intelligence*, Vol. 15, No. 1, February 2001.

[3] I. Bazzi, R. Schwartz, and J. Makhoul, "An Omnifont Open-Vocabulary System for English and Arabic," Vol. 21, No. 6, pp. 495-504, June 1999.

[4] R. Schwartz, C. LaPre, J. Makhoul, C. Raphael, and Y Zhao, "Language-Independent OCR Using a Continuous Speech Recognition System," *Proc. Int. Conf. on Pattern Recognition*, Vienna, Austria, pp. 99-103, August 1996.

[5] L. Nguyen, T. Anastakos, F. Kubala, C. LaPre, J. Makhoul, N. Yuan, G. Zavaliagkos, and Y. Zhao, "The 1994 BBN/BYBLOS Speech Recognition System," *Proc. ARPA Spoken Language Systems Technology Workshop*, pp. 77-81, January 1995, Morgan Kaufmann Publishers.
[6] T Sato et al., "Video OCR: Indexing Digital News Libraries by Recognition of Superimposed Caption," *ACM Multimedia Systems Special Issue on Video Libraries, 7(5): 385-395, 1999.*
[7] Huiping Li, et al., "Automatic Text detection and Tracking in Digital Video," *Language and Media Processing Laboratory technical report*, http://documents.cfar.umd.edu/LAMP/Media/Publications/Papers/huiping98b/Text2.ps.Z
[8] J.D. Hobby and H.S. Baird, "Degraded Character Image Restoration", *Fifth Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, Nevada, pp. 233-245, April 15-17, 1996.
[9] L. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *Proc. IEEE*, Vol. 77, No. 2, pp. 257-286, Feb. 1989.
[10] G. Kopec and P. Chow, "Document Image Decoding Using Markov Source Models", *IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 16, No. 6, pp 602-617, 1994.*

Recent Work in the Document Image Decoding Group at Xerox PARC

Thomas M. Breuel and Kris Popat
Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304

1 Overview

Speed Enhancements to DID (Section 2) When Document Image Decoding (DID) was proposed [15], its attractiveness lay primarily in its potential for high recognition accuracy, owing to its communications-theoretic framework, and well defined models and objective function (posterior probability). In its initial implementations it suffered from high computational cost relative to commercial OCR methods. We will summarize recent progress made on reducing its computational cost. Importantly, these speed enhancements do not come at the expense of accuracy; they are guaranteed to result in the same recognition output as DID without the enhancements.

DID with Language Models (Section 3) Until recently, DID achieved its high recognition accuracy without the benefit of linguistic knowledge. Recent work on the incorporation of linguistic knowledge in DID's search procedure will described.

Grayscale DID (Section 4) The document image decoding framework is quite general, but for computational reasons previous work has focused primarily on binary images. The emergence of alternative image acquisition devices motivates its extension to grayscale. We consider one approach and present preliminary results.

Layout Analysis (Section 5) Layout analysis infers document structure from the arrangement of text and graphical elements in documents and uses that structure for higher-level tasks like matching, segmentation-by-example, layout based retrieval, and document indexing. Many existing systems attempt to find a single representation of document layout prior to solving the high level task. We have developed an approach to document layout analysis that is based on explicitly exploring the space of segmentation parameters as part of the overall segmentation task. In particular, the approach considers all geometrically dissimilar layouts in tasks like layout-based retrieval and segmentation-by-example. This new approach promises to make segmentation-by-example and layout-based retrieval systems considerably more robust than previous approaches.

OCR By Clustering (Section 6) We have re-examined a well-known technique in OCR, recognition by clustering followed by cryptanalysis, from a Bayesian perspective. The advantage of such techniques is that they are font-independent, but they appear not to have offered competitive performance with other pattern recognition techniques in the past. Our analysis suggests a novel approach to OCR that is based on modeling the sample distribution as a mixture of Gaussians. Results suggest that such an approach may combine the advantages of cluster-based OCR with the performance of traditional classification algorithms.

Classification by Probabilistic Clustering (Section 7) Extending and generalizing our prior work on OCR by clustering, we have developed novel methods for probabilistic clustering. These methods promise to make OCR more robust to font variations and novel document degradation conditions, and they also have applications in other classification problems where the distribution of test samples may differ from the distribution of training samples. We describe some experiments demonstrating that the approach outperforms traditional classification methods in an OCR task.

2 Speed Enhancements to Document Image Decoding

Document image decoding involves searching a trellis for a best path that explains the observed text image, where the nodes in the trellis correspond to locations in the image, and where the edges in the trellis are labeled with a score of matching a hypothesized character beginning at that location. At each node, the number of outgoing edges is equal to the

number of characters in the font, plus special whitespace characters. The best-path search has traditionally been carried out using the Viterbi algorithm, a form of dynamic programming.

In the past, the computational cost of performing the best-path search was dominated by the computation of the match scores to be assigned to the trellis edges. Three recent innovations have reduced this cost.

First, the one-pass Viterbi search has been replaced by an iterative scheme which involves repeatedly finding a best path by Viterbi, but initially using inexpensively computed upper bounds on the match scores. On each iteration, any edges labeled with upper-bound scores along the path found are re-labeled with their (expensive) true values. Eventually, a path will be found in which all of the edge labels are the true match score values; since this path has beaten all other optimistically scored paths, it must be a truly highest-score path. The savings comes about because the vast majority of true scores need never be computed; only those determined to be promising on the basis of the upper-bound scores are kept alive. The specific upper bounds used initially in this approach were best-case matches determined from counts of foreground pixels in text-line columns. This general approach, dubbed the Iterated Complete Path (ICP) algorithm, traces its roots to work in separable Markov source modeling [13] and was extended to within-line decoding more recently [17].

Second, when ICP is used as described above, portions of the path found in one iteration are re-used without re-computation in the next iteration, when it can be determined that the boundary conditions of the path segment are such that the best path in that segment cannot change. Specifically, when it is noticed that the cumulative scores attached to the nodes in the current iteration differ from those in the previous iteration by a constant value that persists over a run of pixels exceeding the maximum character width, the best path will not differ in that segment from the one found on the previous iteration, until an edge is encountered that has been re-scored. Empirically, we have noticed that most of the path doesn't change during the vast majority of the ICP iterations, so the savings thus accrued is substantial. The technique of re-using path segments in this manner is referred to as *Incremental Viterbi* and is also described in [17].

A final speed enhancement results by modifying the upper-bound used in ICP to group multiple columns together, which amounts to horizontal subsampling. The best-case match is computed not for each column individually, but rather for two, three, or four columns in the aggregate. The efficacy of grouping multiple columns in this way depends on many factors, including the scan resolution. Details will be provided in a forthcoming publication.

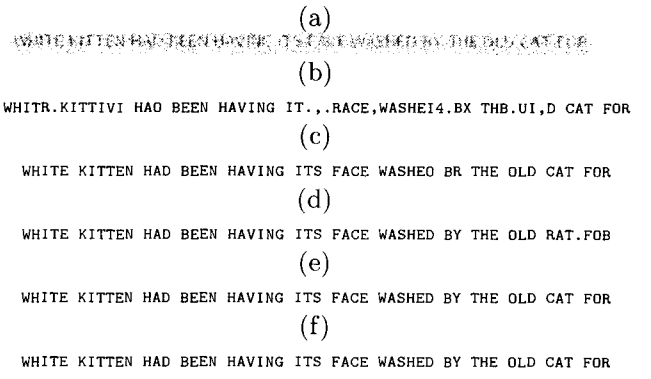


Figure 1: Example of the effect of integrating a language model into document image decoding, using several different strategies. (a) degraded, subsampled grayscale synthetic text line image; (b) decoding without a language model; (c) unigram language model via Viterbi; (d) Stack algorithm; (e) generalized ICP algorithm; (f) ground truth.

Combined, the above enhancements have been found to improve the speed of DID by a factor of about forty on a small set of standard text images.

3 Document Image Decoding with Language Models

Until recently, DID had no mechanism to express prior preference for linguistically valid strings as recognized output over invalid strings. We have considered several approaches, settling on a class of approaches in which soft linguistic constraints are expressed by a sequentially predictive probability distribution over characters, conditioned on a fixed number of previous characters (typically four). This probability distribution is called a *language model.* Paths now have their edges scored with both a match component and a language model component. In principle, the trellis must be vastly expanded so that nodes can now encapsulate linguistic context in addition to position in the image. One approach is to think of the expanded trellis as a full tree, and apply an approximate search procedure to find a nearly-best path. The approximation comes about because of the practical necessity of avoiding searching the full tree of all possible messages. We have examined one such technique, the Stack algorithm, which is widely used in speech recognition [11] and in convolutional decoding [12], and found it to be promising [21].

We have also developed an iterative algorithm, much in the spirit of the ICP algorithm described in Section 2. We will refer to it here as a *general-*

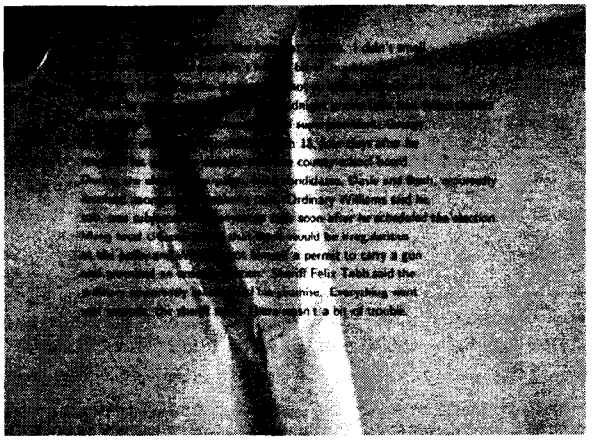


Figure 2: Deliberately wrinkled document image acquired in low-light conditions by a handheld digital camera, used to test the extension of DID to grayscale described in [19].

ized ICP algorithm. Rather than re-score edges on each iteration, nodes are added to encapsulate additional linguistic context along paths that are deemed promising, based on lower-order upper bounds on the language model scores. As the context approaches the full context exploitable by the language model, the upper bound scores approach and ultimately reach the true language model scores. When a path is found having only true language model scores on each edge, that path can be concluded to have the highest score among all paths, and the algorithm terminates. This algorithm has the advantage over the Stack algorithm and other approximate-search algorithms that it results in a true best path, but has the disadvantage that its computational complexity is strongly data-dependent. On the other hand, it can be set up to remember the best path seen so far, and to output that upon early termination. In other words, to limit computational complexity, it can be set up as an *any time* algorithm for approximate best-path search.

Figure 1 shows how language modeling in its various forms can influence DID recognition accuracy. The text line used in this example was severely corrupted by additive noise to make the error rate high enough so that the differences would be clear. The text line shown in (a) is for illustration and is actually *less* noisy than the one used for recognition, which is visually unintelligible. Both the Stack algorithm and generalized ICP yield high accuracy in this example. For more details on these approaches, see references [21] and [20].

4 Grayscale Document Image Decoding

The emergence of low-cost handheld digital cameras as a viable means of document image acquisition motivates the extension of the DID to function on relatively low-resolution grayscale images. Doing so involves significantly generalizing the channel model used by DID. One approach [19] involves carrying out the search in a high-resolution hypothesis image domain, and simulating the physical sampling and noise processes to match against the observed image. Initial results on a challenging test case are promising; the technique was found to perform favorably relative to the simple alternative of adaptive thresholding followed by application of a standard commercial OCR product. Figure 2 shows the test image used for this experiment. The edit distance between ground truth and the result of grayscale DID (with unit weighting for substitutions, insertions, and deletions) was seventy, versus ninety-one for binarization followed by commercial OCR. While preliminary, these results are felt to be encouraging.

5 Layout Analysis using the Document Scale Space[1]

The layout of elements on a printed page conveys a wealth of information about a document. Much of the research on document layout analysis attempts to recover a single representation of the layout of a document. In many commercial OCR applications, the goal of representing document layout is to be able to recover the layout sufficiently well for making the document editable and presentable in a word processor or HTML. Another important application of layout analysis is in information extraction from documents for the purpose of document retrieval, as well as appearance based retrieval (reviewed in [6]).

A large number of different approaches to layout analysis have been described in the literature ([16] contains a list of references). Many systems perform a non-probabilistic bottom-up analysis based on the distances between connected components (e.g. [18]) or based on an analysis of the whitespace (e.g., [10]). Such systems generally require a number of numerical thresholds and parameters to be picked; e.g. thresholds at which characters are merged into lines, thresholds at which lines are merged into paragraphs, etc. Sometimes, these parameters are picked globally for the whole page image, but they can also depend on the local context.

More recently, Liang [16] has described a Bayesian approach that starts with similar primitives but computes a statistically optimal segmentation of the complete page, taking into account higher order constraints among layout elements. Another approach to document layout analysis assumes that there is a known, underlying logical and/or hierarchical model that describes the document (e.g., SGML, HTML,

[1] This section is based on, and contains excerpts from, a paper presented at the DAS '2000 workshop[2].

or TeX source) and attempt to match such models against the physical representation of a document [23, 5]. When the underlying physical segmentation does not correspond well to the given logical model, conflicts are resolved in some cases using backtracking search.

This work proposes an approach to document layout analysis that differs in several ways from these other approaches. The key ideas are:

- The space of all possible physical segmentations is explored and represented efficiently and completely as a *document scale space.*
- The document scale space, rather than a single documentation, is used in layout matching tasks.
- The approach integrates the exploration of different segmentations directly into tasks like layout matching or segmentation by example.
- The approach is motivated using a Bayesian analysis of the layout matching problem.

Two applications for this work are *apperance-based retrieval* and *segmentation by example.* Retrieval of documents from document databases based on their physical or logical layout has been described, for example, by Doermann *et al.* [7]. The idea is to first perform a layout analysis of the documents in the database and the query document and then to compare the layouts for the purposes of retrieval. As we will see below, layout based retrieval can benefit significantly from incorporating the segmentation step directly into the layout matching process.

Appearance-based retrieval can also be used for segmentation by example tasks. The basic idea is to match an unsegment query document against a database of manually segmented documents in a database. The segmentation of the best match found in the database can then be used to segment the query document. Segmentation-by-example tasks occur frequently in legacy conversions of company memos, patent documents, scientific journals, and medical data sheets. For such tasks, it would be desirable if unskilled users could indicate regions of interest on a few sample pages and the system could then use those samples to identify reliably corresponding regions in the document database.

5.1 Document Scale Space

To see how we can compute a document scale space efficiently, we need to look in more detail at how traditional document layout analysis methods work.

Single-Parameter Case A common approach to document layout analysis is based on choosing a threshold θ on the minimum Euclidean distance between connected components found in a document. Connected components that are closer to one another than the given threshold θ are grouped together into *layout components.* This thresholding operation partitions the set of connected components into a collection of disjoint sets. We call this collection of disjoint sets a *segmentation* or a *physical layout analysis* of the document.

Different thresholds θ give rise to different segmentations. If we look at the the different segmentations parameterized by θ, we obtain a structure similar to the scale space widely used in computer vision. We refer to this as a *single parameter document segmentation scale space.* It should be noted that, unlike in the computer vision case, there are only a finite number of distinct segmentations in the document segmentation scale space.

It is generally assumed that the hierarchy implied by a logical layout description of the page (page > column > paragraph > line > word) is paralleled in the document segmentation scale space. In practice, this assumption seems to be fairly well satisfied for some classes of documents, but in general, it clearly is not satisfied by many document layouts. Determining the actual threshold parameters to the different levels of logical layout themselves also involves some experimentation and heuristics. The methods described in this paper address both these problems.

Multiparameter Case We can extend the notion of a document segmentation scale space that we developed above to a case where thresholds and distances are evaluated differently in the x and y directions. In particular, for each pair of connected components, let us measure two distances, their horizontal distance and their vertical distance. We define the horizontal distance between two connected components to be infinity unless their vertical extent overlaps. If their vertical extents overlap, their distance is simply the distance between their bounding boxes. We can make an analogous definition for the vertical distance of two connected components. A segmentation is now given by picking two thresholds, θ_x on the horizontal distance and θ_y on the vertical distance. We can apply the same arguments as above and see that there are at most N different choices for each θ_x and θ_y, so there are at most N^2 different two parameter segmentations of the input.

5.2 Computing Document Scale Space

To compute and represent the document scale space, we use the following approach. First, the bound-

ing boxes of the connected components are stored in a trie data structure, which allows us to determine quickly the nearest neighbors of each connected component in the horizontal and vertical directions. Using this neighborhood information, we construct a graph, in which the connected components are the nodes and the nearest neighbor relationships define the edges. We can now take the set of all horizontal edges and all vertical edges and sort them by their distance.

For a particular choice of horizontal and vertical thresholds, we retain all edges in the neighborhood graph corresponding to distances less than the chosen thresholds. We then compute all the connected components of the neighborhood graph efficiently using a union-find algorithm.

To speed up this process further, we can use incremental updates of the data structures; that is, if we have computed the segmentation for given horizontal and vertical thresholds, if we increase either threshold, we do not have to restart the computation from scratch but update data structures incrementally. When applied to real document images, we obtain a few thousand distinct segmentations per document image in this way. The collection of these segmentations (together with their associated threshold parameters) is a complete representation of document scale space.

While fast enough for analyzing indidvidual documents, when searching through a large document database, performance is proportional to the size of the document scale space. To improve performance further, we would like to come up with a more compact representation. We can achieve this by decimating the document scale space and retaining only representatives that are "substantially different". If we consider segmentations that differ by less than 5% from one another (meaning, they have approximately the same overall structure and the areas of the different segmentation components differ by less than 5%), the number of distinct segmentation is reduced from 1000 or more to under 50 for most documents.

5.3 Applications

Based on these ideas, a prototype system was implemented that allows layout-based (appearance-based) retrieval of documents from the University of Washington Database 1. There is currently no widely used benchmark for document retrieval based on layout or physical appearance, but representative examples of queries and top matches are shown in Figure 3. Searches run at the speed of about 3.7 seconds per 1000 models on a IBM ThinkPad 600E (400MHz, RedHat Linux 6.1).

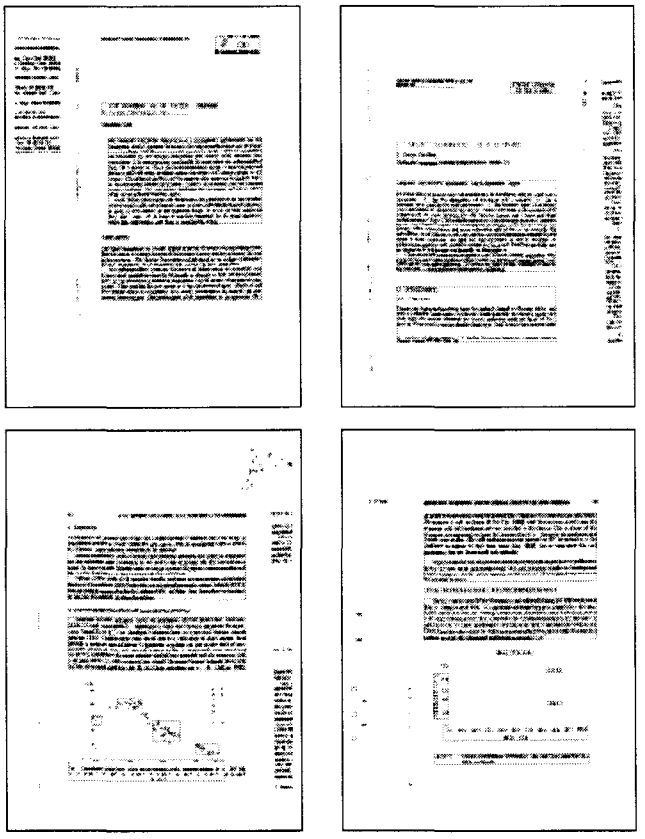


Figure 3: Layout-based retrieval.

5.4 Conclusions and Future Work

This work describes a number of ideas important for layout analysis:

- task-driven segmentation: the segmentation parameters are selected in an integrated way with the overall task (layout-based retrieval, segmentation by example, matching against a logical layout model, etc.);
- anisotropic, multi-parameter segmentations;
- document segmentation scale space;
- the use of representative sets of segmentations to speed up task-driven segmentation.

The paper motivated these ideas with geometric and Bayesian arguments.

These approaches help to address two fundamental problems in document analysis: that of unnecessary early commitment to a (possibly erroneous) bottom-up segmentation, and the dependence of many layout analyis methods on empirical, highly database dependent threshold parameters.

Further experiments need to be carried out to evaluate the performance of task-driven segmentation on layout-based retrieval and other tasks. However, the preliminary experiments presented in this

work suggest that the approach can be both efficient and gives reasonable results on a commonly used database of documents.

6 OCR By Clustering[2]

This work examines the effects of clustering character images prior to recognition in optical character recognition (OCR) of printed documents. This approach has a long history in OCR, and prior work has addressed the questions of how to build a clustered representation quickly [4], as well as how to label the resulting clusters. Clustering, mixture models, and mixture-based Bayesian recognition itself, of course, has a long history in statistics and pattern recognition. In this work, we make a connection between the two approaches. The key point is that the clustering of the character templates is, in effect, a mixture density estimation of the sample distribution. This connection allows us to reexamine issues of cluster validity, style adaptation [22], and cluster label assignment within a Bayesian framework.

6.1 The OCR Problem

For the purposes of this work, we will define the OCR problem in the following simplified manner. We assume that there is a fixed, finite set of characters (digits, lower case letters, upper case letters, special characters). Furthermore, we assume that there is an open-ended set of possible styles, where the notion of style encompasses character properties like font, size, and idiosyncracies of the particular rendering engine used. Picking a character and a style uniquely determines an idealized image (bitmap) for the character. During document creation, this idealized bitmap is printed on a piece of paper. When the document is scanned back in again, a degraded bitmap of the character, is obtained, usually by the addition of noise, blurring, thresholding, sampling error, and various forms of geometric distortions [14]. The core function of an OCR system is (roughly) to find the most likely character and style corresponding to such a degraded character image.

Traditionally, OCR systems perform this task by estimating posterior probabilities like $P(\text{char}, \text{style}|\text{bitmap})$, say, using a neural network or a Gaussian mixture model. However, estimating such probability distributions requires a large number of example characters. In practice, however, training data for many styles (fonts, degradation parameters) is not available at all.

[2]This section is based on, and contains excerpts from, a paper presented at SPIE '2001[1].

6.2 OCR by Solving a Cryptogram

An alternative approach proposed in the literature [4] is based on the idea of clustering similar character shapes and then assigning character labels to the resulting clusters. Such an approach is attractive because the clustering process itself is font independent, and cluster labels can, ideally, be assigned independent of the actual bitmap representation of the characters. This, on ideal data, such an approach is completely font independent and automatically generalizes to arbitrary unknown fonts. Clustering is also attractive because of the emergence of token-based compression methods that already represent documents as a collection of tokens. If we can carry out recognition directly on these clusters, we can perform OCR directly on token-compressed data.

However, in practice, such methods for carrying out OCR by clustering have not been very successful. The reason is, this paper argues, that the clustering methods used have modeled the actual statistical nature of the recognition problem poorly. This work describes how to begin combining the advantages of font independence of clustering OCR systems with the robustness of statistical methods used in current commercial OCR systems.

6.3 Gaussian Mixture Models

Let us assume, for the purpose of illustration, that each character image in the input document is represented by a feature vector $\tilde{v}$ that is derived from a prototype feature vector $v_{c,s}$ representing character c and style s corrputed by and additive error G with zero mean and Gaussian distribution. In such a framework, the class conditional densities are $P(v|c,s)$ are then Gaussians. If we take a supervised pattern recognition approach, we estimate the class conditional densities (or, equivalently, priors and posteriors) from training data, derive discriminant functions, and use those to classify each unknown feature vector $\tilde{v}$ in a Bayes-optimal sense as one of the different classes c, s.

The problem with this approach is that estimating the class conditional densities depends on a representative sample of degraded feature vectors over character classes c and styles s. If we do not have such a representative sample, our class conditional density estimates are going to be poor and recognition accuracy suffers.

We can, however, take a different approach using a partially unsupervised method involving the sample distribution. In a real OCR problem, we are usually given not a single character to classify, but many thousands of samples, one for each character in the document. Of course, these characters are not labeled, so we cannot derive the class conditional densities from this sample. However, what

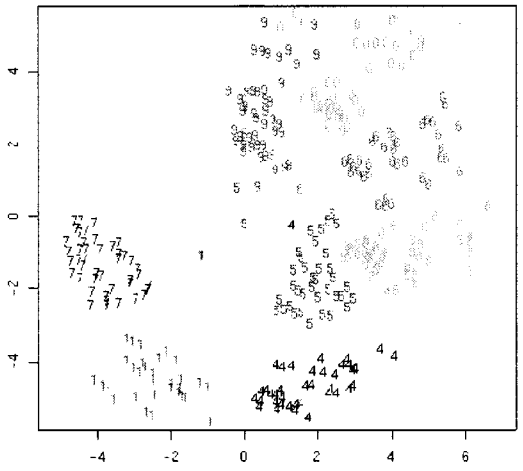


Figure 4: Low-dimensional representation of character feature vectors.

we can do is model the sample distribution, that is, the distribution of degraded feature vectors $\tilde{v}$, ignoring their class labels. If we assume that the class conditional densities are Gaussians, then the sample distribution is going to be a Gaussian mixture component (the frequency of the classes, c, s, are the mixture parameters). If we then try to recover the mixture components of this Gaussian mixture, we recover the individual class conditional distributions. We still do not necessarily know the classes corresponding to each such class conditional distribution, but we have a lot more information at our disposal to assign such labels than if we tried to classify characters one-by-one.

This idea is illustrated in Figure 4, which shows a two-dimensional representation (a Sammon mapping) of the feature vectors representing severely degraded samples of digits from a single font. In the figure, class assignments of the different samples are indicated by colors and labels. Looking at clusters with this information corresponds to estimating the class conditional density given labeled training data. But it is clear that even if we do not have labels available, the data still falls into fairly distinct clusters; recovering these clusters without using class labels corresponds to the clustering OCR described in this paper. (In their original, high dimensonal space, these clusters are considerably better separated than in the low-dimensional non-linear mapping shown in the figure.)

If we compare this to previous methods for OCR by clustering, what it means practically is that we replace the ad-hoc, non-statistical clustering methods used in the literature [4] with a Gaussian mixture extimation algorithm. Furthermore, statistics can also help us with additional questions that the original OCR approaches left unanswered, most importantly: how many clusters should there be?

6.4 Experiments

To study the feasibility of this approach to OCR, a simple prototype system based on the ideas described in this paper was implemented and applied to 1500 images of digits in the `cmr6` font from the Bell Labs database of severely degraded character images found on the University of Washington Database I. The input data was divided into 1000 training samples and 500 test samples. The images were centered, convolved with a Gaussian of $\sigma = 1$ and subsampled to a size of 10×10. The resulting image was treated as a raw feature vector and 7 principal components were extracted. These PCA feature vectors were then used as input to a sample-distribution based classifier, as well as a mixture discriminant analysis-based classifier (MDA; [9]).

The clustering OCR system performs its unsupervised clustering using the method described by Fraley and Raftery[8] and implemented by the `mclust` package for the `R` statistical system. Clusters in this approach are represented as Gaussian distributions. The method first performs hierarchical clustering and follows it by Expectation-Maximization (EM) steps to optmize the cluster shapes. In these experiments, the cluster shapes considered by the algorithm were "spherical" (all clusters have spherical covariance matrices), "uniform" (all clusters have the same covariance matrix), and "unconstrained".

In this way, the clustering OCR represents the unlabelled sample distribution as a mixture of 15 Gaussians. By assumption of the method, each Gaussian corresponds to a single digit label. When the assignment of labels to clusters is correct (either based on cryptanalysis or based on a non-specific classifier), the error rate of the sample distribution based recognizer on test data is 0.7% (N=500) in these experiments.

To compare the performance of the clustering OCR with a traditional approach to character recognition, a Mixture Discriminant Analysis (MDA) model[9] was trained. An MDA model represents likelihood functions as mixtures of Gaussians and uses Bayes rule to perform classification. The Gaussian mixtures are estimated using the Expectation Maximization (EM) algorithm. In the experience of the author, as well as based on results reported in the literature[9], MDA performance is roughly comparable to the performance of other, commonly used classifiers like neural networks and radial basis function methods. The `R` implementation of MDA (available from the `R` web site) was used for the experi-

ment. When the MDA classifier was trained on the training set (N=1000), its error rate on the test set was 0.6% (N=500)

6.5 Discussion

Ideas of clustering and style in OCR are not new and have been explored by a number of authors explicitly or implicitly. What this work contributes is a re-examination of clustering OCR methods from the point of Bayesian statistics, Gaussian mixtures, and mixture density estimation of the sample distribution. This helps both understand why and how clustering OCR methods work, and helps us improve them. The long term promise of this work is to arrive at classification methods that are considerably more robust to statistical differences between training and test data than traditional pattern recognition methods. The initial experiments presented above suggest that such an approach is feasible; more sophisticated implementations are needed to demonstrate that it delivers superior performance in real-wold situations. For OCR systems in particular, this translates into much more robust recognition when novel fonts or document degradation conditions are encountered.

7 Classification by Probabilistic Clustering[3]

7.1 Introduction

Current classifiers for the recognition of handwriting, printed characters, phonemes, and similar signals can achieve very high performance (often exceeding that of humans) when given sufficiently large and representative training sets. Techniques have also been used to synthesize additional training examples from a given training set to further increase the effective training set (and ability to generalize) for the classifier. A key limitation of such approaches is still that they can be sensitive to novel data whose distribution is significantly outside the training set.

In the work described above, we have seen how modeling the sample distribution using Gaussian mixtures can achieve comparable font independent performance to existing classification methods. That work was also motivated by the application of clustering OCR methods to OCR applied in the compressed domain for document images compressed using token-based methods. This work describes a technique that builds on those ideas but uses a novel, non-parametric probabilistic clustering technique. The approach is based on modeling, using a multilayer perceptron (MLP), the probability that two given images represent the same character. These probabilities are then integrated into an overall interpretation of a document using the maximum likelihood assignment of character identities to the individual images in the maximum entropy distribution compatible with the pairwise probability estimates derived from the MLP. Some experimental results are presented that demonstrate superior performance on a font-independent recognition task compared to traditional pattern recognition problems.

[3]This section is based on, and contains excerpts from, a paper to be presented at ICASSP '2001[3].

7.2 Pairwise Probabilities

In recognition by probabilistic clustering, rather than estimating $P(c, f|v)$, we estimate the pairwise probabilities $P(c = c', f = f'|v, v')$, i.e., the probabilities that two feature vectors represent the same character. The motivation for this approach is that we can imagine that determining whether two character images are similar or different may be considerably easier to perform in a font-independent manner than determining whether a given character image actually represents a particular character. For example, empirically, a simple but already fairly good statistic for determining the identity of two bilevel characters is to look at the minimum of the total area of their symmetric difference under arbitrary translations, normalized by the area of the larger of the two characters. This statistic can be computed completely independently of the font and distinguishes characters in a wide variety of fonts well.

If characters were perfectly distinguishable from their feature vectors, so that this probability only assumes values of 0 or 1, this would allow us to divide the set of feature vectors corresponding to characters on a page into equivalence classes. Each such equivalence class would then correspond to a single character class. Of course, we would still have to determine the identity of this equivalence class using some other means.

If $P(c = c', f = f'|v, v')$ can assume values other than zero or one, then the interpretation is more complex. An optimal interpretation of the whole document would be based on the joint conditional probability $\prod_i P(c_i, f_i|v_i)$ for all characters in the document. The conditional probability $P(c_i = c_j, f_i = f_j|v_i, v_j)$ is a marginal probability of this distribution, and we can use this to compute the maximum entropy joint conditional probability. In practice, however, since we are only using estimates of the pairwise probabilities, there is almost always no probability distribution that is consistent with the estimates for the pairwise probabilities. In order to address this problem, we need to formulate the problem of assigning classes to the different feature vectors as an optimization problem. A

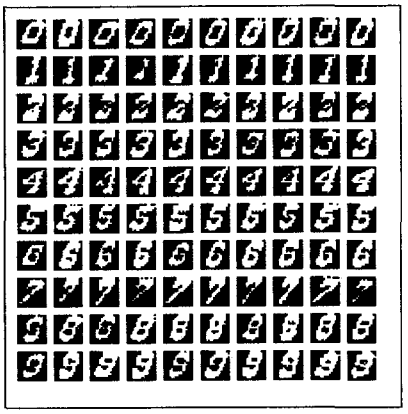


Figure 5: Examples of degraded characters used in the experiments.

simple method that suggests itself is to find an assignment of class labels to feature vectors that maximizes the product of all the pairwise probability estimates (we will not attempt a formal justification in this work). We can solve this optimization problem simply by simulated annealing, which appears to converge quickly in our experiments.

7.3 The Method

Recognition by probabilistic clustering therefore can be described as follows:

- estimate $P(c = c'\hat{f} = f'|v, v')$ based on a set of training examples $\{(c = c'\hat{f} = f', v, v'), ...\}$ (for many different fonts)
- when faced with the problem of recognizing a new collection of feature vectors v_i, compute $\hat{P}(c_i = c_j\hat{f}_i = f_j|v_i, v_j)$ for each pair of feature vectors v_i, v_j
- assign cluster labels χ_i to the feature vectors v_i such as to maximize $\prod \hat{P}(\chi_i = \chi_j|v_i, v_j)$, for example using simulated annealing
- determine the correspondence between the cluster labels χ_i and the actual classes (and fonts, if desired)

7.4 Experiments

The dataset used in these experiments consisted of 71700 images of digits derived from 717 TrueType fonts from a commercial collection of type fonts (examples are shown in Figure 5). This dataset was split into 64600 training images representing 10 degraded samples of each digit from each of 646 fonts, and 7100 test images representing 10 degraded samples of each digit from each of 71 fonts. The character images were rendered using the Freetype engine, which performed antialiased rendering of greyscale images of characters under affine transformation. Character images were degraded using the Baird character degradation model[14] with its standard settings, a widely used and studied model for modeling degradation of printed text under a variety of common document imaging conditions. Characters were rescaled to fit into a 16×16 square, giving rise to a 256 dimensional feature vector.

In a first step, to characterize the dataset, this feature vector was used as input to a multilayer perceptron The MLP had 256 input units, 15 hidden units, and 10 output units. The test set error of the MLP was 9.46%. This may appear like a high error rate for OCR, but it is important to keep in mind that this test is different from most traditional OCR tests, since it (deliberately) involves a very wide diversity of fonts that are severely degraded and represented at low resolution.

For classification based on probabilistic clustering, the probability $P(c = c'|v, v')$ was estimated as follows. For each font in the dataset, the 4950 pairs of non-identical character images representing the same digits, as well as a random set of 4950 pairs of non-identical character images representing different digits were selected. Two 16×16 images were computed for each pair: the absolute difference between the two images, and the sum of the two images (at an offset that minimized the difference). These two images were used as a 512 dimensional feature vector and input into a MLP. The MLP had 512 input units, 15 hidden units, and 1 output unit. Training proceeded by training the output unit to "1" for pairs of character images representing the same digit and to "0" for pairs of character images representing different digits. It is well known that this training procedure will asymptotically converge to an estimate of the conditional probability that $c = c'$ given the input feature vector.

For testing, the input to the system consisted of 100 digit images from each font. For each pair of digit images, the pairwise probabilities $P(c = c'|v, v')$ was computed. For the simulated annealing step, a classification derived from the "traditional" classifier $\hat{P}(c|v)$ (modeled by the MLP described above) was used as the starting configuration. When this procedure was carried out for the test set, the performance of the system improved from 9.46% for the traditional MLP-based classifier to 7.66% for the clustering classifier.

7.5 Discussion

This work describes an approach to classification based on the estimation of a class-independent probabilistic model of the similarity of two feature vectors, followed by a probabilistic clustering method. Future work will include better cluster assignment

methods, a more formal analysis and better parametric models of character similarity, and automatic ways of assessing cluster validity. Perhaps most importantly, the assignment of labels to clusters by initializing the simulated annealing process is suboptimal because its performance is limited intrinsically by the quality of the traditional classifier (significantly incorrect initial assignments will result in permuted label assignments in the output). Several better methods offer themselves: use of the traditional classifier as a prior, greedy assignment of cluster labels based on predominant classifications of the members of each cluster, and the use of statistical language models.

While it will be desirable to design experiments more specifically to explore and demonstrate the ability of the approach to handle variations in font, degradation, and robustness to samples outside the training set, the results presented in this work, generalization of classification to a varied and difficult test set of novel degraded fonts, already suggest classification by clustering holds the promise of being a general approach to addressing problems that are hard for traditional classifiers: coping with stylistic variations and generalization to samples outside the training set.

References

[1] T. M. Breuel. Modeling the Sample Distribution for Clustering OCR. In *SPIE Conference on Document Recognition and Retrieval VIII*, 2001.

[2] Thomas M. Breuel. Layout analysis by exploring the space of segmentation parameters. In *Proceedings of the 4th IAPR Workshop on Document Analysis Systems (DAS 2000)*, December 2000.

[3] Thomas M. Breuel. Classification by probabilistic clustering. In *Proceedings of the 2001 International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2001)*, Salt Lake City, Utah, May 2001. IEEE. To appear.

[4] R. Casey, S. K. Chai, and K. Y. Wong. Unsupervised construction of decision networks for pattern classification. In *Proc. ICPR-7*, July 1984.

[5] A. Dengel. About the logical partitioning of document images. In *3rd Symposium on Document Analysis and Information Retrieval, Las Vegas*, pages 209–218, 1994.

[6] D. Doermann. The indexing and retrieval of document images: A survey. Technical Report CS-TR-3876, University of Maryland CS Department, 1998.

[7] D. Doermann, C. Shin, A. Rosenfeld, H. Kauniskangas, J. Sauvola, and M. Pietikainen. The development of a general framework for intelligent document image retrieval. In *Document Analysis Systems*, pages 605–632, 1996.

[8] C. Fraley and A. E. Raftery. How many clusters? which clustering method? answers via model-based cluster analysis. Technical Report No. 329, Dept. of Statistics, U. of Washington, Februrary 1998.

[9] T Hastie and R Tibshirani. Discriminant analysis by gaussian mixtures. Technical report, AT&T Bell Laboratories, 1994.

[10] D. Ittner and H. Baird. Language-free layout analysis, 1993.

[11] Frederick Jelinek. *Statistical Methods for Speech Recognition.* MIT Press, Cambridge, Massachusetts, 1997.

[12] Rolf Johannesson and Kamil Sh. Zigangirov. *Fundamentals of Convolutional Coding.* IEEE Press, 1999.

[13] Anthony C. Kam and Gary E. Kopec. Document image decoding by heuristic search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):945–950, September 1996.

[14] T Kanungo, H Baird, and R Haralick. Estimation and validation of document degradation models. In *Proc. 4th Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, NV, April 1995.

[15] Gary E. Kopec and Philip A. Chou. Document image decoding using Markov source models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):602–617, June 1994.

[16] Jishen Liang. *Document Structure Analysis and Performance Evaluation.* PhD thesis, University of Washington, 1999.

[17] Thomas P. Minka, Dan S. Bloomberg, and Kris Popat. Document image decoding using iterated complete path heuristic. In *Proceedings of IS&T/SPIE Electronic Imaging 2001: Document Recognition and Retrieval VIII*, January 2001.

[18] L. O'Gorman. The document spectrum for page layout analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1162–1173, 1993.

[19] Kris Popat. Decoding of text lines in grayscale document images. In *Proceedings of the 2001 International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2001)*, Salt Lake City, Utah, May 2001. IEEE. To appear.

[20] Kris Popat, Dan Bloomberg, and Dan Greene. Adding linguistic constraints to document image decoding. In *Proceedings of the 4th international workshop on document analysis systems.* International Association of Pattern Recogni-

tion, December 2000.

[21] Kris Popat, Dan Greene, Justin Romberg, and Dan S. Bloomberg. Adding linguistic constraints to document image decoding: Comparing the iterated complete path and stack algorithms. In *Proceedings of IS&T/SPIE Electronic Imaging 2001: Document Recognition and Retrieval VIII*, January 2001.

[22] P Sarkar. *Style Consistency in Pattern Fields.* PhD thesis, Rensselaer Polytechnic Institute, May 2000.

[23] A. L. Spitz. Style-directed document recognition. In *Workshop on Document Layout Interpretation and its Applications (DLIA)*, 1999.

Additional Submissions

Integrating OCR and Machine Translation for Non-Traditional Languages

Chris Schlesiger **Luis Hernandez**
Melissa Holland
U.S. Army Research Lab

Abstract

An integration of OCR with machine translation (MT) will be demonstrated for "non-traditional" languages, using ARL's FALCon prototype. While FALCon originally focused on languages for which commercial MT and OCR components were available, recent integration has dealt with languages that lack commercial development and have little available OCR or MT software – for example, Behasa Indonesian and Tetum. Strategies for finding and integrating components for these languages will be discussed, such as use of word-substitution algorithms for MT and adoption of OCR for related languages.

1 Commercial Components in FALCon

The Forward Area Language Converter (FALCon) is an end-to-end system developed for the U.S. Army that integrates optical character recognition (OCR) with machine translation (MT). The purpose is to let a user with no foreign language training convert a foreign language document into an approximate English translation [2]. Thus, troops in the field can screen captured or open-source documents and separate those deemed relevant to send to a trained linguist for full translation and analysis. In this way, trained linguists, a scarce resource in the field, are saved from dealing with irrelevant documents.

FALCon began with a set of core languages, including western European and Russian. The integration of these languages drew on conventional components: commercial OCR packages linked with commercial machine translation. The MT software produces translations on the basis of (a) extensive bilingual lexicons and (b) parsing of source-language sentences to extract underlying structure. The parsed structures and the bilingual word or phrase equivalents are then used in generating target-language (English) sentences. This approach to MT, based on syntactic analysis of sentences, is often called the "transfer method" [5].

Integration architecture is broadly depicted in Figure 1 for FALCon with traditional commercial languages. The Windows-based interface, developed at ARL, is government-off-the-shelf (GOTS), while the other components are commercial-off-the-shelf (COTS).

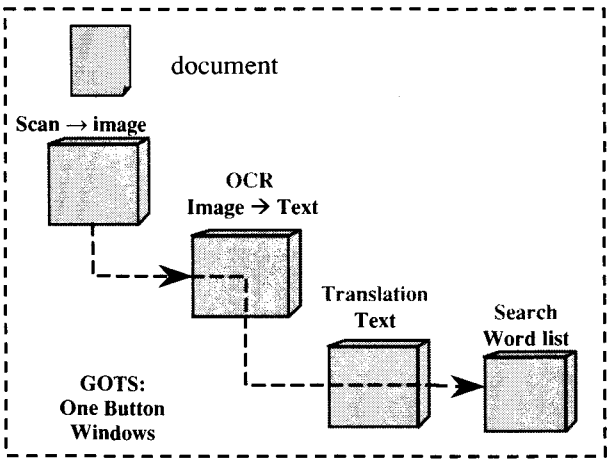


Figure 1: Integration scheme for FALCon with commercial languages.

2 The Need for Non-Commercial Languages

Recent requests from military commands in the Pacific call for integration of languages that have immature markets and, consequently, little commercial development – for example, Behasa Indonesian, Tagalog, Cambodian, Laotian, and Tetum. These might be called "non-traditional" languages because they tend to be absent in commercial language products and infrequent in language learning curricula (where they are known as "less commonly taught languages").

3 Integration Solutions for Non-Commercial Languages

Without commercial software to draw on, we have used other means to select components for integration into FALCon. For MT, we have adapted government-off-the-shelf (GOTS) software developed by the intelligence community. This software dispenses with parsing and instead relies on bilingual lexicons to perform word-by-word replacement. While this approach gives a less accurate and less fluent translation than that provided by parser-based software, it is faster and less costly to develop. This approach to MT is often called the "direct method" [5].

For OCR of languages not available in commercial packages, we have borrowed OCR from languages that

share fonts with the languages we need. Although not ideal, this solution has been investigated for minority (very low-density) languages such as Haitian Creole. For example, tests of both English and French OCR showed that French OCR yielded better recognition of Haitian-Creole text [4]. This solution was then used with adaptive spelling corrections to input native documents to experimental, example-based MT for Haitian Creole [3].

3.1 Word-replacement MT

We obtained GOTS word-replacement software for Indonesian, Tetum (the language of East Timor), and Tok Pisin (New Guinea) as well as experimental, partially GOTS word-replacement software for Thai. This software works by substituting English words for source-language words, drawing from a bilingual lexicon. Since sentence context and word order are ignored, this method words best with languages such as the three just named, whose word order is close to English and which have simple morphology (i.e., the words require little inflection). Users of word-replacement MT rely on the presence of keywords in the English output, rather than making sense of sentences. Thus, it is better suited for screening and broadly categorizing documents than for more demanding uses of translation [1, 6].

3.2 OCR for non-OCR Languages

We found a commercial OCR package for Indonesian, which integrated easily with word-replacement MT. Here is a sample of word-replacement MT working on a *ground truth (non-OCR)* text:

> introduce product new in the form of five a process special for car fast patrol craft which is slight, light and PORTABLE. with TEKMOLOGI process 0,18 MIKRON, enable process work to tension 1,1 VOLT and merging technology intelligence SPEEDSTEP
>
> and technology OUICKSTART, become process this only MENGKONSUMSI capacity less than 1 WATT without look at type application which is be used.

The MT produced identical English when applied to *OCR input* of the same Indonesian text. Capitalized words are those not found in the lexicon. Review of several samples of English produced by the MT on OCR text suggested this integration would be sufficient for screening (e.g., the above sample was categorized as about a new technology – a category confirmed by native speakers). This integration will be tested for the screening task with users located in the Pacific.

No OCR package was available for Tetum and Tok Pisin. Since these languages, like Indonesian, use roman font, we experimented with OCR from languages that share written origins with Tetum and Tok Pisin, including English, Dutch, and Indonesian. We will demonstrate the results of OCR-MT integration for all three Pacific languages. We will also discuss how we used a short-cut method of comparing (a) English output from OCR with (b) English output from groundtruth, to observe implications for a screening task and thereby select an OCR.

References

[1] Church, K., Hovy, E.: Good Applications for Crummy Machine Translation. In: Neal, J., Walter, S. (eds.) *Proceedings of the Natural Language Processing Systems Evaluation Workshop*. Calspan-UB Research Center (1991.

[2] J. DeHart, C. Schlesiger, M. Holland. Issues in OCR for Army machine translation. In *Proceedings of SDIUT99*, Laurel, MD (1999) 213-216.

[3] C. Hogan. OCR for minority languages. In *Proceedings of SDIUT99*, Laurel, MD (1999) 235-244.

[4] C. Schlesiger, L Decrozant. *Comparison of two French OCR packages – How they handle Haitian-Creole text.* Internal report. U.S. Army Research Lab, Adelphi MD (1998).

[5] H.L. Sommers. The current state of machine translation. In *Proceedings of MT-Summit*, San Diego, CA (1997) 115–123.

[6] Taylor, K., White, J.: Predicting What MT is Good for: User Judgments and Task Performance. In: Farwell, D. et al. (eds.), *Machine Translation and the Information Soup: Proceedings of the Association for Machine Translation in the Americas Annual Meeting.* Springer-Verlag (1998) 364-373.

Creating a Digital Library from Newspaper Archives

S.L. Mantzaris, B. Gatos and N. Gouraros
Department of Digital Technologies, Lambrakis Press Archives,
8, Heyden Str., 104 34 Athens, Greece, smantzaris@lpa.gr

Abstract

Newspapers are considered to be the first draft of history, while at the same time, make a significant part of a country's cultural heritage. By converting newspaper archives to digital resources we achieve digital preservation in terms of preventing paper deterioration as well as providing full utilization of the archives by all interested parties. In this demonstration, we present a series of applications pertaining to the retro-conversion of newspapers, i.e. the conversion of newspaper pages into digital resources, as well as to the transformation of the printed material to an accessible digital archive. These applications constitute an integrated system that provides solutions to problems related to digitization, verification and quality control of newspaper images, manual and automatic article clipping, and, finally, information retrieval in intranet and internet environment.

1 Introduction

Lambrakis Press Archives (LPA – www.lpa.gr) is the sector of Lambrakis Press S.A. dealing with the conservation and digital preservation of printed publications as well as the design and the development of an archival digital library consisting of digitized printed material. LPA is divided into two departments: the department of Digital Technologies and the department of Cataloguing. The department of Digital Technologies is responsible for the development of a workflow for the electronic preservation of newspaper pages and the construction of a digital library. The department of Cataloguing deals with the classification and indexing of newspapers and magazines. LPA focuses on creating electronic versions of printed material, as well as on developing prototypes for the creation of an integrated digital library supported by full retrieval capabilities.

Our main task is to create a Newspaper digital archive that can provide full access to all articles of the newspaper issues. Firstly, the newspaper material is gathered and prepared for digitization. Then, a Newspaper Issues Catalogue is created by examining all archival material. After checking for missing, incomplete or deformed issues, we proceed with digitization, image preprocessing and a visual check of all digitized material. The preprocessing task mainly involves image filtering for the improvement of image quality, as well as skew correction in order to restore horizontal image status. Then, we proceed to article clipping and indexing by marking all articles and inserting all necessary article metadata. When necessary, an archival update procedure is applied. The final task of our workflow concerns the construction of a web-search module that will provide access to all articles in intranet and Internet environment. In this demonstration, we will focus on article clipping which is implemented manually or automatically, as well as on the web-search module construction.

2 Manual Article Clipping

At the manual clipping module, all articles are marked and indexed electronically (see Figure 1). The main steps involved are: (a) the user marks the articles of a page using isothetic polygons (polygons having only horizontal and vertical edges), (b) for each of these articles all the necessary cataloging information is given and stored in a database, and (c) article continuities are defined as links between two or more article parts that are located on different pages.

The manual clipping module is at a production phase (every day approximately 1500 articles are clipped and indexed, in 84 human hours). After article marking, the user inserts all the metadata that describes the article (title, over-title, subtitle, running headlines, category, authors etc.).

3 Automatic Article Clipping

A number of algorithms for page segmentation have been already proposed (e.g. [1], [2]) in the literature. However, such algorithms are not directly applicable to newspaper scans. The most significant problems identified include the complex layout of newspaper pages, particularly the oldest ones, where text columns are located very close to each other in a haphazard way, as well as the poor scanning results derived from paper material that was originally of low print quality or has deteriorated through time. Another important problem for news page segmentation is the layout design that seems to have changed throughout the newspaper's life. We have implemented a new technique for automatic

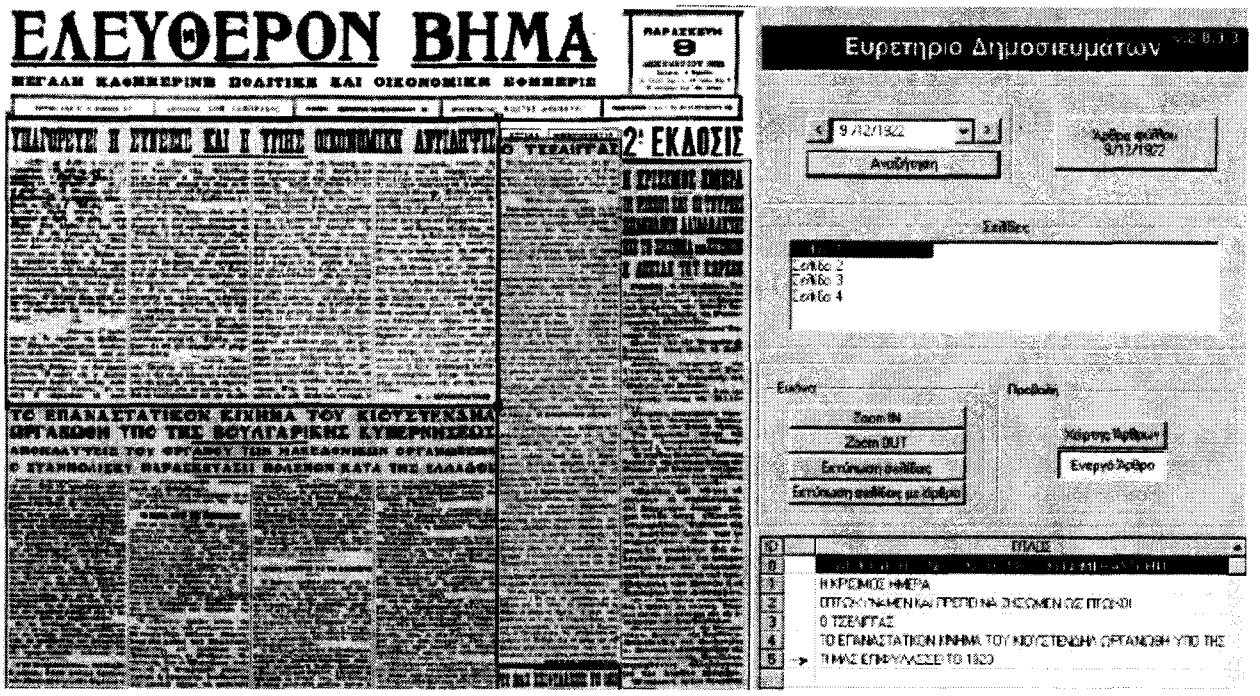


Figure 1: Clipping/Cataloguing module.

Figure 2: Web search module.

newspaper page segmentation based on gradual extraction of newspaper image components in the following order: Lines, images and drawings, background lines, special symbols, text and title blocks ([3], [4], [5]). In order to restrict all different components inside simple geometric shapes we use isothetic polygons with minimum number of vertices ([6]).

Individual articles are traced and automatically recognized using a suitable optical character recognition module. For article tracking, we follow a novel rule based approach, which exploits the segment relationships that exist in the page layout format of newspaper pages ([4],[5]). Currently, we are experimenting with the integration of a novel technique is used for the linking of the textual parts of an article that can be found on different pages of a newspaper issue ([7]). The automatic clipping module is at a testing - evaluation phase. We expect to conclude soon for a supervised automatic clipping module that will increase production significantly.

4 Web Search Module

An article can be located using a combination of metadata information (created during the clipping and cataloguing of articles), and textual information (full text or titles). Our corporate users investigation has shown that the search mechanism should satisfy the following requirements: support users with different search mentalities and computer experience; provide simple ways in order to help users express their information needs and view the results.

We believe that a broader spectrum of users would have similar requirements. Other issues that we have taken into account in order to design our search engine are: (a) The articles have a great thematic variety. (b) The articles cover a broad time period of over 100 years. This has implications on the language, for example the stems of some words have morphological differences. (c) Easily updating available information. (d) Portability and use of standard tools.

Our metadata are stored in an RDBMS while a low resolution version of images is stored on file systems. Every requested image is transformed to pdf and watermarked on the fly. For each requested article, a pdf image having the article outlined is presented to the user. Figure 2 presents our newspaper web search module. A user can enter information in any of the fields (i.e. date, page number, author, text. etc). To support more elaborate user queries, an advanced search dialog box is currently under development.

References

[1] P. Chauvet, J. Lopez-Krahe, E. Taflin and H. Maitre, "System for an Intelligent Office Document Analysis, recognition and description", *Signal Processing*, Vol. 32, pp. 161-190, 1993.

[2] K. Fan, C. Liu and Y. Wang, "Segmentation and Classification of Mixed Text/Graphics/Image Documents", *Pattern Recognition Letters*, Vol. 15, pp. 1201-1209, 1994.

[3] B. Gatos, N. Gouraros, S. Mantzaris, S. Perantonis, A. Tsigris, P. Tzavelis and N. Vassilas, "A new Method for Segmenting Newspaper Articles" , Proc. of the *Second European Conference On Research and Advanced Technology for Digital Libraries (ECDL'98)*, pp. 695-696, Heraklion, Crete, Greece, September 1998.

[4] B. Gatos, S. L. Mantzaris, K. V. Chandrinos, A. Tsigris and S. J. Perantonis, "Integrated Algorithms for Newspaper Page Decomposition and Article Tracking", Proc. of the *Fifth International Conference on Document Analysis and Recognition (ICDAR'99)*, pp. 559-562, Bangalore, India, September 1999.

[5] B. Gatos, S. L. Mantzaris, S. J. Perantonis and A. Tsigris, "Automatic page analysis for the creation of a digital library from newspaper archives", *International Journal on Digital Libraries (IJODL)*, Vol. 3(1), pp. 77-84, 2000.

[6] B. Gatos and S. L. Mantzaris, "A novel recursive algorithm for area location using isothetic polygons", Proc. of the *15th International Conference on Pattern Recognition (ICPR2000)*, pp. 496-499, Barcelona, Spain, September 2000.

[7] S.L. Mantzaris, B. Gatos, N. Gouraros and S.J. Perantonis, "Linking Article Parts for the Creation of a Newspaper Digital Library", Proc. of the *Content-Based Multimedia Information Access International Conference (RIAO2000)*, pp. 997-1005, Paris, France, April 2000.

Standard Metadata for Multimedia Content

Wo Chang*+
wchang@nist.gov

Image Visual Group
Information Access Division
Information Technology Laboratory (ITL)
National Institute of Standards and Technology
Gaithersburg, Maryland 20899, U.S.A.

*Active member of ISO/MPEG Standards Committee and W3C SYMM WG
+ Co-chair of MPEG-7 Application and Promotion to Industry (MAPI)
Co-chair of MPEG-7 XM Software Development
Chair of MPEG-4 Reference Software Architecture

Abstract

The XML metadata technology of describing online documents has emerged as a dominant mode of making information available both for human and machine consumptions. To realize this promise, many online Web applications are pushing this concept to its full potential. However, a good metadata model does require a good standardization effort so that the metadata content and its structure can reach its maximum usage between various applications. An effective document content understanding technology should also use standard metadata structures especially when dealing with multimedia contents. A new metadata technology called MPEG-7 content description has taken off from the ISO MPEG standards body with the charter of defining standard metadata to describe audiovisual content. This abstract session will give an overview of MPEG-7 technology and what impact it can bring forth to the next generation of multimedia indexing and retrieval applications.

1. Background

As technology moves into a new millennium, the most promising technologies include: wireless communication – G3-IMT2000 [1], Bluetooth [2], HomeRF [3], WAP [4], portable devices such as PalmPilot [5] and Handspring [6], streaming multimedia – MPEG-4 [7], QuickTime [8] RealNetwork Format [9], Window Media Format [10], and etc. All of these technologies have one ultimate goal: to provide and deliver Internet multimedia content. History tells us, unless the media content and the delivery protocol are standardized, any proprietary solutions will not last. It was not until the HTML markup language and HTTP transport protocol of the Web technology was defined that millions and millions of existing documents came online and in searchable format. This is exactly the same situation applies to multimedia content. Many great captions, images, audio, and video sequences have been created by either professional or amateur home end-users, however, these contents cannot be easily retrieved and accessed because there is no standard way to represent and to describe the content. Until recently, the international ISO/IEC MPEG [11] standards body was in the process of defining and developing the next generation content description technology named MPEG-7 [12]. It would have been impossible to index and retrieve large and diversified multimedia contents without this standard. This paper is to give an overview of MPEG-7 and show what possible impact this technology can bring to develop the next generation of multimedia indexing and retrieval applications.

2. MPEG-7 Content Description Technology

The MPEG-7 standard also known as "Multimedia Content Description Interface" aims at providing standardized core technologies allowing description of audiovisual data content in multimedia environments. This technology is being designed by a range of experts including broadcasters, manufacturers, content creators,

publishers, intellectual property rights managers, telecommunication service providers, academia, government, etc., to:

- Define a rich set of standardized tools to describe audiovisual content,
- Create good storage solutions, high-performance content identification, fast, accurate, personalized filtering, searching, and retrieval data structure/formats,
- Enable both human users and automatic systems to process the encoded audiovisual content descriptions.

Basically, the MPEG-7 standard community wants to standardize the metadata structure and its attributes/values definition to describe the audiovisual content, as shown in Figure 1.

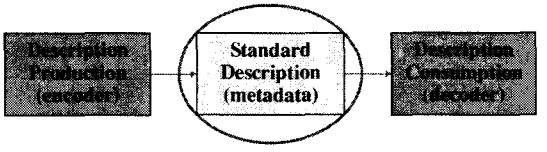


Figure 1: MPEG-7 Standard Content Description

The most challenging task of MPEG-7 is the broad spectrum of requirements and targeted multimedia applications, and the large number of important audiovisual features from various emerging application domain technology. In order to satisfy these broad requirements, MPEG-7 will need to standardize common content description components as shown in Figure 2.

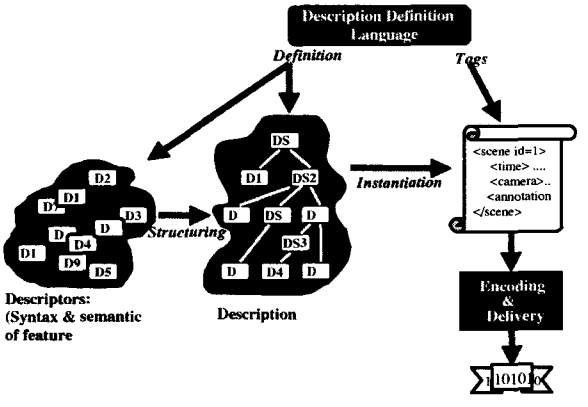


Figure 2: MPEG-7 Metadata Components

- **Datatype** - A description element that is not specific to the multimedia domain that corresponds to a reusable basic type or structure employed by multiple Descriptors and Description Schemes.
- **Descriptors (D)** – define the syntax and the semantics of each feature representation. A Feature is a distinctive characteristic of the data, which signifies something to somebody.
- **Description Schemes (DS)** – specify the structure and semantics of the relationships between their components, which may be both Ds and DSs.
- **Description Definition Language (DDL)** – a language allowing the creation of new DSs, and possibly Ds, able to extend and modify any existing DSs.
- **Systems tools** – support multiplexing of descriptions or description and data, synchronization issues, transmission mechanisms, file format, binary encoding, etc.

From the above datatypes, DSs, Ds, and DDL, MPEG-7 can be grouped into five major categories as shown in Figure-3.

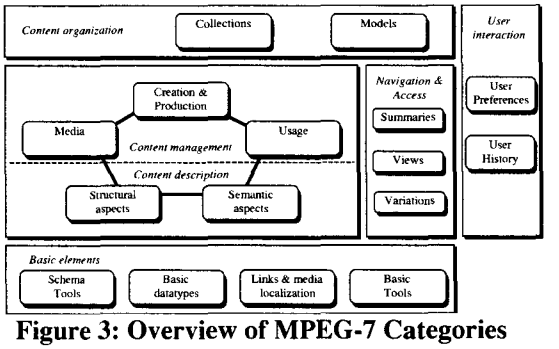


Figure 3: Overview of MPEG-7 Categories

Basic Elements – deals with root, top-level elements, packages for Schema Tools; time, duration, media locations for Link & Media Localization; and textual annotation (free text or structured annotation, etc.), agent, place, graph, etc. for Basic DSs.

Content Management & Description – deals with format, coding, instances, identification, transcoding, hint, etc. for Media; title, creator, creation, location & date, purpose, classification, genre, etc. for Creation & Production; rights

holder, access right, usage record, financial aspects for Content Usage; spatial & temporal structure and elementary semantic information for Structural; and events, objects, abstract concepts, and their relation for the Conceptual.

Navigation & Access – deals with discovery, browsing, navigation, visualization for Summary; and adaptation to terminal, network, or user preferences for Variation.

Content Organization – deals with description and organization of collection of documents for Collection & Classification; and statistical functions and structures to describe sample of AV content and classes of descriptors as in probability model and definition of cluster, classes and models to associate a semantic label as to a set of data.

User Interaction – deals with user identification and preferences, filtering, searching for User Preferences; and usage history for User Preferences.

All the MPEG-7 components are based on W3C 's eXtensible Markup Language (XML) [13] technologies which include: XML Schema [14], XPointer [15], XPath [16], Document Object Model (DOM) [17], Simple API for XML (SAX) [18], etc.. XML allows document structures to be defined, levels of subdivisions to be created, and content data to be stored and retrieved hierarchically. XML is data-centric, whereas HTML is more display-centric. XML focuses on how data is structured rather than on how data is displayed. XML provides mechanisms to define descriptors and description schemes that, in turn, are used to describe audiovisual content descriptions.

XML Schema specifies the structure of instance documents and their datatype for each element and attribute. XML Schema are far more advanced than its predecessor Document Type Definition (DTD). Examples such as the datatypes, XML Schema provides 30+ more datatypes than DTD; can extend or restrict a type; support multiple elements, etc. whereas XPointer allows document referencing within a document versus XPath which deals with external document referencing. The goal is be able to create application domain datatypes, their elements and attributes, and to be able to reference any part of the document either internally or externally. DOM provides the means for manipulating XML documents and SAX provides a standardized interface to DOM.

3. Standard Interoperable Metadata

The main objective for ISO MPEG to develop all MPEG related (MPEG-1/-2/-4/-7/-21) standards is to ensure that all products are interoperable, either at the hardware chip level or software application level. This is particularly true for MPEG-7 since it covers a wide spectrum of multimedia applications. To do this, MPEG-7 experts are hard at work on the topics of MPEG-7 conformance and interoperability. MPEG-7 may use the same approach as in MPEG-4 to define profiles and levels as the checkpoints when dealing with different capabilities of different terminals and systems. This is because the platform configurations may vary in several areas such as: memory size, number of DSs and Ds support, processing power, dynamic update on schemas, etc..

How to implement interoperability checkpoints within MPEG-7 is a tough question for MPEG-7 experts to answer, since the checkpoint can be anywhere from level, profile, or at the system levels. However, since MPEG-7 is standardizing the content descriptions on DSs, Ds, and DDL, it is possible to unambiguously interchange MPEG-7 descriptions in the textual and binary representation formats. These formats are well-defined standard metadata, such that the reconstructed DS/D structure is the same for all conformant description consuming applications and systems.

4. Conclusion

The concept of using international ISO MPEG-7 content description technology is presented. It is vital to use standard metadata to describe multimedia content; otherwise, the media content information will be incompatible and not interoperable between different applications and systems. Furthermore, unstructured matedata will hinder the users' efficient usage of multimedia content and will be a stumbling block for audiovisual related communities such as content providers, publishers, researchers, and most important the end-users.

The goal of this abstract session is to share ISO MPEG-7 development and to promote more standard metadata activities in the areas of multimedia contents and applications.

5. Disclaimers

NIST does not endorse or recommend any of the mentioned standards, products, companies, or sites in this paper and such mentions do not imply that the cited standards, products, companies, or sites are better or worse than similar standards, products, companies, or sites.

6. Acknowledgements

I would like to thank my colleagues at ITL for their help and encouragement, in particular, Joyce Myrick for her unceasing support and Charles Wilson, Manager of the Image Group at NIST for his support and encouragement.

7. Reference

[1] International Telecommunication Union (ITU) on G3 IMT2000 Technology http://www.itu.int

[2] The Official Bluetooth SIG Website http://www.bluetooth.com

[3] Home RadioFrequency Website http://www.homerf.org

[4] Wireless Application Protocol Forum http://www.wapforum.org

[5] PalmPilot Website http://www.palm.com

[6] Hadnspring Website http://www.handspring.com

[7] MPEG-4 Industry Forum Wetsite http://www.m4if.org

[8] Apple's QuickTime Website http://www.apple.com/quicktime

[9] RealNetwork Website http://www.real.com

[10] Microsoft Windows Media Website http://www.microsoft.com/windows/windowsmedia

[11] General MPEG Website http://www.cselt.it/mpeg

[12] MPEG-7 Website http://www.mpeg-7.com

[13] T. Bray, J. Paoli, C. Sperbrg-McQueen (editors). eXtensible Markup Language. http://www.w3.org/TR/REC-xml

[14] Henry S. Thompson, et al (editors). XML Schema. http://www.w3.org/XML/Group/Schemas.html

[15] Steve DeRose, et al (editors). XML XPointer http://www.w3.org/TR/WD-xptr

[16] James Clark and Steve DeRose (editors). XML XPath http://www.w3.org/TR/xpath

[17] Document Object Model (DOM) http://www.w3.org/TR/WD-DOM

[18] Simple API for XML (SAX, Version 2) http://www.megginson.com/SAX

A recognition method of the machine-printed monetary amounts based on the two-dimensional character segmentation

Masashi Koga **Ryuji Mine**
Hiroshi Sako **Hiromichi Fujisawa**
Central Research Laboratory, Hitachi, Ltd
1-280 Higashi-Koigakubo, Kokubunji-shi
Tokyo 185-8601, Japan

Abstract

A new method to recognize the monetary amounts printed by checkwriters is proposed. In conventional methods, character segmentation module segments the image only along the direction of the character line. Thus, it is difficult to extract character segments if there are many noises near the text line, or characters are fragmented into many small pieces. This new method segments the image both horizontally and vertically so that correct character segments are included in the set of the candidate segments. We used a neural network based classifier to recognize characters. A parsing module detects the optimal sequence of candidate segments using linguistic knowledge, and interprets the results of character classification as a monetary amount. In our method, a context-free grammar describes the linguistic constraints in the monetary amounts. We devised a new bottom-up parsing technique that can handle the two-dimensional input. We tested the validity of the new method using 1,314 images, and found that it improves the recognition accuracy significantly.

1 Introduction

The character segmentation is frequently problematic in the character recognition system. The difficulty comes from the dilemma that a system cannot extract sub-images each of which corresponds to a character (character segments) without recognizing characters. Multiple hypotheses-testing of segmentation approach is one of the method to solve this problem [1][2][3]. In this approach, the process consists of three steps; the pre-segmentation, the character classification, and the final decision. In the pre-segmentation step, the system extracts candidate segments without identifying the categories. Here, a candidate segment is a sub-image of the input image that is plausible as a character. If there are plural ways to split the input image and the system is not able to decide which is the best, it extracts candidate segments based on plural hypotheses of segmentation. Secondly, in the character classification step, the system infers to which character each candidate segment belongs and evaluates the credibility of the inference. Finally, the system detects the sequence of candidate segments optimal as a character string in the final decision step.

However, most of the conventional methods assume that the input is a correctly segmented line image. Therefore, the system segments the image only along the direction of the line. That is why it sometimes cannot segment the image correctly if there are noises in the image or the characters are fragmented. [4] presents a method in which the system segments the image both horizontally and vertically. It finds a plausible sequence of segments by minimizing the cost of the character classification, the word matching, and the spatial arrangement analysis. So far as we know, nobody tested the utilization of the grammar like [5] in this kind of two-dimensional segmentation method.

We propose a method to recognize monetary amounts on forms documents printed by checkwriters. The input of the system is a noisy binary image where characters are often broken into many connected components. The pre-segmentation process of the new method works two-dimensionally so that the set of the candidate segments includes correctly segmented sub-images of all the characters in the input. We used a context-free grammar to represent the linguistic constraints in the expression of the monetary amounts. A newly developed bottom-up parser interprets the character classification results mapped to a graph that represents the positional relationship among the candidate segments.

In this paper, we describe the features of the machine-printed monetary amounts in section 2. Section 3 describes the details of the new method. Section 4 presents the experimental results that show significant improvements of the recognition accuracy by the method. Finally, conclusions and discussions are given in section 5.

2 The Features of Monetary Amounts Printed by Checkwriters

The examples of the inputs of the proposed method are shown in Fig. 1. They are binary images of 240 dpi. The characters are printed by checkwriters. A checkwriter prints characters with sets of pins or edges. Therefore, sometimes the characters in the images are fragmented into many dots or short lines. There area many pepper noises in the images generated by thresholding the textured background. Sometimes the characters are not clear because the portion of ink is not enough or the printing pressure is uneven. Thus the precise line segmentation is often impossible. The character segmentation and the recognition are also difficult.

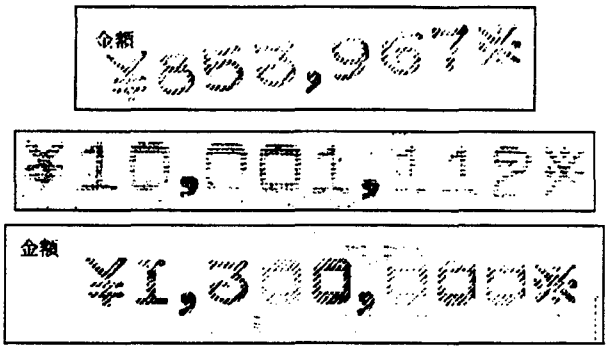


Figure 1: Examples of input images

On the other hand, there are linguistic constraints in the expression of the monetary amounts. For example, they follow the rules as below in Japanese forms:

- A mark "¥" is printed at the head.
- There is(are) special character(s) such as "*" or ",*" at the end.
- A comma is printed every 3 digits.

It is easy to represents these rules by a context-free grammar as explained in the following section.

3 Proposed Method

3.1 System Overview

The essential components in our method are the two-dimensional pre-segmentation, the character classification, and the bottom-up parsing. Moreover, we used a pre-processing module that we devised to improve the recognition accuracy in this application. Figure 2 shows the overview of the system based on the new method.

The input is a binary image of a field in a form in which a monetary amount is supposed to be printed. The pre-processing module cleans the image, and the two-dimensional pre-segmentation module extracts candidate segments. Character classifier infers the categories of candidate segments and calculates their credibility. Here the set of categories consists of digits, comma, "¥", and "*". Finally, the bottom-up parser interprets the results of the character classification as an amount.

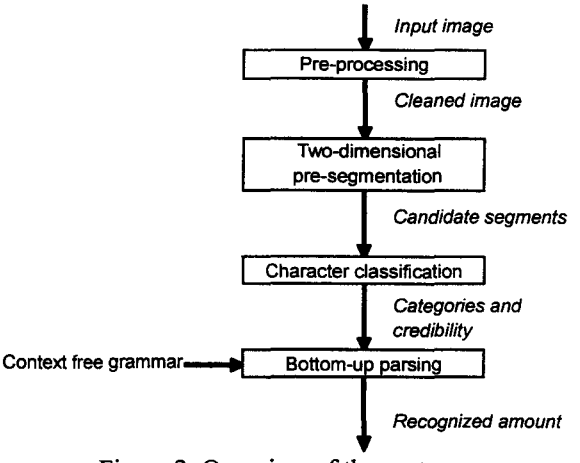


Figure 2: Overview of the system

3.2 Pre-processing

In pre-processing stage, the system cleans the input image by a morphological closing using 3x3 isotopic structure element to smear the fragmented characters, and a noise elimination that detects isolated black dots and erases them. These can make the pre-segmentation and character classification easy. The system processes the input image in three ways as below.

- Noise elimination + closing
- Closing
- No pre-processing (Raw image)

The system feeds these images to following processes one by one until an amount is recognized.

3.3 Two-dimensional Pre-segmentation

In the two-dimensional pre-segmentation, the system extracts candidate segments at first by splitting the input image horizontally based on the connected component analysis as conventional methods. Then if height of a candidate segment is larger than a predefined value and there is a gap in it, the system splits it vertically and adds the split segments to the set of the candidate segments.

The positional relationships among the candidate segments are represented by a directed graph named "segmentation graph". Here, the segmentation graph is represented by a set of records p_j

$$E_p=\{p_j=(a_j,b_j,t_j,r_j)\} \quad (1)$$

Each p_j corresponds an edge in the graph and stores information on a candidate segment. a_j and b_j are the integer variables that identify the nodes incident to p_j. t_j is a list of n_j connected components $(c^{tj}{}_1, c^{tj}{}_2, .., c^{tj}{}_{nj})$ in the segment. r_j stores the results of the character classification explained below.

Followings are the flow of the two-dimensional pre-segmentation:

1) Extract connected components from the input image.
2) Test the size of all the connected components. If width of a connected component is larger than w_{max} (=56 pixels), then it is split at the point w_{cut} (=44 pixels) right from the left edge.
3) Sort the connected components according to the x coordinates of the left boundary. Let n_c is the number of the connected components, and c_i is the ith connected component.
4) Let w_{ij} is width of the object consisting of connected components c_i, c_{i+1},,, and c_{j-1}. For all combinations of i and j ($0 < i < j \leq n_c+1$), test if w_{ij} satisfies $w_{ij} < w_{max}$ (=56 pixels) and $w_{ij} > w_{min}$ (=8 pixel). If w_{ij} satisfies the conditions, generate a new record p_k where $a_k= i$, $b_k= j$, $t_k= (c_i, c_{i+1},,,c_{j-1})$. Here some heuristics are used to prevent separating connected components extremely close to each other.
5) Test all candidate segments if the height of p_j is larger than H_{max} (= 40 pixels) and it has a gap of vertical projection wider than 2 pixels, then split p_j vertically at the gap. Add the split segments to candidate segments. Here a and b of the new segments p_k are equal to the ones of p_j. t_k stores the lists of the connected components that belong to p_k.

2) is a step to split touching characters. As easily expected, sometimes it cannot find the boundaries of touching characters correctly. Some papers such as [1][6] present more sophisticated methods.

3.4 Character Classification

In the character classification, the system extracts a feature vector from each candidate segment, and classifies it by a neural network.

Firstly, it fits a square to each candidate segment so the center and the height of the square are equal to the ones of the bounding box of the candidate segment respectively. Then, it calculates the densities of black pixels around 64 sampling points of the 8x8 grid on the square. Finally, it feeds 64 of the density values to a neural network to infer the category of the candidate segment. Here we used a multi-layer neural network with a hidden layer that has 32 hidden units. 13 output nodes are assigned to digits, "¥", "*", and comma respectively. The output value of the each node in the output layer is used as the credibility of each category. If the credibility of output node is higher than a threshold (=0.6), the system adds the pair of the character code and the credibility to r_j.

We printed 37,252 characters with 8 checkwriters for the training of the classifier. The error rate of the 5-fold cross-validation test using the training samples was 0.19%. However, font styles in some of the images in section 4 were different from the training samples, and the recognition accuracy was not so high as this test.

3.5 Bottom-up Parsing

3.5.1 Grammar

In our method, a context-free grammar represents the linguistic constraints in the monetary amounts. Formally, a grammar (G) is a quadruple,

$$G = (V_n, V_t, P, S), \quad (2)$$

where V_n is a set of symbols called non-terminal symbols, V_t is a set of symbols called terminal symbols. Let V^* denotes the set of all words built from the terminal and/or non-terminal symbols. The set $P \in V_n \times V^*$ is the set of the substitution rules such as "A -> a" and "B -> eDE" where a capital letter represents a non-terminal symbol, a small letter represents a terminal symbol, and an operator "->" represents the substitution. The set $S \in V_n$ is the set of the start symbols. The set of all arrays of terminal symbols that can be generated from the start symbols by the substitution rules is called language $L(G)$.

In our method, we used a context-free grammar G in which elements of V_t correspond to the characters used in the monetary amounts, and $L(G)$ covers all the expressions of amounts. Figure 3 shows the substitution rules used in our method. Here, capital letters represent the non-terminal symbols and "S" represents the start symbol. Digits, "¥", "*" and comma are the terminal symbols.

S -> ¥N*	B -> DDD	C -> 6
S -> ¥N,*	C -> 1	C -> 7
N -> N,B	C -> 2	C -> 8
N -> C	C -> 3	C -> 9
N -> CD	C -> 4	D -> 0
N -> CDD	C -> 5	D -> C

Figure 3: Substitution rules in the new method.

3.5.2 Symbol Graph

Usually the input of a parser is one-dimensional. For example, CYK algorithm [7] is a typical bottom-up parsing method. Its input is a totally ordered set of terminal symbols $W=\{w_i | 1 \leq i \leq n\}$. It utilizes a Chomsky normal form grammar G_c, where only substitutions of one non-terminal symbol by one terminal symbol or two non-terminal symbols exist in P. A $n \times n \times (|V_t|+|V_n|)$ table T stores credibility of the array $\{w_i, w_{i+1},,, w_{i+j}\}$ as a symbol $v_k \in V_n \cup V_t$ in the cell $T[i][j][k]$. The CYK algorithm is a kind of the dynamic programming that calculates $T[i][j][k]$ from $T[i][i'][k']$ and $T[i'][j][k']$ when the substitution v_k -> $v_k v_{k''}$ is an element of P.

The input of our bottom-up parser is the results of the character classification stored in the segmentation graph. They are not a totally ordered set, and conventional methods are not applicable. Thus, we developed a new bottom-up parsing method that utilizes a graph named "symbol graph" instead of the table T of the CYK algorithm.

In the system, a set of records e_j as shown below represents symbol graph.

$$E = \{e_j = (a_j, b_j, s_j, c_j, l_j)\}. \quad (3)$$

Each e_j corresponds an edge in the graph, and a_j and b_j are the integer variables that identify the nodes in the

graph. $s_j \in V_n \cup V_t$ stores a symbol that is assigned to the edge, and c_j is its credibility value. $l_j=(e^{lj}_1, e^{lj}_2, .., e^{lj}_{nj} \mid e^{lj}_t \in E)$ stores a list of edges in E.

3.5.3 Algorithm

The algorithm of the bottom-up parsing is shown below. Here, the operator "Σl_j " represents the concatenation of lists l_j.

1) Generate a symbol graph in which s_j and c_j of each edge correspond to the category and its credibility of a candidate segment respectively given by the character classification. a_j and b_j are equal to the ones of the corresponding candidate segment.
2) Apply the "reverse substitutions" to partial paths in G_s. Let $e.x$ denote the variable x in a record e. If $path = (e^{path}_1, e^{path}_2, .., e^{path}_n path)$ is a sequence of edges in the graph that satisfies $e^{path}_j.b = e^{path}_{j+1}.a$ $(1 \le j \le n^{path}-1)$, and if the sequence of symbols $e^{path}_1.s\ e^{path}_2.s \ldots e^{path}_n path.s$ is equal to the right side of a substitution rule, then generate new edge e_k where $a_k = e^{path}_1.a$, $b_k = e^{path}_n path.b$, $c_k = \Sigma_j e^{path}_j.c$, $l_k = \Sigma_j e^{path}_j.l$, and s is equal to the left side of the substitution rule.
3) If there are more than one edges that have same values of a, b, and s respectively, and they have same path from a to b in the segmentation graph, then delete them except for the one that has the largest c.
4) Repeat step 2) and 3) until the system cannot generate a new edge any more in step 2).
5) Find edges whose symbols (s) are the start symbols and that satisfy the condition of the layout described below. Find e_{best} whose credibility c_j is the largest among them. Output the array of symbols in l_{best} of the e_{best} $(e^{lbest}_1.s\ e^{lbest}_2.s \ldots)$ as the result of the amount recognition.

Figure 4 schematically illustrates a process of the bottom-up parsing.

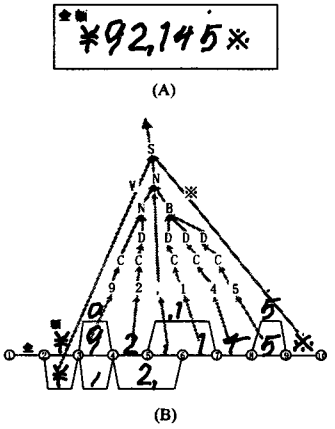


Figure 4: Schematic representation of a bottom-up parsing. (A): Example of an input Image, (B): The segmentation graph (below) and the reverse substitution process (gray arrows).

3.5.4 Utilization of Layout Information

In step (5) above, the system tests the layout of the candidate segments corresponding to the elements of l to eliminate errors. Firstly, the system calculated the upper and lower boundary of the text line by the least square error fitting. Then, it tests if the candidate segments satisfy the following condition.

a) Height of a comma should be smaller than the half of the line height. The upper boundary of the comma also should be lower than the one third from the bottom of the line.

b) Height of a character other than comma is larger than two third of the line height.

The system selects the final result among the edges that satisfy these conditions.

4 Experimental Results

We tested 1,314 images for the experiments. There was a monetary amount printed by a checkwriter in each image.

We used two programs for comparison of the new method and a conventional method. The program of the conventional method is based on the segmentation-based recognition. It utilized extensive heuristics to eliminate noises and to recover fragmented characters instead of the new method. When we simply omitted the step 5) in the two-dimensional segmentation from the new method without such heuristics, about half the images were rejected

Table 1: Comparison of the recognition accuracy

	Conventional method	New method
Correct rate	64.1%	72.7%
Error rate	1.6%	0.9%

The experimental results in table 1 show the significant improvement of the recognition accuracy by the new method. Major reasons of rejection and error were touching characters, noises, unknown font styles, and fragmented characters. The average processing time of the new method using a 450 MHz machine was 54 msec per image.

4 Conclusions

A new method to recognize monetary amounts printed by a check writer is proposed. Here we showed a way to utilize grammatical constraints in the two-dimensional character segmentation approach. Experimental results show that the method improved the recognition accuracy significantly.

The bottom-up parser worked well in this experiment. However, it is not clear so far that it is the best technique to interpret the result of the character

classification. Moreover, further improvement of the pre-segmentation and the character classification is necessary.

References

[1] R.G. Casey *et al.*, A survey of methods and strategies in character segmentation, *IEEE Trans. Pattern Analysis and Machine Intelligence,* **18** (1996) 690 – 706.

[2] H. Fujisawa *et al.*, Segmentation methods for character recognition: From segmentation to document structure analysis, *Proc. of IEEE,* **80** (1992) 1079 – 1092.

[3] M.J. Shridhar *et al.*, Segmentation-based cursive handwriting recognition in *Handbook of character recognition and document image analysis*, H. Bunke and P.S.P Wang (Eds.) (World Scientific, Singapore, 1997).

[4] E. Ishidera *et al.*, Unconstrained Japanese address recognition using a combination of spatial information and word knowledge, in *Proc 4th International Conference on Document Analysis and Recognition,* Ulm, Aug. 18-20, 1997, 1016 - 1022

[5] H. Ikeda *et al.*, A Context-free grammar-based language model for document understanding, in *Proc 4th International Workshop on Document Analysis Systems,* Rio de Janeiro, Dec. 10-13, 2000, 135 – 146

[6] S.W. Lee *et al.*, A new methodology for gray for gray-scale character segmentation and recognition, *IEEE Trans. Pattern Analysis and Machine Intelligence,* **18** (1996) 1045 – 1050.

[7] D. Jurafsky *et al.*, Speech and language processing, (Prentice Hall, New Jersey, 2000).

The architecture of TRUEVIZ: A groundTRUth/metadata Editing and VIsualiZing toolkit

Chang Ha Lee and Tapas Kanungo

Language and Media Processing Laboratory
Center for Automation Research
University of Maryland
College Park, MD 20742
{chlee,kanungo}@cfar.umd.edu

Abstract

Tools for visualizing and creating groundtruth and metadata are crucial for document image analysis research. In this paper we describe TrueViz [LK00, KLCB01], which is a tool for visualizing and editing groundtruth/metadata. We first describe the groundtruthing task and the requirements for any interactive groundtruthing tool. Next we describe the system design of TrueViz and discuss how a user can use it to create groundtruth. TrueViz is implemented in the Java programming language and works on various platforms including Windows and Unix. TrueViz reads and stores groundtruth/metadata in XML format, and reads a corresponding image stored in TIFF image file format. Multilingual text editing, display, and search modules based on the Unicode representation for text are also provided. This software is being made available free of charge to researchers.

This research was funded in part by the Department of Defense under Contract MDA0949-6C-1250, Lockheed Martin under Contract 9802167270, the Defense Advanced Research Projects Agency under Contract N660010028910, and the National Science Foundation under Grant IIS9987944.

1 Introduction

In the document image analysis (DIA) research area, the term 'groundtruth' refers to various attributes associated with the text on the image — bounding box coordinates of words, lines, characters; font type; character size; direction of text; etc. Groundtruth data is crucial for document image analysis because it is impossible to train and test Optical Character Recognition (OCR) algorithms without it. Since groundtruth is created manually in most cases, tools for annotating and visualizing groundtruth are very important. In fact, at the MLOCR99 international workshop [mlo99] the consensus in the corpus working group was that our community needs i) a protocol for groundtruthing documents, ii) an XML-based groundtruth representation format, iii) a public-domain multilingual/multiplatform visualization and data-entry tool, and iv) a consortium for managing and distributing datasets.

In this paper we address two of the four issues raised by the working group: i) We describe an XML-based groundtruth representation format, and ii) we describe TrueViz, which is a public domain[1] annotation tool that we have developed at the University of Maryland.

This paper is organized as follows. In Section 2 we describe various existing annotation tools used in document image analysis and in related areas such as speech recognition, linguistics, and information retrieval. The desirable features of a document image groundtruthing tool are described in Section 3. In Section 4 we discuss design and implementation issues related to editing, visualization, and search. The XML data format for groundtruth is discussed in Section 5, where we also provide representative samples of XML files. The multilingual data entry, visualization, and search features of TrueViz are quite unique and are discussed in Section 6. Finally, in Section 7 we list the things that we hope the international DIA community will add to the public domain system.

2 Previous Work

There are many annotation and visualization tools in various domains. In this section we describe a few annotation tools commonly used in document image analysis, speech recognition, linguistics, information retrieval, video analysis, geographic systems, and statistics. In Table 1 we provide a comparison of these tools.

2.1 Document Image Visualization Tools

Visualization tools for displaying or editing a document image and groundtruth metadata have been developed for evaluating algorithms, creating document groundtruth, or browsing documents.

Pink Panther [YV98] is an environment for creating segmentation groundtruth files and for page segmentation benchmarking. Page segmentation is the process of decomposing a document page image into structural and logical units, such as images, paragraphs, headlines, tables, etc. The performance of a page segmentation algorithm is evaluated

[1] TrueViz is available at `http://www.cfar.umd.edu/~kanungo/software/software.html`

Table 1: Comparison of Visualization Tools

Name	Platform	Data Format	Domain
PinkPanther	Unix/X Windows System	ASCII	Document Image Groundtruth
Illuminator	Unix/X Windows System	DAFS	Document Image Groundtruth
Oulu Database Browser	Multi-Platform/Java	ASCII	Document Image Groundtruth
TrueViz	Multi-Platform/Java	XML Format	Document Image Groundtruth
Transcriber	Unix/Windows NT	XML Format	Speech Annotation
ATLAS	Unix/Windows NT	XML Format	Linguistic Annotation
Alembic Workbench	Unix system	SGML/PTF Format	Linguistic/Named Entities Annotation
ViPER	Multi-Platform/Java	ASCII	Video Sequence Groundtruth
XGobi	Unix/X Windows System	S Data Format/ASCII	Statistical Data
S-PLUS	Windows 95/98	Customized Data	Statistical Data
CLASP	Unix/Macintosh	Commonly Used Formats	Statistical Data
Mondrian	Multi-Platform/Java	ASCII/Databases	Categorical/ Geographical Data
PolyPaint+	SunOS/Solaris	netCDF	Geographical Data
Spotfire	MS Windows	Database/Spreadsheet/ ASCII	Decision Making by Data Analysis
Slicer Dicer	MS Windows	Binary/ASCII/ Commonly Used Formats	Medical/Scientific Data Defined on Grids

by running the algorithm on a set of document images, and comparing the output for each document to corresponding groundtruth metadata. Pink Panther consists of two parts: Grounds-Keeper and Cluzo. Grounds-Keeper is a tool for creating groundtruth metadata. It visualizes a document image and the corresponding metadata, and also allows users to zone the document image and specify the information for each zone. Groundtruth metadata created by Grounds-Keeper is stored in an ASCII file format. Cluzo is a benchmarking tool for collecting the locations, types and severities of segmentation errors on a page as well as information on segmentation performance. Pink Panther is implemented on the Unix and X Windows platforms and is written in C. While Grounds-Keeper allows the user to enter segmentation groundtruth, entering text groundtruth is not possible.

Illuminator [Fru95] is an editor developed by RAF Technology, Inc. for building document understanding test and training sets, for correction of OCR (Optical Character Recognition) errors, and for reverse-encoding the essential information and attributes of a document. Illuminator visualizes or edits a document image and its entities, which are specific regions of the image and the associated metadata. It is configured to handle text in major European languages and Japanese. Illuminator uses the DAFS (Document Attribute Format Specification) file format [Fru95] to store the document image and metadata. DAFS provides a format for breaking down a document into entities which have hierarchical structure, and for defining entity boundaries and attributes. Illuminator is implemented on the Unix and X Windows platforms and is written in C.

The MediaTeam Oulu Document Database [SK98] is a collection of scanned documents with corresponding groundtruth for the physical and logical structure of the documents. It was developed by the University of Oulu MediaTeam. The document database browser is a visualization tool for exploring the contents of the database. The browser is written in the Java programming language and allows visualization of document images and corresponding metadata simultaneously. The browser can explore the database and select particular documents for visualization. The browser also provides a window to list attributes of the document. Document images which were originally stored in TIFF image format are stored in JPEG image format and metadata is stored in an ASCII file format.

Pink Panther and Illuminator work only on the Unix platform. Because there are many tools that are executable only on the Windows platform, this is a limitation. The Oulu document database browser is written in the Java programming language, and can be run on various platforms. However, the Oulu document database supports JPEG image format only, while TIFF is the most popular image format for document images. Furthermore, the file representation of the groundtruth is non-standard. In fact, all the above tools store document metadata in their own file formats. To provide data compatibility, a standard file format, or a file format to which other file formats can be easily converted, is needed.

A prototype system for visualizing and editing groundtruth is currently being built at the University of Fribourg, Switzerland [HRI00]. This system allows users to edit the hierarchical structure of the document. However, the system does not provide a compatible OCR evaluation package to visualize OCR segmentation results.

2.2 Other Visualization Tools

We surveyed visualization tools in other data domains to find out the best way to provide multi-platform and data compatibility. In this section we summarize features of visualization tools in various domains such as statistical, categorical, geographical, and medical data as well as linguistic data and speech signals.

Transcriber [BGWL00, GBBW00, BGWL98] is a tool for segmenting, labeling and transcribing speech signals. It supports most common audio formats and stores the transcription in XML format. It was developed in the Tcl/Tk and C programming languages, and works on Unix and Windows NT platforms.

ATLAS [BDH+00] is an architecture and tool for linguistic analysis based on a formal model for annotating linguistic artifacts. It uses an XML-based ATLAS Interchange Format (AIF) for storing annotated corpora, and was developed in the C++, Perl, Tcl/Tk and Java programming languages.

Alembic Workbench [DAH+97] is a new set of integrated tools that uses a mixed-initiative approach to bootstrapping the manual tagging process with the goal of reducing the overhead associated with corpus development. The Alembic Workbench is developed using the Tcl/Tk, Perl, C and Lisp programming languages, and works on the Unix platform. Alembic uses the SGML and PTF (Parallel Tag File) formats for source text and annotations.

ViPER (Video Processing Evaluation Resource) [DM00] consists of three main components: ViPER-GT, ViPER-PE, and ViPER-Viz. ViPER-GT contains modules for configuring and producing groundtruth information which describes a video sequence. The ViPER-PE module provides performance evaluation capabilities for comparing computed results with appropriate groundtruth information. ViPER-Viz enables a user to visualize groundtruth, analysis results, performance evaluation results, or an entire video clip. ViPER was developed in the Java programming language, and groundtruth and results are stored in ASCII file format.

XGobi [SCB98, SHB91, SCB92] is an X Window application for interactively exploring statistical data. Its current functionalities include brushing, identification, and editing of connected lines, as well as rotation and the grand tour, with several interactive projection pursuit indices. Several functions can be linked so that actions in one window are promptly reflected in another.

S-PLUS [VR99] is a desktop data analysis tool that provides data analysis and visualization capabilities to identify trends in data. It allows data import and export from spreadsheets such as Excel, as well as from a wide range of relational and other data sources.

The Common Lisp Analytical Statistics Package (CLASP) [AWC+95] is a tool for visualizing and statistically analyzing data. CLASP provides an interactive environment for data manipulation and statistical analysis and a variety of descriptive and hypothesis-testing statistics. It includes many features that facilitate exploratory data analysis.

Mondrian [Uni] is a data-visualization system written in Java. Its main emphasis is on visualization techniques for categorical data and geographical aata. Mondrian provides various plots such as mosaic plots, maps, barcharts, and parallel coordinates, which are fully linked and allow various interrogations.

PolyPaint+ [Nat] is an interactive scientific visualization tool that displays complex structures within three-dimensional data fields. It provides color shaded-surface display, as well as simple volumetric rendering in either index or true color. PolyPaint+ routines first compute the polygon set that describes a desired surface within the 3D data volume, and these polygons are then rendered as continuously shaded surfaces. Objects rendered volumetrically may be viewed along with shaded surfaces. Additional data sets can be overlaid on shaded surfaces by color coding the data according to a specified color map.

Spotfire [AS94] is a decision analysis workspace that uses the connectivity of the Web to provide a workspace in which to access large amounts of complex data from wherever it resides, to visually explore and analyze the data, and to share results.

Slicer Dicer [PIX] provides tools for analysis, interpretation and documentation of complex data defined in three or more dimensions. It helps in exploring the data visually by "slicing and dicing" to create arbitrary orthogonal and oblique slices, rectilinear blocks and cutouts, isosurfaces, and projected volumes. It also provides animation sequences featuring continuous rotation, moving slices, blocks, parametric variation (time animation), oblique slice rotation, and varying transparency.

A more detailed review and taxonomy of visualization tools can be found in an article by Shneiderman [Shn96], and a good general reference for user interfaces is Shneiderman's book [Shn98].

3 Desired GUI Functionalities

Since TrueViz will be used by different researchers for different tasks, we first summarize the functionalities that are desired of such a tool. The simplest task that the tool could be used for is to visualize and input multilingual text. Next, it could be used to mark regions of a scanned document image as text or graphics, and assign labels to regions. A researcher wanting to look at the results obtained by a DIA system might want to search for all the incorrectly recognized characters and then zoom into the image at those locations. A researcher interested in extracting the logical structure of a document might want to label the reading order of the text areas, or the hierarchy of the text regions corresponding to sections and subsections.

After studying the various tasks for which a user might want to use the to-be-designed tool, we formulated the following set of requirements for the graphical user interface:

Entities: Users should be able to visualize and edit zone-, line-, word-, and character-level geometric groundtruth. Furthermore, they should be able to establish their own entity structure. For each entity, they should be able to define attributes (e.g. bounding boxes) and specify their values.

Scale: Users should be able to zoom in and out of the image and overlaid groundtruth so that they can study the image and OCR error results at the page, paragraph, line, word, or character level.

Color: It should be possible to display entities that have different attributes in different colors. For example, image zones could be shown in one color and table or text

zones in another. Thus if a DIA system incorrectly recognizes a table zone as an image zone, the error would be easily identifiable from the color coding.

Logical information: The visualization tool should allow users to visualize and edit the logical reading order of text zones, and also to specify the hierarchy of the text zones. For example, it should be possible to visually specify that a subsection is contained in a section.

Multilingual Visualization: Since DIA systems are being developed for various languages and scripts, users should be able to visualize groundtruth text in these languages and scripts. The use of a standard encoding such as Unicode is highly desirable.

Multilingual Data Entry: While regular English text can be entered by regular keyboards, keyboard mappings that allow other languages and scripts to be entered should also be available.

XML-based Representation: The XML markup language would be ideal for representing page layout groundtruth since it is the current industry standard and various parsers, syntax checkers and editors are publicly available for it.

Converters: Converters to convert standard datasets such as the University of Washington dataset (in DAFS format) into the XML representation would help bootstrap research by providing seed datasets.

Search: Users should be able to search for strings in the groundtruth and find the locations where they appear in the image. The search module should work in any language and users should be able to specify edit distances for approximate searching, which is essential when searching for strings in noisy OCR text.

Evaluation: The tool should have a built-in OCR evaluation module or should be compatible with one, so that users are able to visualize OCR evaluation results easily.

Multiplatform: Since researchers and data entry persons work on various platforms such as UNIX, PC and Mac, the tool should be platform-independent so that users need not spend time learning how to use it on a platform that they are not familiar with.

Public Domain: In order for the community to take full advantage of it, the tool should be freely available.

4 Design and Implementation

4.1 Overview

The TrueViz display is vertically split into two panels (see Figure 1). The left panel is an image panel for displaying a document image and corresponding geometric metadata, and the right panel is a tree view for displaying textual metadata structure.

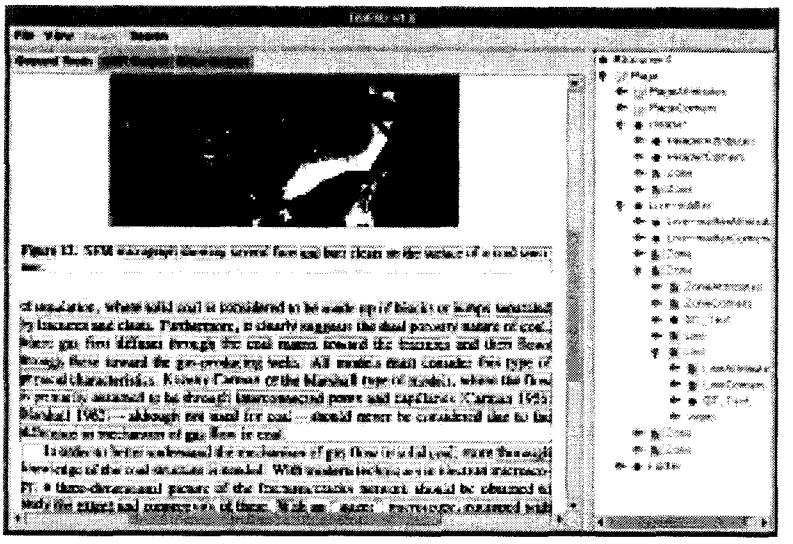


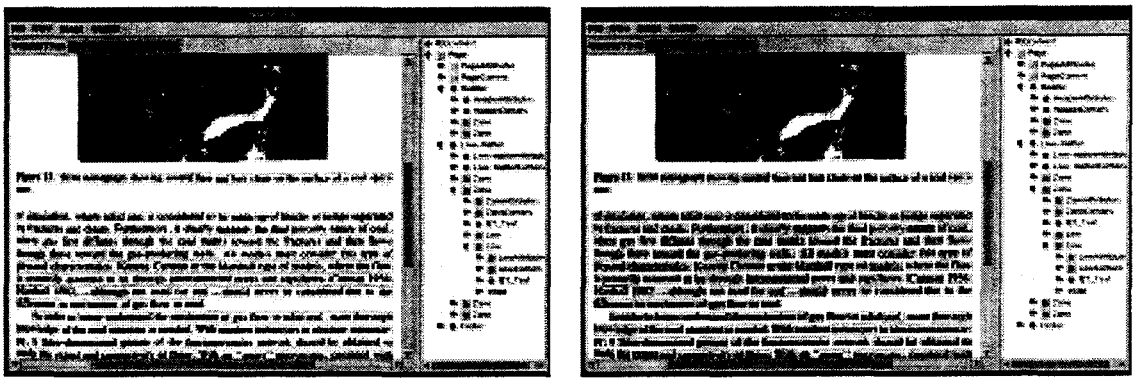
Figure 1: TrueViz consists of an image panel (left) and a tree view (right).

The image panel displays a document image and overlays geometric metadata on the image. Currently, three kinds of geometric metadata can be visualized: Bounding boxes, logical relationships, and an Infopanel. The bounding box of an entity is visualized as a polygon whose color represents the type of the entity. "Logical relationship" refers to logical reading order, and is visualized using an arrow from one entity to the next. The Infopanel is a small window for displaying a few important attributes of the entity. The image and metadata visualization can be scaled to various resolutions.

The tree view displays the XML-based groundtruth metadata in a tree structure of expandable and collapsible nodes. The attribute values can be edited in the tree nodes and the groundtruth text can be edited in the separate multilingual text editor.

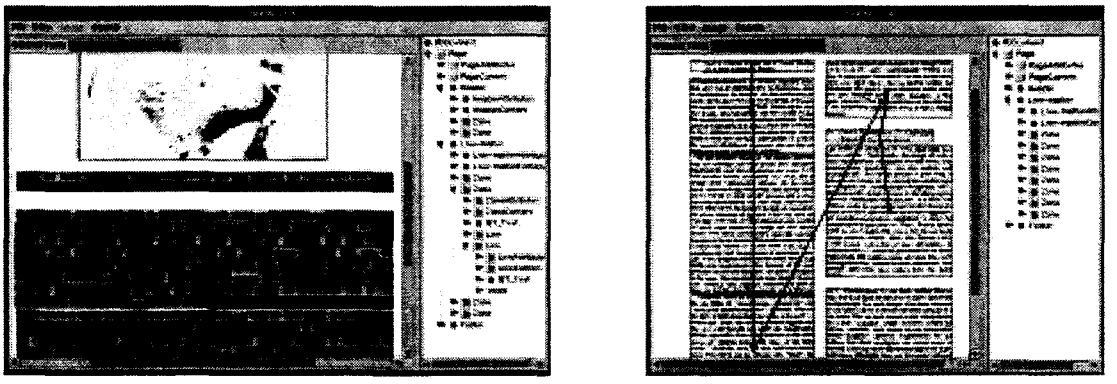
4.1.1 Metadata Visualization

Entities can be classified into four categories: Zones, Lines, Words and Characters. Entities are hierarchical in nature, so a Zone is contained within a Page, a Line is contained within a Zone, a Word is contained within a Line, and a Character is contained within a Word. Because of the hierarchical nature of the entities, it is necessary to change views in order to view specific portions of the structure. There are five views: Image Only, Page, Zone, Line, Word and Character. The Image Only view shows only the image without any groundtruth visualization. The Page view shows metadata for all entities, from the highest level to the lowest level. This view is not editable or selectable. The Zone view shows only Zone metadata. A Zone's data can be accessed by clicking on the Zone. This causes the Zone to be active (selected) and highlighted, and the Infopanel to pop up. The Infopanel is a small window for displaying important metadata for the active entity (see Figure 8). The corresponding node in the tree view will also be selected. Similarly, the Line view shows all Line metadata (see Figure 2 (a)), the Word view shows all Word metadata (see Figure 2 (b)), and the Character view shows all Character metadata. As



(a) Line view displays all Line entities. (b) Word view displays all Word entities.

Figure 2: Hierarchical display.



(a) Fill Bounding Boxes. (b) Logical Relations.

Figure 3: View options

in the Zone view, metadata can be selected, and the Infopanel for the active entity is popped up.

There are two options for views: 'Fill Bounding Boxes' and 'Logical Relations'. If the 'Fill Bounding Boxes' option is checked, all entities are painted in colors corresponding to their types (see Figure 3 (a)). Otherwise, entities are displayed using polygonal outlines whose colors also represent their types. This option is useful when the document is displayed at a large scale, because a user can see the type of an entity from its color even if the bounding box is too large to fit on the screen. If the 'Show Logical Relations' option is selected, the logical reading order relations are visualized using arrows from each entity to the next logical entity (see Figure 3 (b)).

4.1.2 Metadata Editing

Groundtruth metadata can be edited in two ways: graphical editing and text editing. All metadata can be edited within the attribute value node in the tree view. Because the

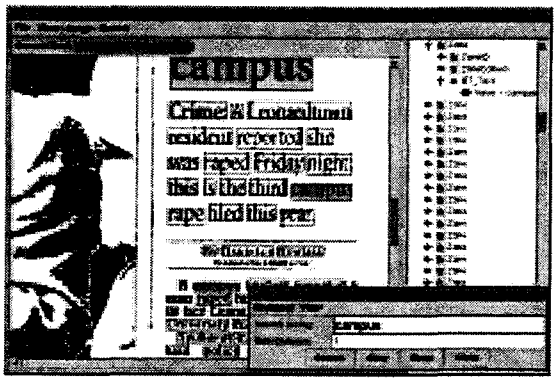


Figure 4: Search string "campus" and edit distance 1 are specified in the search window (in the lower right corner), and the matching entities are highlighted.

groundtruth text may contain multilingual text, it is edited in the separate multilingual text editor. The metadata visualized in the image panel can also be edited graphically. It is very difficult to correct bounding boxes of entities by editing their coordinates. Therefore, TrueViz enables users to change the coordinates of bounding boxes graphically. In addition to the bounding boxes, the logical relationships can be changed graphically. The image panel can also be used used to create and delete entities.

4.2 Search

TrueViz provides a multilingual approximate search functionality. A search string and edit distance can be specified in the search window. TrueViz provides multilingual input for a search string. The edit distance is the minimum number of substitutions, insertions and deletions required to transform one string into another. The maximum edit distance allowed during the search can be specified [Gus97]. After the search is finished, all entities containing the search string within the specified edit distance are highlighted (see Figure 4).

5 The Data Format

5.1 Overview

Groundtruth metadata is stored in XML file format [Har99, BPSM98, WHA+99, McL00] (see Figure 5), and document images are stored in TIFF image file format. The tree view reflects the XML data file, and an internal data structure is created to visualize the groundtruth metadata. The internal data structure consists of Region of Interest (ROI) nodes. A ROI is a generic term used to describe any area of the image that the user deems of interest. The internal data forms a directed acyclic graph with ROIs as nodes and hierarchical or logical links as edges.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Page SYSTEH "Trueviz.dtd">
<Page>
        <PageID Value="P000"> </PageID>
        <PageType Value="Journal"> </PageType>
        <PageNumber Value="1"> </PageNumber>
        <PageColumns Value="1"> </PageColumns>
        <Font Size="9-12" Spacing="Undefined" Style="Normal" Type="Serif"> </Font>
        <Zone>
                <ZoneID Value="Z000"/>
                <ZoneNext Value="Z001"/>
                <CharacterOrientation Type="String" Value="up-right"/>
                <DominantFontSize Type="String" Value="9-12"/>
                <DominantFontSpacing Type="String" Value="proportional"/>
                <DominantFontStyle Type="String" Value="plain"/>
                <DominantFontType Type="String" Value="serif"/>
                <Language Type="String" Value="English"/>
                <TextAlignment Type="String" Value="justified"/>
                <TextReadingDirection Type="String" Value="left-right"/>
                <ZoneCorners>
                        <Vertex x="1281" y="3136"></Vertex>
                        <Vertex x="1296" y="3136"></Vertex>
                        <Vertex x="1296" y="3169"></Vertex>
                        <Vertex x="1281" y="3169"></Vertex>
                </ZoneCorners>
                <GT_Text Value="a"></GT_Text>
                <Line>
                        <LineID Value="Z000L000"/>
                        <LineCorners>
                                <Vertex x="1281" y="3136"></Vertex>
                                <Vertex x="1296" y="3136"></Vertex>
                                <Vertex x="1296" y="3169"></Vertex>
                                <Vertex x="1281" y="3169"></Vertex>
                        </LineCorners>
                        <GT_Text Value="a"></GT_Text>
                        <Word>
                                <WordID Value="Z000L000W000"/>
                                <WordCorners>
                                        <Vertex x="1281" y="3136"></Vertex>
                                        <Vertex x="1296" y="3136"></Vertex>
                                        <Vertex x="1296" y="3169"></Vertex>
                                        <Vertex x="1281" y="3169"></Vertex>
                                </WordCorners>
                                <GT_Text Value="a"></GT_Text>
                                <Character>
                                        <CharacterID Value="Z000L000W000C000"/>
                                        <CharacterCorners>
                                                <Vertex x="1281" y="3136"></Vertex>
                                                <Vertex x="1296" y="3136"></Vertex>
                                                <Vertex x="1296" y="3169"></Vertex>
                                                <Vertex x="1281" y="3169"></Vertex>
                                        </CharacterCorners>
                                        <GT_Text Value="a"></GT_Text>
                                </Character>
                        </Word>
                </Line>
        </Zone>
        <Zone>
                <ZoneID Value="Z001"/>
                <ZoneNext Value=""/>
                <CharacterOrientation Type="String" Value="up-right"/>
                <DominantFontSize Type="String" Value="9-12"/>
                <DominantFontSpacing Type="String" Value="proportional"/>
                <DominantFontStyle Type="String" Value="italic"/>
                <DominantFontType Type="String" Value="serif"/>
                <Language Type="String" Value="English"/>
                <TextAlignment Type="String" Value="justified"/>
                <TextReadingDirection Type="String" Value="left-right"/>
                <ZoneCorners>
                        <Vertex x="2281" y="3136"></Vertex>
                        <Vertex x="2296" y="3136"></Vertex>
                        <Vertex x="2296" y="3169"></Vertex>
                        <Vertex x="2281" y="3169"></Vertex>
                </ZoneCorners>
                <GT_Text Value="b"></GT_Text>
        </Zone>
</Page>
```

Figure 5: An example XML file.

5.2 XML Data Format

The groundtruth data is organized in a hierarchical structure. The highest-level and therefore most inclusive entity is the Document. A Document is, in its simplest form, a collection of individual units, known as Pages, which are related to or support a specific topic or purpose (e.g. a report or manual). A Page is the next level down in the hierarchy and represents individual units of a Document. Each Page has an associated image that represents the original hard copy. A Page contains one or more Zones. A Zone is usually a rectangular area definable by its horizontal and vertical coordinates within a page. The purpose of a Zone is to identify a key area of the page such as title, heading, graphic, page number, etc. Each Zone may contain one or more Lines. A Line is an individual line of text. A Line can be broken down into one or more Words, each of which may contain one or more Characters. Each tag in the XML file represents an entity or attribute. An entity name can be any alphanumeric word, but the only entities that can be graphically edited in TrueViz are Zone, Line, Word and Character.

An entity's attributes can be listed under the entity's tag in the XML file. While any attribute name can be listed, some built-in attributes are crucial for the visualization of groundtruth data.

ID: ID is the identification of the entity. The attribute name for ID is combined with the entity name. For example, the ID of a Zone entity is represented as ZoneID, and similarly we use LineID for Line, WordID for Word, and CharacterID for Character.

Corners: Corners represent the bounding box of the entity. The upper left, upper right, lower right, and lower left vertices are listed inside a Corners tag in order. Like the ID, the attribute name is combined with the entity name.

Next: Next stores the ID of the logically following entity. As with the ID, the attribute name is combined with the entity name.

GT_Text: GT_Text stores the groundtruth text of the entity.

The following example shows a simple entity.

```
<Zone>
  <ZoneID Value="Z001"/>
  <ZoneNext Value="Z002"/>
  <GT_Text Value="Hello, world">
  <ZoneCorners>
    <Vertex x="10" y="10"/>
    <Vertex x="100" y="10"/>
    <Vertex x="100" y="30"/>
    <Vertex x="10" y="30"/>
  </ZoneCorners>
<Zone>
```

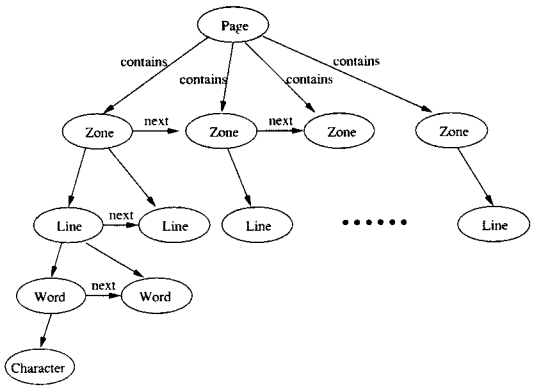


Figure 6: Entity structure.

5.3 Internal Data Structure

The groundtruth metadata is stored in XML file format, which is essentially a tree. The entities, on the other hand, form a directed acyclic graph structure. Each entity contains child entities and has a next logical entity. The graph representing the entity structure can be expressed by equation (1) (see Figure 6):

$$G = (V, E) \quad \text{where} \quad V = \{Zone, Word, Line, Character\},\ E = \{contains, next\} \tag{1}$$

Because of the difference between the entity structure and the XML structure, TrueViz has an internal data structure that is a little different from the XML structure. The internal data structure consists of Region of Interest (ROI) nodes, and the ROIs form a directed acyclic graph as described in equation (1). A next logical entity is stored as an attribute of an entity in the XML file, and is converted into a link from a ROI to the next ROI in the internal data structure.

For parsing XML files and converting XML structures into internal structures, Java APIs (Application Program Interfaces) were used. Two kinds of Java APIs can be used for XML parsing: SAX (Simple API for XML) and DOM (Document Object Model) [McL00]. SAX is an event-based framework for parsing XML data. It reads through the XML document, breaks down the data into usable parts, and defines the events that occur at each step of the process. DOM provides a data representation of an XML document as a tree, which can be traversed and manipulated. A DOM parser was used in TrueViz because TrueViz has an internal data structure that needs to be kept in memory.

5.4 Flexible Entity Structure

Various entity hierarchies are used [AS99, KA99, dli99], depending on the type of document. Users may want to build document metadata using their own structures. TrueViz provides flexible entity structures so that users can build their own entity structures and DTD (Document Type Definition) files for defining and verifying the structures of their

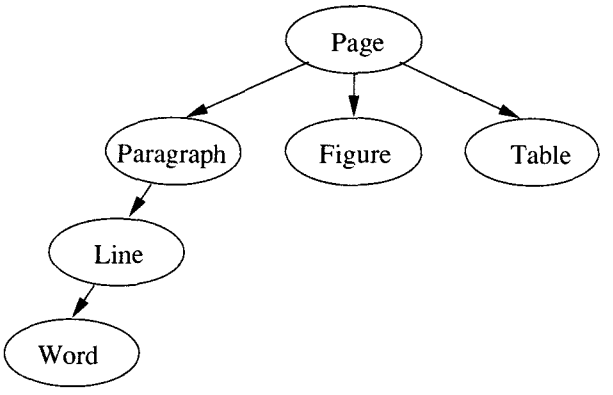


Figure 7: User-defined entity structure.

XML files. The entity structure is extracted from the XML file, and the DTD file can be used to verify that the XML file conforms to the corresponding entity structure. The DTD file can be created and edited using any existing public domain editor.

If an element has an "Entity" attribute and its value is "True," the element is recognized as an entity when the XML file is parsed. An entity structure for an XML file is automatically built by TrueViz from the recognized entities and their level information. The following is an example XML file with a user-defined entity structure, and Figure 7 is the entity structure extracted from the XML file. If there are no elements with the attribute "Entity," the default entity structure (see Section 5.2) is used.

```
<Page>
  <Paragraph Entity="True">
    <Line Entity="True">
      <Word Entity="True">
        <Character Entity="True">
        </Character>
      </Word>
    </Line>
  </Paragraph>
  <Figure Entity="True">
  </Figure>
  <Table Entity="True">
  </Table>
</Page>
```

6 Multilingual Features

6.1 Multilingual Text Data

Java programs running on JDK1.1 or JDK1.2 can display any Unicode [Con97] character which can be rendered with a host font. TrueViz displays multilingual text using Java Unicode facilities (see Figure 8). TrueViz can read Unicode characters from the XML

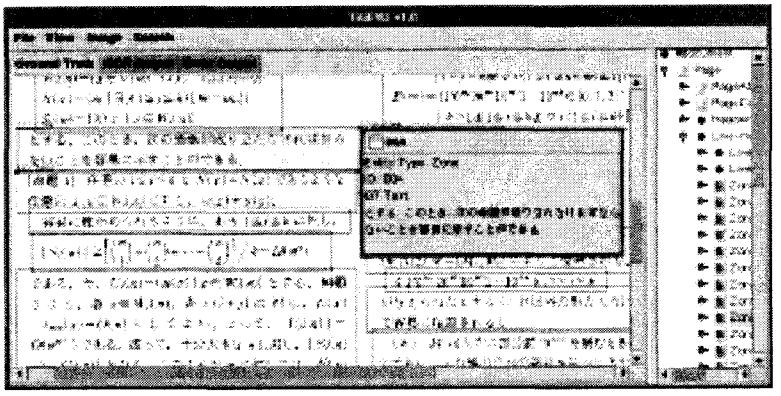


Figure 8: Infopanel and multilingual display.

file, and saves the XML file in the Unicode UTF8 format [Con97]. However Java does not provide a multilingual input method. We therefore developed such a method, which is described in Section 6.2.

6.2 Multilingual Input System

TrueViz provides a multilingual input system. Some languages like Chinese, Japanese, or Korean use more characters than can be input by a regular keyboard. To handle such languages, a sequence of several characters needs to be typed to construct a single character. While this composition process is going on, the input system accepts the sequence of characters, and produces composed text and committed text. The composed text is the intermediate text which is being processed to produce the intended text. The final text is called committed text (see Figure 10). Input capabilities for various languages can be easily added using this common interface. In addition to the default English language input, Russian input is also currently implemented. The input system can be used anywhere multilingual text input is needed (see Figure 9). For example, TrueViz supports multilingual text input in the search window for multilingual search. For people who are not familiar with the keyboard mapping, TrueViz provides a keyboard mapping display. In addition to keyboard input, TrueViz provides Unicode character input using a code table, so that any Unicode character can be selected and inserted into a text.

6.3 Adding New Input Capabilities

Input capabilities for various languages can be easily added using the common interface *did.gui.DIDInputMethod*. A new input capability can be added by implementing the following member functions of the interface.

public DIDKeyBDisplay getKeyBDisplay(): The function for getting the keyboard mapping display.

public String getComposingText(): The function for getting the current composed text.

(a) (b)

Figure 9: Multilingual input. (a) Russian input in search. For users who are not familiar with the keyboard mapping, a keyboard mapping display window is provided. (b) Russian input in groundtruth editor.

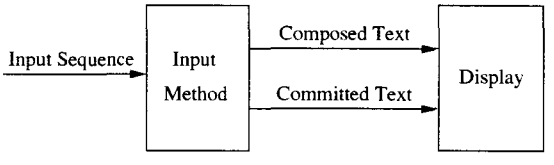


Figure 10: Input system.

public String getCommittedText(): The function for getting the current committed text.

public void keyTyped(char ch): The function for sending a typed character to the input.

public void showKeyboard(): The function for showing keyboard mappings.

7 Future Directions

TrueViz provides basic OCR groundtruthing functionalities. We hope that researchers in the international community will volunteer to add other features.

Currently TrueViz provides only English and Russian input. Other languages such as Korean, Japanese, and Chinese would be useful in multilingual OCR. A public-domain Java package with keyboards for various scripts/languages [LLW98a, LLW98b] that could be incorporated into TrueViz would be of great benefit to researchers.

Since tables are not trees, existing XML validation programs cannot verify table groundtruth data. Thus convenient ways for representing, annotating, and validating tables is needed.

Converters for DAFS to XML and XML to DAFS are currently implemented. This makes the XML representation compatible with the public domain performance evaluation toolkit PSET [MK00c, MK00b, MK01, MK00a], and allows researchers to visualize segmentation evaluation results using TrueViz. Converters from SGT format (produced by the Pink Panther groundtruthing tool), XDOC (the Xerox representation for groundtruth), and the Caere representation would be helpful.

Document images contain huge amounts of data, and XML files can require more disk space than binary-formatted files. If compressed XML files could be saved and read, the file size of the XML files would not be a concern.

Zooming is a very integral part of any document image groundtruth visualization tool. A more "zoom-centric" design using the zoomable user interface package Jazz [BMG00] could be explored.

TrueViz was tested by several members of our research group. A more thorough quantitative user evaluation using questionnaires would be desirable [CDN88].

Since the OCR community currently does not have annotation standards similar to the Corpus Encoding Standard [Exp], it would be beneficial to start a working group to build such a standard and also ensure that TrueViz is compatible with this new encoding standard.

Acknowledgments

The authors would like to thank the participants of MLOCR99 for valuable discussions; Jeff Czorapinski and Ivan Bella for their help in the initial phases of this project; Song Mao for discussion and user testing; Ben Bederson for comments on the user interface; Thomas Baby for suggesting the method of handling flexible entity structures; and Azriel Rosenfeld for editorial comments.

This research was funded in part by the Department of Defense under Contract MDA0949-6C-1250, Lockheed Martin under Contract 9802167270, the Defense Advanced Research Projects Agency under Contract N660010028910, and the National Science Foundation under Grant IIS9987944.

References

[AS94] C. Ahlberg and B. Shneiderman. Visual information seeking: Tight coupling of dynamic query filters with starfield displays. In *Proceedings of the ACM CHI94 Conference*, pages 313–317, Boston, MA, April 1994. `http://www.spotfire.com/products/spotfire_net.asp`.

[AS99] R. B. Allen and J. Schalow. Metadata and data structures for the historical newspaper digital library. In *Proceedings of the Eighth International Conference on Information Knowledge Management*, pages 147–153, Kansas City, MO, November 1999.

[AWC+95] S. D. Anderson, D. L. Westbrook, A. Carlson, D. M. Hart, and P. R. Cohen. *Common Lisp Analytical Statistics Package: User Manual.* University of Massachusetts, 1995. `http://eksl-www.cs.umass.edu/clasp.html`.

[BDH+00] S. Bird, D. Day, J. Garofolo J. Henderson, C. Laptun, and M. Liberman. ATLAS: A flexible and extensible architecture for linguistic annotation. In *Proceedings of the Second International Language Resources and Evaluation Conference*, pages 1699–1706, Athens, Greece, May 2000.

[BGWL98] C. Barras, E. Geoffrois, Z. Wu, and M. Liberman. Transcriber: A free tool for segmenting, labeling and transcribing speech. In *Proceedings of the First International Conference on Language Resources and Evaluation*, pages 1373–1376, Granada, Spain, May 1998.

[BGWL00] C. Barras, E. Geoffrois, Z. Wu, and M. Liberman. Transcriber: development and use of a tool for assisting speech corpora production. *Speech Communication special issue on Speech Annotation and Corpus Tools*, 33(1-2), January 2000.

[BMG00] B. Bederson, J. Meyer, and L. Good. Jazz: An extensible zoomable user interface graphics toolkit in Java. Technical Report CS-TR-4137, UMIACS-TR-2000-30, University of Maryland, College Park, MD, May 2000. `http://www.cs.umd.edu/hcil/jazz/`.

[BPSM98] T. Bray, J. Paoli, and C. M. Sperberg-McQueen. *Extensible Markup Language (XML).* W3C, 1998. `http://www.w3.org/TR/REC-xml`.

[CDN88] J. P. Chin, V. A. Diehl, and K. L. Norman. Development of an instrument measuring user satisfaction of the human-computer interface. In *Proceedings of SIGCHI '88*, pages 213–218, New York, NY, October 1988. `http://www.lap.umd.edu/QUIS/index.html`.

[Con97] The Unicode Consortium. *The Unicode Standard, Version 2.0.* Addison Wesley Developers Press, 1997.

[DAH+97] D. Day, J. Aberdeen, L. Hirschman, R. Kozierok, P. Robinson, and M. Vilain. Mixed-initiative development of language processing systems. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, Washington, DC, March-April 1997.

[dli99] *Proceedings of the IAPR Workshop on Document Layout Interpretation and its Applications*, Bangalore, India, September 1999.

[DM00] D. Doermann and D. Mihalcik. Tools and techniques for video performance evaluation. In *Proceedings of the 15th International Conference on Pattern Recognition*, pages 167–170, Barcelona, Spain, September 2000. `http://documents.cfar.umd.edu/LAMP/Media/Projects/ViPER/`.

[Exp] Expert Advisory Group on Language Engineering Standards. *Corpus Encoding Standard - Document CES 1, Version 1.5.* `http://www.cs.vassar.edu/CES/`.

[Fru95] T. Fruchterman. DAFS: A standard for document and image understanding. In *Proceedings of the Symposium on Document Image Understanding Technology*, pages 94–100, Bowie, MD, October 1995.

[GBBW00] E. Geoffrois, C. Barras, S. Bird, and Z. Wu. Transcribing with annotation graphs. In *Proceedings of the Second International Conference on Language Resources and Evaluation*, pages 1517–1521, Athens, Greece, May-June 2000.

[Gus97] D. Gusfield. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology.* Cambridge University Press, 1997.

[Har99] E. R. Harold. *XML Bible.* IDG Books, Foster City, CA, 1999.

[HRI00] O. Hitz, L. Robadey, and R. Ingold. An architecture for editing document recognition results using XML technology. In *Proceedings of the Fourth IAPR International Workshop on Document Analysis Systems*, pages 385–396, Rio de Janeiro, Brazil, December 2000.

[KA99] T. Kanungo and R. B. Allen. Full-text access to historical newspapers. Technical Report CS-TR-4014, Laboratory for Language and Media Processing, University of Maryland, College Park, MD, April 1999.

[KLCB01] T. Kanungo, C. H. Lee, J. Czorapinski, and I. Bella. TRUEVIZ: a groundtruth/metadata editing and visualizing toolkit for OCR. In *Proceedings of the SPIE Conference on Document Recognition and Retrieval*, pages 1–12, San Jose, CA, January 2001.

[LK00] C. H. Lee and T. Kanungo. *TRUEVIZ User's Manual*, August 2000.

[LLW98a] K. Y. Leong, H. Liu, and O. P. Wu. Java input method engine. In *Proceedings of the Seventh International World Wide Web Conference*, Brisbane, Australia, April 1998. `http://www7.scu.edu.au/programme/fullpapers/1915/com1915.htm`.

[LLW98b] K. Y. Leong, H. Liu, and O. P. Wu. Web internationalization and Java keyboard input methods. In *Proceedings of INET 98*, pages 21–24, Geneva, Switzerland, July 1998.

[McL00] B. McLaughlin. *Java and XML.* O'Reilly, Sebastopol, CA, 2000.

[MK00a] S. Mao and T. Kanungo. Empirical performance evaluation of page segmentation algorithms. In *Proceedings of the SPIE Conference on Document Recognition and Retrieval*, pages 303–314, January 2000.

[MK00b] S. Mao and T. Kanungo. PSET: A page segmentation evaluation toolkit. In *Fourth IAPR International Workshop on Document Analysis Systems*, pages 451–462, Rio de Janeiro, Brazil, December 2000.

[MK00c] S. Mao and T. Kanungo. Software architecture of PSET: A page segmentation evaluation toolkit. Technical Report CAR-TR-955, University of Maryland, College Park, MD, September 2000. `http://www.cfar.umd.edu/~kanungo/pubs/trpset.ps`. Software is available at `http://www.cfar.umd.edu/~kanungo/software/software.html`.

[MK01] S. Mao and T. Kanungo. Empirical performance evaluation methodology and its application to page segmentation algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):242–256, 2001. (To appear.).

[mlo99] Working group notes, international workshop on performance evaluation issues in multilingual OCR, September 1999. `http://www.cfar.umd.edu/~kanungo/workshop/reco.html`.

[Nat] National Center for Atmospheric Research/Mesoscale and Microscale Meteorology. *PolyPaint User Manual, Version 3.0.* `http://lasp.colorado.edu/polypaint/home.html`.

[PIX] PIXOTEC, LLC. *Slicer Dicer.* `http://www.slicerdicer.com/`.

[SCB92] D. F. Swayne, D. Cook, and A. Buja. XGobi: interactive dynamic graphics in the X window system with a link to S. In *Proceedings of the American Statistical Association Meetings*, 1992. `http://www.research.att.com/areas/stat/xgobi/`.

[SCB98] D. F. Swayne, D. Cook, and A. Buja. Xgobi: Interactive dynamic data visualization in the X window system. *Journal of Computational and Graphical Statistics*, 7, 1998. `http://www.research.att.com/areas/ stat/xgobi/`.

[SHB91] D. F. Swayne, N. Hubbell, and A. Buja. XGobi meets S: Integrating software for data analysis. In *Proceedings of the Symposium on the Interface*, pages 430–434, 1991. `http://www.research.att.com/areas/stat/xgobi/`.

[Shn96] B. Shneiderman. The eyes have it: A task by data type taxonomy of information visualizations. In *Proceedings of the IEEE Symposium on Visual Languages*, pages 336–343, September 1996. `http://otal.umd.edu/Olive/`.

[Shn98] B. Shneiderman. *Designing the User Interface.* Addison Wesley, Reading, MA, 1998.

[SK98] J. Sauvola and H. Kauniskangas. *MediaTeam Oulu Document Database.* MediaTeam, University of Oulu, Finland, 1998. `http://www.mediateam.oulu.fi/MTDB/`.

[Uni] University of Augsburg. *Mondrian.* `http://jetta.math.uni-augsburg.de/Mondrian/`.

[VR99] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S-Plus.* Springer, Berlin, 1999. `http:// www.splus.mathsoft.com/`.

[WHA+99] L. Wood, A. L. Hors, V. Apparao, L. Cable, M. Champion, J. Kesselman, P. L. Hegaret, T. Pixley, J. Robie, P. Sharpe, and C. Wilson. *Document Object Model (DOM).* W3C, 1999. `http://www.w3.org/TR/DOM-Level-2/`.

[YV98] B. A. Yanikoglu and L. Vincent. Pink Panther: A complete environment for ground-truthing and benchmarking document page segmentation. *Pattern Recognition*, 31:1191–1204, 1998.

Software Architecture of PSET: A Page Segmentation Evaluation Toolkit

Song Mao and Tapas Kanungo

Language and Media Processing Laboratory
Center for Automation Research
University of Maryland, College Park, MD

Abstract

Empirical performance evaluation of page segmentation algorithms has become increasingly important due to the numerous algorithms that are being proposed each year. In order to choose between these algorithms for a specific domain it is important to empirically evaluate their performance. To accomplish this task the document image analysis community needs i) standardized document image datasets with groundtruth, ii) evaluation metrics that are agreed upon by researchers, and iii) freely available software for evaluating new algorithms and replicating other researchers' results.

In an earlier paper (SPIE Document Recognition and Retrieval 2000) we published evaluation results for various popular page segmentation algorithms using the University of Washington dataset. In this paper we describe the software architecture of the PSET evaluation package, which was used to evaluate the segmentation algorithms. The description of the architecture will allow researchers to understand the software better, replicate our results, evaluate new algorithms, experiment with new metrics and datasets, etc. The software is written using the C language on the SUN/UNIX platform and is being made available to researchers at no cost.

This research was funded in part by the Department of Defense under Contract MDA 9049-6C-1250, Lockheed Martin under Contract 9802167270, the Defense Advanced Research Projects Agency under Cooperative Agreement N660010028910, and the National Science Foundation under Contract IIS9987944.

1 Introduction

It is important to quantitatively monitor progress in any scientific field. The information retrieval community and the speech recognition community, for example, have yearly competitions in which researchers evaluate their latest algorithms on clearly defined tasks, datasets, and metrics. To make such evaluations possible, researchers have access to standardized datasets, metrics, and freely available software for scoring the results produced by algorithms [18, 1].

In the Document Image Analysis area, regular evaluations of OCR accuracy have been conducted by UNLV [3]. Page segmentation algorithms, which are crucial components of OCR systems, were at one time evaluated by UNLV based on the final OCR results, but not on the geometric results of the segmentation. Recently [14], we empirically compared various commercial and research page segmentation algorithms, using the University of Washington dataset. We used a well-defined (geometric) line-based metric and a sound statistical methodology to score the segmentation results. Furthermore, unlike the UNLV evaluations, we trained the segmentation algorithms prior to evaluating them.

In this paper we describe in detail the software architecture of the package called PSET, which we used in [14] to evaluate page segmentation algorithms. This package was developed by us at the University of Maryland and will be made available to researchers at no cost. Publication of the package will allow researchers to implement our five-step evaluation methodology and evaluate their own algorithms.

Software architecture can be described using methods such as Petri Nets and Data Flow Diagrams [8]. We describe the architecture of PSET, the I/O file formats, etc., using Object-Process Diagrams (OPDs) [5], which are similar in spirit to Petri Nets.

The package, called the Page Segmentation Evaluation Toolkit (PSET), is modular, written using the C language, and runs on the SUN/UNIX platform. The software has been structured so that it can be used at the UNIX command line level or compiled into other software packages by calling API functions. The description in this paper will aid users in using, updating, and modifying the PSET package. It will also help users to add new algorithm modules to the package and to interface it with other software tools and packages. The PSET package includes three research page segmentation algorithms; [1] a textline-based benchmarking algorithm; and a Simplex-based optimization algorithm for estimating algorithm parameters from training datasets.

This paper is organized as follows. In Section 2, we discuss the page segmentation problem. In Section 3, we present our five-step page segmentation performance evaluation methodology. In Section 4, we describe the architecture and file formats of our PSET package in detail and show how to implement each step of our five-step performance evaluation methodology. In Section 5, we give the hardware and software requirements for using the PSET package. In Section 6, we discuss our future work. Finally in Section 7, we give a summary of the article. A detailed description of our textline-based metric is given in an Appendix for completeness.

[1] We implemented the X-Y cut algorithm [15] and the Docstrum algorithm [16]. Kise [11] provided us the C implementation of his Voronoi-based algorithm.

2 The Page Segmentation Problem

There are two types of page segmentation, physical and logical. Physical page segmentation is a a process of dividing a document page into homogeneous zones. Each of these zones can contain one type of object. These objects can be of type text, table, figure, halftone image, etc. Logical page segmentation is a process of assigning logical relations to physical zones. For example, reading order labels order the physical zones in the order in which they should be read. Similarly, assigning section and sub-section labels to physical zones creates a hierarchical document structure. In this paper, we focus on physical page segmentation and refer to it simply as page segmentation hereafter.

Page segmentation is a crucial preprocessing step for an OCR system. In many cases, OCR engine recognition accuracy depends heavily on page segmentation accuracy. For instance, if a page segmentation algorithm merges two text zones horizontally, the OCR engine will recognize text across text zones and hence generate unreadable text. Page segmentation algorithms can be categorized into three types: top-down, bottom-up, and hybrid approaches. Top-down approaches iteratively divide a document page into smaller zones according to some criterion. The X-Y cut algorithm developed by Nagy *et al.* [15] is a typical top-down algorithm. Bottom-up approaches start from document image pixels, and iteratively group them into bigger regions. The Docstrum algorithm of O'Gorman [16] and the Voronoi-based algorithm of Kise *et al.* [11] are representative bottom-up approaches. Hybrid approaches are usually a mixture of top-down and bottom-up approaches. The algorithm of Pavilidis and Zhou [17] is an example of the hybrid approach that employs a split-and-merge strategy.

3 Performance Evaluation Methodology

In order to objectively evaluate page segmentation algorithms, a performance evaluation methodology should take into consideration the performance metric, the dataset, the training and testing methods, and the methodology of analyzing experimental results. In this section, we introduce a five-step methodology that we proposed earlier [14, 12, 13]. The PSET package is an implementation of this methodology.

Let $\mathcal{D}$ be a given dataset containing (document image, groundtruth) pairs (I, G), and let $\mathcal{T}$ and $\mathcal{S}$ be a training dataset and a test dataset respectively. The five-step methodology is described as follows:

1. Randomly divide the dataset $\mathcal{D}$ into two mutually exclusive datasets: a training dataset $\mathcal{T}$ and a test dataset $\mathcal{S}$. Thus, $\mathcal{D} = \mathcal{T} \cup \mathcal{S}$ and $\mathcal{T} \cap \mathcal{S} = \phi$, where ϕ is the empty set.

2. Define a computable performance metric $\rho(I, G, R)$. Here I is a document image, G is the groundtruth of I, and R is the OCR segmentation result on I. In our case, $\rho(I, G, R)$ is defined as textline accuracy, as described in the Appendix.

3. Given a segmentation algorithm A with a parameter vector $\mathbf{p}^A$, automatically search for the optimal parameter value $\hat{\mathbf{p}}^A$ for which an objective function $f(\mathbf{p}^A; \mathcal{T}, \rho, A)$

assumes the optimal value on the training dataset $\mathcal{T}$. In our case, this objective function is defined as the average textline error rate on a given training dataset:

$$f(\mathbf{p}^A; \mathcal{T}, A, \rho) = \frac{1}{\#\mathcal{T}} \left[\sum_{(I,G)\in\mathcal{T}} 1 - \rho(G, Seg_A(I, \mathbf{p}^A)) \right].$$

4. Evaluate the segmentation algorithm A with the optimal parameter $\hat{\mathbf{p}}^A$ on the test dataset $\mathcal{S}$ by

$$\Phi\left(\{\rho(G, Seg_A(I, \hat{\mathbf{p}}^A))|(I,G) \in \mathcal{S}\}\right)$$

where Φ is a function of the performance metric ρ on each (document image, groundtruth) pair (I,G) in the test dataset $\mathcal{S}$, and $Seg_A(\cdot,\cdot)$ is the segmentation function corresponding to A. The function Φ is defined by the user. In our case,

$$\Phi\left(\{\rho(G, Seg_A(I, \hat{\mathbf{p}}^A))|(I,G) \in \mathcal{S}\}\right) = 1 - f(\hat{\mathbf{p}}^A; \mathcal{S}, \rho, A),$$

which is the average of the textline accuracy $\rho(G, Seg_A(I, \hat{\mathbf{p}}^A))$ achieved on each (document image, groundtruth) pair (I,G) in the test dataset $\mathcal{S}$.

5. Perform a statistical analysis to evaluate the statistical significance of the evaluation results, and analyze the errors to identify/hypothesize why the algorithms perform at their respective levels.

4 Architecture, File Formats, and Evaluation Methodology

In this section, we first describe the software architecture of the PSET package and the formats of the files used to communicate with the package. Next we show how this software package can be used to implement the five steps of the page segmentation evaluation methodology described in Section 3. Generic file format descriptions as well as specific examples are provided, for clearer understanding. This description of the architecture and file formats will allow users to i) understand the working of the PSET package, ii) replicate our results, iii) modify the parameter files for datasets, metrics, etc., and conduct their own evaluation experiments, iv) understand, maintain and improve the software, and v) evaluate new algorithms and compare the results with existing algorithms. The PSET package has been used to evaluate five page segmentation algorithms [14, 13].

4.1 Architecture and File Formats

The PSET package can be used to i) automatically train a given page segmentation algorithm, i.e., automatically select optimal algorithm parameters on a given training dataset, and ii) evaluate the page segmentation algorithm with the optimal parameters found in i) on a given test dataset. Figure 1 shows the overall architecture of the PSET package and illustrates these two functionalities.

The overall architecture shows all the input files that are needed to conduct the training and testing experiments for a given page segmentation algorithm, and all the

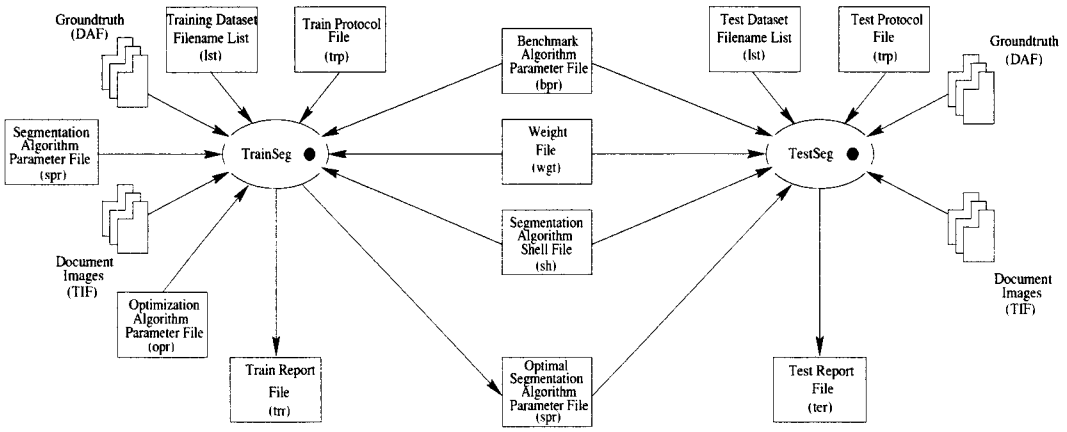


Figure 1: Overall PSET architecture. The left half of the architecture represents the training phase; the right half represents the testing phase. Note that in the testing phase, the optimal page segmentation parameter found in the training phase is used. The training and testing phases use the same performance metric related input files (benchmark algorithm parameter file (bpr) and weight file (wgt)) and the same segmentation algorithm shell file (sh).

Table 1: Summary of the file formats in the PSET package.

File Type	Extension	Description
Dataset List File	lst	It saves the root name of each image in a dataset.
Train Protocol File	trp	It saves the protocol parameters of the training experiment.
Test Protocol File	tep	It saves the protocol parameters of the testing experiment.
Segmentation Algorithm Parameter File	spr	It saves the parameters of a page segmentation algorithm that are to be trained.
Benchmarking Algorithm Parameter File	bpr	It saves all parameters of a benchmarking algorithm.
Optimization Algorithm Parameter File	opr	It saves all parameters of an optimization algorithm.
Groundtruth File	DAF	It saves document images and their groundtruth information.
Segmentation Result File	dafs	It saves document images and their segmentation results.
Train Report File	trr	It saves the training result of a segmentation algorithm.
Test Report File	ter	It saves the test result of a segmentation algorithm.
Weight File	wgt	It saves a set of weights for a set of error measures.
Segmentation Algorithm Shell File	sh	It saves a shell command for running segmentation algorithm executable. It is a Bourn shell program.

output files generated by the training and testing procedures. Table 1 lists all the files used, their purposes, and their file name extensions.

Input files include various initial algorithm parameter files (an optimization algorithm parameter file (opr), a page segmentation algorithm parameter file (spr), and a benchmark algorithm parameter file (bpr)), dataset files (lst), a shell file (sh), and experimental protocol files (training protocol file (trp) and test protocol file (tep)). Users need to provide these files to the PSET package to conduct training or testing experiments. The output files of the training phase include a training report file (trr) and an optimal segmentation algorithm parameter file (spr). The training report file (trr) records intermediate as well as final training results of the training experiment. The optimal segmentation algorithm parameter file (spr) records the optimal segmentation algorithm parameter values found in the training phase. The output of the testing phase is a testing

report file (ter), which records a set of error measures, timing and performance scores for each image in the test dataset, and a final average performance score over all images in the test dataset. Figure 2 shows various input file formats. Figure 3 shows the training report file format and Figure 4 shows the test report file format.

```
# [comments]

DATASET             = <dataset file name>
GROUNDTRUTH_DIR     = <groundtruth directory name>
IMG_DIR             = <image directory name>
GT_SUFFIX           = <groundtruth file suffix>
SG_SUFFIX           = <segmentation result file suffix>
IMG_SUFFIX          = <image file suffix>
TRAIN_RESULT_DIR    = <training result file location>
OPT_ALG             = <optimization algorithm name>
BEN_ALG             = <benchmark algorithm name>
SEG_ALG             = <page segmentation algorithm name>
```

(a)

```
# [comments]

DATASET             = <testing dataset file name>
GROUNDTRUTH_DIR     = <groundtruth directory name>
IMG_DIR             = <image directory name>
GT_SUFFIX           = <groundtruth file suffix>
SG_SUFFIX           = <segmentation result file suffix>
IMG_SUFFIX          = <image file suffix>
TEST_RESULT_DIR     = <testing result file location>
BEN_ALG             = <benchmark algorithm name>
SEG_ALG             = <page segmentation algorithm name>
```

(b)

```
# [comments]
<parameter 1 name>   =   <value>
<parameter 2 name>   =   <value>
        .            =       .
        .            =       .
        .            =       .
<parameter N name>   =   <value>
```

(c)

File Attribute Name	Description
DATASET	The filename of a list file that saves the root name of each image in a dataset.
GROUNDTRUTH_DIR	The location of the groundtruth files.
IMG_DIR	The location of the image files.
GT_SUFFIX	The suffix of a groundtruth filename, e.g. the suffix of groundtruth file "A001.DAF" is ".DAF".
SG_SUFFIX	The suffix of a segmentation result filename, e.g. the suffix of segmentation result file "A001.dafs" is ".dafs".
IMG_SUFFIX	The suffix of an image filename, e.g. the suffix of image file "A001BIN.TIF" is "BIN.TIF".
TRAIN_RESULT_DIR	The location of the training result files generated by a training experiment.
TEST_RESULT_DIR	The location of the testing result files generated by a test experiment.
OPT_ALG	The name of the optimization algorithm that is to be used.
BEN_ALG	The name of the benchmarking algorithm that is to be used.
SEG_ALG	The name of the page segmentation algorithm that is to be used.

(d)

Figure 2: Input file formats. The training protocol file format is shown in (a), the test protocol file format is shown in (b), and the algorithm parameter file format is shown in (c). The description of the attributes in (a) and (b) is given in (d).

The parameter values in the parameter files are first read into the corresponding data structures inside the TrainSeg and the TestSeg modules as shown in Figure 5. The Train module shown in Figure 5(a) is shown at a finer level of detail in Figure 6, where the interaction of the optimization algorithm and the objective function computation module is illustrated. A detailed view of the *Objective Function Genscore* showing the interaction between the segmentation algorithm module and the performance metric computation module is shown in Figure 7(a). Finally, a blown-up view of the Test module shown in Figure 5(b) is shown in Figure 7 (b).

4.2 Implementing the Evaluation Methodology

In this section we show how a user can implement each step of the five-step evaluation methodology described in Section 3. Each variable in the methodology is mapped to a specific parameter file and each step is mapped to a specific group of modules in the package.

1. The training dataset $\mathcal{T}$ is specified in the image root name list file (lst). The file name and location of the list file and the location of the image and groundtruth files

[experimental environments]
#

# Feval	p[1]	p[2]	. . .	p[n]	score	timing	plow[1]	plow[2]	. . .	plow[n]	Flow
1	<data>	<data>	. . .	<data>	<data>	<data>	<data>	<data>	. . .	<data>	<data>
2	<data>	<data>	. . .	<data>	<data>	<data>	<data>	<data>	. . .	<data>	<data>
.	.	.	. . .	.	.	.	.	.	. . .	.	.
.	.	.	. . .	.	.	.	.	.	. . .	.	.
.	.	.	. . .	.	.	.	.	.	. . .	.	.
M	<data>	<data>	. . .	<data>	<data>	<data>	<data>	<data>	. . .	<data>	<data>

Optimal_Parameter_Vector = <param 1> <param 2> . . . <param N>
Optimal_Performance_Value = <data>

End of the training.

(a)

Item Name	Description
Feval	Number of objective function evaluations.
p[1], p[2], ... , p[n]	Current objective function parameter vector value; here the objective function parameter vector is the page segmentation parameter vector being trained. n is the dimensionality of the parameter vector.
score	Current performance measure, in this case, textline error rate.
timing	The time it takes to obtain the current score.
plow[1], plow[2], ... , plow[n]	The objective function parameter vector value that gives the best score so far.
Flow	The best score so far — in this case, the minimum textline error rate so far.

(b)

Figure 3: The training report file format. The format is shown in (a) and the description of each column entry in (a) is shown in (b).

<experimental environments>
#

#Img	nSpl	nMrg	nFA	nSplL	nMrgL	nMisL	nErrL	nGtL	score	timing
<img_root_name 1>	<data>	<data>	<data>	<data>	<data>	<data>	<data>	<data>	<data>	<data>
<img_root_name 2>	<data>	<data>	<data>	<data>	<data>	<data>	<data>	<data>	<data>	<data>
.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.
<img_root_name M>	<data>	<data>	<data>	<data>	<data>	<data>	<data>	<data>	<data>	<data>

The average textline accuracy = <data>

End of testing.

(a)

Column Entry	Description
Img	The root name of the current image file.
nSpl	The number of split errors.
nMrg	The number of horizontal merge errors.
nFA	The number of false alarm errors.
nSplL	The number of split textlines.
nMrgL	The number of horizontally merged textlines.
nMisL	The number of mis-detected textlines.
nErrL	The number of error textlines (textlines that are either split, horizontally merged or mis-detected).
nGtl	The number of groundtruth textlines.
score	The performance measure (textline error rate) on current image.
timing	The time taken to obtain the score.

(b)

Figure 4: The test report file format. The format is shown in (a) and the description of each column entry in (a) is shown in (b).

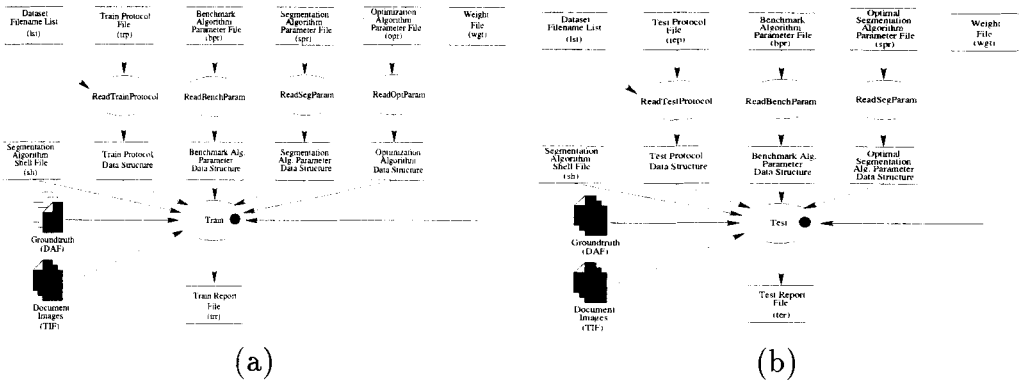


Figure 5: Parameter reading stage of the training phase (a) and the testing phase (b). At this level, various parameter files are read into their corresponding data structures which are fed into the Train and Test modules.

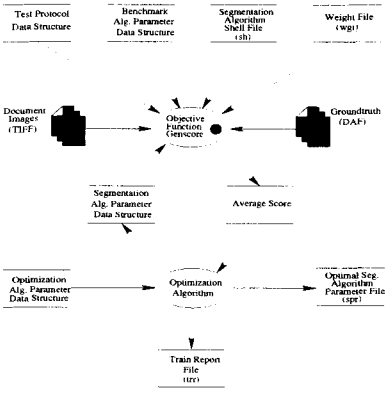
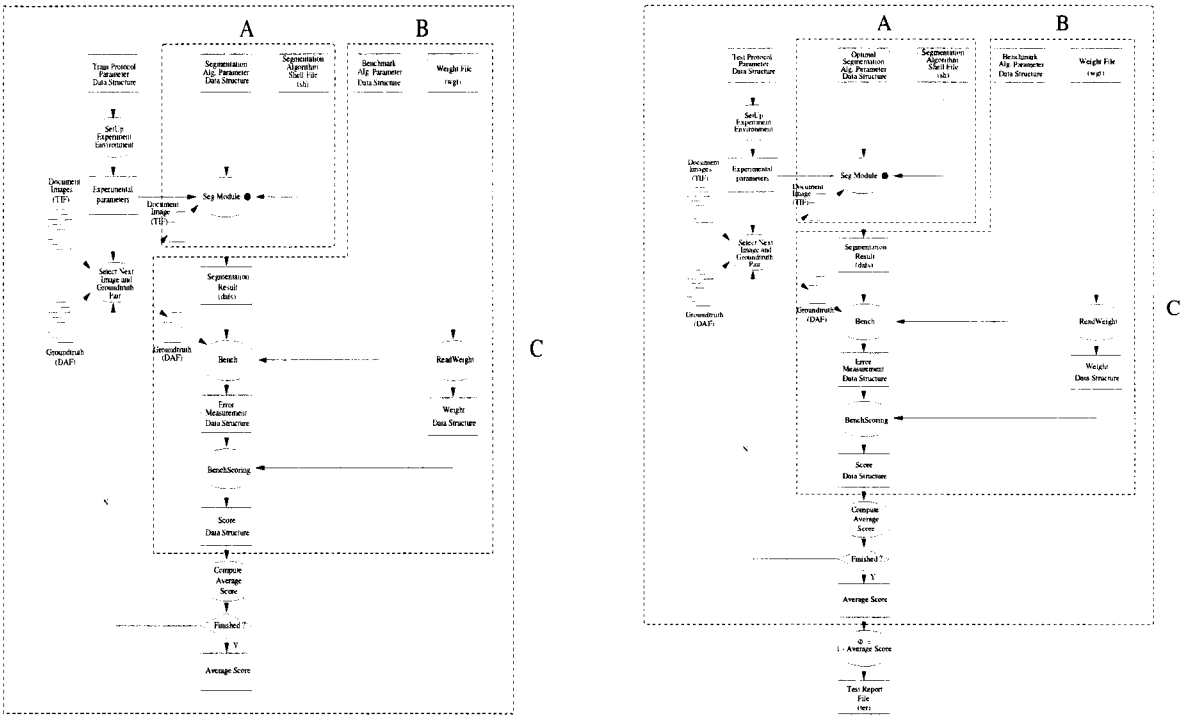


Figure 6: The Train module. In this module, the objective function is optimized over a given training dataset. Two files are generated by this module, a train report file (trr) and an optimal segmentation algorithm parameter file (spr).

are specified in the training protocol file (trp). This information is later read into the Train Protocol Parameter Data Structure as shown in Figure 5(a). Similarly, a test dataset $\mathcal{S}$ is specified in another image root name list file (lst). The file name and location of the list file and the location of image and groundtruth files are specified in the test protocol file (tep). This information is later read into the test protocol parameter data structure as shown in Figure 5(b). Other experimental protocol parameters such as file suffix and algorithms used are also specified in the training protocol file (trp) and test protocol file (tep). Figures 2(a) and (b) show generic formats for these two files and Figure 8 shows samples of these two files.

2. The performance metric $\rho(I, G, R)$ is computed in module B, shown in Figures 7(a) and (b). (I, G) is an (image, groundtruth) pair, which is represented by two single pages in the architecture, and R is the segmentation result file represented by Segmentation Result (dafs). The error counter algorithm for generating a set of error measures is implemented in the *Bench* module. In the *BenchScoring* module,



(a) (b)

Figure 7: Software architectures of the objective function module and the test module. Module A represents the page segmentation algorithm module, module B represents the page segmentation error counter and scoring module, and module C represents the objective function module. The test module in (b) has sub-modules similar to those in (a). It also has a module for computing a final testing performance score (average textline accuracy).

a weighted error measure $1 - \rho(I, G, R)$ is computed. The formal definitions of error measures and performance metrics are given in the Appendix. To compute a performance metric, two input files, a benchmark algorithm parameter file (bpr) and a weight file (wgt), are required. Examples of these two files are shown in Figure 13. Users can substitute their own performance metrics and error counters in place of these two modules. However, this also requires that the users write a new *ReadBenchParam* module and define a new benchmark algorithm parameter data structure as shown in Figure 5.

3. The objective function $f(\mathbf{p}^A; \mathcal{T}, A, \rho)$ is represented by the module C in Figure 7(a), where page segmentation algorithm A is represented by module A, the training dataset $\mathcal{T}$ is specified in the train protocol parameter data structure, the computation of performance metric ρ is conducted in module B, and objective function parameter vector $\mathbf{p}^A$ is represented by the segmentation algorithm parameter data structure in the architecture. The optimization procedure is shown in Figure 6 in a simplified representation. In addition, a benchmark algorithm parameter file (bpr), weight file (wgt), shell file (sh), list file (lst), training protocol file (trp),

```
# Training experiment protocal
# By: Song Mao
# Feb. 21, 2000
# LAMP, UMCP

DATASET             = train.lst
GROUNDTRUTH_DIR     = /fs/mirak2/LAMP/UWIII/ENGLISH/LINEWORD/DAFS/
IMG_DIR             = /fs/mirak2/LAMP/UWIII/ENGLISH/LINEWORD/IMAGEBIN/
GT_SUFFIX           = .DAF
SG_SUFFIX           = .dafs
IMG_SUFFIX          = BIN.TIF
TRAIN_RESULT_DIR    = ./
OPT_ALG             = simplex
BEN_ALG             = textline_based
SEG_ALG             = docstrum
```

(a)

```
# Test experiment protocal
# By: Song Mao
# Feb. 21, 2000
# LAMP, UMCP

DATASET             = test.lst
GROUNDTRUTH_DIR     = /fs/mirak2/LAMP/UWIII/ENGLISH/LINEWORD/DAFS/
IMG_DIR             = /fs/mirak2/LAMP/UWIII/ENGLISH/LINEWORD/IMAGEBIN/
GT_SUFFIX           = .DAF
SG_SUFFIX           = .dafs
IMG_SUFFIX          = BIN.TIF
TEST_RESULT_DIR     = ./
BEN_ALG             = textline_based
SEG_ALG             = xycut
```

(b)

Figure 8: Sample protocol files. From both the train protocol file (a) and the test protocol file (b), we can see that the list files of the training dataset and test dataset are *train.lst* and *test.lst* respectively, the optimization algorithm used is the *Simplex* algorithm, the benchmarking algorithm used is the *Textline-based* algorithm, the page segmentation algorithm trained is the *Docstrum* algorithm, and the page segmentation algorithm tested is the *X-Y cut* algorithm. We can also find the locations of the groundtruth files, image files and training and test result files. Moreover, the suffixes for various files are given for file name manipulation in the PSET API.

optimization algorithm parameter file (opr) and segmentation algorithm parameter file (spr) are required to conduct objective function optimization. Samples of opr and spr are shown in Figure 9. The generic file format of these sample files is shown in Figure 2.

```
# The Simplex Optimization
# Algorithm Parameters
NDIM      =   4
CRIFLG    =   nelder-mead
NMAX      =   500
FTOL      =   0.000001
ALPHA     =   1.0
BETA      =   0.5
GAMMA     =   2.0
SIGMA     =   0.5
P         =   100,80,100,50
SCALE     =   20,20,20,20
```

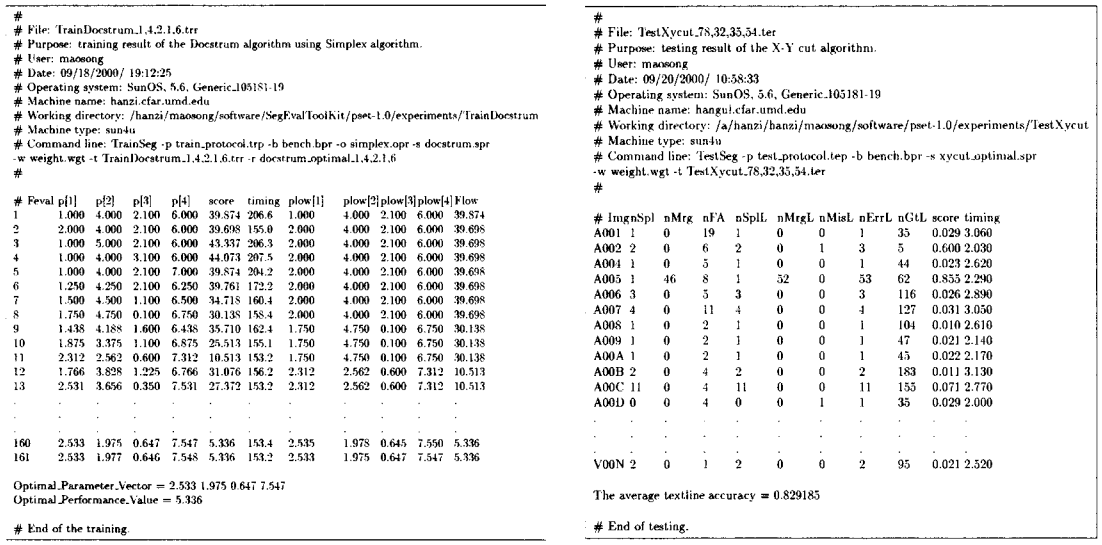
(a)

```
# The X-Y Cut Page Segmentation
# Algorithm Parameters
ALG_MODE    = func_call
TNX         = 100
TNY         = 80
TCX         = 100
TCY         = 50
```

(b)

Figure 9: Samples of an optimization algorithm parameter file (opr) and a segmentation algorithm parameter file (spr). A sample file for the Simplex optimization algorithm is shown in (a) and a sample file for the X-Y cut segmentation algorithm is shown in (b). Their detailed parameter descriptions can be found in [12].

The optimal objective function parameter vector $\hat{\mathbf{p}}^A$ is stored in the optimal segmentation algorithm parameter file (spr) shown in Figure 6. Users can substitute their own objective function in place of the architecture shown in Figure 7(a) and their own optimization algorithm module in the place of the *Optimization Algorithm* module shown in Figure 6. Again, they need to write new parameter reading functions and define corresponding data structures. This step generates two files,



```
#
# File: TrainDocstrum_1.4.2.1.6.trr
# Purpose: training result of the Docstrum algorithm using Simplex algorithm.
# User: maosong
# Date: 09/18/2000/ 19:12:25
# Operating system: SunOS, 5.6, Generic_105181-19
# Machine name: hanzi.cfar.umd.edu
# Working directory: /hanzi/maosong/software/SegEvalToolKit/pset-1.0/experiments/TrainDocstrum
# Machine type: sun4u
# Command line: TrainSeg -p train_protocol.trp -b bench.bpr -o simplex.opr -s docstrum.spr
-w weight.wgt -t TrainDocstrum_1.4.2.1.6.trr -r docstrum_optimal_1.4.2.1.6
#

# Feval p[1]    p[2]   p[3]   p[4]   score  timing  plow[1]   plow[2] plow[3] plow[4] Flow
1       1.000   4.000  2.100  6.000  39.874 206.6   1.000     4.000   2.100   6.000   39.874
2       2.000   4.000  2.100  6.000  39.698 155.0   2.000     4.000   2.100   6.000   39.698
3       1.000   5.000  2.100  6.000  43.337 206.3   2.000     4.000   2.100   6.000   39.698
4       1.000   4.000  3.100  6.000  44.073 207.5   2.000     4.000   2.100   6.000   39.698
5       1.000   4.000  2.100  7.000  39.874 204.2   2.000     4.000   2.100   6.000   39.698
6       1.250   4.250  2.100  6.250  39.761 172.2   2.000     4.000   2.100   6.000   39.698
7       1.500   4.500  1.100  6.500  34.718 160.4   2.000     4.000   2.100   6.000   39.698
8       1.750   4.750  0.100  6.750  30.138 158.4   2.000     4.000   2.100   6.000   39.698
9       1.438   4.188  1.600  6.438  35.710 162.4   1.750     4.750   0.100   6.750   30.138
10      1.875   3.375  1.100  6.875  25.513 155.1   1.750     4.750   0.100   6.750   30.138
11      2.312   2.562  0.600  7.312  10.513 153.2   1.750     4.750   0.100   6.750   30.138
12      1.766   3.828  1.225  6.766  31.076 156.2   2.312     2.562   0.600   7.312   10.513
13      2.531   3.656  0.350  7.531  27.372 153.2   2.312     2.562   0.600   7.312   10.513
.       .       .      .      .      .      .       .         .       .       .       .
.       .       .      .      .      .      .       .         .       .       .       .
.       .       .      .      .      .      .       .         .       .       .       .
160     2.533   1.975  0.647  7.547  5.336  153.4   2.535     1.978   0.645   7.550   5.336
161     2.533   1.977  0.646  7.548  5.336  153.2   2.533     1.975   0.647   7.547   5.336

Optimal_Parameter_Vector = 2.533 1.975 0.647 7.547
Optimal_Performance_Value = 5.336

# End of the training.
```

(a)

```
#
# File: TestXycut_78,32,35,54.ter
# Purpose: testing result of the X-Y cut algorithm.
# User: maosong
# Date: 09/20/2000/ 10:58:33
# Operating system: SunOS, 5.6, Generic_105181-19
# Machine name: hangul.cfar.umd.edu
# Working directory: /a/hanzi/hanzi/maosong/software/pset-1.0/experiments/TestXycut
# Machine type: sun4u
# Command line: TestSeg -p test_protocol.tep -b bench.bpr -s xycut_optimal.spr
-w weight.wgt -t TestXycut_78,32,35,54.ter
#

# ImgnSpl nMrg  nFA  nSplL  nMrgL  nMisL  nErrL  nGtL  score timing
A001  1   0     19   1      0      0      1      35    0.029 3.060
A002  2   0     6    2      0      1      3      5     0.600 2.030
A004  1   0     5    1      0      0      1      44    0.023 2.620
A005  1   46    8    1      52     0      53     62    0.855 2.290
A006  3   0     5    3      0      0      3      116   0.026 2.890
A007  4   0     11   4      0      0      4      127   0.031 3.050
A008  1   0     2    1      0      0      1      104   0.010 2.610
A009  1   0     2    1      0      0      1      47    0.021 2.140
A00A  1   0     2    1      0      0      1      45    0.022 2.170
A00B  2   0     4    2      0      0      2      183   0.011 3.130
A00C  11  0     4    11     0      0      11     155   0.071 2.770
A00D  0   0     4    0      0      1      1      35    0.029 2.000
.     .   .     .    .      .      .      .      .     .     .
.     .   .     .    .      .      .      .      .     .     .
.     .   .     .    .      .      .      .      .     .     .
V00N  2   0     1    2      0      0      2      95    0.021 2.520

The average textline accuracy = 0.829185

# End of testing.
```

(b)

Figure 10: Samples of a training report file format (a) and a test report file format (b). The comment lines provide experimental environment information about the training and test experiments. They are automatically generated by calling various GNU C functions. They are crucial for replicating experimental results. In the data area, both intermediate information and final results are recorded. This information can be used to analyze the convergence properties of the training process and to study the statistical significance of the test experiment results. A detailed description of each column entry can be found in Figure 3(b) and Figure 4(b).

a training report file (trr) and an optimal segmentation algorithm parameter file (spr). Figure 10(a) shows a sample training report file.

4. After the optimal objective function parameter vector $\hat{\mathbf{p}}^A$ has been found, the page segmentation algorithm is evaluated on a given test dataset $\mathcal{S}$. Figure 7(b) shows the architecture of the test procedure. The test dataset $\mathcal{S}$ is specified in the test protocol parameter data structure. Performance metric ρ is computed in module B. Note that module C here has the same architecture as module C in Figure 7(a). The computation of the final performance value Φ is represented in module Φ. Users can define their own Φ function by changing the *Bench, BenchScoring, Compute Average Score,* and Φ modules in Figure 7(b). This step generates a test report file (ter) which records a performance score for each image in the test dataset as well as a final average performance score over all images in the test dataset. Figure 10(b) shows a sample test report file.

5. The statistical analysis of the test experimental results can be conducted using a standard statistics software package such as S-PLUS [4] or SPSS [6].

4.3 Algorithm Calling Mode in the Segmentation Algorithm Module

An important feature of the PSET package is that there are two page segmentation algorithm calling modes: function call and shell call. If the source code of a segmentation algorithm is available as a function, the user can link the function into the training and testing modules. In many cases, however, source code of a segmentation algorithm is not available, but executable code is. In such cases the shell calling mode can be used to run the segmentation algorithm from within the training or testing module. Furthermore, if a segmentation algorithm source code is not well debugged, e.g., if it leaks memory after each function call, the leaked memory can accumulate after many function calls and can finally cause algorithm crash at some point. The shell call mode is a good solution to this problem since in this case the executable code is used, and after each call all leaked memory is freed. The disadvantage of the shell call mode is that it can be slower than the function call mode. Figure 12 shows the architecture of the software implementation of these two calling modes. A shell file is required in the page segmentation algorithm shell call mode. A sample shell file is shown in Figure 11.

```
#! /bin/sh

Docstrum -t $1 -p $2 -u $3 -d $4 $5 $6 $7
```

Figure 11: A sample shell file.

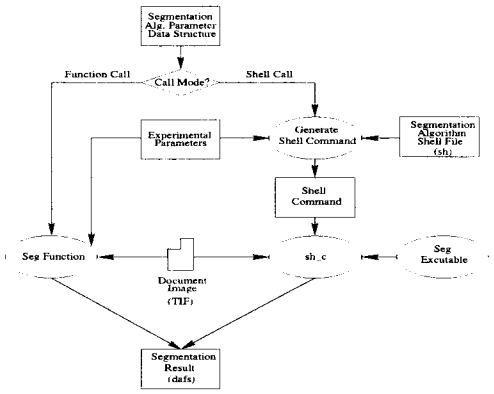


Figure 12: Page segmentation algorithm calling modes: function call and shell call. The left half represents the function calling mode and the right half represents the shell calling mode. The shell calling mode can be used only when the algorithm executable is available; otherwise the function calling mode can be used. Note that the executable is called by the function *sh_c*.

5 Hardware and Software Requirements

The PSET package has been developed in ANSI C on SUN Ultra 1, 2, and 5 workstations running the Solaris 2.6 operating system. The compiler used was GNU gcc 2.7.2. Two

public-domain libraries, DAFS and TIFF, were used in PSET and have been included in the distribution. The DAFS data structure library [7] was used for manipulating intermediate datatypes and the TIFF library [2] was used for image I/O.

6 Future Work

We are currently generalizing the PSET package to include i) other metrics, ii) other training/optimization algorithms, and iii) non-text region evaluation. Once the package is in the public domain, we expect that the international community will add other segmentation algorithms to the package. We are also porting the package to the Linux platform. A visualization tool called TRUEVIZ [10] that can display the segmentation and evaluation results of our PSET package is under development. For example, different types of errors can be visualized in various colors. TRUEVIZ can also be used for creating groundtruth for segmentation. Furthermore, we are developing an XML-based representation for zone groundtruth and intend to migrate to this representation from the current DAFS representation.

7 Summary

We have described the architecture and the file formats of a page segmentation evaluation toolkit (PSET). The overall architecture and the file formats were described to illustrate two major functionalities of the PSET package: i) automatically train a given page segmentation algorithm on a given training dataset and ii) evaluate the page segmentation algorithm with the optimal parameters found in i) on a given test dataset. The details of the architecture and samples of file formats were then described as an implementation of our five-step performance evaluation methodology. This paper is intended to assist users in understanding, using, updating and modifying the PSET package. It will also aid programmers who intend to add new algorithm modules to the package and interface it with other software tools.

A Textline-Based Error Measures and Error Metrics

In the following sections, we define page segmentation, a set of textline-based error measurements, and a performance metric that we used in our previous evaluation of page segmentation algorithms [14, 13], These definitions are based on set theory and mathematical morphology [9]. We then define a general metric that users can customize for their individual tasks.

A.1 Page Segmentation Definition

Let I be a document image, and let G be the groundtruth of I. Let $Z(G) = \{Z_q^G, q = 1, 2, \ldots, \#Z(G)\}$ be a set of groundtruth zones of document image I where $\#$ denotes the cardinality of a set. Let $L(Z_q^G) = \{l_{qj}^G, j = 1, 2, \ldots, \#L(Z_q^G)\}$ be the set of groundtruth textlines in groundtruth zone Z_q^G. Let the set of all groundtruth textlines in document image I be $\mathcal{L} = \cup_{q=1}^{\#Z(G)} L(Z_q^G)$. Let A be a given segmentation algorithm, and $Seg_A(\cdot, \cdot)$ be

the segmentation function corresponding to algorithm A. Let R be the segmentation result of algorithm A such that $R = Seg_A(I, \mathbf{p}^A)$ where $Z(R) = \{Z_k^R | k = 1, 2, \ldots, \#Z(R)\}$.

Let $D(\cdot) \subseteq \mathcal{Z}^2$ be the domain of its argument. The groundtruth zones and textlines have the following properties: 1) $D(Z_q^G) \cap D(Z_{q'}^G) = \phi$ for $Z_q^G, Z_{q'}^G \in Z(G)$ and $q \neq q'$, and 2) $D(l_i^G) \cap D(l_{i'}^G) = \phi$ for $l_i^G, l_{i'}^G \in \mathcal{L}$ and $i \neq i'$.

A.2 Error Measurements and Metric Definitions

In this section, we define four error measurements and a metric. Let $T_X, T_Y \in Z^+ \cup \{0\}$ be two length thresholds (in pixels) that determine if the overlap is significant or not. Each of these thresholds is defined in terms of an absolute threshold and a relative threshold. The absolute threshold is in pixels and the relative threshold is a percentage. T_X and T_Y are defined as follows:

$$T_X = \min\{HPIX, (100 - HTOL) \cdot h/100\} \quad (1)$$

$$T_Y = \min\{VPIX, (100 - VTOL) \cdot v/100\} \quad (2)$$

where $HPIX$ and $VPIX$ are the the two thresholds in pixels, $HTOL$ and $VTOL$ are the two thresholds in percentages, and h, v are the minimum width and height (in pixels) of two regions that are tested for significant overlap. Users must specify the $HTOL, VTOL, HPIX$ and $VPIX$ parameter values in the benchmark algorithm parameter file (bpr). Figure 13(b) shows a sample benchmark algorithm parameter file.

```
# The Textline-Based Benchmark
# Algorithm Parameters

HTOL    = 90
VTOL    = 80
HPIX    = 11
VPIX    = 8
```

(a)

```
# weight file

wSpl        =   0
wMrg        =   0
wMis        =   0
wFA         =   0
wSplLine    =   1
wMrgLine    =   1
wMisLine    =   1
wFAZone     =   0
```

(b)

Figure 13: Samples of a benchmark algorithm parameter file (bpr) (a) and a weight file (wgt) (b).

Let $E(T_X, T_Y) = \{e \in Z^2 | -T_X \leq X(e) \leq T_X, -T_Y \leq Y(e) \leq T_Y\}$ be a region of a rectangle centered at $(0, 0)$ with a width of $2T_X + 1$ pixels, and a height of $2T_Y + 1$ pixels where $X(\cdot)$ and $Y(\cdot)$ denote the X and Y coordinates of the argument, respectively. We now define two morphological operations: dilation and erosion [9]. Let $A, B \subseteq Z^2$. Morphological *dilation* of A by B is denoted by $A \oplus B$ and is defined as $A \oplus B = \{c \in Z^2 | c = a + b \text{ for some } a \in A, b \in B\}$. Morphological *erosion* of A by B is denoted by $A \ominus B$ and is defined as $A \ominus B = \{c \in Z^2 | c + b \in A \text{ for every } b \in B\}$.

We now define three types of textline based error measurements:

1) Groundtruth textlines that are missed:

$C_L = \left\{ l^G \in \mathcal{L} | D(l^G) \ominus E(T_X, T_Y) \right.$

$\subseteq (\cup_{Z^R \in Z(R)} D(Z^R))^c\Big\}$,

2) Groundtruth textlines whose bounding boxes are split:

$S_L = \Big\{ l^G \in \mathcal{L} | (D(l^G) \ominus E(T_X, T_Y)) \cap D(Z^R) \neq \phi,$
$(D(l^G) \ominus E(T_X, T_Y)) \cap (D(Z^R))^c \neq \phi,$
for some $Z^R \in Z(R)\Big\}$,

3) Groundtruth textlines that are horizontally merged:

$M_L = \Big\{ l^G_{qj} \in \mathcal{L} | \exists l^G_{q'j'} \in \mathcal{L}, Z^R \in Z(R), q \neq q',$
$Z^G_q, Z^G_{q'} \in Z(G)$ such that
$(D(l^G_{qj}) \ominus E(T_X, T_Y)) \cap D(Z^R) \neq \phi,$
$(D(l^G_{q'j'}) \ominus E(T_X, T_Y)) \cap D(Z^R) \neq \phi,$
$((D(l^G_{qj}) \ominus E(0, T_Y)) \oplus E(\infty, 0)) \cap D(Z^G_{q'}) \neq \phi,$
$((D(l^G_{q'j'}) \ominus E(0, T_Y)) \oplus E(\infty, 0)) \cap D(Z^G_q) \neq \phi\Big\}$.

4) Noise zones that are falsely detected (false alarm):

$$F_L = \Big\{ Z^R \in Z(R) | D(Z^R) \subseteq (\cup_{l^G \in \mathcal{L}} (D(l^G) \ominus E(T_x, T_Y)))^c \Big\}$$

Let the number of groundtruth error textlines be $\#\{C_L \cup S_L \cup M_L\}$ (mis-detected, split, or horizontally merged), and let the total number of groundtruth textlines be $\#\mathcal{L}$. We define the performance metric $\rho(I, G, R)$ as textline accuracy:

$$\rho(I, G, R) = \frac{\#\mathcal{L} - \#\{C_L \cup S_L \cup M_L\}}{\#\mathcal{L}}.$$

In the PSET package, we also define some other error measurements. Table 2 shows the error measurements, the metric defined in the PSET package, and the corresponding symbols used in the above discussion.

Table 2: Summary of error measurements and the corresponding symbols defined in this section.

Error Measure Defined in the PSET package	Equivalent Term in this Section	Description
nSpl	none	The number of split errors.
nMrg	none	The number of horizontal merge errors.
nFA	$\#F_L$	The number of false alarm errors.
nSplL	$\#S_L$	The number of split textlines.
nMrgL	$\#M_L$	The number of horizontally merged textlines.
nMisL	$\#C_L$	The number of mis-detected textlines.
nErrL	$\#\{C_L \cup S_L \cup M_L\}$	The number of error textlines (textlines that are either split, horizontally merged or mis-detected).
nGtl	$\#\mathcal{L}$	The number of groundtruth textlines.

In general, the performance metric can be any function of the error measures shown in Table 2. In the PSET package, a performance metric can be defined as a weighted sum of these error measures in function *BenchScoring*. Let *wSpl* be the weight of the error measurement *nSpl*. The weights of other error measurements are defined similarly. A general performance metric is defined as follows:

$$
\begin{aligned}
N &= wSpl * nSpl + wMrg * nMrg + wFA * nFA + wSplL * nSplL \\
&\quad + wMrgL * nMrgL + wMisL * nMisL, \\
D &= wSpl + wMrg + wFA + wSplL + wMrgL + wMisL, \\
\rho^*(I, G, R) &= \frac{N}{D}.
\end{aligned} \tag{3}
$$

Figure 14 gives a set of possible errors as well as an experimental example.

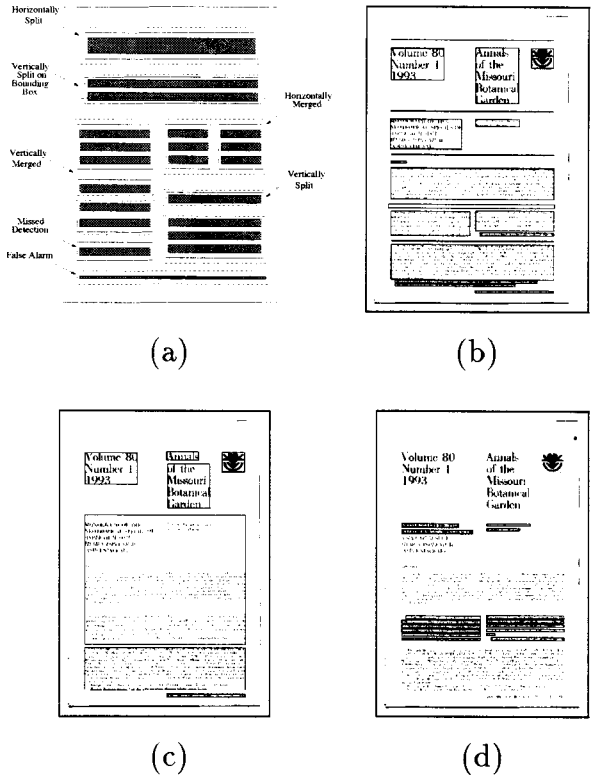


Figure 14: (a) This figure shows a set of possible textline errors. Solid-line rectangles denote groundtruth zones, dashed-line rectangles denote OCR segmentation zones, dark bars within groundtruth zones denote groundtruth textlines, and dark bars outside solid lines are noise blocks. (b) A document page image from the University of Washington III dataset with the groundtruth zones overlaid. (c) OCR segmentation result on the image in (b). (d) Segmentation error textlines. Notice that there are two horizontally merged zones just below the caption and two horizontally merged zones in the middle of the text body. In OCR output, horizontally split zones cause reading order errors whereas vertically split zones do not cause such errors.

Acknowledgement

We would like to thank Dr. Kise of Osaka Prefecture University for providing us with a software implementation of his segmentation algorithm and modifying it for our evalua-

tion purposes; Glenn van Doren of the Department of Defense for supporting this effort; and Dr. Azriel Rosenfeld of the University of Maryland for his comments.

This research was funded in part by the Department of Defense under Contract MDA 9049-6C-1250, Lockheed Martin under Contract 9802167270, the Defense Advanced Research Projects Agency under Contract N660010028910, and the National Science Foundation under Grant IIS9987944.

References

[1] *DARPA Broadcast News Workshop*, Herndon, VA, Feburary 1999. `http://www.itl.nist.gov/iaui/894.01/publications/darpa99/index.htm`.

[2] Aldus Corporation. *TIFF*. `ftp://sgi.com/graphics/tiff/`.

[3] A. D. Bagdanov. The fourth annual test of OCR accuracy. In A. D. Bagdanov, editor, *Annual Report*. Information Science Research Institute, University of Nevada, Las Vegas, NV, 1995.

[4] R. A. Becker, J. M. Chambers, and A. R. Wilks. *The New S Language*. Wadsworth & Brooks/Cole, Pacific Grove, CA, 1988.

[5] D. Dori, I. Phillips, and R. M. Haralick. Incorporating documentation and inspection into computer integrated manufacturing: An object-process approach. In S. Adiga, editor, *Applications of Object-Oriented Technology in Manufacturing*. Chapman & Hall, London, UK, 1994.

[6] J. J. Foster. *Data Analysis Using SPSS for Windows — A Beginner's Guide*. SAGE Publications, London, UK, 1998.

[7] T. Fruchterman. DAFS: A standard for document and image understanding. In *Proceedings of Symposium on Document Image Understanding Technology*, pages 94–100, Bowie, MD, October 1995.

[8] C. Ghezzi, M. Jazayeri, and D. Mandrioli. *Software Engineering*. Prentice Hall, Englewood Cliffs, NJ, 1991.

[9] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision*. Addison-Wesley, Reading, MA, 1992.

[10] T. Kanungo, C. H. Lee, J. Czorapinski, and I. Bella. TRUEVIZ: A groundtruth/metadata editing and visualizing toolkit for OCR. In *Proceedings of SPIE Conference on Document Recognition and Retrieval*, San Jose, CA, January 2001.

[11] K. Kise, A. Sato, and M. Iwata. Segmentation of page images using the area Voronoi diagram. *Computer Vision and Image Understanding*, 70:370–382, 1998.

[12] S. Mao and T. Kanungo. A methodology for empirical performance evaluation of page segmentation algorithms. Technical Report CAR-TR-933, University of Maryland, College Park, MD, December 1999. `http://www.cfar.umd.edu/kanungo/pubs/trsegeval.ps`.

[13] S. Mao and T. Kanungo. Automatic training of page segmentation algorithms: An optimization approach. In *Proceedings of International Conference on Pattern Recognition*, pages 531–534, Barcelona, Spain, September 2000.

[14] S. Mao and T. Kanungo. Empirical performance evaluation of page segmentation algorithms. In *Proceedings of SPIE Conference on Document Recognition and Retrieval*, pages 303–314, San Jose, CA, January 2000.

[15] G. Nagy, S. Seth, and M. Viswanathan. A prototype document image analysis system for technical journals. *Computer*, 25:10–22, 1992.

[16] L. O'Gorman. The document spectrum for page layout analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:1162–1173, 1993.

[17] T. Pavlidis and J. Zhou. Page segmentation and classification. *Graphical Models and Image Processing*, 54:484–496, 1992.

[18] E. M. Voorhees and D. K. Harman, editors. *The Seventh Text REtrieval Conference (TREC 7)*. National Institute of Standards and Technology, 1998. `http://trec.nist.gov/pubs.html`.

VIPER: Tools and Techniques for Video Performance Evaluation Applied to Scene and Document Images

David Doermann and David Mihalcik
Laboratory for Language and Media Processing
University of Maryland, College Park, Maryland 20742

Abstract

In this work we outline a reconfigurable VIdeo Performance Evaluation Resource (ViPER), which provides an interface for ground truth generation, metrics for evaluation and tools for visualization originally used for video analysis results, that is now begin used for document analysis. A key component is that the approach provides the basic infrastructure, and allows users to configure data generation and evaluation.

This document describes the system and its application for primarily from the video point of view (temporal and spatial), but by removing the temporal component, and adding a few extra tools, it is being activly used for ground truthing and evaluation of scene and document images, as will be described in the talk.

1 Introduction

An important requirement of any system that tries to automate content based analysis is a method to evaluate performance. Such evaluation is often carried out by comparing *Results* obtained from a given algorithm against *Ground Truth* — a set of results determined a priori to be correct. In video, the combination of spatial and temporal dimensions makes applying traditional evaluation methodologies difficult since we need to localize in both time and space. Although much work has been in computer vision evaluation, the evaluation work in video has focused primarily on evaluation of specific tasks such as motion estimation, or on segmentation [1, 2, 3], rather then on more general tasks of object detection, localization and classification. For a scene image, we can simply remove the temporal component, and for document images, we include models for which the relationships are both spatial and hierarchical.

The first goal of ViPER is the creation of a flexible ground truth format which facilitates the representation of both static and dynamic descriptors of the video. ViPER provides a segment-based view of the video, where attributes of descriptors are recorded for arbitrary sets of consecutive frames. The instances of descriptors and attributes are identified by the user developing the ground truth, and groups which develop ground truth for a specific class of problems are encouraged to develop guidelines for representation.

The second goal is to provide tools to easily create and share ground truth data. The process of creating ground truth can be tedious, especially in the video domain, since it can involve substantially similar content from frame to frame and require repeatedly scanning sequences of frames. We have developed a GUI that can be used to record the requisite information in a single scan of the video content. The system operates using the configuration files and data formats described below, with the configuration file being the only information that is provided a priori.

Finally, the third goal is to provide metrics which can be used to evaluate both the temporal and spatial aspects of video. We provide a core set of metrics for each of the datatypes currently implemented, as well as detection and precision and recall computation.

2 Representation and Interface

In ViPER, the ground truth (and subsequent results) are stored in files as sets of *descriptor* records. Each descriptor annotates an associated range of frames by instantiating a set of attributes for that range. So that applications can interpret the descriptors and render attributes appropriately, users provide a *configuration* which serves as a comprehensive baseline of what can appear in each record. For each valid descriptor type, there is a single *configuration record* in the GT formatted as follows:

```
descriptor-type descriptor-name
     attribute1 : attribute-type [default value]
     attribute2 : attribute-type [default value]
     ...
     attributeN : attribute-type [default value]
```

where the descriptor-type is either CONTENT (general properties about a range of frames) or OBJECT (instances

Table 1. ViPER Attribute Types

TYPE	DESC/REP	MEASURE	RNG
bvalue	boolean	bool equal	[0,1]
dvalue	integer	abs diff	[0-∞]
fvalue	float	abs diff	[0-∞]
svalue	string	Levenstein	[0-∞]
		equality	[0,1]
point	(x, y)	Eucl dist	[0-∞]
circle	(x, y, r)	OC/DC/ED	[0,1]
bbox	(ulx, uly, h, w)	OC/DC/ED	[0,1]
obox	(ulx, uly, h, w, o)	OC/DC/ED	[0,1]
lvalue	enumerated list	set equality	[0,1]
relation	obj IDs		

components in the scene). Current attribute-types are shown in Table 1.

Each instance of a record is then formatted as:

```
descriptor-type descriptor-name id sframe:eframe
     attribute1 : attribute-value
     attribute2 : attribute-value
     ...
     attributeN : attribute-value
```

The ViPER interface is shown in Figure 1. Individual frames are controlled with the use of step-increment and step-decrement arrows, a slider or by entering a frame number directly. When the video is moved to a given frame, changes in the objects and their attributes are reflected in tables linked back to the frame. For a given frame, users can select a cell representing a spatial attribute (point, bbox, obox or circle) in the content or object panels and a drawing panel is provided to view individual frames and to edit spatial attributes. Lists are entered via pull-downs, booleans via toggles and other attributes can be manually entered for each object as text. Each object instance can also be propagated attribute across multiple frames which is especially helpful for spatial attributes that do not change much across frames.

3 Evaluation

The problem of performance evaluation of video is a difficult and often subjective task. Since we are not necessarily dealing with a strict classification problem, we need to consider whether two descriptions are "close enough" to satisfy a particular set of constraints. This may include, for example, constraints on the temporal range over which the description is valid, on the spatial location of objects detected in scene or on other properties of the scene or objects extracted by the system. In continuing with our record-based philosophy, we provide a mechanism through which we can

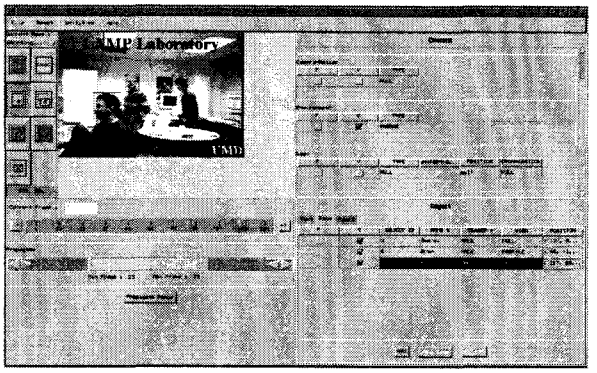


Figure 1. Layout of the ViPER-GT GUI.

match *candidate* records from the results with *target* records from the ground truth.

Target: An object or content record delineated temporally in the Ground Truth along with a set of attributes (possibly spatial).

Candidate: An object or content record delineated temporally in the Results along with a set of attributes (possibly spatial).

Our evaluation is based on a hierarchy of matching in both time and space and is split into two interdependent concepts: detection and localization. Ultimately, constraints on the localization will form a basis for "correct" detection. We will first consider detection based on the range of frames over which a pair of records is valid, then use localization constraints on both the temporal range and on attributes to judge the correctness of detection.

3.1 Detection

> ... a decision as to whether a particular object or content descriptor is adequately identified, either temporally, spatially or both.

A target record is said to be *minimally detected* if there exists at least one matching record in the candidate set. At the lowest level, we can ignore any localization constraints and say that a candidate minimally *matches* a target if the temporal range of the candidate and the temporal range of the target correspond on at least one frame. Localization constraints, both temporal and spatial, can then be used to constrain the definition of a match.

3.2 Localization

> ... a measure of how well a given target is identified.

As part of the detection process, each target has associated with it a set candidate matches, which we can then constrain. In general there are two ways we can constrain the initial set, with temporal localization on the range or with attribute localization - constraints on the individual attributes.

For example, the temporal correspondence may be required meets a certain tolerance with respect to the number or percentage of frames in common. At a frame level, we may require the difference between the attributes of corresponding frames be within a given tolerance (e.g. the face overlap >= X). At an attribute level, we can even introduce attribute localization constraints by requiring that the overall deviation of a candidates attribute from the target over the entire range be within some tolerance (The average overlap is at least 75%).

3.2.1 Temporal Localization

When matching objects temporally, we consider only metrics on the range of frames over which the target is valid. Although we can have a match mode as one-to-one, many-to-one, one-to-many, many-to-many between targets and candidates, it suffices to consider only the one-to-one case for now.

For a given pair of ranges (one from the target and one from the candidate), we will define three range metrics[1], but users can define additional metrics:

OVERLAP_COEF - the fraction of frames in the target range which are also in the candidate range.

$$OC = \frac{|Range_{target} \cap Range_{candidate}|}{|Range_{target}|}$$

Note this measure is not symmetric and does not in any way penalize for excessively large candidate. Nevertheless, if your only goal is to make sure the target is detected, it is a simple and effective metric.

DICE_COEF - a normalized measure of the number of frames in common, providing a similarity measure between [0,1]:

$$DC = \frac{2 * |Range_{target} \cap Range_{candidate}|}{|Range_{target}| + |Range_{candidate}|}$$

This coefficient rewards ranges which not only have a large number of frames in common, but also have minimal extra frames which are not in common. It is computed as twice the intersection divided by the sum of the candidate and target ranges.

[1] These 1D metrics are later extended to 2D as spatial metrics

EXTENT_COEF - the difference in start and end point correspondence between the target range and the candidate range.

$$\begin{aligned} \alpha &= |End_{target} - End_{candidate}| \\ &\quad + |Start_{target} - Start_{candidate}| \\ EC &= 1 - e^{-\alpha} \end{aligned}$$

This measure is useful for example, when it is necessary to precisely specify the start and end of a descriptor as it considers only the deviation of the endpoints and not how much of the candidate or target where correctly detected. It is simply the difference in end range positions.

Given a target/candidate pair whose ranges overlap we define a "correct" detection based only on the temporal range as one whose range metric meets a given tolerance. All correct detections, again, will be reported as a single or set of matched candidates.

3.2.2 Attribute Localization

Attribute localization, like temporal localization, may be computed by considering one-to-one, many-to-one, one-to-many or many-to-many correspondences. Each data-type will have associated with it a distance measure or set of distance measures which can be applied between corresponding attributes of a descriptor (Table 1).

On a frame by frame basis, *attribute localization* can be defined by a distance (or dis-similarity) measure as a function of the instances of attributes. A tolerance can be set on this distance to define a "close enough" attribute localization and subsequent correct detection. For 2D spatial attributes including bounding boxes (bboxes), oriented boxes (oboxes) and circles, the OVERLAP and DICE coefficients are extended from the definitions above by simply considering the overlap in the 2D plane which is easily computed geometrically.

3.3 Correctness

When evaluating performance, there we must be able to subject the measures defined above to various constraints. For each descriptor, we will allow the specification of a tolerance (and where appropriate a metric) both on the range metric and on individual attribute metrics. Furthermore, we will provide a number of fundamental "levels" of matching between a candidate and a target to define correct detection. We note that each of these levels is subsequently more restrictive, not in magnitude but in the types of features and metrics it considers. Our software will explicitly implement each of these levels.

Level 0: any candidate/target combination which has at least one frame in common is a level 0 *temporal correspondence.*

Level 1: any candidate/target combination for which the number and distribution of **corresponding** frames meets a specified tolerance is a level 1 *temporal match.* A temporal tolerance and metric (defined above) used to compute the match is specified on a per descriptor basis.

Level 2: any candidate/target combination for which the number and distribution of **valid** corresponding frames meet a given tolerance is a level 2 *frame-constrained temporal match.* A pair of corresponding frames in a temporal match is valid if all instances of attributes at that frame meet their respective tolerances. Level 2 extends level 1 by considering the effect of frame by frame tolerances of the attributes on the temporal or range tolerance.

Level 3: any candidate/target combination for which all attributes are valid is considered a level 3 *attribute-constrained temporal match.* An attribute is considered valid iff either the 1) average, 2) minimum or 3) median computed over all pairs of corresponding frames in a temporal match meets a given tolerance. The type of metric (average, minimum or median) as well as the tolerance is specified by the user.

Level 0 is useful for simple detection of a descriptor, level 1 requires a minimal temporal overlap, level 2 further constrains the attributes in individual frames and level 3 constrains the attributes across time.

4 Implementation

We have implemented all parts of the system in Java. The evaluation software is configurable so that different metrics can be used, both the ground truth and results data can be filtered by object type and by range of attributes. In this way, we can cycle through the parameter space and compare performance for a single algorithm with different metrics or different subsets of the data or multiple algorithms on the same data.

We have used for a number of tasks including detection of scene changes, detection of text is scene images and tracking of faces in video. The back end of our system processes raw detections and provides explicit associations as well as precision and recall summaries based on thresholds at a descriptor and/or attribute level. Graphical summaries of the detections and localizations are also produced from the raw results.

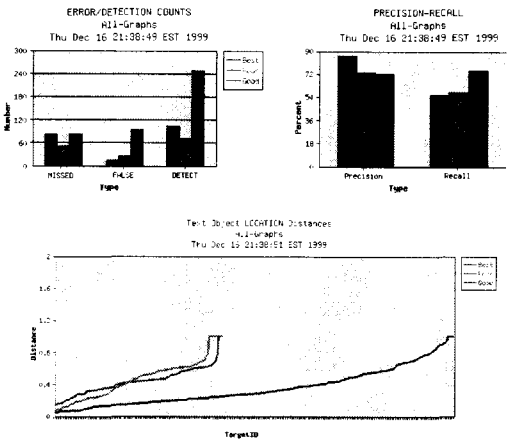


Figure 2. Examples of Bar Chart Summaries and Distance Graphs

For evaluating our system for the detection and tracking of text in video sequences[4], the ground truth contains information such as the position of text blocks, the type of motion, the text blocks content, and the quality. Runs will evaluate both detection and recognition results for either different algorithms on the same parameters or the same algorithm as a function of text quality. Figure 2a shows the overall detection results as a function of quality, Figure 2b shows the precision and recall, while Figure 2c shows the localization for the same case. In this case quality was based only on text clarity, it appears our detection algorithms are not dependent on text quality.

5 Discussion

We have presented a framework for performance evaluation which combines both temporal and spatial aspects of detection. The approach is reconfigurable with respect to both the evaluation and performance criteria, and can easily be extended to incorporate new temporal, spatial and attribute metrics. The system is being adapted to provide XML I/O, enhanced graphical capabilities and ranked retrieval metrics. Although the system was originally developed for video, it is also being used successfully for still images and document images. The software system is available for research use and can be down-loaded from http://lamp.cfar.umd.edu/.

References

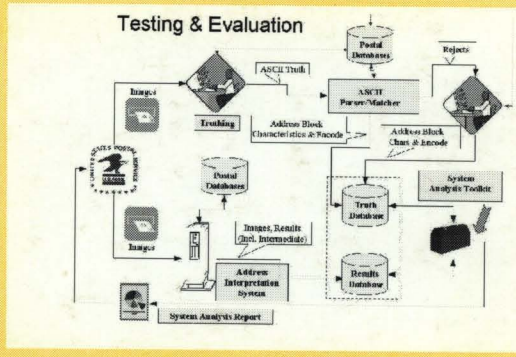
[1] G. Ahanger and T. Little. A survey of technologies for parsing and indexing digital video. *JVCIR, Special Issue on Digitial Libraries*, 7(1):28–43, 1996.

[2] J. Boreczky and L. Rowe. Comparison of video shot boundary detection techniques. In *SPIE 2670*, pages 170–179, 1996.

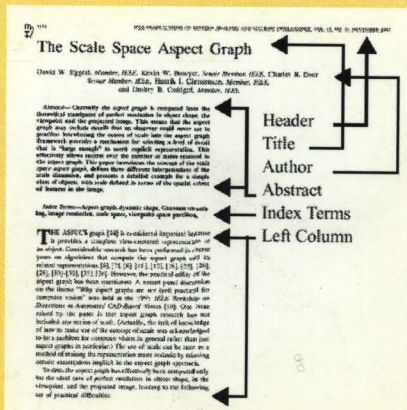
[3] U. Gargi, R. Kasturi, and S. Antani. Performance characterization and comparison of video indexing algorithms. In *CVPR*, pages 559–565, 1998.

[4] H. Li, D. Doermann, and O. Kia. Automatic text detection and tracking in digital video. *IEEE Transactions on Image Processing - Special Issue on Image and Video Processing for Digital Libraries*, pages 147–155, 1999.

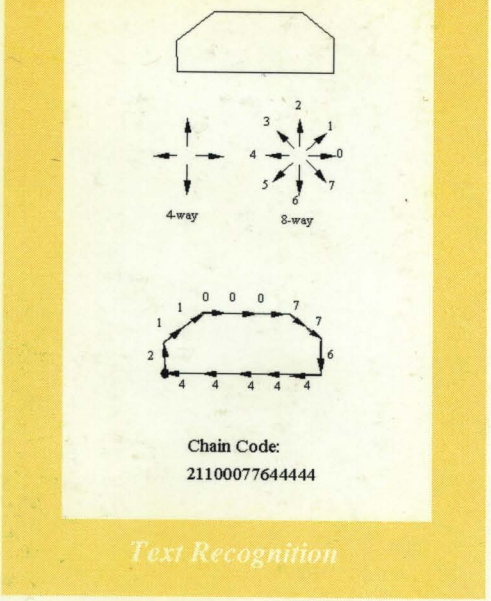
Author Index



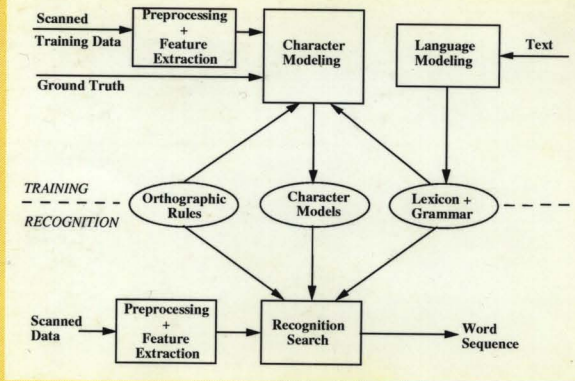
Performance Evaluation



Page Analysis and Classification



Text Recognition



Foreign Language Information Retrieval