Background Line Detection with A Stochastic Model

Yefeng Zheng, Huiping Li and David Doermann
Language and Media Processing Laboratory
Institute for Advanced Computer Studies
University of Maryland, College Park, MD 20742-3275
E-mail: {zhengyf, huiping, doermann}@cfar.umd.edu

Abstract

Background lines often exist in textual documents. It is important to detect and remove those lines so text can be easily segmented and recognized. A stochastic model is proposed in this paper which incorporates the high level contextual information to detect severely broken lines. We observed that 1) background lines are parallel, and 2) the vertical gaps between any two neighboring lines are roughly equal with small variance. The novelty of our algorithm is we use a HMM model to model the projection profile along the estimated skew angle, and estimate the optimal positions of all background lines simultaneously based on the Viterbi algorithm. Compared with our previous deterministic model based approach[15], the new method is much more robust and detects about 96.8% background lines correctly in our Arabic document database.

1 Introduction

When we process documents it is not uncommon that background lines exist in the documents, touching or mixing with text. Figure 1(a) shows an Arabic document with background lines and handwriting. These lines are originally printed on the paper to help writers to guide their writing. After digitization they will, however, touch text and cause problems for segmentation and recognition. It is important that those lines can be detected and removed before we feed the text to the Optical Character Recognition (OCR) engine.

1.1 Related Work

Line detection is widely used in table detection and interpretation [14, 16], engineering graph interpretation [1], and bank check/invoice processing [13]. The line detection algorithms presented in these works can be broadly classified as: Hough transform or vectorization based [9]. The Hough transform is a global approach with the capability to detect dashed and mildly broken lines, but is extremely time consuming [5]. To reduce the computation cost, a projection based method is proposed in [7] to search lines within a small range of angles around 0^o or 90^o. The algorithm is much faster than the Hough transform based approach, however, it can only detect roughly horizontal or vertical lines. Vectorization based algorithms, such as BAG [14] and SPV methods [9], extract vectors from the image first, then merge vectors into lines. Recently Zheng presented a novel vectorization based algorithm called the Directional Single-Connected Chain (DSCC) method [16]. Each extracted DSCC represents a line segment and multiple non-overlapped DSCCs are merged into a line based on rules.

These line detection algorithms work well on relatively clean documents with solid or mildly broken lines. In our task there are two challenges: 1) the lines are severely broken due to the low image quality, and 2) the lines are mixed with text, making the separation difficult. Figure 1(c) shows the line detection result using the DSCC algorithm. We can see only few lines are partially detected. It is very difficult, if not impossible, to detect these lines without contextual information.

In the form analysis, most form cells are rectangular which can be used as a priori knowledge to correct the low level line detection errors [2, 16]. In both approaches, the contextual information is incorporated into pre-defined rules in an ad hoc way. In this paper we present a novel model-based approach to systematically incorporate high level information.

1.2 Outline of the Approach

We observed that 1) background lines are parallel, and 2) the vertical gaps between any two neighboring lines are roughly equal. In our previous work [15], we presented a

The support of this research by the Department of Defense under contract MDA-9040-2C-0406 is gratefully acknowledged.

1063-6919/03 $17.00 © 2003 IEEE

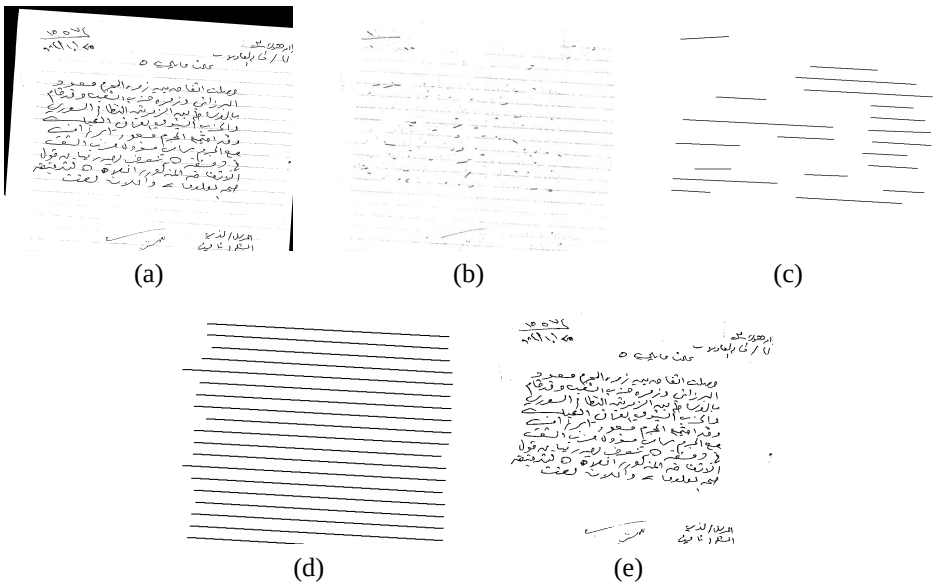


Figure 1. An example of background parallel line detection. (a) Original document image; (b) after filtering; (c) DSCC based line detection result; (d) model based line detection result; (e) text after line removal.

model $\mathcal{M}(\theta, g, y_1)$ to detect a group of parallel lines, where θ is the skew angle, g is the vertical line gap between two neighboring lines, and y_1 is the vertical translation (the vertical position of the first line). However, the deterministic model we used has several limitations: 1) The model parameters (θ, g, y_1) are estimated sequentially and the estimation error will accumulate and propagate; 2) The model is deterministic with the assumption that the vertical line gaps between any two neighboring lines are equal without the consideration of the variance of these gaps.

In this paper, a stochastic model, $\mathcal{M}(\theta, y_1, y_2, \ldots, y_N)$, is proposed (as shown in Figure 2), where N is the number of lines on the document, and $y_i, i = 1, 2, \ldots, N$ is the vertical position of the i^{th} background lines, which can be modeled by a HMM model well. Similar to our previous approach, we first estimate the skew angle θ, then perform a coarse estimation of the vertical line gap $\bar{g}$, from the auto-correlation of the projection profile along θ. Our novelty is to use the Viterbi algorithm to search the optimal position of background lines simultaneously from the projection profile. The estimation error of $\bar{g}$ and variance between vertical line gaps are all compensated by the Viterbi decoding of the HMM model. The detection accuracy increases from 94% [15] to about 96.8% based on this new stochastic model.

2 Pre-Processing

First we extract horizontal line segments using the DSCC based algorithm [16]. A horizontal DSCC is an array of connected vertical run-length, which can be a line segment, a text stroke or noise. We only preserve those DSCCs with small skew angles and large aspect ratios, which are likely to be horizontal line segments. From the filtered image (Figure1(b)) we can see most text strokes are filtered and the background line segments are preserved. We then merge neighboring DSCCs into lines, as shown in Figure 1(c). Based on this initial detection result, we use a two-step method to estimate the skew angle θ from coarse to fine. Some other methods surveyed in [3] estimate the skew angle based on text (printed text or handwriting) on the document. Then we do horizontal projection along the estimated angle. A coarse estimation of average vertical line gap $\bar{g}$ is calculated from the auto-correlation of the projection profile along θ. The details are addressed in [15].

3 Model-based Line Detection

3.1 HMM Model for Line Position

The Markov property of a sequence of events is well studied in literature [12]. Consider a system that stays at one of a set of N distinct states, $S_1, S_2, \ldots, S_N$, at any sampling time t. It undergoes a change of state according to a

set of probabilities associated with the state during the period between two successive sampling time. For a Markov chain (the first order), the probability of staying at state q_t only depends on previous state q_{t-1}:

$$P[q_t = S_{k_t}|q_{t-1} = S_{k_{t-1}}, q_{t-2} = S_{k_{t-2}}, \ldots, q_1 = S_{k_1}] = P[q_t = S_{k_t}|q_{t-1} = S_{k_{t-1}}] \quad (1)$$

If the state transition probability is independent of time t, then the Markov chain is said to be homogeneous:

$$P[q_t = S_j|q_{t-1} = S_i] = a_{ij}, 1 \leq i, j \leq N \quad (2)$$

If the state is not observable, the resulting model (which is called a hidden Markov model) is a doubly embedded stochastic process with an underlying stochastic process that is not observable, but can only be observed through another set of stochastic process that produces the sequence of observations. The elements of a HMM model are:

1) N, the number of the states in the model.

2) M, the number of distinct observation symbols per state.

3) The state transition probability distribution matrix:
$A = \{a_{ij}\}$.

4) The probability distribution matrix of the observation symbol:
$B = \{b_{ij}\}$.

5) The initial state distribution π.

HMM models can model some 1-D signals well, and has achieved great success in speech [12] and handwriting recognition [11]. In our approach we use it to model the sequence of vertical line position y_i.

$$P(Y_i|Y_1, Y_2, \ldots, Y_{i-1}) = P(Y_i|Y_{i-1}) \quad (3)$$

Here we use uppercase characters to represent random variables (e.g. Y_i), and lowercase characters to represent the value of the random variables, (e.g. y_i). The actual y_i is not observable. Instead we can only observe the projection profile $h_k, k = 1, 2, \ldots, T$, where T is the dimension of the profile.

$$P(H_i|Y_1 = y_1, \ldots, Y_N = y_N) = \begin{cases} P(H_k|\exists i, k = y_i) & \text{A line is on } k \\ P(H_k|\forall i, k \neq y_i) & \text{No lines are on } k \end{cases} \quad (4)$$

There are two states: line state S_l and non-line state S_n. A standard HMM model for our problem is shown in Figure 3(a).

One weakness of conventional HMMs is the modeling of state duration. The inherent duration probability density $p_i(d), d = 1, 2, \ldots$, associated with state S_i, with self transition coefficient a_{ii} is of the form:

$$p_i(d) = (a_{ii})^{d-1}(1 - a_{ii}) \quad (5)$$

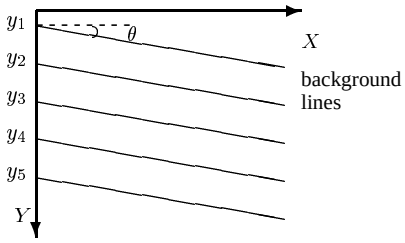


Figure 2. A group of background lines.

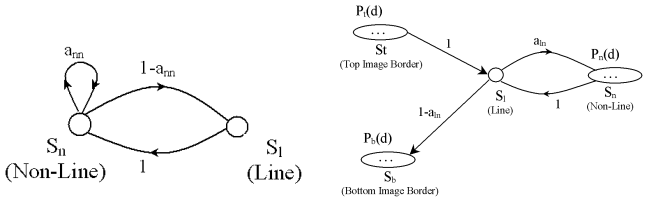


Figure 3. HMM Models for the horizontal projection profile. (a) A standard HMM model; (b) A HMM model with explicit duration.

For our problem, the duration of non-line state S_n is around the average vertical line gap with minor variance. The exponential state duration density is inappropriate. Instead we explicitly model the duration density. We add two more states: S_t and S_b for top and bottom image borders. The model with explicit state duration is shown in Figure 3(b). For some applications, the quality of the modeling is significantly improved when explicit state duration densities are used [6].

3.2 Model Parameter Estimation

After explicitly modeling the state duration, the state transition probability is very simple and labeled on Figure 3(b). We set a_{ln} to 0.5 and the detection result is not sensitive to a_{ln}. The observation is the projection profile h_k, which takes values between $[0, w]$, where w is the width of the image. We quantize h_k to five levels. Ideally, background lines form peaks on the projection profile. Therefore, we set $h_k = 0$ if there are no local peaks at k, and quantize the local peaks into four levels using the following quantization levels: $w/16, w/8, w/4$. The observation probability distribution matrix B, as listed in Table 1, can be easily estimated from the groundtruth of the background lines. We observed that 1) due to the severely brokenness, the horizontal projections of about 80% background lines are less than 1/4 of the image width; 2) 4.9% background lines do not form peaks; and 3) the peaks with small height

Table 1. Observation probability distribution matrix B

	0 Non-peak	1 $(0, \frac{w}{16}]$	2 $(\frac{w}{16}, \frac{w}{8}]$	3 $(\frac{w}{8}, \frac{w}{4}]$	4 $(\frac{w}{4}, w]$
S_l	192 (4.9%)	607 (15.4%)	663 (16.8%)	1,645 (41.7%)	835 (21.2%)
S_u, S_b, S_n	328,201 (98.8%)	3,530 (1.1%)	283 (0.1%)	97 (0.03%)	19 (0.006%)

Table 2. The duration probability distribution of state S_n. The distribution is roughly symmetric around $\bar{g}$. The row of *distance* lists the difference to $\bar{g}$.

Distance	-8	-7	-6	-5	-4	-3	-2	-1	0	1
Occurring	2	5	12	17	46	39	47	510	2,223	687
Distance	2	3	4	5	6	7	8	9	10	11
Occurring	74	45	17	17	13	5	0	0	2	1

are most likely formed by text strokes or noise (3,530 instances), rather than by background lines (607 instances). We need to use high level contextual information to achieve reasonable detection results for such severely broken background lines.

The major drawback of the explicit duration HMM model is that it greatly increases computational cost. With the traditional forward-backward training algorithm (a type of EM algorithm), the re-estimation problem is more difficult for the variable duration HMM than for the standard HMM. The training time may increase by a factor of 300 [12]. One solution is to use a parametric state duration density instead of the non-parametric $p_i(d)$, or assume a uniform duration distribution. In our case, we can directly get the HMM parameters from groundtruth since the states have explicit physical meaning. We set states S_t and S_b to be uniformly distributed on $[0, \bar{g} - 1]$. The duration probability of state S_l is estimated directly from the groundtruth, as shown in Table 2. Initially, the system stays at state S_t with probability 1.

3.3 Decoding of HMM model

Given the observation sequence $O = h_k, k = 1, 2, \ldots, T$, and the HMM model λ, we want to search an optimal state sequence $Q = q_1 q_2 \ldots q_T$, to maximize $P(Q|O, \lambda)$, which is equivalent to maximizing $P(Q, O|\lambda)$. Normally, the Viterbi algorithm, based on dynamic programming methods, is used to decode HMM models. We define the quality

$$\delta(t) = \max_{q_1, q_2, \ldots, q_{t-1}} P[q_1, q_2, \ldots, q_t = S_l, h_1, h_2, \ldots, h_t|\lambda] \tag{6}$$

as the best decoding score at time t, which accounts for the first t observations and ends in state S_l. The sequence $q_1, q_2, \ldots, q_{t-1}$ which maximizes the probability in Equation 6 is the best decoding result until time t. The complete procedure of decoding is stated as follows:

1. Clear all entries of δ.

2. For $1 \leq i \leq \bar{g}$, decode the first $i - 1$ observations as S_t (the top image border) and observation i as S_l:

$$\delta(i) = \frac{1}{\bar{g}} P(q_i = S_l|h_i) \prod_{j=1}^{i-1} P(q_j \neq S_l|h_j), \tag{7}$$

where $P(q_i = S_l|h_i)$ is the probability to decode observation i as state S_l, given observation h_i, and $\prod_{j=0}^{i-1} P(q_j \neq S_l|h_j)$ is the probability to decode the first $i - 1$ observations as S_t.

3. Set $t = 1$.

4. For $-\Delta_- \leq j \leq \Delta_+$, where Δ_- and Δ_+ are the maximal variance on two sides of $\bar{g}$ (as shown in Table 2, $\Delta_- = 8$ and $\Delta_+ = 11$).

$$\delta'(t + \bar{g} + j) = \delta(t) a_{ln} P_n(d = \bar{g} + j) \times P(q_{t+\bar{g}+j} = S_l|h_{t+\bar{g}+j}) \prod_{k=t+1}^{t+\bar{g}+j-1} P(q_k \neq S_l|h_k) \tag{8}$$

$$\delta(t + \bar{g} + j) = \max\{\delta(t + \bar{g} + j),\ \delta'(t + \bar{g} + j)\} \tag{9}$$

where a_{ln} is the transition probability from S_l to S_n, and $P_n(d = \bar{g} + j)$ is the probability of staying at state S_n with $\bar{g} + j$ consecutive times.

5. Decode the remaining states as S_b, the bottom image border, if $t > T - \bar{g}$:

$$\delta'(T) = \delta(t) \frac{1}{\bar{g}} (1 - a_{ln}) \prod_{k=t+1}^{T} P(q_k \neq S_l|h_k) \tag{10}$$

$$\delta(T) = \max\{\delta(T),\ \delta'(T)\} \tag{11}$$

6. If $t < T$, then $t = t + 1$, and goto step 4.

For each $t, t = 1, \ldots, T$, the algorithm remembers the best decoding path at time t. Therefore, we can get the best decoding sequence $q_1, q_2, \ldots, q_T$ after the program ends. With the empirically estimated parameters, the background line detection accuracy is about 95.6%.

3.4 Optimization of the Model Parameters

The HMM parameters estimated directly from the groundtruthed database is not optimal since: 1) The data is sparse. Some entries in Table 2 do not appear in the training data, and many entries only appear very few times; 2) the o^{th} entry in Table 2 is much larger than other entries. The estimation error of the average vertical line gap can be around 1-2 pixels, which will significantly deteriorate the decoding result; and 3) the Viterbi algorithm searches the hidden state sequence which is with the highest probability given the observation sequence and the model. However, such optimization criteria does not minimize the final detection error, especially when the model mis-matches.

To reduce the effect of sparse data, we smooth the duration distribution $p_n(d)$ of state S_n. Suppose the state duration is symmetric around the average vertical line gap, we perform the following averaging:

$$p_n(\bar{g}+i) = p_n(\bar{g}-i) = \frac{p_n(\bar{g}+i) + p_n(\bar{g}-i)}{2} \quad (12)$$

After averaging, we set the empty entries to the minimal value of all non-zero entries. So the allowed variance of vertical line gap is within $[-11, 11]$. Our ultimate goal is to search the optimal HMM model parameters to minimize the line detection error. Unfortunately, the error criterion is a very complex function of the model parameters without a close form. A direct searching algorithm can be used to solve the optimization problem in low dimensional space. In our case, the simplex search method proposed by Nelder and Mead is used to minimize the detection error [10]. Among many parameters of our model, we only optimize the observation probability matrix B and the 0^{th} entry of the state duration of S_n. Experiments show the detection accuracy increases to 97.3% in the training set after optimization.

4 Post-Processing

After identifying the vertical position of a line, our next step is to detect the left and right end points. We first group the broken line segments together. At each detected position of a line, those DSCCs within the strip of 10 pixels above and below the line are merged into a line [16]. If there are less than 50 pixels on the line, then it is removed.

An ideal straight line can be presented with two parameters a and b as $y = a \times x + b$. For a real line with points $(x_i, y_i), i = 0, 1, \ldots, n-1$, parameters a and b can be estimated using the minimum mean square error criterion (MMSE):

$$\bar{x} = \sum_{i=0}^{n-1} x_i/n \qquad \bar{y} = \sum_{i=0}^{n-1} y_i/n$$

$$a = \frac{\sum_{i=0}^{n-1}(x-\bar{x})(y-\bar{y})}{\sum_{i=0}^{n-1}(x-\bar{x})^2}$$
$$b = \bar{y} - a \times \bar{x} \quad (13)$$

For most background lines, this approximation is good enough. However, due to the distortion introduced by the photocopying and scanning, some background lines are cursive, and can not be approximated by two end points. In this case, we use a polyline to represent the line as follows:

1. Calculate the average approximation error of a line as follows:

$$\delta y_i = |y_i - a \times x_i - b| \quad (14)$$

$$e = \sum_{i=0}^{n-1} \delta y_i / n \quad (15)$$

2. If e is smaller than the average line width (often 2-4 pixels), keep it with two end points representation, and exit.

3. Otherwise, split the whole line into left and right segments from the middle. Estimate the line parameters a and b for each segment respectively, as described in Equation (13).

4. For each segment, goto step 1 and repeat.

A polyline is described as a sequence of junction points $(P_0, P_1, \ldots, P_m)$. Experiments show 2 or 3 segments are sufficient to represent most cursive lines.

5 Experiments

5.1 Evaluation Protocol

Line detection accuracy can be evaluated at the pixel level and the line level [8]. The pixel level evaluation compares the difference of the pixels between groundtruthed and detected lines. It is straightforward and objective, but groundtruthing at the pixel level is extremely expensive when lines are broken, distorted and/or overlapped with text. Therefore, we evaluate the algorithm at the line level. Two metrics, the vertical and horizontal distance, are defined. Vertical distance is defined as a modified Hausdorff distance. The Hausdorff distance between two point sets is:

$$H(A, B) = \max\{h(A, B), h(B, A)\} \quad (16)$$

where,

$$h(A, B) = \max_{a \in A} \min_{b \in B} ||a - b|| \quad (17)$$

and $||.||$ is some underlying norm (e.g., the L_2 or Euclidean distance). The function $h(A, B)$ is called the *directed* Hausdorff distance from A to B. It identifies the point $a \in A$ that

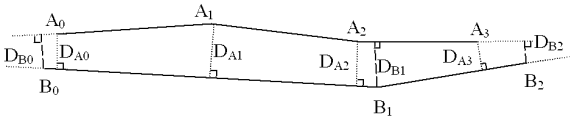


Figure 4. Vertical distance between two polylines.

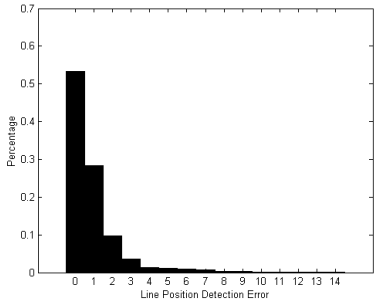


Figure 5. Histogram of vertical detection distance of matched groundtruth background lines.

is the farthest from any point of B and measures the distance from a to its nearest neighbor in B [4]. For severely broken lines, it is very hard to define the left and right end points exactly. The detection accuracy of left and right points are measured by the following horizontal distance. And the vertical distance is defined only to evaluate the detection accuracy at the vertical position. Therefore a modified Hausdorff distance is used instead. Suppose A and B are two polylines, and we extend them to infinity to generate two new polylines A' and B', as shown in Figure 4. The vertical distance $vd(A, B)$ is:

$$vd(A, B) = \max\{h(A, B'), h(B, A')\} \tag{18}$$

The original method to compute the Hausdorff distance is very time consuming [4]. In our case, the Hausdorff distance can be easily calculated due to no two pixels on a polyline having the same horizontal coordinate. Suppose polyline A and B are represented as $(A_0, A_1, \ldots, A_m)$ and $(B_0, B_1, \ldots, B_n)$ respectively, Then Equation (18) is simplified as:

$$vd(A, B) = \max\{D_{A0}, D_{A1}, \ldots, D_{Am}, D_{B0}, D_{B1}, \ldots, D_{Bn}\} \tag{19}$$

where $D_{Ai}, i = 0, \ldots, m$ is the distance from A_i to extended polyline B', and $D_{Bi}, i = 0, \ldots, n$ is the distance from B_i to extended polyline A', as shown in Figure 4.

The horizontal distance, $hd(A, B)$, and the horizontal matching rate are defined to measure the detection accuracy of left and right end points. Suppose the horizontal coordinates of the left and right end points of polyline A and B are (L_A, R_A) and (L_B, R_B) respectively. Then, the horizontal distance $hd(A, B)$ are defined as:

$$hd(A, B) = \frac{|L_A - L_B| + |R_A - R_B|}{2} \tag{20}$$

And the horizontal matching rate is defined as:

$$m = \frac{\min\{R_A, R_B\} - \max\{L_A, L_B\}}{\max\{R_A, R_B\} - \min\{L_A, L_B\}} \tag{21}$$

A detected line matches a groundtruthed line if the vertical distance is less than $\bar{g}/3$. If there are more than one detected lines matching a groundtruthed line, or vice verse, then the match with the minimal distance is kept. The vertical detection distance of the groundtruthed line is defined as the vertical distance to the matched detected line. If a groundtruthed line can not match any detected line, it is mis-detected. For a matched groundtruth line, if the detection distance is within 5 pixels, then it is said to be detected correctly. Otherwise it is regarded as partially detected. A false alarm happens if a detected line does not match any groundtruthed lines.

5.2 Experimental Results

We obtained 168 Arabic document images with a total of 3,870 groundtruthed lines, most of which are severely broken. We use 100 images to train the HMM model, and the remaining 68 images as the test set. The detection results are shown in Table 3. On the test set, 96.8% lines are detected correctly and only 2 lines are missed. The histogram of the vertical detection distance for all matched groundtruth lines is shown in Figure 5. For the matched lines, most vertical detection distance is less than 8 pixels. The maximum vertical detection distance is 14 pixels. The false alarm rate is about 2.3%. Most of the false alarms are generated because our model detected severely broken lines which are not groundtruthed based on the subjective judgment of the groundtruther.

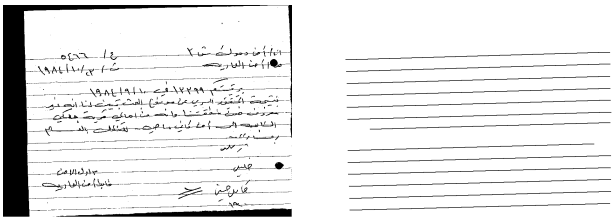
For correctly detected lines we evaluate the left and right end points detection accuracy using the horizontal distance defined in Equation 20. The average horizontal detection distance is 6 pixels and the matching rate is 99.1%.

Figure 1(d) shows the model based line detection result. Compared with Figure 1(c), we can see with contextual information the result is significantly improved. In Figure 6(a), we remove 35 rows of the image (about half of the average vertical line gap of this document). The corresponding detection result is shown in Figure 6(b), with only one line missed due to the anomalous vertical line gap.

Our final experiment is to test the robustness of our algorithm. We select a document with relatively good quality. We manually separate the document into two layers:

Table 3. Background line detection result.

	Ground-truthed Lines	Detected Lines	Correct	Partial Correct	Missed	False Alarm
Training Set	2,274	2,319	2,212 (97.3%)	56 (2.5%)	6 (0.3%)	51 (2.2%)
Test Set	1,596	1,631	1,545 (96.8%)	49 (3.0%)	2 (0.1%)	37 (2.3%)



(a) (b)

Figure 6. An example of a model mis-match. (a) A document image with 35 image rows removed; (b) Line detection result of (a).

text and background lines, as shown in Figure 7(b) and (c) respectively. We randomly flip some black pixels in background lines to white, and then merge the degraded background lines with the original text. Figure 7(d) and (f) are the degraded images with only about 10% and 5% pixels in the line preserved. With 10% pixels preserved, our algorithm can detect all background lines, as shown in Figure 7(e). The algorithm fails when only about 5% pixels are preserved, as shown in Figure 7(h), due to the wrong estimation of the vertical line gap.

The average processing time is about 0.4 second for an image with the size of $1{,}700 \times 1{,}800$ pixels on a PC with 1.8GHZ CPU, 1GB memory.

6 Conclusion and Future Work

In this paper we present a novel approach to detect severely broken parallel lines in documents. Our method is based on a stochastic model to incorporate high level constraints into a general line detection algorithm. Compared with our previous deterministic model, we use HMM to model the stochastic property of the vertical line gaps. Instead of detecting single line sequentially, we use the Viterbi algorithm to detect all background lines simultaneously. Our method can detect 96.8% lines in the database we collected. Some challenging examples demonstrated the robustness of our approach. The other contributions of this paper include: 1) DSCC filtering to remove most text strokes to facilitate the following line detection; and 2) polyline representation of distorted lines.

After line detection, we can remove these detected lines to achieve a cleaned version of the document. Figure 1(e) is the result of Figure 1(a) after we remove the black pixels on the line and filter the noise. While the result is encouraging, we find some text strokes touching the detected lines are removed erroneously. We are investigating a more robust algorithm to remove the lines and reserve the text strokes. And we will evaluate the line removal algorithm when an OCR engine is available.

References

[1] D. Dori, Y. Liang, and J. Dowell. Sparse-pixel recognition of primitives in engineering drawings. *Machine Vision and Application*, 6:69–82, 1993.

[2] O. Hori and D. Doermann. Robust table-form structure analysis based on box-driven reasoning. In *Proc. Int'l Conf. Document Analysis and Recognition*, pages 218–221, 1995.

[3] J. J. Hull. Document image skew detection: Survey and annotated bibliography. In J. J. Hull and S. L. Taylor, editors, *Document Analysis Systems II*. Word Scientific, 1998.

[4] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge. Comparing images using the hausdorff distance. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(9):850–863, 1993.

[5] J. Illingworth and J. Kittler. A survey of the Hough transform. *CVGIP*, 44:87–116, 1988.

[6] S. E. Levinson. Continuously variable duration hidden markov models for automatic speech recognition. *Computer, Speech and Language*, 1(1):29–45, 1986.

[7] J. Liu, X. Ding, and Y. Wu. Description and recognition of form and automated form data entry. In *Proc. Int'l Conf. Document Analysis and Recognition*, pages 579–582, 1995.

[8] W. Liu and D. Dori. A protocol for performance evaluation of line detection algorithms. *Machine Vision and Application*, 9(5/6):57–68, 1997.

[9] W. Liu and D. Dori. From raster to vectors: Extracting visual information from line drawings. *Pattern Analysis and Application*, 2(1):10–21, 1999.

[10] J. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[11] R. Plamondon and S. N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(1):63–84, 2000.

[12] L. R. Rabiner. A tutorial on hidden markov models and selected application in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[13] Y. Y. Tang, C. Y. Suen, and C. D. Yan. Finanial document processing based on staff line and description language. *IEEE Trans. Systems, Man and Cybernetics*, 25(5):738–753, 1995.

[14] B. Yu and A. K. Jain. A generic system for form dropout. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(11):1127–1131, 1996.

Figure 7. An experiment on document degradation. (a) Original image, which is separated into two layers: text (b) and background lines (c) manually; (d) About 10% pixels of the background lines are preserved; (e) Line detection result of (d); (f) About 5% pixels of the background lines are preserved; (h) Line detection result of (f).

[15] Y. Zheng, H. Li, and D. Doermann. A model-based line detection algorithm in documents. In *Proc. Int'l Conf. Document Analysis and Recognition*, (to appear), 2003.

[16] Y. Zheng, C. Liu, and X. Ding. Form frame line detection with directional single-connected chain. In *Proc. Int'l Conf. Document Analysis and Recognition*, pages 699–703, 2001.