Image and Vision Computing 20 (2002) 25–35

www.elsevier.com/locate/imavis

# Efficient skew estimation and correction algorithm for document images

H.K. Kwag*, S.H. Kim, S.H. Jeong, G.S. Lee

*Department of Computer Science, Chonnam National University, 300 Yongbong-Dong, Bug-Gu, Kwangju 500-757, South Korea*

Received 11 May 2000; received in revised form 28 March 2001; accepted 6 June 2001

**Abstract**

In this paper, we propose a fast skew estimation and correction algorithm for English and Korean documents based on a BAG (Block Adjacency Graph) representation. BAG is one of the most efficient data structures for extracting various information concerning connected components; the image rotation for skew correction is performed rapidly using the block information in the BAG. The proposed skew estimation algorithm uses a coarse/refine strategy based on the Hough transformation of connected components in the image. The skew correction algorithm then generates a non-skew image by rotating the blocks, rather than the individual pixels. An experiment using 2016 images from various English and Korean documents demonstrates how the proposed method is superior to conventional ones. © 2002 Elsevier Science Ltd All rights reserved.

*Keywords:* Skew estimation; Skew correction; BAG; Hough transform; Block rotation

## 1. Introduction

The volume of paper-based documents continues to grow at a rapid rate in spite of the use of electronic documents. As a result, both the transformation of a paper document to its electronic version, and its subsequent image processing and understanding have become an important application domain in computer vision and pattern recognition researches. Document analysis and character recognition are usually performed through several phases: scanning, image enhancement, skew estimation and correction, segmentation and character classification. The skew estimation and correction of document images is particularly crucial among the document processing operations as it affects the subsequent understanding of the document.

Document skew has been recognized as a universal problem of document imaging. Hand placement or automatic document feeder mechanisms normally create 1–3° of skew, due to the document's incorrect placement, or to a slight variation in roller speed. In some cases, the skew can reach as much as 10°. When the skew angle is 2–3°, the accuracy of document analysis and OCR is reduced; when it is more than 5°, however, the result becomes unreliable. As a result, the skew estimation and correction of document images needs to be carried out before segmentation and classification.

Many skew estimating techniques have been developed during the last several decades. These can be classified into three general categories: projection profile methods [1–3], Hough transform methods [4–10] and nearest-neighbor methods [11–13]. Projection profile methods are the commonly used techniques, and usually work well for text-only documents. Hough transform methods can also achieve high estimation accuracy. A common weakness to both of these approaches, however, is that the computational complexity is proportional to the degree of desired accuracy. Most projection profile and Hough transform methods thus limit their detection ranges, typically to within ±15°. The nearest-neighbor clustering methods also have a relatively high accuracy, but they are costly, especially when there are many connected components involved. Some other approaches using Fourier transform [1], neural networks [14], or cross-correlation [15,16] methods have also been reported in the literature.

Once the skew angle of a document image has been estimated, the image should then be corrected to generate a non-skew version. Most skew correction methods reported so far are pixel-oriented, i.e. each pixel in the original image is rotated in the output image. Pixel-oriented methods can be categorized into direct [7,17,18] and indirect [1,2,5,6,8] methods depending on the type of rotational equation used. The direct method has the drawback of rounding

* Corresponding author. Center for Artifical Intelligence, Korea Advanced Institute of Science and Technology, 373-1 Kusung-Dong, Yusung-gu, Taejon, 305-701 Korea, South Korea. Tel.: +82-62-530-0430; fax: +82-62-530-3439.

*E-mail address:* hkkwag@ai.kaist.ac.kr (H.K. Kwag).

0262-8856/02/$ - see front matter © 2002 Elsevier Science Ltd All rights reserved.
PII: S0262-8856(01)00071-3

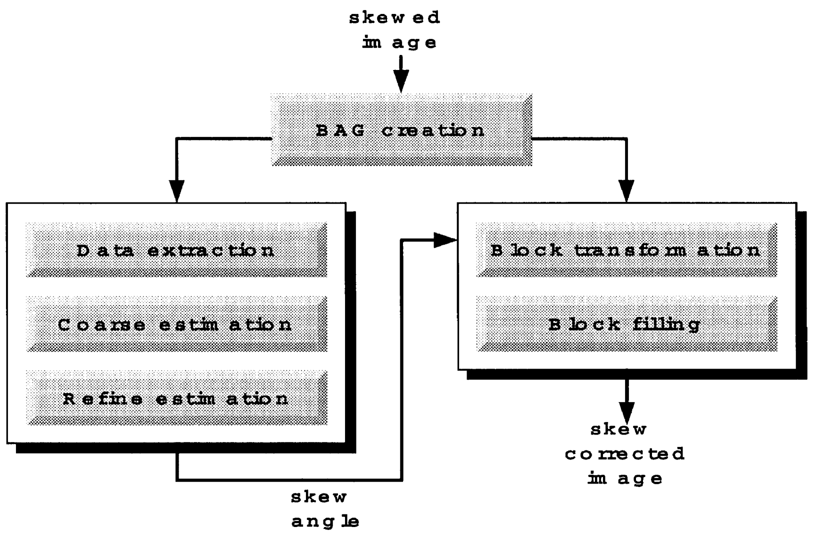

Fig. 1. Block diagram of the proposed skew estimation and correction algorithm.

problem, which produces undesirable holes in the skew-corrected images. The indirect method, on the other hand, does not have a rounding problem, but it is a time-consuming process, as every pixel needs to be considered. Another approach based on object contour has been published, but there still remains a rounding problem [19].

In this paper, we describe a fast skew estimation algorithm based on a BAG (Block Adjacency Graph) representation of binary document images. The BAG is an efficient data structure for extracting both the contours of the connected components in the image, and the lower and upper center points of the surrounding bounding boxes. The skew estimation algorithm consists of three basic steps: data extraction, coarse estimation and refine estimation. In the first step, some connected components corresponding to noise, drawings or tables are filtered out, based on the size of their bounding boxes. In the second step, the entire image is divided into several rectangular regions, and one of these is then chosen as a region of interest, along with a rough estimation angle $\theta_C$ for the region. In the third step, a Hough transform is applied to the lower (or upper) center points of the bounding boxes in the selected region to compute a final skew angle. The search range of the Hough transform is confined to $\theta_C \pm 1°$.

We also propose a fast skew correction algorithm using the block in the BAG. Since the rotational transformations for skew correction are performed with the block unit, the processing speed of our method is faster than that of pixel-oriented methods. The proposed algorithm consists of two basic steps: block transformation and block filling. In the first, the four boundary coordinates of each block are transformed using a rotational matrix. In the second, the pixels inside the rotated block are identified. Fig. 1 shows a block diagram of the proposed skew estimation and correction algorithm.

This paper is organized as follows: a review of the related work is contained in Section 2; the details of our skew estimation and correction algorithm are described in Sections 3 and 4; some experimental results with various English and Korean document images are discussed in Section 5; and our conclusive remarks are given in Section 6.

## 2. Related work

### 2.1. Skew estimation

#### 2.1.1. Projection profile methods

The projection profile is the most popular method for skew estimation. Its straightforward use computes profiles at a number of angles close to the expected orientation [1]. For each angle, a measure is made for the variation in the bin heights along the profile, and the one with the maximum variation provides the skew angle. Since scan lines are aligned to text lines, the projection profile has maximum-height peaks for text, and valleys for between-line spacing at the correct skew angle. Baird [2] has proposed a modified projection profile method, which improves the speed and accuracy of skew estimation. A connected component (CC) analysis is first applied to the document and then the bottom midpoint of each CC is projected onto an imaginary accumulator line perpendicular to the different projection angle. A measurement of the total variation, such as peak-to-valley difference in height, is determined for the various projection angles. The peak of these measurements comes very close to the true skew angle. The estimation error of this method is $\pm 0.5°$. Akiyama and Hagita [3] divide a page into columns and then calculate horizontal projection profiles for each column. Each peak corresponds to text line in the column. The skew angle is obtained by calculating the arc-tangent connecting the peaks of the adjoining projection profiles.

### 2.1.2. Hough transform methods

Hough transform is a well-known technique for detecting lines and curves in an image. In order to find the skew from text-line components, a set of points in Cartesian space $(x - y)$ are mapped to sinusoidal curves in the Hough space $(\rho - \theta)$ using the following transform:

$$\rho = x\cos\theta + y\sin\theta.$$

An accumulator array is then used to count the number of intersections at various $\rho$ and $\theta$ values. The cells with high values in the accumulator array correspond to lines in the original image.

Srihari and Govindaraju [4] use the Hough transform technique on all black pixels, and their method suffers from the extreme computational load. Hinds et al. [5] apply a vertical run-length analysis for images. A gray-scale burst image is created from the black runs that are perpendicular to the text lines by placing the length of the run in the bottom-most pixel. The Hough transform is then applied to this burst image. Unlike the standard approach of incrementing the accumulator cells by one, the cells are incremented by the value of the burst image. Le et al. [6] apply a CC analysis to the original image. Large CCs such as images, or small CCs such as noise, are excluded from the calculation to minimize the estimation error. The pixels of the last black runs of each CC are then extracted, and the Hough transform is applied only to these pixels. Yu and Jain [7] propose an algorithm based on a hierarchical Hough transform. This algorithm extracts the centroids of CCs based on a graph data structure, called a BAG, and detects the skew angle by using a hierarchical Hough transform to perform the skew estimation at two different angular resolution levels.

In the related works, we can observe the following two aspects for improving the speed and accuracy of the skew estimation. Most of these methods attempt to minimize the effect of the non-text elements, such as noise, images and tables. They also group individual pixels into CCs, runs, or blocks, to extract meaningful data values.

### 2.1.3. Nearest-neighbor methods

In these approaches, connected components are first prepared. The nearest neighbor of each component is then found, and the angle between the centroids of the nearest-neighbor components is calculated. All angles for the nearest-neighbor connections are accumulated in a histogram, and the histogram peak indicates the dominant direction—that is, the skew angle [11]. An extension of this nearest-neighbor approach is to obtain not just one neighbor for each component but $k$ neighbors, where $k$ is typically 4 or 5 [12]. A new skew detection algorithm that focuses the clustering process to a subset of plausible candidates from all nearest-neighbors has been proposed [13]. A least-square line fitting is performed on these plausible neighbors, and the skew angle associated with the computed straight line is used to build up a histogram. The peak in the histogram is regarded as the skew angle of the input document image.

## 2.2. Skew correction

### 2.2.1. Direct method

Given a skew angle $\theta$, the direct method de-skews the image by rotating the black pixels by $(-\theta)$ [7,17,18]. A black pixel $p$ in the input image is transformed into $p'$ by multiplying the coordinates of $p$ by a rotational matrix:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos(-\theta) & -\sin(-\theta) \\ \sin(-\theta) & \cos(-\theta) \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix},$$

where $(x,y)^t$ are the coordinates of $p$ in the input image, and $(x',y')^t$ are the coordinates of $p'$ in the output image. One of major weaknesses of this method is a rounding problem in the discrete plane, where neighboring connected components are merged and isolated components are split. As a result, some illegal holes may appear inside the skew-corrected objects. This undesirable transformation causes a number of problems in the subsequent document processing.

### 2.2.2. Indirect method

The indirect method is the opposite of the direct method [1,2,5,6,8]. For a pixel $(x',y')^t$ in the goal image, the indirect method finds the corresponding pixel $(x,y)^t$ in the original image, and then sets a value of $(x',y')^t$ with that of $(x,y)^t$. The correspondence is computed by applying the inverse rotational matrix to $(x',y')^t$:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \cdot \begin{pmatrix} x' \\ y' \end{pmatrix}.$$

This method can resolve the rounding problem, but it suffers from a computational burden incurred from considering all the pixels in the goal image. In this context, the indirect method produces better images, but is more time-consuming than the previous method.

### 2.2.3. Contour-oriented method

In the contour-oriented method [19], the de-skewing is applied to the connected components using their contour descriptions. Every connected component can be defined with its color, start corner, and contour-string. The contour-string is a description of the connected component boundary. A real rotation for a connected component is carried out by: (1) computing the location of the start corner in the rotated image; (2) rotating every contour edge in the contour-string; and then (3) computing the final new position of the start corner and rebuilding the contour-string. This is an efficient method in that it rotates only the relevant information; due to rounding mistakes, however, the contour-string must always be closed manually.

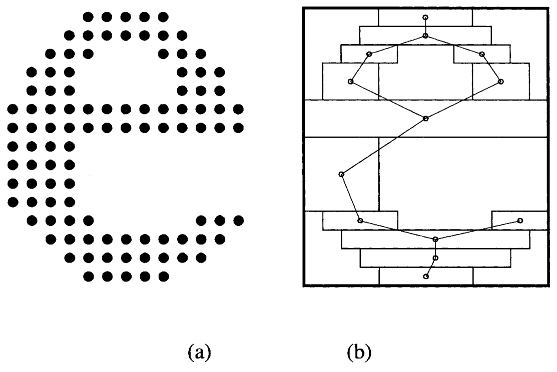

Fig. 2. The creation of a BAG and its bounding box: (a) a connected component; (b) its BAG and bounding box.

### 2.3. Block adjacency graph

A text line is a group of characters, symbols, and words that are adjacent, relatively close to each other, and through which a straight line can be drawn. Most previous skew estimation methods, therefore, have made it a rule to undertake a connected component analysis before the estimation. Yu et al. [7] describe a BAG to extract the contour of the connected components in a binary image. The graph data structure is already known as simple iterative procedure and can be created concurrently when the document is being scanned row by row. Therefore, we do not need a large amount of processing time. Instead, some amount of memory space to store the block information is needed. Fig. 2 shows a connected component and its BAG, in which 13 blocks form a bounding box. Note that the block is defined as a set of one or several runs connected adjacently, and aligned exactly on both sides. An algorithm for BAG creation is given in Fig. 3. In the algorithm, the numbers of blocks can be reduced by given a tolerance instead of zero, and then we can speed up the processing time.

The BAG is an efficient data structure for skew estimation since the connected components and their bounding rectangles are easily obtained by searching the graph. Also, it is very efficient to use block information for skew correction, since a rotational transformation is more quickly performed with the block unit [21].

## 3. Skew estimation algorithm

The proposed skew estimation algorithm uses the Hough transform. The performance of Hough transform method, however, is very sensitive to estimation accuracy and range. To overcome this problem, we consider three factors which influence both the speed and accuracy of the skew estimation. First, some non-text components, such as noise, tables, figures, etc. are excluded from the estimation. Second, the Hough transform is applied only to a set of lower or upper center points of the bounding boxes, instead of to the individual pixels or runs. Third, a coarse estimation technique is embedded before determining the final skew angle of the document to reduce the search range of the Hough transform. The proposed algorithm consists of three steps: data extraction, coarse estimation, and refine estimation.

### 3.1. Data extraction step

The more data used in the skew estimation, the better the result will be, but the longer the process will take. The selection of small but representative data from the document is therefore extremely important in the skew estimation. A data extraction step excludes the non-text CCs and chooses a minimum possible number of text components to be used for the skew estimation.

A bounding box is a minimum rectangle surrounding a CC. Let us first define a vertical bounding box whose height is greater than its width. In English documents, most of isolated (non-touched) characters form a vertical bounding box, except for 'w', 'm', etc. Also, the vertical bounding boxes for a series of characters in a text line are aligned along the baseline, except for 'g', 'p', etc. We can therefore take a lower center point of the vertical bounding box for the isolated character as the representative data.

In case of Korean documents, a character has two or more bounding boxes because it is composed of consonants and vowels. Also, we have observed that most Korean characters have a vertical bounding box, and the vertical bounding boxes from a series of characters in a text line are aligned along the upper text line. So, we can also take an upper

```
Each run in the first row of the input image is regarded as a block.
For each successive row in the image {
    For each run $r_c$ in the current row {
        If $r_c$ is 8-connected to a run in the preceding row {
            If $r_c$ is 8-connected to only one run $r_l$ and the difference of the horizontal
positions of their beginning and ending pixels are both zero, then $r_c$ is merged into the block
involving $r_l$.
            Else, $r_c$ is regarded as a new block, initialized with links to those blocks,
which are 8-connected to $r_c$.
        }
        Else, $r_c$ is regarded as a new block, initialized with no links.
    }
}
```

Fig. 3. Algorithm for generating a BAG.

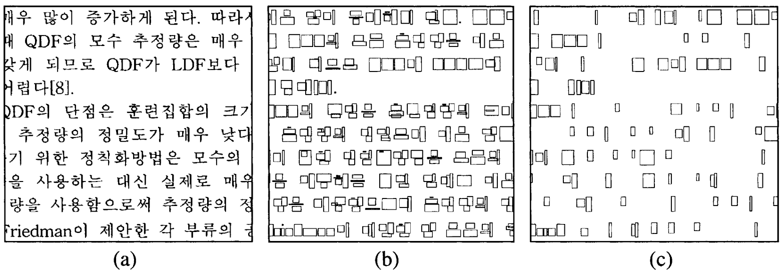
우 많이 증가하게 된다. 따라
QDF의 모수 추정량은 매우
게 되므로 QDF가 LDF보다
렵다[8].
DF의 단점은 훈련집합의 크기
추정량의 정밀도가 매우 낮다
기 위한 정칙화방법은 모수의
을 사용하는 대신 실제로 매우
량을 사용함으로써 추정량의 정
riedman이 제안한 각 부류의
(a)
(b)
(c)

Fig. 4. An example result of the data extraction step: (a) original image; (b) bounding boxes; (c) selected bounding boxes.

center point of the vertical bounding box from the Korean character as the representative data.

According to these observations, a data extraction step produces a set of points as the representative data for a given document image—the lower and upper center points of the vertical bounding boxes. The parameters used in the data extraction step are:

| | |
|---|---|
| $h_B$ | Bounding box height |
| $w_B$ | Bounding box width |
| $h_{freq}$ | The most frequent height of the vertical bounding boxes |
| $T_{min}$ | The minimum threshold of a vertical bounding box |
| $T_{max}$ | The maximum threshold of a vertical bounding box |

We first construct a height frequency array acc_H[ ], whose elements are initially set to zero, by applying the following condition for every bounding box:

if ($h_B > w_B$) increment acc_H[$h_B$] by one.

The maximum value in the array is given for the most frequent height of the vertical bounding boxes, $h_{freq}$. The $h_{freq}$ is used to describe the dominant height information of texts in a document image. As a result, the two thresholds, $T_{min}$ and $T_{max}$, are set to ($T_{min} = h_{freq} \times 0.5$) and ($T_{max} = h_{freq} \times 1.5$), respectively. If the height of a bounding box, $h_B$, does not exist within the range of [$T_{min}$,$T_{max}$], we then remove the bounding box from further processing.

Fig. 4(b) shows a set of bounding boxes created from a Korean document image in Fig. 4(a). We can observe that a Korean character is composed of two or more bounding boxes. The vertical bounding boxes selected by our algorithm are given in Fig. 4(c). The lower and upper center points of the remaining vertical bounding boxes are therefore extracted as the salient features for the skew estimation.

## 3.2. Coarse estimation step

The objective of this step is to reduce the search range of the Hough transform in the refine estimation. It produces: (1) a coarse skew angle $\theta_C$ measured within a range of $\pm 45°$ with a $\pm 1°$ of maximum error; (2) a region of interest which gives the coarse skew angle; and (3) a flag designating

```
(1) For each bounding box $B_i$ {
  (2) Find the corresponding region $R_k$ for $B_i$. ($1 \le k \le K$)
  (3) For each bounding box $B_j$ in the region $R_k$ {
    (4) Compute the angle between the lower center points of $B_i$ and $B_j$.
        $Q_{lp} = \tan^{-1}\left(\frac{y_{lj} - y_{li}}{x_{lj} - x_{li}}\right)$
        increment acc_lp[$R_k$][ $Q_{lp}$] by one. (acc_lp: the lower accumulator)
    (5) Compute the angle between the upper center points of $B_i$ and $B_j$.
        $Q_{up} = \tan^{-1}\left(\frac{y_{uj} - y_{ui}}{x_{uj} - x_{ui}}\right)$
        increment acc_up[$R_k$][$Q_{up}$] by one. (acc_up: the upper accumulator)
      }
}
(6) Find the maximum in the $2 \times K$ accumulators.
(7) Determine the coarse skew angle $\theta_C$, a region k, and a flag $l$ (lower) or $u$ (upper).
```

Fig. 5. An algorithm for the coarse estimation step.

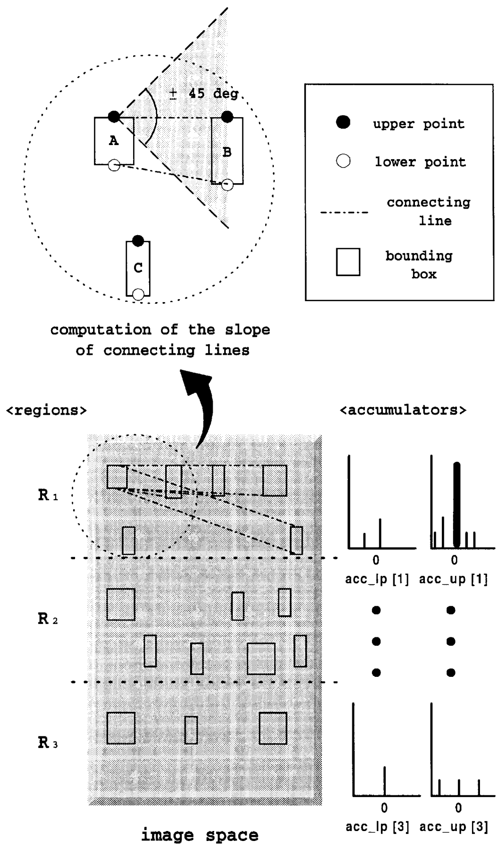

Fig. 6. An example of the coarse skew estimation step.

whether the coarse skew angle is obtained from the lower or upper points of the bounding boxes.

To begin with, we divide the entire image into $K$ regions; each region contains $m$ text lines. If the $m$ is too large, the computational burden increases. If the $m$ is too small, the estimation accuracy can decrease. We determine ($m = 10$) through trial and error. The height of each region, $H$, and numbers of regions, $K$, are determined as:

$H = \mu_h \times 2 \times m$ ($\mu_h$: the average height of the vertical bounding boxes)

$K = \lceil(\text{the height of the entire image})/H + 0.5\rceil$
$K$ regions: $R_1, R_2, \ldots, R_K$
$R_1$: $[0,H)$, $R_2$: $[H,2H), \ldots, R_K$: $[(K-1)H, KH)$.

A region $R_k$ ($1 \le k \le K$) therefore refers to a sub-area of the image consisting of the scan lines from $(k-1)H$-th row to $kH$-th row.

The coarse skew angle is estimated locally for each region within a range of $\pm 45°$ with $\pm 1°$ of maximum error. Given a region $R_k$, we first consider only the lower center points of the vertical bounding boxes in this region. The slope $\theta$ of the line connecting any two lower points are measured, and then the lower accumulator element acc_lp[$k$][$\theta$] is incremented by one if $\theta$ is within the range of $\pm 45°$. The value of slope $\theta$ is then truncated to an integer to facilitate the frequency counting. So, the maximum error of estimation is $\pm 1°$.

The same work is performed with the upper center points of the bounding boxes in $R_k$ to complete the accumulator array acc_up[$k$]. After all the regions are examined, $2K$ accumulators are constructed. The maximum frequency in the $2K$ accumulator arrays is searched to produce the final output. For example, if the maximum frequency is found in acc_lp[$k$][$\theta_C$], the final coarse skew angle is $\theta_C$, the region of interest is $R_k$, and the flag is set to $l$. On the contrary, if the maximum frequency is found in acc_up[$k$][$\theta_C$], the flag is set to $u$. The flag $l$ means that the lower points are the more reliable data for the next step, and the flag $u$ means the opposite. The algorithm for the coarse estimation is given in Fig. 5.

Fig. 6 shows an example of coarse skew estimation. Assume that three vertical bounding boxes, A, B and C, exist in region $R_1$. The local skew angles between the lower and upper center points of A and B are used to increment the accumulator arrays. The connecting lines between A and C, however, are not considered, since bounding box C is out of the $\pm 45°$ range from A. As can be seen from Fig. 6, regions $R_1$ and $R_2$ contain six and seven bounding boxes, respectively. The maximum is found in the upper accumulator of region $R_1$, acc_up[1][0]. We therefore determine the

```
$R_k$: the region,
$\theta_C$: the skew angle, and
$u$ (or $l$): the flag determined by the coarse step.
(1) For all upper (or lower) points $(x, y)$ in the region $R_k$ {
  (2) For all $\theta$ in the range of $\theta_C - 1 \le \theta \le \theta_C + 1$ {
    (3) $\rho = x\cos\theta + y\sin\theta$
    (4) Increment the cell of the accumulator acc [$\rho$][$\theta$] by one.
    (5) $\theta$ += 0.1
  }
}
(6) Find the maximum in the array, acc [$\rho$][$\theta$].
```

Fig. 7. An algorithm for the refine estimation step.

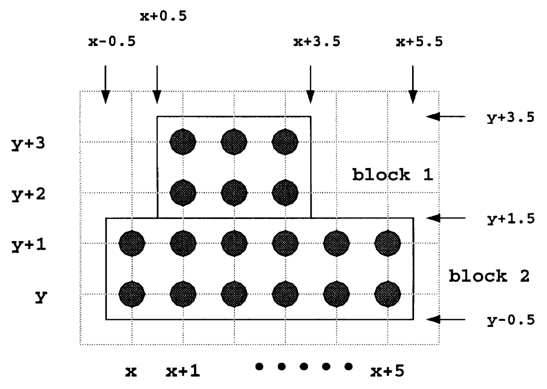

Fig. 8. Two adjacent blocks and corner coordinates.

coarse skew angle as 0°, the region of interest as $R_1$, and the flag as $u$.

### 3.3. Refine estimation step

Given a coarse skew angle $\theta_C$, the refine estimation step computes a fine skew angle $\theta_R$ within a range of $\theta_C \pm 1°$ with $\pm 0.1°$ of maximum error. This is performed by applying a Hough transformation to a set of selected points in the selected region of interest.

In contrast to other methods using a Hough transform, our method applies the time-consuming transformation only to a small set of data points, i.e. the lower or upper center points of the bounding boxes in the selected region according to the flag. Also, the search range of the skew angles to be considered is significantly reduced. These facts lead to superior speed and accuracy of the proposed algorithm. An algorithm for refine skew estimation is given in Fig. 7. An accumulator array is used to discover the most frequent orientation in the $(\rho - \theta)$ space.

## 4. Skew correction algorithm

The skew correction is performed by rotating the input image by the skew angle $\theta$, which was obtained in the skew estimation stage. We propose a simple and fast block-oriented algorithm. As mentioned in Section 2, the pixel-oriented methods have a rounding problem, or a heavy computational burden. The proposed method resolves these problems by rotating the blocks instead of the individual pixels. For each block in the BAG, only four corner points are multiplied by rotational matrix. The proposed skew correction algorithm consists of two steps: block transformation and block filling.

### 4.1. Block transformation

A block in a BAG can be represented with its two corner points. The coordinates of the lower $(x_{\min},y_{\min})$ and upper right corner points $(x_{\max},y_{\max})$ are:

$$(x_{\min}, y_{\min}) = (L - 0.5, B - 0.5),$$

$$(x_{\max}, y_{\max}) = (R + 0.5, T + 0.5),$$

where $L$, $B$, $R$ and $T$ are the furthest left, bottom, right and top pixel coordinates, respectively. Note that the coordinates of the corner points are formed by adding $(-0.5)$ or $(+0.5)$ to the furthest pixel coordinates. This enables our algorithm to avoid a rounding problem, since the two adjacent blocks share the same coordinate at their boundary. Hence they are not disconnected by the geometric rotation. Fig. 8 shows two adjacent blocks and the coordinates of their corner points. We can see that $y_{\min}$ of block1 and $y_{\max}$ of block2 share the same coordinate $y + 1.5$.

The four corner points of each block in the original image are transformed by multiplying their coordinates with a rotational matrix to get the corresponding coordinates of the block in the rotated image. Given a skew angle $\theta$, the transformation of rotating by (-$\theta$°) is:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos(-\theta) & -\sin(-\theta) \\ \sin(-\theta) & \cos(-\theta) \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix},$$

where $(x,y)^t$ and $(x',y')^t$ are the coordinates in the input and output image respectively. This is very efficient in that the rotational transformations are performed just four times for each block—this means that the number of floating-point operations is significantly reduced. Consequently, the processing speed of the proposed method is faster than that of conventional pixel-oriented methods.

### 4.2. Block filling

Filling rotated blocks with black pixels is not an easy task as they are not parallel to $x$- or $y$-axis. A scan-line approach is typically used in a general graphics package for polygon

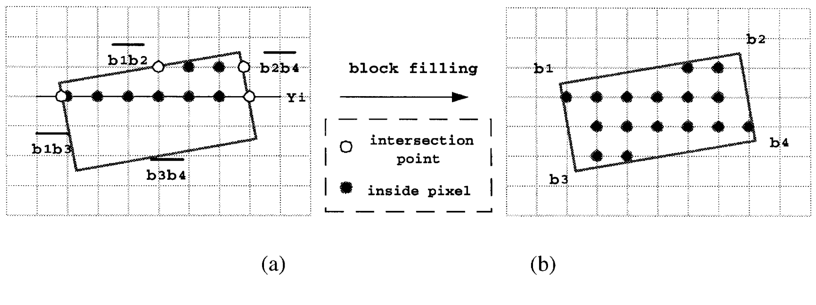

Fig. 9. An example of block filling based on the intersecting point calculation: (a) the intersecting points and inside pixels; (b) the filled block.

Table 1
Number of document images for nine different kinds of skew angles

| Skew angle (°) | − 29 | − 10 | − 5 | − 0.5 | 0 | 5 | 10 | 27 | 43 | Total |
|---|---|---|---|---|---|---|---|---|---|---|
| English magazines | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 |
| Korean magazines | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 |
| English journals | | 18 | 18 | 18 | 18 | 18 | 18 | | | 108 |
| Korean journals | | 18 | 18 | 18 | 18 | 18 | 18 | | | 108 |
| Total | 200 | 236 | 236 | 236 | 236 | 236 | 236 | 200 | 200 | 2016 |

filling [20]. Filling an area determines the intervals for the scan lines that overlap the area. For each scan line crossing a block, the filling algorithm locates the intersection points of the scan line with the block boundaries. The interior pixels between each intersecting pair are colored black.

An example of filling the rotated block is shown in Fig. 9. In Fig. 9(a), there are two white balls intersecting the scan line $Y_i$. The intersection point is obtained by solving a linear system of two equations: $\overline{b_1b_3}$ and $\overline{b_2b_4}$ for the scan line $Y_i$. The black balls in Fig. 9(b) are the interior pixels, which are obtained by checking the interval without any floating-point operations.

## 5. Experimental results

We have measured the performance of the proposed algorithm by an experiment with 2016 document images. We have also proven the superiority of our algorithm by comparing its performance with other well-known methods. In comparing skew estimation performance, three methods adapting the Hough transform (Hinds et al. [5], Le et al. [6] and Yu et al. [7]) were chosen. In case of skew correction, two types of pixel-oriented approaches—direct and indirect methods—were considered. All of these algorithms, including ours, were implemented in a C language on a Pentium 200 MHz PC.

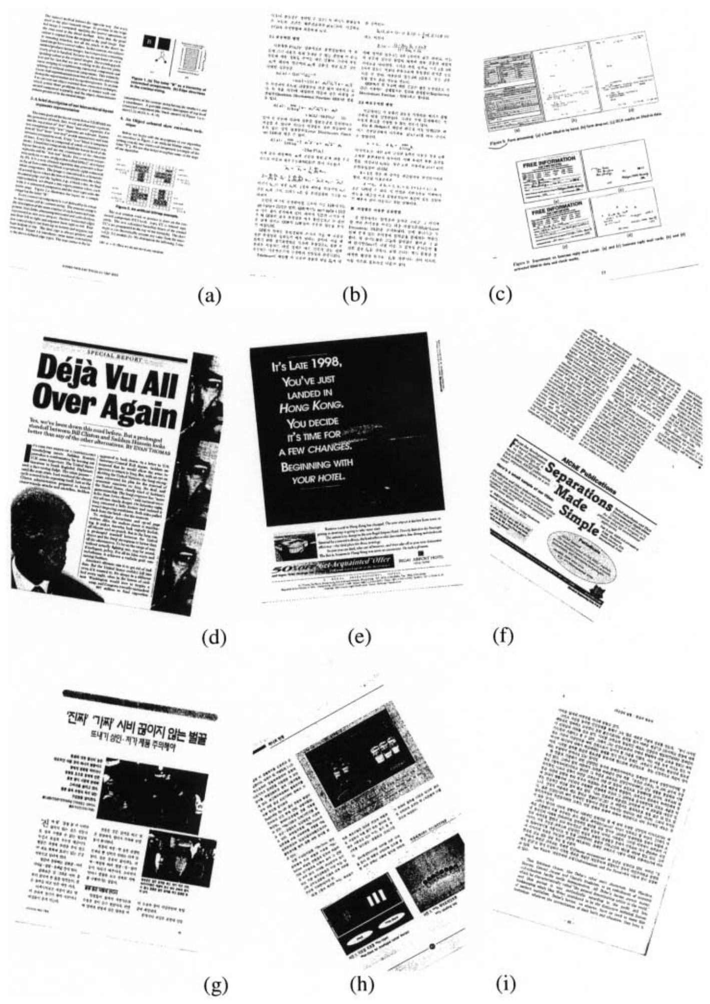

Fig. 10. Some of the original skew images.

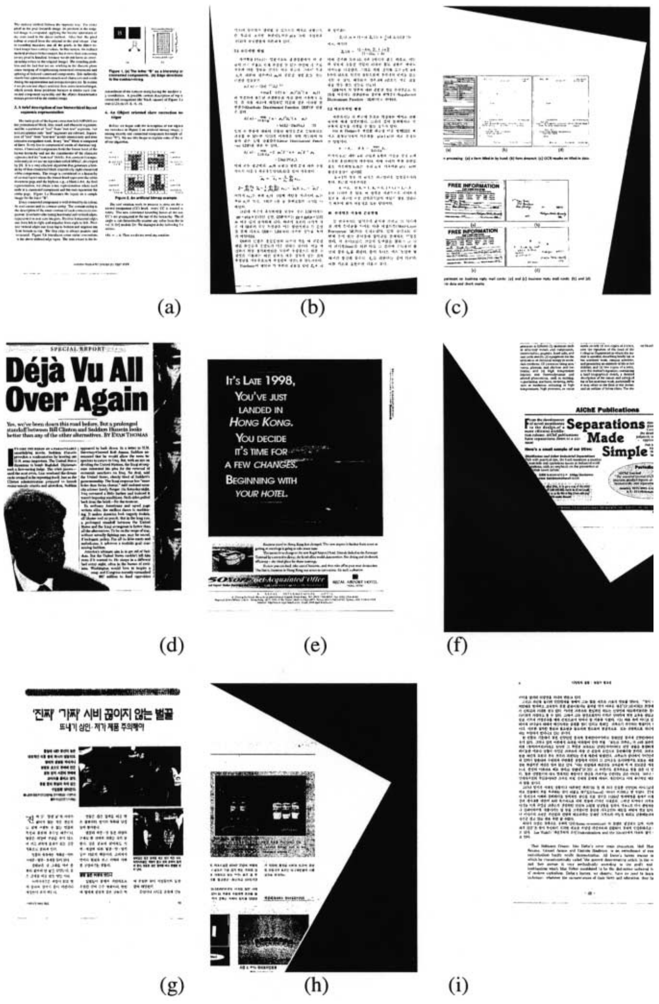

Fig. 11. Some of the resulting images generated by the proposed algorithm.

The 2016 images have been obtained by scanning 236 English and Korean documents, such as journal papers and magazines, at a resolution of 300 dpi. All the A4-size documents were scanned several times with different amount of skew (see Table 1). Some of the test images are shown in Fig. 10, and the corresponding skew-corrected images produced by our algorithm are given in Fig. 11.

## 5.1. Skew estimation

The performance of a skew estimation algorithm can be evaluated in terms of speed and accuracy. The speed is measured in CPU time on a Pentium 200 MHz PC, and the accuracy is computed as the difference between the true skew angle and the estimated one. For all 2016 images, the average processing time of our algorithm is 0.19 s and the average estimation error is 0.06°.

Another three skew estimation algorithms, Hinds et al. [5], Le et al. [6], and Yu et al. [7] have also been implemented under the same computing environment. In the implementation of the Hough transform, we set the estimation range as [−15°, + 15°] and the angular resolution as

Table 2
Performance comparison of the skew estimation methods

| | Average processing times (s) | Average error (°) | The accuracy within the error tolerance (%) | | |
|---|---|---|---|---|---|
| | | | ≤ 0.1° | ≤ 0.2° | ≤ 0.5° |
| Our method | 0.19 | 0.06 | 98.69 | 99.34 | 99.67 |
| Yu et al. | 0.42 | 0.18 | 82.43 | 95.94 | 96.95 |
| Le et al. | 2.33 | 0.08 | 93.13 | 96.72 | 97.38 |
| Hinds et al. | 31.80 | 0.04 | 90.19 | 99.16 | 99.66 |

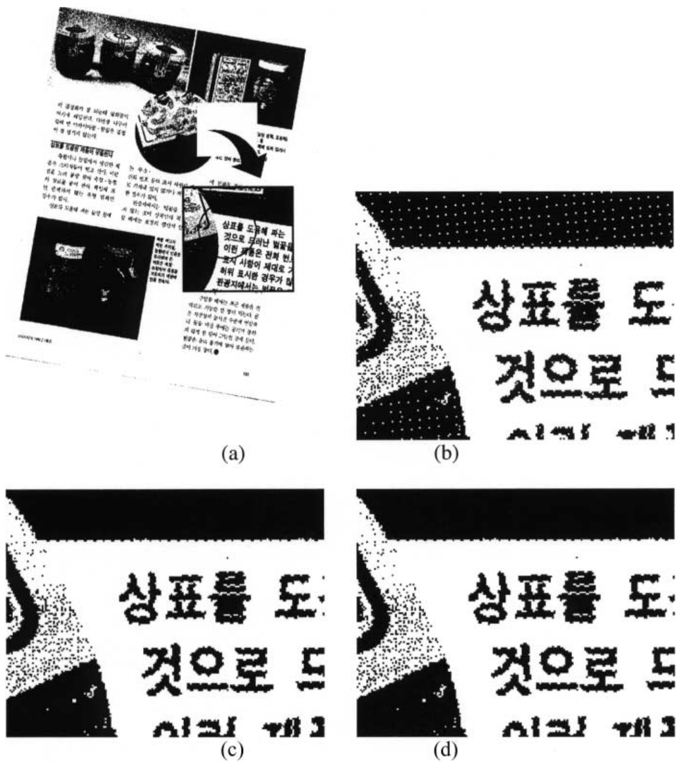

Fig. 12. Examples of skew-corrected images: (a) original input images (skewed by +10°); (b) direct method; (c) indirect method; (d) the proposed method.

0.1°. In the case of the hierarchical Hough transform of Yu et al., the first angular resolution was set to 3° and the second one to 0.1°. The accumulator cell in Hinds et al.'s algorithm was incremented by the amount of run length, which varied from 1 to 25.

Table 2 shows the speed and accuracy of the four skew estimation algorithms. The first column shows the average processing time, and the second one shows the average error. The last three columns show the successful estimation percentage when the error tolerance was 0.1, 0.2 and 0.5°, respectively. Our method is faster than the other methods. As shown in Table 2, the coarse/refine strategy—introduced in our method and Yu et al.'s method—reduces the time complexity more than the original Hough transform. The proposed method, where the Hough transform is applied only to a small region of interest, also reduces the processing time significantly. In terms of accuracy, Hinds et al.'s method shows the smallest average error; our method, Le et al.'s, and Yu et al.'s method follow in this order. Our method, however, shows the highest degree of successful

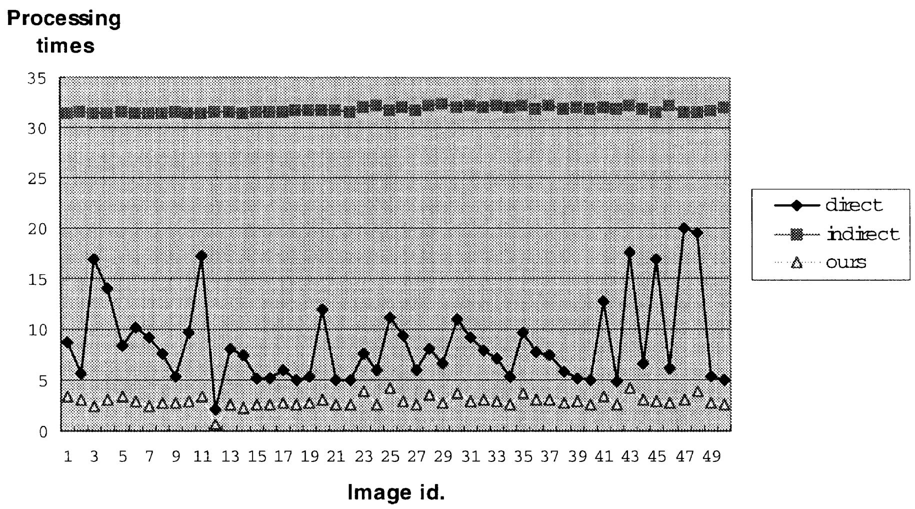

Fig. 13. The processing time of the three methods for 50 images from English magazines.

estimation under an error tolerance. Also, it is indicated that the accuracy of the coarse estimation is more than 99.67%. In short, the proposed method is more reliable than the other three. Considering the performance in terms of speed and accuracy, our method is therefore superior to the other three using the Hough transform.

### 5.2. Skew correction

We also have compared the performance of our skew correction algorithm with two conventional pixel-oriented algorithms—the direct and indirect methods. The processing time of our method averaged 1.8 s for all 2016 documents, while the direct and indirect methods averaged 4.1 and 31.8 s, respectively. The time complexity of our block-based method thus is much lower than that of the pixel-oriented methods. The indirect method, in particular, is more time-consuming, as all the pixels of the output image have to be considered for the rotational transformation.

Fig. 12 shows some examples of skew-corrected images obtained by the three algorithms. Fig. 12(a) is an original image skewed by $+10°$. Fig. 12(b) is part of a skew-corrected image obtained by the direct method. Some periodic holes are incurred from the rounding problem. Fig. 12(c), from the indirect method, and Fig. 12(d), from our method, have good quality and look similar, and there is no rounding problem.

Fig. 13 compares the processing time of the three methods for 50 images from English magazines skewed by $+10°$. The average processing times of the three methods are 2.9, 8.5 and 31.8 s, respectively. In addition to faster processing, we can see that our algorithm shows little variation in processing time among the various document images compared to the direct method.

## 6. Conclusions

In this paper, we have proposed a fast and accurate skew estimation and correction algorithm for English and Korean documents based on a BAG representation. Our skew estimation method is superior to other well-known methods in the literature in terms of speed and accuracy as we confine the region of interest to a small and plausible text area, and confine the search range of Hough transform to $\theta_C°$, where $\theta_C$ is a coarse skew angle. The skew correction method is also much faster than pixel-oriented methods, over three times faster than the direct method, and over 16 times faster than the indirect method, because the time-consuming rotation operations are performed on the blocks in the BAG rather than on the individual pixels. The superiority of our algorithm was proven by an experiment with 2016 images of various English and Korean documents.

## References

[1] W. Postl, Detection of linear oblique structures and skew scan in digitized documents, Proceedings of the Eighth International Conference on Pattern Recognition, Paris, 1986, pp. 687–689.

[2] H.S. Baird, The skew angle of printed documents, Proceedings of the SPSE Fortieth International Symposium on Hybrid Imaging Systems, New York, 1987, pp. 21–24.

[3] T. Akiyama, N. Hagita, Automated entry system for printed documents, Pattern Recognition 23 (11) (1990) 1141–1154.

[4] S.N. Srihari, V. Govindaraju, Analysis of textual images using the Hough transform, Machine Vision and Applications 2 (1989) 141–153.

[5] S.C. Hinds, J.L. Fisher, D.P. D'Amato, A document skew detection method using run length encoding and the Hough transform, Proceedings of the Tenth International Conference on Pattern Recognition, Atlantic City, 1990, pp. 464–468.

[6] D.X. Le, G. Thoma, H. Weschler, Automated page orientation and skew angle detection for binary document image, Pattern Recognition 27 (10) (1994) 1325–1344.

[7] B. Yu, A.K. Jain, A. robust, and fast skew detection algorithm for generic documents, Pattern Recognition 29 (10) (1996) 1599–1629.

[8] A. Amin, S. Fischer, A.F. Parkinson, R. Shiu, Comparative study of skew detection algorithms, Journal of Electronic Imaging 5 (4) (1996) 443–451.

[9] U. Pal, B.B. Chaudhuri, An improved document skew angle estimation technique, Pattern Recognition Letters 17 (1996) 899–904.

[10] H.F. Jiang, C.C. Han, K.C. Fan, A. fast, approach to the detection and correction of skew documents, Pattern Recognition Letters 18 (1997) 675–686.

[11] A. Hashizume, P.S. Yeh, A. Rosenfeld, A. method, of detecting the orientation of aligned components, Pattern Recognition Letters 4 (1986) 125–132.

[12] L. O'Gorman, The document spectrum for page layout analysis, IEEE Trans. Pattern Anal. Mach. Intell. 15 (11) (1993) 1162–1173.

[13] X. Jiang, H. Bunke, D.W. Kljajo, Skew detection of documents images by focused nearest-neighbor clustering, Proceedings of the Fifth International Conference on Document Analysis and Recognition, Bangalore, 1999, pp. 629–632.

[14] N. Rondel, G. Burel, Cooperation of multi-layer perceptrons for the estimation of skew angle in text document images, Proceedings of the Third International Conference on Document Analysis and Recognition, Montreal, 1995, pp. 1141–1144.

[15] H. Yan, Skew correction of document images using interline cross-correlation, CVGIP: Graphical Models and Image Processing 55 (6) (1993) 538–543.

[16] Avanindra, S. Chaudhuri, Robust detection of skew in document images, IEEE Trans. Image Processing 6 (2) (1997) 344–349.

[17] C.L. Yu, Y.Y. Tang, C.Y. Suen, Document skew detection based on the fractal and least squares method, Proceedings of the Third International Conference on Analysis and Recognition, Montreal, 1995, pp. 1149–1152.

[18] C. Sun, D. Si, Skew and slant correction for document images using gradient detection, Proceedings of the Fourth International Conference on Document Analysis and Recognition, Ulm, 1997, pp. 142–146.

[19] M. Ali, An object/segment oriented skew-correction technique for document images, Proceedings of the Fourth International Conference on Document Analysis and Recognition, Ulm, 1997, pp. 671–674.

[20] D. Hearn, M.P. Baker, Computer Graphics, 2nd ed, Prentice Hall; New Jersey, 1994, pp. 117–124.

[21] H.K. Kwag, K.C. Lee, S.H. Kim, S.W. Jeong, Fast skew correction using block transformation, Third IAPR Workshop on Document Analysis Systems, Nagano, 1998, pp. 210–213.