

Document skew estimation without angle range restriction

Oleg Okun*, Matti Pietikäinen, Jaakko Sauvola

Machine Vision and Media Processing Unit, Infotech Oulu and Department of Electrical Engineering, University of Oulu, P.O.Box 4500, FIN-90401 Oulu, Finland; e-mail{oleg,mkp,jjs}@ee.oulu.fi

Received October 10, 1998 / Revised version September 9, 1999

Abstract. The existing skew estimation techniques usually assume that the input image is of high resolution and that the detectable angle range is limited. We present a more generic solution for this task that overcomes these restrictions. Our method is based on determination of the first eigenvector of the data covariance matrix. The solution comprises image resolution reduction, connected component analysis, component classification using a fuzzy approach, and skew estimation. Experiments on a large set of various document images and performance comparison with two Hough transform-based methods show a good accuracy and robustness for our method.

Key words: Image processing – Document image analysis – Skew estimation

1 Introduction

Document skew is a distortion that often occurs during scanning or copying of a document or as a design feature in the document's layout. This mainly concerns the orientation of text lines, where a zero skew occurs when the lines are horizontal or vertical, depending on the language. Document skew is an unavoidable effect, in many cases affecting negatively the accuracy of page segmentation/classification and OCR, because they often require properly aligned images before their application. Several methods exist for page segmentation/classification without a need for skew estimation and correction (see [2, 9], for example). However, they either intentionally restrict the possible angle range, or they are considered in isolation from the subsequent processing operations, such as character segmentation and recognition. As a result,

* He is also a senior researcher at the Institute of Engineering Cybernetics, National Academy of Sciences, Surganov street 6, 220012 Minsk, Belarus

Correspondence to: M. Pietikäinen

the simplification of one solution leads to more sophisticated and therefore more time-consuming techniques for another task. In general, there can be three types of skew within a page: a global skew, when all page blocks have the same orientation; a multiple skew, when certain blocks have a different slant than the others; and a non-uniform text line skew, when the orientation fluctuates within a line, e.g., a line is bent at one or both of its ends, or a line has a wave-like shape. Here, we will focus on the global skew estimation. Currently, there are many global skew estimation techniques [1, 3–8, 11, 13–15, 17–36, 38] (later we will simply call them skew detection or estimation methods). They can be roughly divided into two classes, according to the detectable angle range. The techniques belonging to the first group [1, 3–7, 11, 14, 17–20, 22, 24–33, 35, 36] can detect a skew only for a limited angle range varying usually from $\pm 5^\circ$ to $\pm 45^\circ$, while another class of techniques is suitable for finding any skew angle [8, 13, 15, 21, 23, 34, 38]. In addition to the complicated real and artificially generated document layouts and various document types, there is yet another reason for developing robust skew detection methods. Japanese documents, for example, often contain both horizontal and vertical text lines, and the vertical lines constitute angles with the X-axis which are greater than 90° . An arbitrary skew angle may also appear when digitizing small-sized documents. Therefore eliminating the restrictions on detectable angle range is of great importance, especially in automated document processing. Several methods detect skew for compressed or low-resolution (50 dpi) images [17, 31, 32, 38]. Some of them are developed for specific image formats such as JBIG and CCITT Group IV [17, 31, 32], while others use images scanned at a low resolution as input (see, for example, [38]). The methods [17, 31, 32] limit the detectable angle range, while the method [38] does not. The skew detection methods have been applied to images taken from journals, newspapers, invoices, letters (printed or handwritten), book covers, etc. Journals and newspapers are, however, the main documents processed by many methods, because they contain much printed text information needed for skew estimation. Usually a

particular method can work with one or two classes of document images, though two methods [32,38] recently proposed apparently can be applied to many different classes. In this paper, we propose a new technique for document image skew estimation. The main novelty is a new method for extracting the lines of connected components on low-resolution (50 dpi) images. Each component represents either several merged characters or a whole word in many cases, because we lower a high resolution of the original image instead of scanning it at a low resolution. Our method is based on computing the first eigenvector of the input data covariance matrix, where the input data are pixel coordinates of the connected components on the low-resolution image. The angle between this eigenvector and the X-axis guides a process of collecting the connected components into lines. Our method does not depend on the image format, it can estimate an arbitrary skew angle, and it is still quite fast because only simple operations like computation of the 1st and 2nd-order moments are necessary. In addition, unlike in the Hough transform-based approaches, the accumulator space for our method is 1D, the size of which is significantly smaller than the image size. The rest of the paper has the following organization. Section 2 reviews the document skew detection methods previously developed. Section 3 describes each step of our method in detail. Section 4 presents experimental results, including a comparison with two Hough transform-based skew detection methods. Section 5 gives final conclusions and possible extensions of our work. The necessary mathematical formulae are presented in Appendix 1.

2 Related work

Most of the skew detection methods have the following common features: (1) a prior text/graphics separation is necessary, which may take a significant amount of time, though it can be useful for the next steps; (2) large text areas have to be present on a page for an accurate estimation; (3) many techniques have been designed for high-resolution images ranging from 100 to 300 dpi. Projection profiles for skew detection are used by Akiyama and Hagita [1], Baird [3], Bloomberg et al., [4], Kanai and Bagdanov [17], Komukai and Saiwai [18], Lam and Zandy [19], Messelodi and Modena [21], Pavlidis and Zhou [26], Postl [27], and Spitz [31,32]. The projection profile is a histogram of black pixels or representative (fiducial) points of characters such as centers of lower segments of bounding boxes of connected components [3], for example, along a given direction. The data is projected in a number of directions and the variation in the obtained projection profile is calculated for each direction. The angle corresponding to the maximum variation is the desired skew. This method is simple and well understandable, but the range of detectable angles is restricted because the profile computation for many angles is a time-consuming operation. One paper [21] recently showed that projection profiles in combination with a clustering procedure based on simple heuristics may overcome the problem of the limited angle range.

Although this method is robust to multiple skew and small interline spacing, it was only tested on small-sized (512x512 pixels) images of book covers containing a few text lines. The Hough transform is another popular technique for skew detection (Hinds et al., [14], Jiang et al., [15], Le et al., [20], Nakano et al., [22], Srihari and Govindaraju [33], Sugawara [34], and Yu and Jain [38]). This transform is often applied to a number of representative points of characters such as the lowermost pixels or centers of gravity. Each representative point (x,y) is mapped from the Cartesian space to the points (ρ,θ) in the Hough space by forming a set of lines coming through (x,y) with a slope θ and distance ρ from the origin. The skew corresponds to the angle associated with a peak in the Hough space. The high computational complexity of the Hough transform often imposes restrictions on the possible angle range. Recently, three methods [15,34,38] were mentioned as being independent of these restrictions. Two of them [15,38] use a coarse-to-fine strategy, in which the angle range is first reduced to a narrow interval, and then the Hough transform is applied to the angles belonging to this interval. Narrowing the angle interval is done by a left margin search [8] in [15] and the Hough transform at a rough angular resolution in [38]. No prior document segmentation into text and graphics is required. The third method is the Weighted Hough transform [34], in which variable weights, whose values depend on the number of black pixels inside a given block, are used for accumulating the votes of the centers of gravity of the blocks. This method determines all necessary features at byte level without switching to individual bits. Pal and Chaudhuri [6,24,25] apply a clustering technique to prior detected representative points (uppermost and lowermost pixels of the connected components satisfying the predefined size conditions). As a result, they obtain two lines for each text row and compute the line slopes. The average of all slopes indicates the final skew. The clustering procedure relies on the fact that the points belonging to the same line cannot significantly deviate around this line. The suggested methods require that the text regions occupy a much larger area on a page than the non-text ones. Yan [36] uses an interline cross-correlation for two vertical lines located at a fixed distance d for skew estimation. With a fixed value for d , he computes the cross-correlation function in the region between these lines for pixels in positions (x,y) and $(x+d,y+s)$ by varying s . This function is maximized for $s=s_{opt}$ when the pixels in positions (x,y) and $(x+d,y+s_{opt})$ belong to the same text line. In this case, the desired skew angle is equal to $\arctan(s_{opt}/d)$. The cross-correlation function is computed for an entire image to obtain a better result. This can, however, be time-consuming and the presence of graphics degrades the accuracy. To overcome these problems, Chaudhuri and Chaudhuri [5] choose small windows randomly with a Monte Carlo sampling technique and calculate the correlation only for them by replacing the maximum of s with the median which is more robust. Gatos et al., [11] developed a method which can work with more than two vertical lines to improve the accuracy. The line position is also determined as a by-product of skew esti-

mation. However, all cross-correlation methods require large text regions on a page and the possible angle range is often restricted to speed up processing. Sauvola and Pietikäinen [29] and Sun and Si [35] apply gradient information for skew detection. Their works are based on the observation that the gradient orientation has to be mainly in a direction perpendicular to the text line. However, this approach requires gradient computation that is a time-consuming operation, except when the image resolution is low. Chen et al., [7] and Smith [30] detect a skew using least squares or least median squares techniques. Their methods require connected component detection and text line extraction. While Smith uses information on the proximity of two components belonging to the same line, Chen et al., process an image with recursive morphological opening and closing operations to extract the lines more automatically, though the computational cost of morphological operations is quite high. Fourier transform and neural networks are the basis of the method introduced by Rondell and Burel [28]. The method can be applied to a handwritten text, but it is quite slow due to the Fourier transform and convergence of the neural network learning procedure. Hashizume et al., [13] and O’Gorman [23] find the skew angle by using a nearest neighbor clustering technique. The basic idea is to find k ($k \geq 1$) nearest neighbors for each connected component, compute the angles of segments connecting these neighbors, and collect these angles into a histogram, its peak indicating the document skew. The nearest neighbor search may take much time, depending on image complexity and the number of neighbors to be found. Subparts of characters, noise and interline connections can reduce the accuracy of the method, though this technique is not limited to a specific angle range.

3 Method description

Our method comprises four main steps: 1) image resolution reduction, 2) connected component detection on the obtained low-resolution image, 3) component classification using fuzzy logic, and 4) skew estimation for non-graphic components.

3.1 Reduction of image resolution

We apply one of the simplest compression techniques (OR-rule) for fast processing. Compression with the OR-rule not only reduces the data size but also enables us to preserve the correct visual perception of image structure, especially of text lines. Let us assume that the original image consisting of black (1-valued or object) and white (0-valued or background) pixels is divided into squares of $M \times M$ pixels. According to the OR-rule, each squared block in the original image is replaced with one pixel in the compressed image. The value of this pixel is equal to 1 if there is at least one black pixel inside a given square, otherwise the value is set to 0. Implementation of compression by the OR-rule can be easily optimized without scanning the entire block. That is, when the first

black pixel inside a block is found, it is unnecessary to continue searching for other black pixels, but one can begin processing the next block. Only one parameter has to be set: the image resolution reduction coefficient. In our experiments, we used the value of M equal to 6, because the original image resolution was 300 dpi. To our knowledge, there are only a few techniques (e.g., [17, 31, 32, 38]) using compressed or low-resolution (50 dpi) image representations to determine the skew. However, either they are only designed for particular formats (JBIG and CCITT Group IV) and have limitations on the estimated angle range like in [17, 31, 32], or a processed document is scanned at a low resolution [38]. In the last case, although the method is quite fast (it takes 0.4 s on a SunSparc 20 workstation to process a 413×575 pixels image), we must scan each original document twice (once for skew detection at a low resolution and once for layout analysis and OCR at a high resolution). This is not convenient, if we need to process a bulk of documents automatically. In our method, the image is scanned only once at a high resolution (e.g., 300 dpi) and then we obtain the reduced image by using the OR-rule, while keeping the original one. The former is used for skew detection, while the latter can be employed for subsequent processing steps without extra scanning after skew correction. The memory space allocated for the reduced image is much smaller than that needed for the original image. With $M=6$, it is approximately equal to 1/36th of that needed for the original image.

The principal difference between scanning at 50 dpi (or another low resolution) and obtaining a 50 dpi image by applying the OR-rule is that in the first case, a connected component is usually a character or its part, while in the second case, it is mostly a word or several connected characters, i.e., it is elongated along a particular direction, depending on the skew angle. The way of reducing the resolution we chose is more suitable for our method (reasons will be given in Sect. 3.4). The effect of subsampling with the OR-rule is somewhat similar to that produced by smearing used in the RLSA [16] with a small smoothing threshold, because we aim at merging the neighboring characters belonging to the same word, while trying to keep separately the characters and words in different text lines. Which method is faster depends on the chosen image representation. When the image is an array of pixels (our case), the OR-rule seems to be faster because we do not need to analyze all pixels inside every block, i.e., we do not analyze all image pixels in contrast to the RLSA. However, if run lengths are used for coding of continuous runs of black and white pixels, the RLSA may be faster, because in this case, one only needs to analyze distances between adjacent runs or lengths of runs, which does not require pixel-based processing.

3.2 Detection of connected components

Next, the connected components are detected in the obtained low-resolution image. To separate the components already found from the unprocessed ones, the pixels of

each component are given a unique label. Each component is detected by applying a region growing procedure, where only the nearest vertical and horizontal neighbors of each black pixel are inspected. The area and perimeter of each component are also computed during this process because they are necessary for component classification. The area is defined to be the total number of pixels belonging to a given component, and the perimeter is the number of boundary pixels in it, respectively.

3.3 Classification of connected components

Almost every existing skew detection technique detects the skew angle by using only textual data. We also follow this because it improves the accuracy and reduces the processing time of skew estimation. Next, we have to choose a classification method in order to select useful data. This should be as fast as possible and simultaneously provide valuable information for the next steps, including page segmentation and classification. Those methods based on fuzzy logic [39] satisfy these conditions, and therefore we selected them for classification. One of the main benefits of fuzzy logic is that it is able to encode human knowledge easily and directly by using simple linguistic rules. However, it is necessary to design and tune the fuzzy membership functions that quantitatively define linguistic information.

The details of our fuzzy logic classifier implemented with the MATLAB toolbox [10] are shown in Figs. 1-4 (these pictures were copied from the corresponding MATLAB windows) and in Table 1. The classifier is a two-input, one-output system. The input variables we used are the area of the connected component (“area”) and the shape feature (“shape”) corresponding to the ratio of the area to the squared perimeter of the connected component. The output variable (“class”) is one of the four class labels: “TEXT”, “CHARACTER”, “GRAPHICS”, “LINE”.

Definition 1. “TEXT” - group of merged characters in text paragraphs.

Definition 2. “CHARACTER” - large character in the headings of the articles.

Definition 3. “LINE” - long (non-text) lines such as ruling lines.

Definition 4. “GRAPHICS” - large and often solid (due to binarization and subsampling with the OR-rule) object corresponding to binary graphics or binarized half-tone picture.

Those components of an area less than or equal to 3 pixels are considered to be noisy and are not processed further.

To label the connected components, the following rules are introduced, based on an analysis of images taken from various magazines:

Rule 1. If *area* is **small** AND *shape* is **elongated** then *class* is TEXT.

Table 1. Methods used in our fuzzy classification

Method's name	Type
AND	min
OR	max
Implication	min
Aggregation	max
Defuzzification	som

Rule 2. If *area* is **small** AND *shape* is **non-elongated** then *class* is CHARACTER.

Rule 3. If *area* is **large** AND *shape* is **elongated** then *class* is LINE.

Rule 4. If *area* is **large** AND *shape* is **non-elongated** then *class* is GRAPHICS.

The concepts “large”, “small”, “elongated”, and “non-elongated” are illustrated in Figs. 2-3. The membership functions chosen for input variables are sigmoids being described by the formula $f(x; b, c) = 1 / (1 + e^{-b(x-c)})$, where b and c are parameters. The parameter b controls the steepness of the sigmoid, while the parameter c corresponds to the fuzzy variable value for which the membership function is equal to 0.5. The sigmoids were selected because these curves represent concepts well, such as “very large” or “very small” and similar concepts used in our case. The values of b and c are shown in Table 2.

Table 2. Parameters of the membership functions for fuzzy input variables

Feature	b	c
Area is small	-0.01373	300
Area is large	0.01373	300
Shape is elongated	-500	0.061
Shape is not elongated	500	0.061

The choice of these values was not casual, but was derived from experiments. In Fig. 2, an area equal to 300 pixels separates the textual and non-textual data, while the shape feature set to 0.061 in Fig. 3 is the boundary between elongated and non-elongated objects (the lower the value of the shape feature is, the more elongated object is). The value of 0.061 was determined by computing the shape feature for a number of rectangles elongated along the X-axis, that is, being zero-skewed and having different width to height ratios, where the height was fixed, while the width altered. For each ratio, we computed the rectangle orientation until it became quite far from zero. The width and height values found gave an approximate estimation for which an object can be still considered as elongated along the X-axis. By averaging these estimations for different heights, while changing the width for each of them, we obtained the desired value for c ¹. The membership functions chosen for the output variable are triangles in order to simplify computations.

¹ In general, this value depends of the skew angle. We, however, omitted this fact in order to simplify calculations.

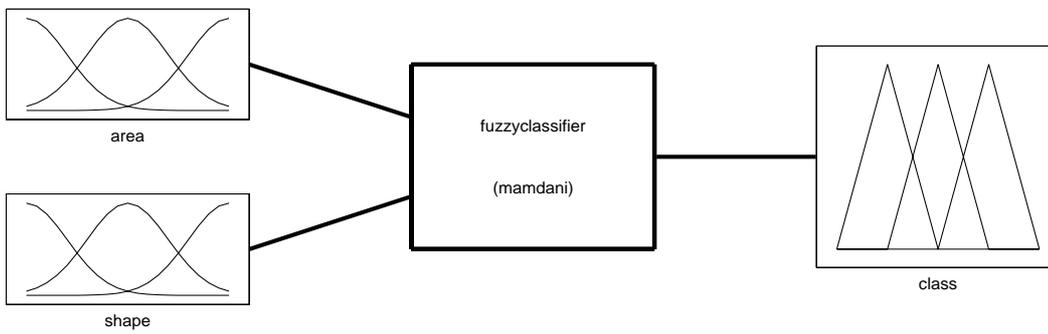


Fig. 1. Fuzzy logic classifier of the connected components

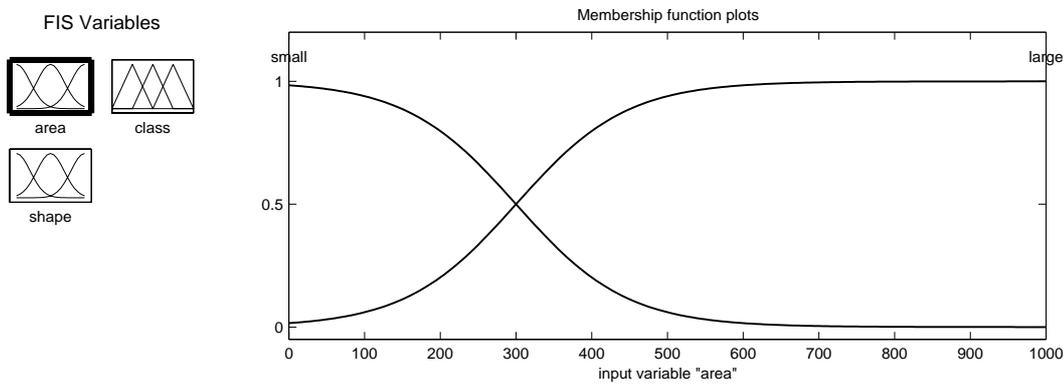


Fig. 2. Membership functions for the input variable “area”

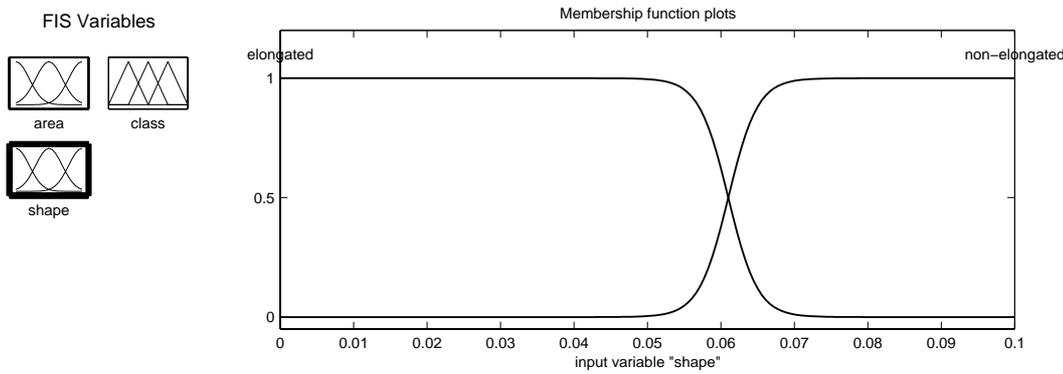


Fig. 3. Membership functions for the input variable “shape”

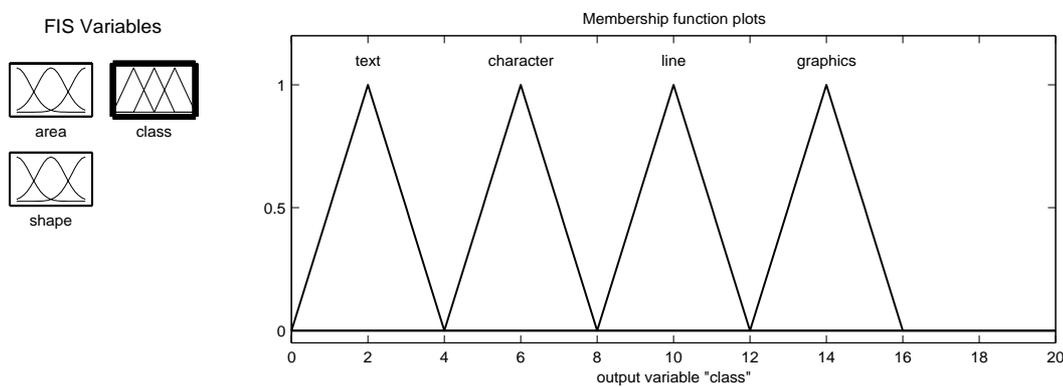


Fig. 4. Membership functions for the output variable “class”

3.4 Skew estimation

The skew is estimated in four optional ways. We will describe them in separate subsections for clarity. At first, we define the main operation used in each method: the determination of the first eigenvector of the input data covariance matrix, where the input data are the pixel coordinates of each connected component. The first eigenvector indicates the direction of maximum data deviation that is associated with the document skew, as in project profile techniques. The angle of the X-axis and the first eigenvector is the skew angle. Computation of the first eigenvector is a trivial task in our case, when it is necessary to resolve the 2nd-order equation, and we do not describe it here. This operation is applied to each connected component. We do not need to limit the angle range, due to the low computational cost of this operation because the 1st and 2nd-order moments are only necessary to compute and processing is done on the low-resolution image whose sizes are much smaller than those of the original high-resolution image. In addition, the skew estimation based on the first eigenvector works better for low-resolution images obtained by lowering a resolution by the OR-rule than the other methods do. This is because there are quite many connected components representing the words or their parts, and they are elongated along a direction close to the document skew. They can therefore give many votes to form a sharp peak in the angle histogram. In contrast, the Hough transform and the nearest neighbor search methods, for example, will fail on low-resolution images obtained with the OR-rule because (1) there will be too few representative points within each text line and the peak in the Hough space will be flat, and (2) the nearest neighbor for a given component can be often found in the adjacent but not in the same text line, which will also produce significant errors in skew estimation.

3.4.1 Method 1. The skew detection is only done for components labeled as “TEXT” or “LINE”. The “LINE” components are included in the computations, because the longer an object is, the more accurate estimation can be obtained by the technique presented in the previous section. The “CHARACTER” components are not processed since they are too large and usually non-elongated. All local angle estimations are accumulated into an angle histogram consisting of 180 or 360 bins, depending on the angular resolution, with the weight factor equal to 1. The angle corresponding to the maximum in the histogram (or the first maximum in the case of several equal maxima) is considered as the skew angle. An important point of this method, and of the methods described below, is that they use a 1D instead of a 2D accumulator necessary in Hough transform-based approaches. The size of this 1D array is very small in comparison with the image size, while this is not the case for the Hough transform methods, even if the real angle range is limited.

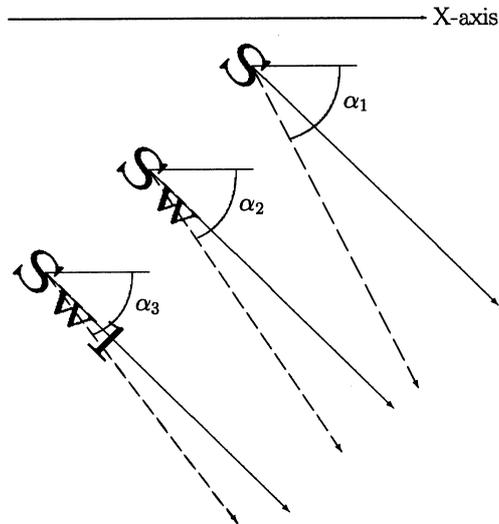


Fig. 5. Schematic explanation of the basic principle of Method 2. Three skewed hypothetical text lines consisting of 1, 2, and 3 characters are shown. Solid lines correspond to the real line orientation, while dashed lines show the approximate orientation by making use of the first eigenvector. One can see that $\alpha_3 < \alpha_2 < \alpha_1$

3.4.2 Method 2. This is performed after finishing Method 1, that is, after the connected component detection and local skew estimation for each component. The key idea is illustrated in Fig. 5. The aim is to collect as many connected components on the same line as possible by utilizing the first eigenvector orientation as an indicator. Each “TEXT” or “CHARACTER” component is used as a starting point for line accumulation. This is done to increase the number of votes in the 1D accumulator and to obtain a more accurate result. Each vote corresponds to the slope of the extracted line and it is weighted by the number of components included in this line. This kind of weighting sharpens the peak in the angle histogram. The skew angle corresponds to the maximum value of the histogram.

In Fig. 5, the estimated line orientations were not explicitly computed (though they are quite close to the practical values) and here they serve only for explanatory goals. One can see that the estimated orientation becomes closer and closer to the real one (and the angle of the X-axis and the first eigenvector, representing a line slope, becomes smaller and smaller) as the number of components in a line increases. The basic principle of Method 2 can be formulated as follows: *a component is considered to belong to a given line, if a line slope determined by the angle of the X-axis and the first eigenvector decreases after including this component in a line.* This rule is used for line extraction. Notice that the angles defining the line slopes are always positive and belong to $[0^\circ, 179^\circ]$.

We use the following notations to describe Method 2:
 C_i - the i -th connected component,
 xc_i, yc_i - coordinates of the center of gravity of C_i ,
 W, H - width and height of the bounding box of C_i ,

x_i^{left}, y_i^{top} - coordinates of the top-left point of the bounding box of C_i ,

$x_i^{right}, y_i^{bottom}$ - coordinates of the bottom-right point of the bounding box of C_i ,

L - set of components included in a given line,

$weight$ - the number of components in L (weight for voting),

S - set of the components being candidates for inclusion in L ,

α_i - local skew angle computed for C_i ,

α_L - current line slope,

α_{L+j} - new line slope after inclusion of C_j in L ,

α_{skew} - resulting skew angle,

$A[\cdot]$ - 1D accumulator.

With these definitions, the kernel of the method consists of the following steps:

Step 1.

Select $C_i \in \{\text{"TEXT"}, \text{"CHARACTER"}\}$;

$L = \emptyset; S = \emptyset; L = L \cup C_i$; /* include a component into a line */

$weight := 1$; /* set the number of components in L to 1 */

$\alpha_L := \alpha_i$; /* set the initial line slope */

Step 2.

Find all $C_j (j \neq i)$ satisfying the following conditions and include them in S , i.e., $S = S \cup C_j$.

(1) $C_j \notin L$,

(2) $C_j \in \{\text{"TEXT"}, \text{"CHARACTER"}\}$,

(3) $\left(x_i^{left} \text{ or } x_i^{right} \in \left[x_j^{left}, x_j^{right} \right] \text{ and } y_i^{top} \text{ or } y_i^{bottom} \in \left[y_j^{top}, y_j^{bottom} \right] \right) \text{ or } \left(x_j^{left} \text{ or } x_j^{right} \in \left[x_i^{left}, x_i^{right} \right] \text{ and } y_j^{top} \text{ or } y_j^{bottom} \in \left[y_i^{top}, y_i^{bottom} \right] \right)$,

where $x_i^{left} = xc_i - W; x_i^{right} = xc_i + W;$

$y_i^{top} = yc_i - H; y_i^{bottom} = yc_i + H.$

/* Condition (3) verifies if the bounding boxes of C_i and C_j either touch or intersect each other, or one of them is surrounded by another box */.

Step 3.

If $S = \emptyset$ then

/* No adjacent components were found in Step 2 */

{ /* Finish current line accumulation */ $A[\alpha_L] := A[\alpha_L] + weight$; go to Step 7; } else go to Step 4.

Step 4.

Find α_{L+j} as if $L = L \cup C_j \forall C_j \in S$ by using Formulae (1-3) (see Appendix 1 for their derivation).

$$\mu_{20,L+j} = \mu_{20,L} + \mu_{20,j} + \frac{n_L n_j}{n_L + n_j} (\bar{x}_L - \bar{x}_j)^2 \quad (1)$$

$$\mu_{02,L+j} = \mu_{02,L} + \mu_{02,j} + \frac{n_L n_j}{n_L + n_j} (\bar{y}_L - \bar{y}_j)^2 \quad (2)$$

$$\mu_{11,L+j} = \mu_{11,L} + \mu_{11,j} + \frac{n_L n_j}{n_L + n_j} (\bar{x}_L - \bar{x}_j)(\bar{y}_L - \bar{y}_j) \quad (3)$$

In these formulae, n is the object area, \bar{x} and \bar{y} are the 1st-order moments (mean values) and μ_{20}, μ_{02} , and μ_{11} are the 2nd-order central moments. The indices 'j', 'L', and 'L+j' correspond to the moments computed for C_j ('j') and L before ('L') and after ('L+j') inclusion of C_j in L , respectively.

Step 5.

Find α_{L+j^*} , where $j^* = \arg \min_j | \alpha_L - \alpha_{L+j} | \forall C_j \in S$. Calculate the new area and mean values with Formulae (4-6).

$$n_{L+j^*} = n_L + n_{j^*} \quad (4)$$

$$\bar{x}_{L+j^*} = \frac{n_L \bar{x}_L + n_{j^*} \bar{x}_{j^*}}{n_L + n_{j^*}} \quad (5)$$

$$\bar{y}_{L+j^*} = \frac{n_L \bar{y}_L + n_{j^*} \bar{y}_{j^*}}{n_L + n_{j^*}} \quad (6)$$

Step 6.

If $\alpha_{L+j^*} \leq \alpha_L$ then

{ /* This means that if a line slope becomes smaller, we have found a component located in the same line as the other components earlier included in the given line. */

$weight := weight + 1$; /* Increase the number of components in L */

$\alpha_L := \alpha_{L+j^*}$; /* Set a new line slope */

assign the values of the moments with the indices 'L+j' and 'L+j*' computed with Formulae (1-6) to those with the index 'L',

assume that the last-found component C_{j^*} will be used as C_i , and go to Step 2. }

else { /* Finish current line accumulation */ $A[\alpha_L] := A[\alpha_L] + weight$; go to Step 7; }

Step 7.

If there are still "TEXT" or "CHARACTER" components not being used as a starting point for line accumulation, then $i := i + 1$ and go to Step 1; else go to Step 8.

Step 8.

$\alpha_{skew} = \arg \max_{\alpha_k} (A[\alpha_k]), k \in [0^\circ, 179^\circ]$.
The important point with this method is that we do not need to recompute the 1st and 2nd-order moments (see Formulae (1)-(6)) each time when a new component is included in a line. All we need are these features for each component and they are computed only once which results in fast processing!

3.4.3 Method 3. This method finds the skew by using two angle histograms obtained using Methods 1 and 2. There are only two possible candidates for skew: these are the angles that are winners in each method. The number of votes for each candidate in both histograms is counted and the maximum of these two values indicates the final winner.

3.4.4 Method 4. This method is similar to the previous one; however, here we combine the votes of both histograms into the same histogram. Every angle represented in this joint histogram is a candidate for skew, and the global histogram maximum determines the desired angle.

4 Experiments

To test our method, we used the database UW-I of the University of Washington. We selected 647 of about 1000 binary images scanned at 300 dpi. The images from this database have been taken from a number of scientific magazines and contain both pure text information of various fonts/sizes and mixtures of text, graphics, tables, mathematical expressions, and pictures. A few images consist of exclusively non-textual blocks, except for figure captions. A quite common feature is that there are wide noisy black strips near page borders and some text lines are slightly curved at their ends due to a document copying process before digitization. We used two Hough transform based techniques [15,34] to compare our approach with others, because this transform is a general and popular tool for line extraction and skew detection. Another reason for this choice was that these techniques have no restrictions on detectable angle range. The method [15] is similar to [38], while the former is simpler for implementation than the latter, and that is why we did not use [38] in tests. In experiments, each tested image was rotated through every angle from 0° to 179° in 1-degree steps (that is, an additional 179 images were derived from each original image), and the skew was estimated and compared with the real angle value. As a result, we had $180 \times 647 = 116460$ images for testing. The angular resolution was set to 0.5° . We assumed that there was no error in skew estimation if the absolute difference of the real and determined angle was less than or equal to 1° , because we believe that modern page segmentation/classification methods can tolerate this value. Let us introduce the following notions before describing the experimental results. M1, M2, M3, and M4 stand for our four skew estimation methods described in Sect. 3.4. M5 and M6 denote the methods proposed in [15] and [34], respectively. Tables 3 and 4 show statistics calculated for all 116460 images. The number of correct skew estimations divided into three groups (0° -, 0.5° -, and 1° -differences Δ of the actual and found skew angles) and the total number of correct estimations for each tested method are given in Table 3, whereas Table 4 contains information on errors. For our approach, the results were obtained with the resolution reduction coefficient set to 6 and parameters of the fuzzy logic classifier chosen as described in Sect. 3.3. We did not reduce the original image resolution when testing M5 and M6. M5 had one predefined parameter: block height determined by the number of image rows (we used the value 100 as advised in [15]). M6 required three parameters: grid size (=24 pixels), weighting function type (=6), and the approximate number of text lines on a page (=1) (for details explaining these parameters, see [34]). The last

parameter was set to 1 because this value significantly varies from image to image, and it would be extremely difficult to determine it each time, especially when we process many images automatically with parameters set only once. In addition, no clear indications were presented in [34] as to how to choose the right parameter values, and we used here a trial-and-error method applied to several images.

By analyzing the data in Tables 3 and 4, one can see that M4 gave the highest score of correct skew estimations and the lowest number of errors. Methods M1–M3 had similar features, though slightly different. M5 produced about 6 times as many errors as M4, and M6 showed very bad results. As our main goal was to develop a method appropriate for automatic document processing, we have to define some criterion of judgement on the method. We assume that a given method can be used for automatic processing, if it gives no more than 5% of errors in the total number of estimations. Methods M1–M4 fully meet this criterion, and M5 approximately does. One reason why M6 shows bad performance is that fine parameter settings are necessary for it. When we adjusted the parameters for particular images, the results were much better. However, such manual tuning restricts its potential benefits, because M1–M5 gave very good results when the same parameters were used for all images. M5 produces errors when there is noise near to the left page border or text lines are not aligned along this border, that is, when there are many indents in paragraphs, and block boundaries coincide with indents. M1–M4 exhibited better than M5, because M5 relies on conditions that text lines have to be aligned along a left page border, which sometimes does not occur, while M1–M4 do not depend on this.

Error distributions for M1–M4 are shown in Figs. 6–9, where the number of errors and their distribution over the range $[0^\circ, 179^\circ]$ obtained from all 116460 images are accumulated in a single histogram for each method. In these figures, the angle range is displayed along the X-axis, while the number of errors is measured along the Y-axis. All distributions have two main peaks, at 2 and 179 degrees. One can see that the majority of errors is concentrated near peaks. There are errors for other angle values as well, but their number is very small (about 10 or a bit more) in comparison with the total number of estimations. These errors are not uniformly distributed over the range and their secondary concentration is around 90° .

At first glance, it seems that M1–M4 showed the same performance. However, the number of errors for M2–M4 can be greatly reduced if one analyses Table 5 presenting the number of errors obtained for the angles close to 180° and compares it with Table 4.

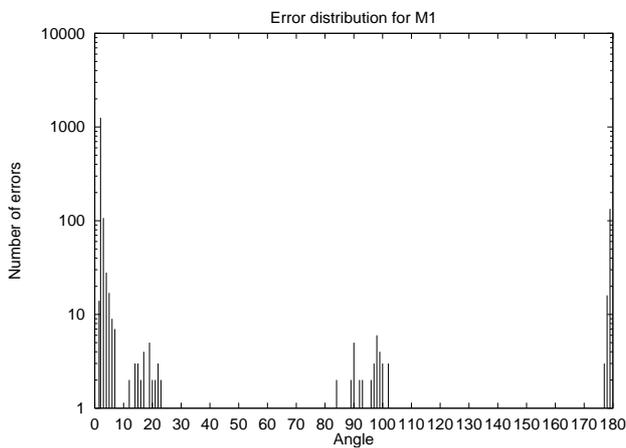
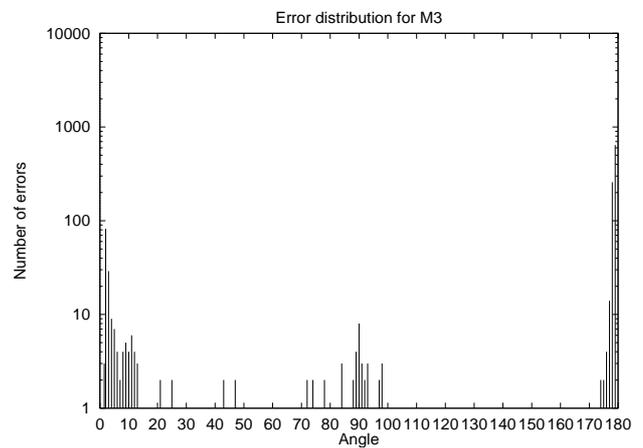
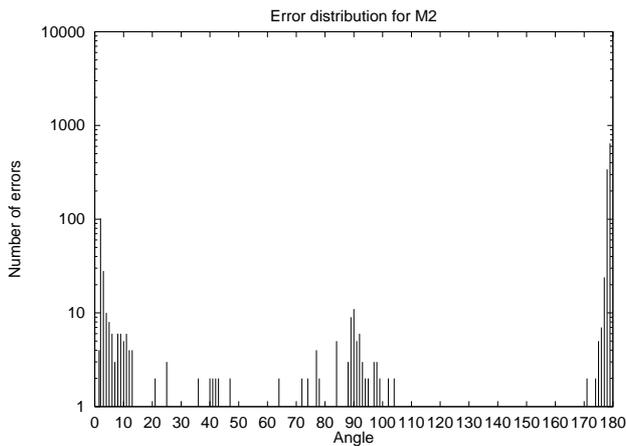
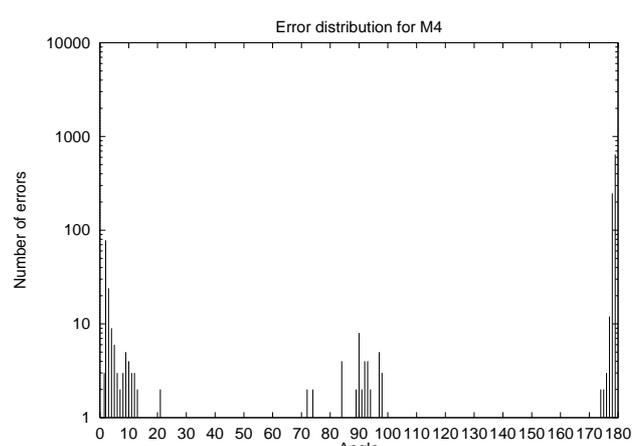
The significant number of errors equal to 178 or 179 degrees appears because it is often difficult to distinguish 0° and angles close to 180° using the techniques described in Sect. 3, and without any information on individual characters. In Table 5, the number of errors for M1 is much less than that for M2–M4. This is because M2–M4 use line extraction that can move the right estimation of 178° or 179° to 180° by fitting a line con-

Table 3. The number of correct skew estimations for all tested images

Δ	M1	M2	M3	M4	M5	M6
0°	93,874	105,898	107,485	107,550	70,335	13,708
0.5°	151	394	204	165	34,142	14,569
1°	20,727	8,794	7,595	7,612	5,349	4,136
TOTAL	114,752	115,086	115,284	115,327	109,826	32,413

Table 4. The number of errors of skew estimation for all tested images

	M1	M2	M3	M4	M5	M6
The total number of errors, nERR	1,708	1,374	1,176	1,133	6,634	84,047
Ratio (in %) of nERR to the total number of estimations	1.47%	1.18%	1.01%	0.97%	5.70%	72.17%

**Fig. 6.** Error distribution for M1**Fig. 8.** Error distribution for M3**Fig. 7.** Error distribution for M2**Fig. 9.** Error distribution for M4**Table 5.** The number of errors equaling to 178 and 179 degrees for all tested images when using methods M1–M4

Error	M1	M2	M3	M4
178°	16	340	258	247
179°	134	644	643	642

sisting of several physically separated components, and 180° is interpreted as 0° in our simulations. In contrast to them, M1 detects skew for each connected component separately, and in this case the chance of obtaining 180° instead of 178° or 179° is smaller. In our experiments, we did not consider how the results of M1 could influence the results of M2–M4. However, if we did so, then even in the worst case, when errors of 178 and 179 degrees occurred both for M1 and for any of M2–M4, we could

Table 6. Improved error rate for M2-M4, when errors of 178 or 179 degrees are eliminated by making decisions based on results for M1. Here, N_{new} and $N_{eliminated}$ stand for the new value of the total number of errors and the number of eliminated errors, respectively

	M1	M2	M3	M4
N_{new}	1,708	540	425	394
$N_{eliminated}$	0	834	751	739

obtain a much better error rate, as shown in Table 6 (derived from Tables 4 and 5), where the advantages of M2-M4 over M1 become visible. Here, we assumed that the final estimation is always 178 or 179 degrees, if it was so for M1, though estimations given by M2-M4 could be equal to 0.

Now let us explain why M4 behaves better than M2 or M3. M2 (as well as M1) has only one candidate for the desired skew angle, M3 makes its choice among two candidates, while M4 explores all possible opportunities (their number is equal to 180, 360 or even more, depending on the angular resolution). Yet, another reason is that M4 makes its decisions based on the histograms combining data from two different approaches. Estimations obtained from several experts, no matter whether the decisions of these experts are independent of each other or not, have been successfully used to solve various sophisticated tasks of pattern recognition such as handwritten text recognition, for example. In M4, a similar approach is applied and it behaves very well too. M1-M4 are not very robust when dealing with small interline spacings and large font sizes in the headings of the articles. A small interline spacing can lead to merging characters belonging to adjacent text lines after resolution reduction. If this effect dominates for a particular image, errors in skew detection are unavoidable. One way to overcome this is to use a smaller resolution reduction. Large font sizes may result in errors because characters of these sizes (even being merged together) will form rectangular-like components for which the technique based on the 1st eigenvector is not robust. A more robust statistical estimator could possibly alleviate this problem at the cost of processing time. Degrada-tions such as split or merged (within the same text line) characters and pepper noise seem not to have a negative effect on the performance of M1-M4. Pepper noise can be filtered out by analyzing the component’s area, while the quality of characters does not matter because we merge, that is, degrade them anyway during resolution reduction. The image content (pure text, mixture text and pictures, graphics, mathematical expressions, tables, etc.) influences, of course, on the accuracy of M1-M4, but this effect is not significant and the error distributions in Figs. 6-9 prove this. In fact, we successfully processed the images containing only from 1 to 3 text lines of figure captions. However, how much text information should be on the image for accurate skew estimation is not definitely clear. More research on this topic is necessary, including experiments with different font sizes.

Table 7. Average processing time for the methods M1-M6 in seconds

Operation	M1	M2-M4	M5	M6
Image unpacking	3.06	3.06	3.06	–
Image compression	0.81	0.81	–	–
Skew estimation	0.09	0.20	0.13	3.94

Finally, we will mention the average processing times for M1-M6 using a Pentium-200 Pro standalone computer and the images from the database of the University of Washington (see Table 7). The image sizes were about 2500×3500 pixels. Here “image unpacking” means a conversion from the input image format like BMP to an array of pixels. This operation had not been optimized because it was not the goal of our research. Notice that no unpacking was necessary for M6 because this method works with whole bytes (not the particular bits comprising them). “Image compression” means reducing the initial image resolution by the OR-rule. The significant processing time for M6 was due to the large sizes of the Hough space equal to 360 multiplied by the squared root of a length of the image diagonal. Of course, if we reduced the angular resolution, this time would be smaller.

5 Conclusions

In this paper, we proposed a new technique for global skew estimation. It is applied to binary document images of magazine pages and comprises four main steps: initial image resolution reduction, detection of connected components, classification of connected components, and skew estimation. A basic operation of skew estimation is the determination of the first eigenvector of the data covariance matrix. The angle between the X-axis and this eigenvector determines the skew.

The skew was found by four different techniques denoted as M1-M4 for short. M1 relies on accumulating local skew estimations, taken from the connected components, into an angle histogram and on detecting the angle associated with the histogram peak as the desired skew. M2 uses a similar approach to find the skew, but it tries to extract text lines and accumulate their slopes into the angle histogram instead of votes on the particular components. It differs from M1 by the principle of vote weighting: the weight is not constant and defined to be equal to the number of components included in a given line. M3 and M4 make decisions on the results obtained by M1 and M2. M3 supposes that there are only two possible candidates for skew and they are the angles selected as winners by M1 and M2, respectively, while M4 searches within an entire angle range.

The main novelty of our approach is a new fast line accumulation strategy applied to low-resolution (50 dpi) images that is able to find the lines of physically separated components. The standard Hough transform and its different modifications applied in computer vision tasks show unsatisfactory performance on thick lines that have a width greater than 1-2 pixels, or they often

cannot even detect these lines. We found only one reference to thick line detection based on the Hough transform [37], but the authors reported that their approach is even slower than the standard transform.

The main benefits of our approach include:

- no restrictions on the detectable angle range and processing time does not depend on the width of this range, unlike in many other approaches,
- skew estimation is quite fast, because compressed, low-resolution images are used for processing, and it is still accurate because we apply a skew detection technique which works better with the images of such a type,
- small 1D accumulator, the size of which is much less than the image sizes (usually 180 or 360 bins are quite enough), unlike in the Hough transform-based methods,
- large text areas on a page are not necessary, though the performance will be better if they are present.

The performance of our method was tested on the large data set containing over 116000 images and compared with two Hough transform-based methods [15,34]. The obtained experimental results (low error rate) clearly show the advantages of our approach over the two Hough transform methods.

The main problems that are necessary to solve are accurate skew estimations for (1) text lines containing the characters of large font sizes, (2) for sparse text lines, and (3) for text lines with small interline spacing.

A possible extension of our work is simultaneous text line extraction and skew estimation based on the principles described in this paper. This will allow us to process multiple skew in addition to global skew and probably can generalize our approach for document images of different classes.

Acknowledgements. The financial support of the Center for International Mobility (CIMO) of Finland and the Technology Development Center (TEKES) of Finland is gratefully acknowledged.

Appendix 1

Here, we derive the formulae used to compute new values of the 1st and 2nd-order moments for the first eigenvector determination when combining two objects with lower indices 'a' and 'b', respectively, into a single object with the lower index 'c'.

At first, we will define the basic formulae we will use (you can also see [12], for example). The formulae (A.1) and (A.2) define the moments of order (p+q) and the central moments of order (p+q), respectively. Here and further x_i and y_i denote x- and y-coordinates of the pixels.

$$m_{pq} = \sum_{i=1}^n x_i^p y_i^q \quad (\text{A.1})$$

$$\mu_{pq} = \sum_{i=1}^n (x_i - \bar{x})^p (y_i - \bar{y})^q \quad (\text{A.2})$$

where n is the object area and

$$\bar{x} = \frac{m_{10}}{m_{00}} = \frac{\sum_{i=1}^n x_i}{n}, \quad \bar{y} = \frac{m_{01}}{m_{00}} = \frac{\sum_{i=1}^n y_i}{n}.$$

The following central moments are necessary to calculate the first eigenvector.

$$\mu_{11} = m_{11} - \frac{m_{01}m_{10}}{m_{00}} = m_{11} - m_{00}\bar{x}\bar{y} = \sum_{i=1}^n x_i y_i - n\bar{x}\bar{y}$$

$$\mu_{02} = m_{02} - \frac{m_{01}m_{01}}{m_{00}} = m_{02} - m_{00}\bar{y}^2 = \sum_{i=1}^n y_i^2 - n\bar{y}^2$$

$$\mu_{20} = m_{20} - \frac{m_{10}m_{10}}{m_{00}} = m_{20} - m_{00}\bar{x}^2 = \sum_{i=1}^n x_i^2 - n\bar{x}^2$$

By using these formulae, we obtain new ones for the mean values (A.3-A.4) when combining two objects into a single one.

$$\begin{aligned} \bar{x}_c &= \frac{\sum_{i=1}^{n_a+n_b} x_i}{n_a+n_b} = \frac{\sum_{i=1}^{n_a} x_i}{n_a+n_b} + \frac{\sum_{i=1}^{n_b} x_i}{n_a+n_b} \\ &= \frac{n_a}{n_a+n_b} \frac{\sum_{i=1}^{n_a} x_i}{n_a} + \frac{n_b}{n_a+n_b} \frac{\sum_{i=1}^{n_b} x_i}{n_b} \\ &= \frac{n_a \bar{x}_a + n_b \bar{x}_b}{n_a+n_b} \end{aligned} \quad (\text{A.3})$$

The analogous formula holds for y.

$$\bar{y}_c = \frac{n_a \bar{y}_a + n_b \bar{y}_b}{n_a+n_b} \quad (\text{A.4})$$

The central moments are presented in (A.5-A.7).

$$\mu_{20,a} = \sum_{i=1}^{n_a} x_i^2 - n_a \bar{x}_a^2, \quad \mu_{20,b} = \sum_{i=1}^{n_b} x_i^2 - n_b \bar{x}_b^2$$

$$\mu_{02,a} = \sum_{i=1}^{n_a} y_i^2 - n_a \bar{y}_a^2, \quad \mu_{02,b} = \sum_{i=1}^{n_b} y_i^2 - n_b \bar{y}_b^2$$

$$\mu_{11,a} = \sum_{i=1}^{n_a} x_i y_i - n_a \bar{x}_a \bar{y}_a, \quad \mu_{11,b} = \sum_{i=1}^{n_b} x_i y_i - n_b \bar{x}_b \bar{y}_b$$

$$\begin{aligned}
\mu_{20,c} &= \sum_{i=1}^{n_a+n_b} x_i^2 - (n_a + n_b)\bar{x}_c^2 = \sum_{i=1}^{n_a} x_i^2 + \sum_{i=1}^{n_b} x_i^2 \\
&\quad - n_a\bar{x}_a^2 - n_b\bar{x}_b^2 + n_a\bar{x}_a^2 + n_b\bar{x}_b^2 - \\
&\quad (n_a + n_b)\bar{x}_c^2 = \mu_{20,a} + \mu_{20,b} + n_a\bar{x}_a^2 + \\
&\quad n_b\bar{x}_b^2 - \frac{(n_a\bar{x}_a + n_b\bar{x}_b)^2}{n_a + n_b} = \\
&\quad \mu_{20,a} + \mu_{20,b} + \frac{n_a n_b}{n_a + n_b} (\bar{x}_a - \bar{x}_b)^2
\end{aligned} \tag{A.5}$$

$$\mu_{02,c} = \mu_{02,a} + \mu_{02,b} + \frac{n_a n_b}{n_a + n_b} (\bar{y}_a - \bar{y}_b)^2 \tag{A.6}$$

$$\begin{aligned}
\mu_{11,c} &= \sum_{i=1}^{n_a+n_b} x_i y_i - (n_a + n_b)\bar{x}_c \bar{y}_c = \sum_{i=1}^{n_a} x_i y_i + \\
&\quad \sum_{i=1}^{n_b} x_i y_i - n_a \bar{x}_a \bar{y}_a - n_b \bar{x}_b \bar{y}_b + n_a \bar{x}_a \bar{y}_a + \\
&\quad n_b \bar{x}_b \bar{y}_b - (n_a + n_b)\bar{x}_c \bar{y}_c = \mu_{11,a} + \mu_{11,b} + \\
&\quad n_a \bar{x}_a \bar{y}_a + n_b \bar{x}_b \bar{y}_b - (n_a + n_b)\bar{x}_c \bar{y}_c = \\
&\quad \mu_{11,a} + \mu_{11,b} + n_a \bar{x}_a \bar{y}_a + n_b \bar{x}_b \bar{y}_b - \\
&\quad \frac{(n_a \bar{x}_a + n_b \bar{x}_b)(n_a \bar{y}_a + n_b \bar{y}_b)}{n_a + n_b} = \mu_{11,a} + \\
&\quad \mu_{11,b} + \frac{n_a n_b}{n_a + n_b} (\bar{x}_a \bar{y}_a + \bar{x}_b \bar{y}_b - \bar{x}_b \bar{y}_a - \bar{x}_a \bar{y}_b) \\
&\quad = \mu_{11,a} + \mu_{11,b} + \frac{n_a n_b}{n_a + n_b} (\bar{x}_a - \bar{x}_b)(\bar{y}_a - \bar{y}_b)
\end{aligned} \tag{A.7}$$

References

1. Akiyama, T., Hagita, N.: Automatic entry system for printed documents. *Pattern Recognition* 23:11, 1141–1154, 1990
2. Antonacopoulos, A., Ritchings, R.T.: Representation and classification of complex-shaped printed regions using white tiles. In: *Proc. 3rd Int. Conf. on Document Analysis and Recognition*, Montréal, Canada, 1995, pp.1132–1135
3. Baird, H.S.: The skew angle of printed documents. In: *Proc. SPSE 40th Symp. on Hybrid Imaging Systems*, Rochester, New York, 1987, pp.21–24
4. Bloomberg, D.S., Kopec, G.E., Dasari, L.: Measuring document image skew and orientation. In: Vincent, L.E., Baird, H.S. (eds.) *Document Recognition II*, Proc. SPIE 2422, San Diego, CA, 1995, pp.302–316
5. Chaudhuri, A., Chaudhuri, S.: Robust detection of skew in document images. *IEEE Trans. on Image Processing* 6:2, 344–349, 1997
6. Chaudhuri, B.B., Pal, U.: Skew angle detection of digitized Indian script documents. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 19:2, 182–186, 1997
7. Chen, S., Haralick, R.M., Phillips, I.T.: Automatic text skew estimation in document images. In: *Proc. 3rd Int. Conf. on Document Analysis and Recognition*, Montréal, Canada, 1995, pp.1153–1156
8. Dengel, A.: ANASTASIL: a system for low-level and high-level geometric analysis of printed documents. In: Baird, H.S., Bunke, H., Yamamoto, K. (eds.) *Structured Document Image Analysis*, Berlin Heidelberg New York: Springer-Verlag, 1992, pp.70–99
9. Etemad, K., Doermann, D., Chellappa, R.: Multiscale segmentation of unstructured document pages using soft decision integration. *IEEE Trans. on Pattern Recognition and Machine Intelligence* 19:1, 92–96, 1997
10. *Fuzzy Logic Toolbox for use with MATLAB: The MathWorks, Inc.* 1995
11. Gatos, B., Papamarkos, N., Chamzas, C.: Skew detection and text line position determination in digitized documents. *Pattern Recognition* 30:9, 1505–1519, 1997
12. Gonzalez, R.C., Woods, R.E.: *Digital Image Processing*. Reading MA: Addison-Wesley, 1993
13. Hashizume, A., Yeh, P.-S., Rosenfeld, A.: A method of detecting the orientation of aligned components. *Pattern Recognition Letters* 4:2, 125–133 (1986)
14. Hinds, S.C., Fisher, J.L., D’Amato, D.P.: A document skew detection method using run-length encoding and the Hough transform. In: *Proc. 10th Int. Conf. on Pattern Recognition*, Atlantic City, NJ, 1990, pp.464–468
15. Jiang, H.-F., Han, C.-C., Fan, K.-C.: A fast approach to the detection and correction of skew documents. *Pattern Recognition Letters* 18:7, 675–686, 1997
16. Johnston, E.G.: Short note: printed text discrimination. *Computer Graphics and Image Processing* 3:1, 83–89 (1974)
17. Kanai, J., Bagdanov, A.D.: Projection profile based skew estimation algorithm for JBIG compressed images. *Int. J. on Document Analysis and Recognition* 1:1, 43–51, 1998
18. Komukai, T.C., Saiwai, K.: Document skew detection based on local region complexity. In: *Proc. 2nd Int. Conf. on Document Analysis and Recognition*, Tsukuba, Japan, 1993, pp.49–52
19. Lam, S.W., Zandy, V.C.: Skew detection using directional profile analysis. In: *Proc. IAPR Workshop on Machine Vision and Application*, Kawasaki, Japan, 1994, pp.95–98
20. Le, D.S., Thoma, G.R., Weschler, H.: Automated page orientation and skew angle detection for binary document images. *Pattern Recognition* 27:10, 1325–1344, 1994
21. Messelodi, S., Modena, C.M.: Automatic identification and skew estimation of text lines in real scene images. *Pattern Recognition* 32:5, 791–810, 1999
22. Nakano, Y., Shima, Y., Fujisawa, H., Higashino, J., Fujinawa, M.: An algorithm for the skew normalization of document images. In: *Proc. 10th Int. Conf. on Pattern Recognition*, Atlantic City, NJ, 1990, pp.8–13
23. O’Gorman, L.: The document spectrum for page layout analysis. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 15:11, 1162–1173, 1993
24. Pal, U., Chaudhuri, B.B.: An improved document skew angle estimation technique. *Pattern Recognition Letters* 17:8, 899–904 (1996)
25. Pal, U., Chaudhuri, B.B.: Automatic separation of words in multi-lingual multi-script Indian documents. In: *Proc. 4th Int. Conf. on Document Analysis and Recognition*, Ulm, Germany, 1997, pp.576–579
26. Pavlidis, T., Zhou, J.: Page segmentation and classification. *Computer Vision, Graphics and Image Processing: Graphical Models and Image Processing* 54:6, 484–496 (1992)

27. Postl, W.: Detection of linear oblique structures and skew scan in digitized documents. In: Proc. 8th Int. Conf. on Pattern Recognition, Paris, France, 1986, pp.687–689
28. Rondell, N., Burel, G.: Cooperation of multi-layer perceptrons for the estimation of skew angle in text document images. In: Proc. 3rd Int. Conf. on Document Analysis and Recognition, Montréal, Canada, 1995, pp.1141–1144
29. Sauvola, J., Pietikäinen, M.: Skew angle detection using texture direction analysis. In: Proc. 9th Scandinavian Conf. on Image Analysis, Uppsala, Sweden, 1995, pp.1099–1106
30. Smith, R.: A simple and efficient skew detection algorithm via text row accumulation. In: Proc. 3rd Int. Conf. on Document Analysis and Recognition, Montréal, Canada, 1995, pp.1145–1148
31. Spitz, A.L.: Skew determination in CCITT Group 4 compressed document images. In: Proc. Symp. on Document Analysis and Information Retrieval, Las Vegas, NV, 1992, pp.11–25
32. Spitz, A.L.: Analysis of compressed document images for dominant skew, multiple skew, and logotype detection. *Computer Vision and Image Understanding* 70:3, 321–334, 1998
33. Srihari, S.N., Govindaraju, V.: Analysis of textual images using the Hough transform. *Machine Vision and Applications* 2:3, 141–153, 1989
34. Sugawara, K.: Weighted Hough transform on a gridded image plane. In: Proc. 4th Int. Conf. on Document Analysis and Recognition, Ulm, Germany, 1997, pp.701–704
35. Sun, C., Si, D.: Skew and slant correction for document images using gradient direction. In: Proc. 4th Int. Conf. on Document Analysis and Recognition, Ulm, Germany, 1997, pp.142–146
36. Yan, H.: Skew correction of document images using interline cross-correlation. *Computer Vision, Graphics and Image Processing: Graphical Models and Image Processing* 55:6, 538–543, 1993
37. Yang, M.C.K., Lee, J.-S., Lien, C.-C., Huang C.L.: Hough transform modified by line connectivity and line thickness. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 19:8, 905–910, 1997
38. Yu, B., Jain, A.: A robust and fast skew detection algorithm for generic documents. *Pattern Recognition* 29:10, 1599–1629, 1996
39. Zadeh, L.A.: Fuzzy sets. *Information and Control* 8, 338–353, 1965

Oleg Okun received the Ph.D. degree in Engineering Science in 1996 from the Institute of Engineering Cybernetics of National Academy of Sciences, Republic of Belarus. Since 1998 he joined the Department of Electrical Engineering and Infotech Oulu of Oulu University, Finland, where he is currently a researcher. He received research grants from Soros Foundation in 1995 and Center for International Mobility of Finland in 1997. His research interests include document image processing and recognition, fuzzy logic, and neural networks. He has over 20 published papers.

Matti Pietikäinen received his Doctor of Technology degree in Electrical Engineering from the University of Oulu, Finland, in 1982. Currently he is Professor of Information Technology, Scientific Director of Infotech Oulu research center, and Director of Machine Vision and Media Processing Unit at the University of Oulu. From 1980 to 1981 and from 1984 to 1985 he was visiting the Computer Vision Laboratory at the University of Maryland, USA. His research interests cover various aspects of machine vision, including texture analysis, color vision and document analysis. Prof. Pietikäinen is Fellow of International Association for Pattern Recognition (IAPR) and Senior Member of IEEE, and serves as Member of the Governing Board of IAPR. He also serves on program committees of several international conferences.

Jaakko Sauvola is a Professor and Director of MediaTeam research group in the University of Oulu, Finland, and a member of affiliated faculty at the LAMP laboratory, Center for Automation Research, University of Maryland, USA. Dr. Sauvola is also a Research Manager in Nokia Telecommunications, where his responsibilities cover value adding telephony services. Dr. Sauvola is a member of several scientific committees and programs. His research interests include computer-telephony integration, media analysis, mobile multimedia, media telephony and content-based retrieval systems.