Image Analysis on a Scanned Journal Page

Yung-Sheng Chen, Pao-Hsien Li, and Chin-Hung Teng

***Abstract*—Document image analysis is of great importance in the field of image processing and pattern recognition, where the scanned document analysis receives much more attention due to its significance in the fast growing digital libraries. Three types of scanned journal pages including one-column, two-column as well as two-column mixed with one-column are studied in this paper. A journal page is basically composed of four lines including texts, figures or tables, isolated mathematical expressions and embedded mathematical expressions. In order to identify these useful lines for future recognition application, an effective approach is presented. The main steps of this approach consist of skew restoration, line extraction, page orientation checking, page inversion checking as well as line detection. The step of line detection is used to classify the detected lines. A series of experiments are conducted and the results show that our system can achieve 90% above accuracy of line detection without the use of OCR and thus confirms the feasibility of the proposed approach.**

***Index Terms*—Document image analysis, Journal page, Mathematical expression, Skew restoration**

I. Introduction

DOCUMENT image analysis (DIA) is a long-lasting research topic in image processing and pattern recognition. Document image analysis means that we want to recognize each object in a document and then understand the document or transfer it to another format. Document image analysis is closely correlated with optical character recognition as if we can recognize each word in a document, then we may understand the content of the document. In the past decades, many optical character recognition and document image analysis technologies have been extensively developed. Among the documents we want to analyze, journal or conference page is one of notable targets since the academic digital archives are growing very fast at present and there is a strong need to digitize a journal page into an editable digital format, not just a "journal image".

Journal page is quite different from ordinary text document as it may contain a number of figures, tables, and mathematical expressions especially for scientific journal page. It is a challenge to identify all of these objects since their positions vary from page to page. Moreover, the journal page may be in the format of one-column, two-column, or two-column mixed with one-column, or the page may be skew or inverted because of inappropriate scan. All of these may complicate the analysis of a journal page. To analyze such a page, an effective approach needs to be developed for segmenting and detecting each object in a journal page, including texts, figures, tables as well as isolated/embedded mathematical expressions. In the literature, some methods are developed for analyzing journal page based on OCR technologies. However, OCR is another significant topic and requires a dictionary for its implementation. Moreover, the OCR for different languages is quite different. Therefore, we get rid of the use of OCR technology so as the complexity of proposed system is greatly simplified. We merely use the bounding boxes and contours of a journal page to analyze the document and perform subsequent line classification. A journal image used in this study is scanned from a journal page and has been binarized as a black-white image, the J-image. The following five inherent properties existing in the J-image are considered for designing our approach.

1) A J-image may possibly be skewed or not.
2) A J-image may possibly be inverted or not.
3) There are possibly tables, graphics, figures, mathematical expressions, and texts involved simultaneously in a J-image.
4) It is possible to mix one-column and two-column formats in a J-image.
5) The mathematical expression may be embedded in a text line (embedded expression) or not (isolated expression) as shown in Fig. 1. The embedded expression is also known as inline expression mixed with ordinary texts within lines, whereas the isolated expression is known as a display expression typed in a separated line.

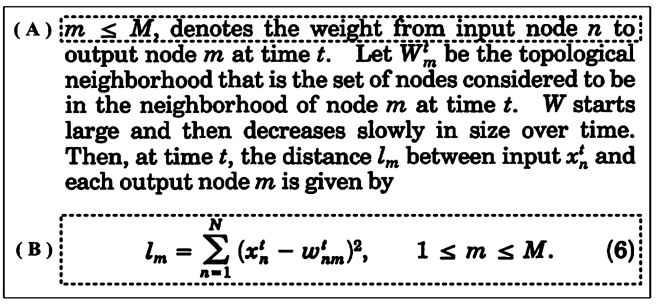


Fig. 1. Illustration of (A) embedded and (B) isolated mathematical expression.

In the following, we first give some related works and then present our approach to dealing with the journal pages with the mentioned properties.

Mathematical expression detection is a very important step for journal page analysis and there are many approaches developed focusing on this issue. Although OCR of mathematical expression is not the main consideration of this paper, it is worthy of noting some researches on the mathematical expression recognition. Guo et al. [15] presented an automatic mathematical expression understanding system. Li et al. [22] devised a baseline structure analysis for studying the mathematical formula recognition. Tian et al. [28] presented

This work was supported in part by the National Science Council, Taiwan, Republic of China, under the grant numbers NSC 101-2221-E-155-056.

Yung-Sheng Chen and Pao-Hsien Li are with the Department of Electrical Engineering, Yuan Ze University, Taoyuan, Taiwan, ROC. (Email: eeyschen@saturn.yzu.edu.tw)

Chin-Hung Teng is with the Department of Information Communication, and Innovation Center for Big Data and Digital Convergence, Yuan Ze University, Taoyuan, Taiwan, ROC. (Email: chteng@saturn.yzu.edu.tw)

a symbol recognition method for mathematical expressions. Chowdhury et al. [7] developed a segmentation algorithm for positioning math-zones from document images. These methods deal only with isolated expressions. Contrarily, Garain et al. [12] presented a method to identify the embedded expressions. Researches on both isolated and embedded mathematical expression segmentation and detection can be found in Refs. [2], [13], [14], [16], [17], [18], [19], [29].

Due to the skewed problem existed in scanned document images, Das and Chanda [8] proposed a method using morphological operations to draw baselines of the text lines, and calculating slopes of these baselines to estimate skew angle, which is used to restore skewed document images. Pal and Chaudhuri [25] presented improved method using bounding box of characters to find mean line and base line, and then to estimate skew angle. Shivakumara and Kumar [27] put forward a similar method which used pixel coordinates of centroid, uppermost and lowermost of characters to estimate skew angle. Although these skew detection methods can work well for documents with pure texts, they cannot deal with complex document images involving simultaneously tables, graphics, figurers, mathematical expressions, and texts.

Recently, many skew detection methods for complex scanned document images have been proposed. Li et al. [21] presented a skew angle estimation algorithm using wavelet decomposition and projection profile analysis. The method proposed by Liu et al. [23] used borderline extraction of connected component to estimate skew angle. Fan et al. [11] proposed a rectangular active contour model to calculate skew angle in document images. Chou et al. [6] proposed using piecewise covering by parallelograms to detect skew angle. This method created non-overlapping slabs and scan lines as features for skew detection. This method was further improved by Dey and Noushath [9]. Dhandra et al. [10] presented using region labeling and image dilation to detect skew angle, in which the detail of the number of times of image dilation is unknown. In addition, Manjunath Aradhya et al. adopted the background growing, thinning, and moments methods for the skew estimation [24]. A short survey of skew detection of scanned document images can be found in [26] and an application to video news system is worthy of reading [20]. However, the issues of page inverted ($180°$) and page orientation ($-90°$ or $90°$) restoration have not been investigated yet in these skew detection methods.

Text line (or "line" in this paper) extraction is a major preprocessing for locating a mathematical expression in a printed document. Some researchers applied the line extraction for mathematical expression detection or lines removal of texts, tables, graphics and figures [1], [2], [16], [19], [30]. Tsujimoto and Asada [30] adopted adjacent connected components to extract segments and defined four thresholds to determine words so that they can merge the words into lines. Similar methods can also be found in Refs. [16], [19], where line locations are determined based on many thresholds and therefore these methods require more processing time. Chang et al. [1] proposed another method which uses projection profile to determine threshold of space between words and finally extracts lines. However, their report lacks detailed statistics analysis for further suggestions.

Typically, extracting isolated mathematical expressions does not require character recognition. Lee and Wang [19] defined some detection rules based on the properties of isolated expression in a printed document, e.g. isolated expressions printed in a separate line usually is taller than ordinary texts. Similar method can be found in Ref. [16], where a number of more detailed rules are defined. Chaudhuri and Garain [2] analyzed the connected component of texts to calculate Standard Deviation (SD) of lines, and then find isolated mathematical expressions that have large SD. Although it can perform well on some cases, this paper did not provide further statistics analysis on how to determine the threshold for the calculated SDs in detecting isolated mathematical expressions, in particular, how to differentiate a mathematical expression from figures since the SD of figures is also very large. Chowdhury et al. [7] used the amount and the positions of superscripts and subscripts to detect isolated expressions. However, all of these mentioned approaches do not consider yet the isolated expression extraction of figures, graphics and tables.

Methods of extracting embedded mathematical expressions can be performed by using or without using character analysis. The former methods [2], [12], [14], [19] recognize each character in each line and perform syntactic analysis to obtain a high accuracy of embedded expression extraction with a higher time complexity. The approaches without character recognition typically produce a lower accuracy but are more efficiency. For example, Jin et al. [16] used horizontal projection analysis of lines to detect the positions of embedded expressions. Kacem et al. [18] adopted the property of superscripts and subscripts in lines and used the relative size and position of their contours for embedded expression extraction.

In this paper, we propose an approach which can effectively restore the skewed and inverted J-images, and identify large regions possibly having tables, graphics, and figures, as well as isolated and embedded mathematical expressions. All algorithms presented in this paper do not use any OCR process like the method presented in [31]. The flowchart of the proposed approach is depicted in Fig. 2, which mainly includes (1) skew restoration, (2) line extraction, (3) page orientation checking, (4) page inversion checking, (5) large region identification, as well as (6) isolated and embedded mathematical expression detection. In what follows, we present the detailed procedures of each block in Fig. 2 in Sections II-VI. The experiments and analyses will be discussed in Section VII. The conclusion and future works are drawn finally in Section VIII.

II. Skew Restoration

A skewed J-image may degrade the performance of line extraction and therefore skew restoration is a necessary preprocessing for extracting lines in a J-image. In this study, we employ a contour based approach to estimate the skew angle of a J-image. By analyzing the histogram of slopes of contours in an image, we may find the orientation of the image. This approach has been well applied in restoring the orientation of Chinese seal images [3] and printed music documents [5]. Since a J-image has many text lines, we can expect that this approach can also be applied in restoring the orientation of a J-image. However, because a J-image may contain many large regions such as tables, figures, and graphics, such a contour analysis method may receive a degraded

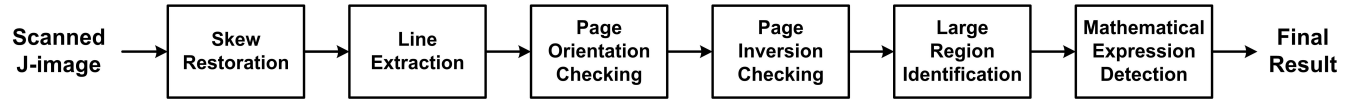
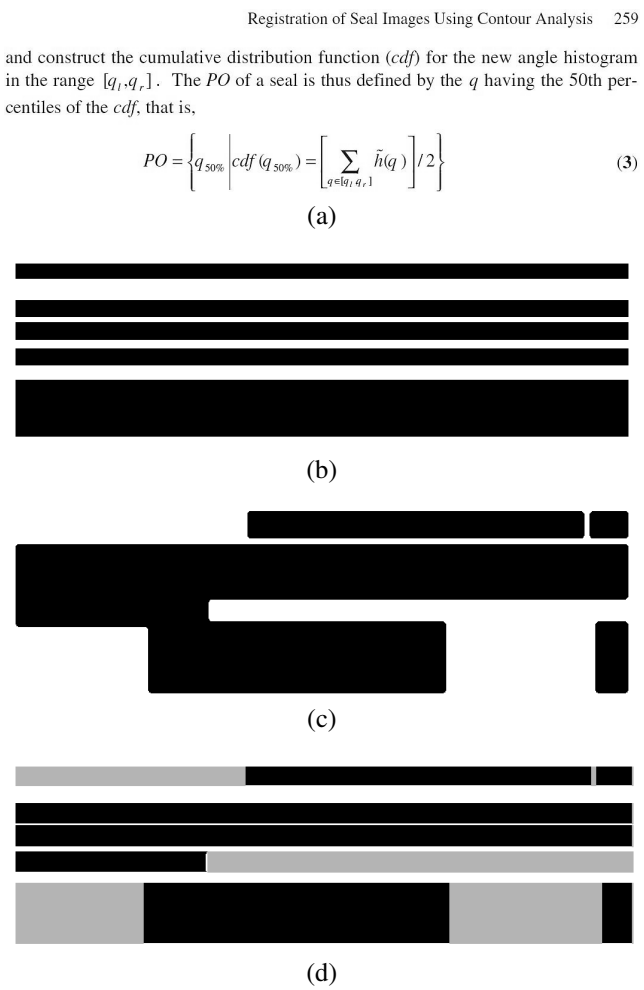


Fig. 2. Flowchart of proposed approach.

performance due to the interference of the contours from these large regions. Hence, for J-image skew restoration, a more dedicated approach should be developed.

In our system, we get rid of the influence of tables, figures, and graphics by first applying morphological dilations to assemble texts into smooth regions. Subsequently, the contour of each region in the image is extracted and the inner contours (i.e. the contour inside another contour) are removed. Only the extreme outer contours are remained for further processing. Following this, the areas of remained contours are calculated and the largest m_{top} contours are removed. These m_{top} contours are typically from tables, figures as well as graphics; and removing them can make the skew angle estimation more accurate. In our analysis, m_{top} is set to 10 and from a series of experiments, the skew estimation approach can produce an accuracy with mean error of skew angle below 1° for arbitrary image rotation as presented in Section VII-A. The reader can refer to Ref. [4] about the detailed procedures of the skew angle estimation, the determination of m_{top}, and the experimental results in our skew restoration method.


Registration of Seal Images Using Contour Analysis 259

and construct the cumulative distribution function (*cdf*) for the new angle histogram in the range $[q_l, q_r]$. The *PO* of a seal is thus defined by the q having the 50th percentiles of the *cdf*, that is,

$$PO = \left\{ q_{50\%} \,\middle|\, cdf(q_{50\%}) = \left[\sum_{q \in [q_l, q_r]} \tilde{h}(q) \right] / 2 \right\} \quad (3)$$

(a)

(b)

(c)

(d)

Fig. 3. (a) Original J-image. (b) $A(x, y)$ of (a). (c) DRDE image of (a). (d) $\tilde{A}(x, y)$ of (a).

III. Line Extraction

Except for the text line, a line in a J-image defined here may also be a table, figure, graphic, isolated or embedded mathematical expression, which will be finally identified as a result of the presented approach. In order to segment a line in a J-image, the vertical projection, $A_v(y)$, and horizontal projection, $A_h(x)$, of a J-image are defined as follows:

$$A_v(y) = \begin{cases} C_b, & \exists J(x, y) = C_b, \forall x = 0, 1, \ldots, W-1 \\ C_w, & \texttt{otherwise} \end{cases} \quad (1)$$

and

$$A_h(x) = \begin{cases} C_b, & \exists J(x, y) = C_b, \forall y = 0, 1, \ldots, H-1 \\ C_w, & \texttt{otherwise} \end{cases} \quad (2)$$

where $J(x, y)$ denotes pixel value at image coordinates (x, y); W and H are image width and height; and C_b and C_w represent the black and white pixels, respectively. $A_v(y)$ and $A_h(x)$ tell us the rough vertical and horizontal positions of a line and by intersecting $A_v(y)$ and $A_h(x)$ we can define a two-dimensional array that includes the blocks of all lines in a J-image. The two-dimensional array, $A(x, y)$, is defined as follows:

$$A(x, y) = \begin{cases} C_b, & \texttt{if } A_v(y) = C_b \texttt{ and } A_h(x) = C_b \\ C_w, & \texttt{otherwise} \end{cases} \quad (3)$$

Figure 3(b) shows an example of the resulting $A(x, y)$ for the J-image in Fig. 3(a). From this figure we can observe that each line in the J-image has been roughly identified. Since $A(x, y)$ may contain some blocks without any information in it (i.e. no black $J(x, y)$ in the block), such a block will be filtered out (i.e. changed to white) before further processing. Moreover, by examining Fig. 3(b) we can find that some blocks are not compact. For example, the final block in Fig. 3(b), which corresponds to the isolated mathematical expression, contains a large white area in the original J-image as shown in Fig. 3(a). To obtain more compact lines, a DRDE image is formed by first identifying the bounding box of each line using the block in $A(x, y)$. Then, for each pixel in the bounding box in the original J-image, if there is a black pixel in that column, the pixel is set to black. This step can fill some small white gaps in the J-image. Subsequently, the resulting J-image is dilated two times, reduced to 15% of its original size, dilated one time and enlarged back to its original size; and the so-called DRDE image is thus obtained. The dilation can prevent the disappearance of small symbols such as '-' and '|', and image size reduction and restoration is used to find the rough layout of a J-image. The reason of selecting 15% image reduction will be discussed in Section VII-B.

Figure 3(c) shows the resulting DRDE image for the J-image in Fig. 3(a). From this figure we can find that by image dilation and reduction, DRDE image is able to roughly locate the regions which contain information of a J-image. By

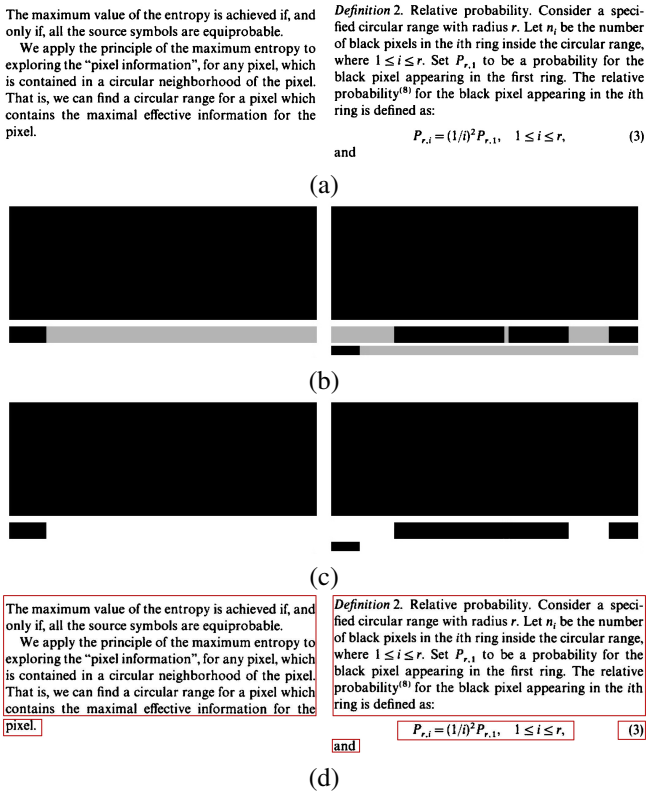


Fig. 4. (a) Original J-image. (b) $\tilde{A}(x,y)$ of (a). (c) $\tilde{A}(x,y)$ after filtering out some gray regions. (d) The J-image as well as the resulting bounding boxes from the extracted blocks.

combining DRDE and $A(x,y)$, an image with more compact blocks of lines is defined as follows:

$$\tilde{A}(x,y) = \begin{cases} C_b, & \texttt{if } A(x,y) = C_b \texttt{ and } DRDE(x,y) = C_b \\ C_g, & \texttt{if } A(x,y) = C_b \texttt{ and } DRDE(x,y) = C_u \\ C_w, & \texttt{otherwise} \end{cases} \tag{4}$$

where C_g indicates a gray pixel. Figure 3(d) shows the resulting $\tilde{A}(x,y)$ for the J-image in Fig. 3(a). The gray regions in $\tilde{A}(x,y)$ is typically generated from the indent of a new line or the white space of an isolated mathematical expression. Hence, removing them can perform a more compact line extraction for a journal page. However, directly removing the gray regions may sometimes result in undesired effects. Figure 4 shows another example for line extraction. In this case if we remove all the gray regions, the isolated mathematical expression will be divided into two parts due to the small gray gap in the line as Fig. 4(b) shows, which is an unwanted situation since these two parts belong to a single mathematical expression indeed. To avoid such a situation, we devise a simple rule to remove the gray regions. For each block, we only remove the largest two gray regions, reserving the other gray regions in the block. Figure 4(c) shows the resulting $\tilde{A}(x,y)$ after removing the largest two gray regions in each block. The resulting line extraction is more accurate for the isolated mathematical expression. According to the resulting $\tilde{A}(x,y)$ we can find the bounding box of each block as shown in Fig. 4(d) where the red rectangles are the extracted bounding boxes.

Examining Fig. 4 we can observe that the text lines are sometimes merged to form a large block. This is not what we want and we can decompose it by checking the pixels of J-image inside the detected bounding box. If there is a row in the bounding box consisting entirely of white pixels, the pixels in the row of corresponding $\tilde{A}(x,y)$ are changed to white. This can divide a large block of $\tilde{A}(x,y)$ into several small blocks, where each block represents a line in the J-image. Following this, the bounding boxes for the resulting $\tilde{A}(x,y)$ are detected again. Figure 5 shows the results of our line extraction for a J-image. We can observe that most lines in the journal page are well extracted by proposed line extraction algorithm.

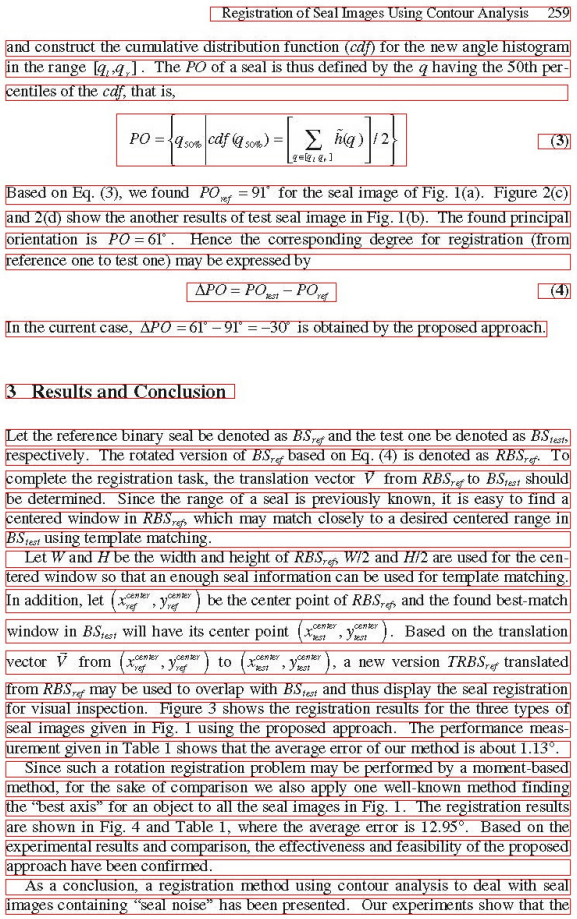


Fig. 5. Results of line extraction. Each line is identified by a red bounding box.

IV. Page Orientation Checking

Page orientation checking indicates that we want to determine the orientation, i.e. portrait or landscape, of a J-image. In most cases, our skew restoration can amend the incorrect page orientation but sometimes an image would be rotated to a landscape orientation because of incorrect scan process. Therefore, we include the page orientation checking to correct such a situation.

Because we have extracted the lines of a journal page in previous stage, we can employ the results of line extraction to check page orientation. A published journal paper is usually printed in portrait orientation and in such a situation the resulting bounding box of each line has typically a larger width than height. Hence, by counting the number of portrait and landscape bounding boxes we can easily determine the orientation of a J-image. Let N_w denote the number of

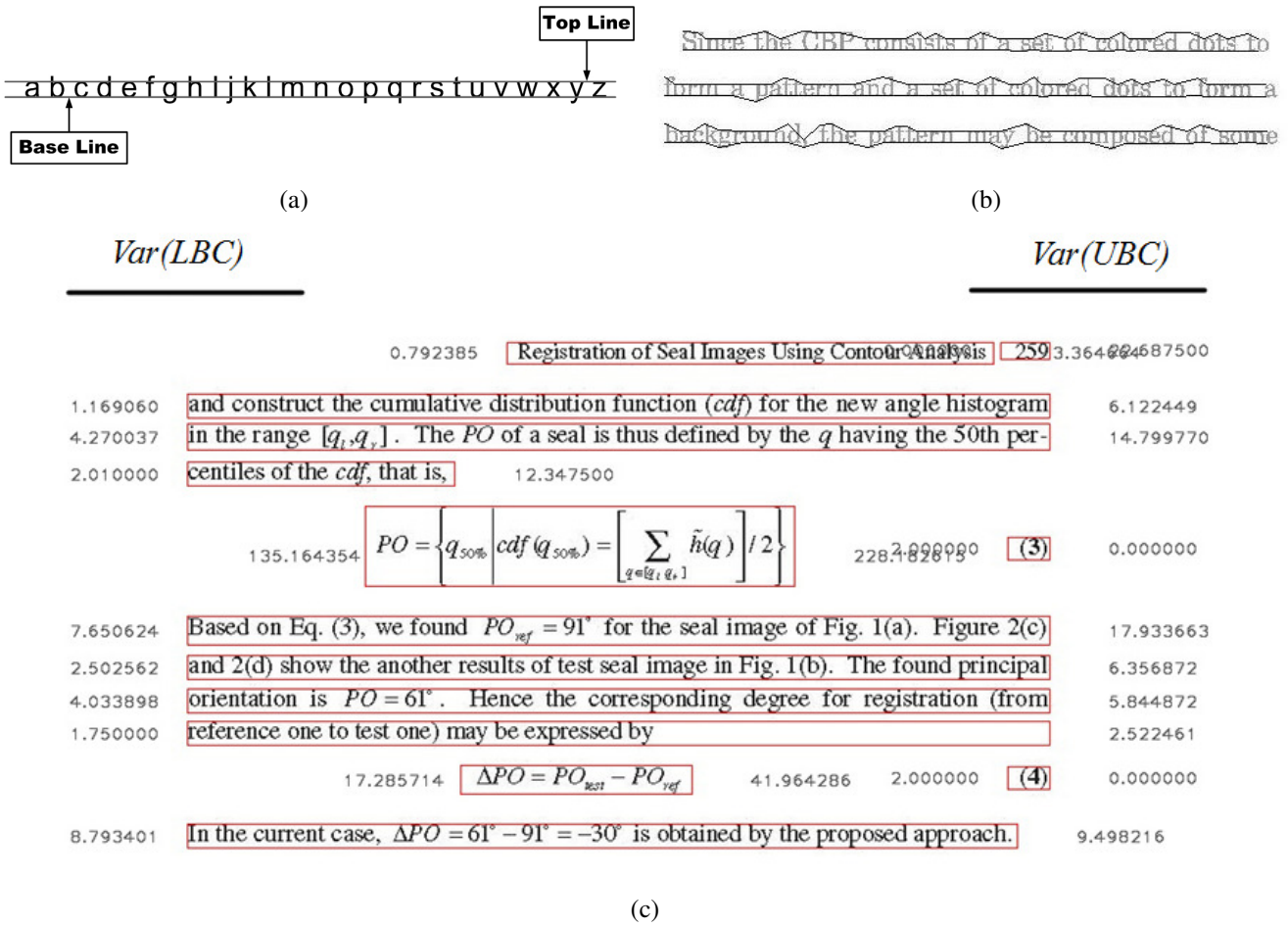


Fig. 6. (a) Base line and top line of English letters. (b) The upper and low bounding curves for three extracted lines. (c) The variances of upper and low bounding curves of extracted lines in a journal page.

bounding boxes of extracted lines whose width is greater than height, and N_h be the number of bounding boxes whose height is greater than width. Then, if $N_h > N_w$, the J-image is in landscape orientation and should be rotated $90°$ to restore it orientation.

V. Page Inversion Checking

Sometimes a journal page may be inverted due to scanning failure or if the page has greater than $180°$ image skew, it may also be inverted after skew restoration. Hence, page inversion detection is necessary to correct such error. In this study, we achieve this by checking the variances of upper and lower bounding curves of extracted lines.

Since most journal pages are typed using lowercase English letters, only the lowercase English letters are considered to design our page inversion checking process. By observing the lowercase English letters, we have the base and top lines as illustrated in Fig. 6(a). For the 26 lowercase English letters, 5 letters, i.e. 'g', 'j', 'p', 'q' and 'y', have strokes below the base line, while the 9 letters, namely 'b', 'd', 'f', 'h', 'i', 'j', 'k', 'l' and 't', have strokes above the top line. Hence, we believe that for a non-inverted J-image the probability of black pixels above top lines is greater than that of black pixels below base lines. However, the base and top lines are not easy to locate in a J-image. Hence, we employ the curves enveloping an extracted line to design our page inversion checking. The upper and lower curves enveloping an extracted line are named the upper bounding curve (UBC) and lower bounding curve (LBC), respectively. Figure 6(b) shows three text lines and the associated UBCs and LBCs. To find UBC and LBC, we first find the contours in a line and then determine the bounding boxes of these contours. For an ordinary text line, a contour is typically from a letter and therefore the UBC is obtained by connecting the left-top corner of the bounding boxes in a line. Similarly, the LBC is obtained by connecting the left-bottom corner of the bounding boxes in the line. If a text line has many black pixels above the top line, the variance of the y-coordinates of its UBC (or the variance of UBC in short) should also be high. Hence, by comparing the variances of UBC and LBC, we can determine whether the page is inverted or not.

Figure 6(c) shows the obtained variances of UBC and LBC for a non-inverted journal page. From this figure we can find that for most lines the variance of UBC is greater than that of LBC. Hence, our page inversion checking is achieved by counting the number of lines whose variance of UBC is greater than that of LBC. If the number of such lines is greater than half of the number of total lines in the page, the page is not inverted. Otherwise, the page is inverted and should be corrected by flipping the image vertically.

To test the effectiveness of proposed page inversion checking process, we collected 859 non-inverted J-images and inverted them to form 859 inverted J-images. These inverted J-images as well as the 859 non-inverted J-images were then

used to test our page inversion checking algorithm. Experimental results showed that all inverted and non-inverted J-images were correctly detected. This confirms the good performance of our page inversion checking method.

VI. Line Detection

We now describe our approach for classifying the extracted lines. They are classified into four categories, namely, ordinary text lines; lines for table, graphic and figure; lines for isolated mathematical expression; and embedded mathematical expression lines. In the process of line detection, some extracted lines may also be combined to form a more correct object. We will first differentiate ordinary text lines from other objects.

A. Text line detection

In Section V, we have defined LBC and use its variance, i.e. Var(LBC), to detect inverted J-image. In fact, the variance of LBC can also be used to distinguish text lines from non-text lines. We found that a text line has typically much smaller Var(LBC) than that of non-text lines. Hence, by simple thresholding on Var(LBC) of detected lines, the text lines can be identified easily.

In order to minimize classification error, we adopted 100 J-images to support the selection of threshold value. The lines in the 100 J-images are detected and then classified into text and non-text lines via human inspection. Subsequently, the Var(LBC) of each line is calculated and the histograms of text and non-text lines are computed and plotted as shown in Fig. 7. This figure tells us that a threshold of 6.0 can yield a minimal classification error. Hence, if a line with Var(LBC) less than 6.0, this line is classified as $\mathbf{O}_{text}$, indicating it is an object of text line. Otherwise, the line is classified as $\mathbf{O}_{non-text}$ for further processing.

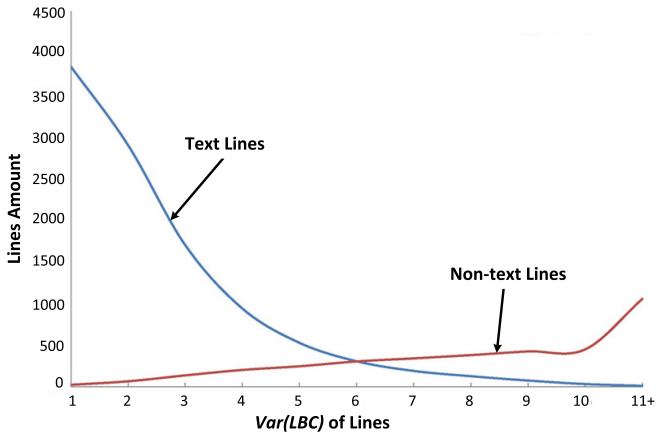


Fig. 7. Histograms of variances for text and non-text lines. The intersection of the two curves is at variance 6.0.

An unexpected situation is however found that sometimes the superscript and subscript of an isolated mathematical expression may be extracted as individual lines as illustrated in Fig. 8. These lines have very small Var(LBC) and therefore are classified as $\mathbf{O}_{text}$. To amend this problem, each line in $\mathbf{O}_{text}$ are re-examined. We found that if a line is extracted from superscripts or subscripts of an isolated mathematical expression, it typically has a lot of white pixels inside the bounding box of that line (see Fig. 8). Hence, by thresholding the density of black pixels of the bounding box of a line, we are able to differentiate the lines of superscripts or subscripts from ordinary text lines. Specifically, let D_k be the density of black pixels for the kth line in $\mathbf{O}_{text}$, and let μ_D and σ_D denote the mean and standard deviation of the density of black pixel for all the lines in $\mathbf{O}_{text}$. Then, if $D_k < \mu_D - 2\sigma_D$, the line is classified as $\mathbf{O}_{script}$, indicating that it is a line from the superscript or subscript of an isolated mathematical expression. The lines in $\mathbf{O}_{script}$ will be further processed in Section VI-C. Typically, it will be combined with the line of an isolated mathematical expression to obtain more accurate classification.

$$l_m = \sum_{n=1}^{N} (x_n^t - w_{nm}^t)^2, \qquad 1 \le m \le M. \tag{6}$$

And the output node c with minimum l_c is obtained by

$$l_c = \min_{m=1}^{M} l_m. \tag{7}$$

Fig. 8. Extracted lines for isolated mathematical expressions. Note that sometimes the subscript or superscript is in an individual line.

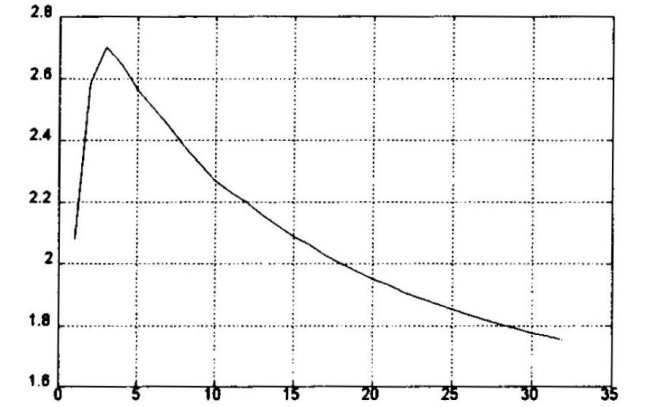
B. Table, graphic, figure, and mathematical expression detection

Now we classify the objects or lines in the class $\mathbf{O}_{non-text}$. We first extract the objects belonging to tables, graphics, and figures, leaving isolated and embedded mathematical expressions in $\mathbf{O}_{non-text}$. Because tables, figures, and graphics are typically higher than the lines of mathematical expressions, we can distinguish them by simply thresholding the height of these objects. Let $\bar{W}$ and $\bar{H}$ denote the average width and height of all extracted lines in a J-image. From our experiments, we found that the height of table, figure, and graphics is usually greater than $3\bar{H}$. Therefore, if the height of an object in $\mathbf{O}_{non-text}$ is greater than $3\bar{H}$, this object is classify as $\mathbf{O}_{fig}$. However, this simple rule cannot completely distinguish mathematical expressions from tables or figures. Some large lines of isolate mathematical expression may be falsely classified as $\mathbf{O}_{fig}$. Hence, the objects in $\mathbf{O}_{fig}$ need further checking to find such isolated mathematical expressions.

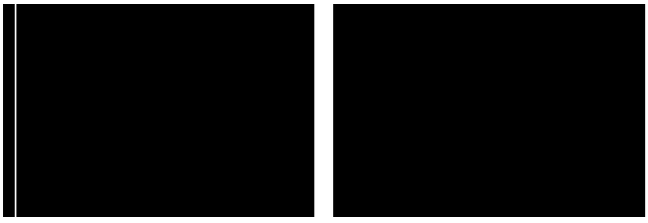
A mathematical expression is typically composed of numbers, variables, and operators, and has many white gaps between these elements. On the other hand, a table or figure is typically more compact with few large and vertical white gaps inside them. Hence, by dividing an object into several parts according to vertical white gaps, we may differentiate a mathematical expression from figures and tables. To achieve this, for each object in $\mathbf{O}_{fig}$ we first perform two dilations on the image. The two dilations can fill some small white gaps of a figure or graphic, while insufficient to fill the large white gaps of a mathematical expression. An example of applying dilations on a figure is shown in Fig. 9, where the figure is no longer divided into two parts after dilations. After performing dilations, we then decompose the object according to vertical white gaps. Subsequently, two rules are designed and if an object from $\mathbf{O}_{fig}$ satisfies these two rules, it is re-classified

as $\mathbf{O}_{iMath}$, the class of isolated mathematical expression. The two rules are listed as follows.

1) If the number of parts of an object after white gap division is greater than 3, the object may be an isolated mathematical expression and should be verified by the second rule. The threshold 3 is obtained from the simplest case, $y = x$, where the number of parts after white gap division should be 3. For more complicated cases, the number of parts is usually greater than 3.
2) Because some symbols in a complex isolated mathematical expression have very large width or height, we further check the average width and height of symbols in an object. We first find the contour of each symbol in the object and then find the bounding boxes of these symbols. Let W_m and H_m denote the average width and height of the bounding boxes of symbols in the object. If H_m is greater than $3\bar{H}$ or W_m is greater than $3\bar{H}$, the object is assigned to $\mathbf{O}_{iMath}$.



(a)



(b) (c)

Fig. 9. (a) A figure in a journal page. (b) The corresponding $\tilde{A}(x, y)$ without dilations. (c) $\tilde{A}(x, y)$ with dilations.

C. Further mathematical expression checking

After detecting table and figure objects in $\mathbf{O}_{non-text}$, the remaining objects in $\mathbf{O}_{non-text}$ are mostly isolated and embedded mathematical expressions. Note that we have a class $\mathbf{O}_{script}$ in which the objects are typically generated from superscripts or subscripts of a mathematical expression. These objects should be combined with an isolated or embedded mathematical expression to obtain a more accurate line detection. Hence, for each object in $\mathbf{O}_{script}$, we compare its position with the objects in $\mathbf{O}_{non-text}$. If two objects have overlapped horizontal position and their shortest vertical distance is smaller than $\bar{H}$, the two objects are combined and put back to $\mathbf{O}_{non-text}$. If we cannot find an object in $\mathbf{O}_{non-text}$ satisfying this condition, the object in $\mathbf{O}_{script}$ is re-classified to the class of $\mathbf{O}_{text}$.

After processing all objects in $\mathbf{O}_{script}$, we then examine the objects in $\mathbf{O}_{non-text}$ to differentiate isolated mathematical expressions from embedded expressions. In this study, a line of embedded mathematical expression indicates that there is a mathematical expression in a text line. We do not segment the mathematical expression from the text line, thus the width of the line is roughly the same as a typical text line. On the other hand, an isolated mathematical expression is located in a separate line and its width is typically less than that of an ordinary text line. Moreover, the height of an isolated mathematical expression is typically greater than that of an ordinary text line. According to these observations, we design the following rule to distinguish isolated mathematical expression from embedded one. For each object in $\mathbf{O}_{non-text}$, we define W_{block} as the horizontal distance between the leftmost and rightmost black pixels in the bounding box of the object. If W_{block} is less than $\bar{W}$ or the height of the object is greater than $1.5\bar{H}$, the object is classified as $\mathbf{O}_{iMath}$. Otherwise, the object is classified as $\mathbf{O}_{eMath}$, the class of embedded mathematical expression.

D. Equal mark checking

In this study, we employ Var(LBC) to differentiate ordinary text lines from other objects. However, the Var(LBC) of an embedded mathematical expression line is quite similar to that of ordinary text lines. Moreover, section titles of a paper have typically large Var(LBC) and may be mis-classified as isolated mathematical expression lines. Hence, a refinement process may be designed to improve the classification accuracy. Fortunately, mathematical expressions often have an equal mark ('=') which can be employed to correct the mentioned classification failure.

We observed that an equal mark is composed of two horizontal lines, and hence we can exploit this property to detect it. We first extract every symbol in an extracted line by detecting contours in the line. Following this, the bounding box of each contour is identified. If the width of a box is larger than 3 times of its height, this symbol is recognized as a horizontal line. If there are two such lines in a local area, an equal mark is identified. We apply this equal mark detection technique to the lines or objects in the classes of $\mathbf{O}_{text}$, $\mathbf{O}_{iMath}$, and $\mathbf{O}_{eMath}$. For an object in $\mathbf{O}_{text}$, if an equal mark is detected, the object is then re-classified to the class of $\mathbf{O}_{eMath}$. Similarly, for a line in $\mathbf{O}_{iMath}$ or $\mathbf{O}_{eMath}$, if we cannot detect an equal mark, the line is re-classified as $\mathbf{O}_{text}$. The refinement process by equal mark detection can further improve our classification accuracy of line detection.

VII. Experiments and Discussion

A. Performance of our skew estimation approach

We used 859 J-images being converted into 200 dpi resolution from the PDFs of technical journals, which include tables, graphics, figures mathematical expressions, and normal text, for the validation of the performance of our skew angle estimation method. Two categories of skewed J-images were used in this validation as listed in Table I. According to our early study in [4], the range of dilation times between 14 and 24 is suggested for use in the skew estimation approach. Hence, for each case, we compute all PTA_D, D = 14, 15, $\cdots$, 24. Then we use (a) $mean_{\forall D}(\text{PTA}_D)$ and

(b) $median_{\forall D}(\text{PTA}_D)$ to represent the final skew estimation result. After performing the skew estimation on all cases, all PTAs are obtained. The plots of mean error between the computed PTA and the corresponding ground truth versus the skewed angles are given in Fig. 10(a) and Fig. 10(b) using $mean_{\forall D}(\text{PTA}_D)$ and $median_{\forall D}(\text{PTA}_D)$, respectively. In these two plots, we found that the small mean errors about 0.1° are apparently located at the range of skewed angles between 1° and 25°, as well as between 70° and 90°. This confirms that our skew estimation method can perform an effective skewed J-image adjustment for normal cases.

TABLE I
TWO CATEGORIES OF SKEWED J-IMAGES WERE USED FOR VALIDATING THE PROPOSED SAE METHOD.

Category 1	Category 2
1°	20°
2°	25°
3°	30°
4°	35°
5°	40°
6°	45°
7°	50°
8°	55°
9°	60°
10°	65°
11°	70°
12°	75°
13°	80°
14°	85°
15°	90°

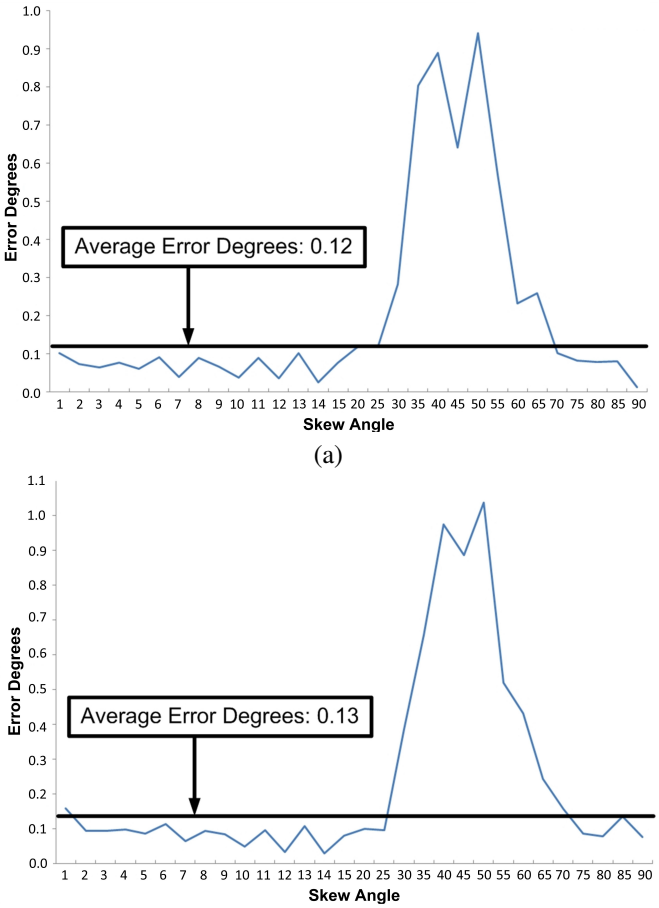


Fig. 10. Plots of mean error between the computed PTA and the corresponding ground truth against the skewed angles for using (a) $mean_{\forall D}(\text{PTA}_D)$ and (b) $median_{\forall D}(\text{PTA}_D)$, respectively

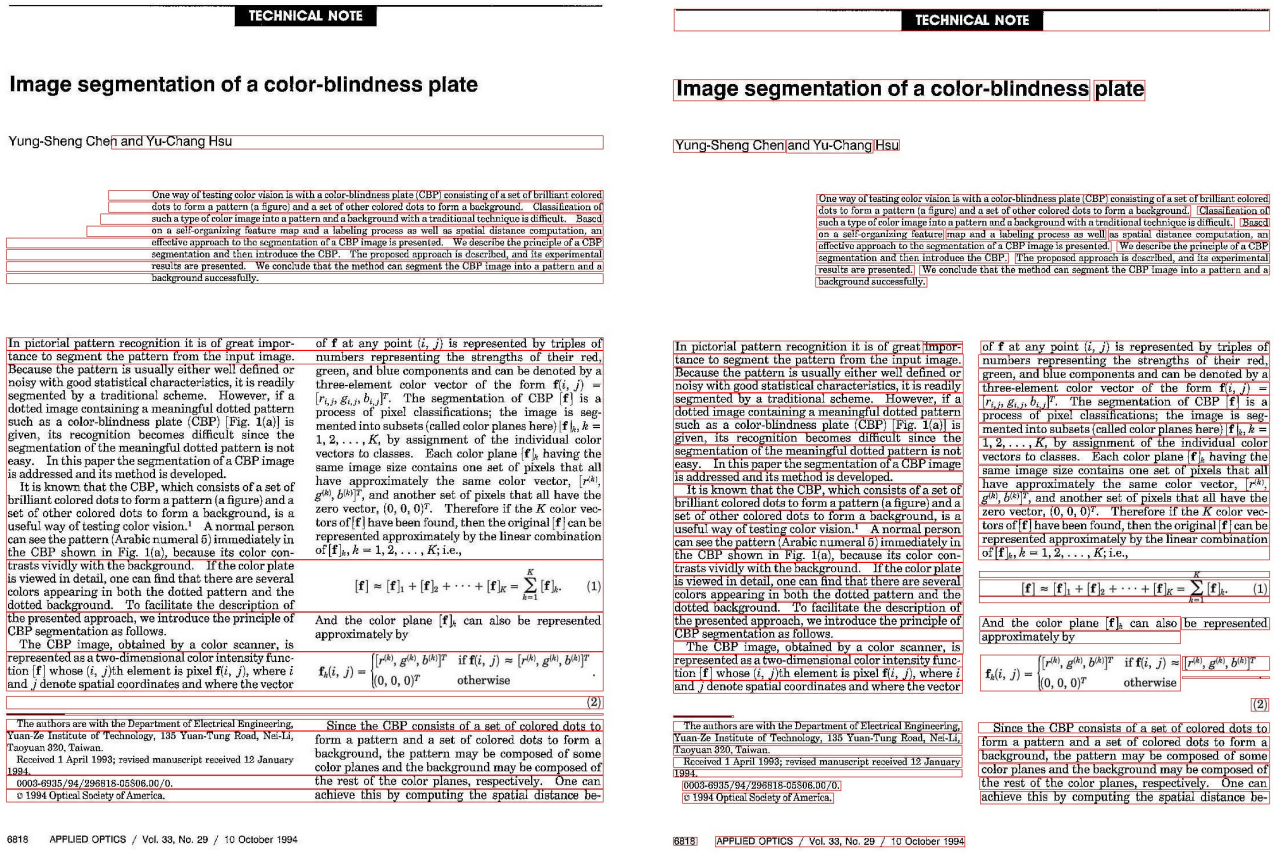
B. Determining image reduction size in line extraction

In our line extraction algorithm in Section III, we used a DRDE image to help us find the rough layout of a J-image. In forming the DRDE image, the original J-image is dilated, reduced in size, dilated again, and finally restored to its original size. Image reduction is a critical step in this algorithm as it can neglect the detailed information of a J-image while preserving the global view of the image. However, the size of reduction has a great impact on the final result of line extraction. If an image is slightly reduced, some holes in the image will be preserved and a single text line may break into several lines. On the other hand, if the reduction is too much, the regions of different lines may merge into a large block and the extraction of individual line may fail. Figure 11 shows two examples of different reductions. We can see that for large image reduction (e.g. reduced to 5% of its original size), some lines are missed and some large blocks are produced; while for slight reduction (e.g. 95% of its original size), some broken lines are detected. Hence, we need a suitable size of image reduction to obtain satisfactory line extraction.

We conducted an experiment to determine the size of reduction. We use three types of J-images, namely one-column, two-column, and two-column mixed with one-column, for the experimental test. The lines on these images are extracted using different size of image reduction. The results are depicted in Fig. 12, where the number of extracted lines is plotted against image reduction from 1% to 95%. As there are several thousands of lines in these images, checking the correctness of these lines is very tedious and time-consuming work. Hence we tried to find the suitable size of image reduction by an observation. If a line is correctly extracted, then it should also be extracted when the size of reduction is slightly changed. In other words, the number of extracted lines should be quite stable at a certain range of size reduction. Figure 12 confirms this observation where the number of extracted lines is very stable at the range of 10% to 20% image reduction. Hence, we select the median of the range, i.e. 15% image reduction, for the generation of DRDE image in our line extraction algorithm. From our experiments, 15% image reduction can produce very satisfactory line extraction results.

C. Line detection results

We now evaluate the performance of proposed line extraction and detection algorithm. 417 journal pages scanned by a Panasonic KV-S3065C Scanner with resolution 200 dpi were used to test the proposed algorithm. In addition to ordinary text, these journal pages also contain many tables, graphics, figures, and mathematical expressions. Figure 13 shows the detection results of three J-images, where different objects are enclosed by rectangles of different colors. Ordinary texts are enclosed by red rectangles, while figures, isolated, and embedded mathematical expressions are enclosed respectively by green, black, and blue rectangles. This figure can let us roughly understand the region and position of extracted



(a) (b)

Fig. 11. Line extraction under different sizes of image reduction. In (a), the J-image is reduced to 5% of its original size, while in (b) the reduction is 95%.

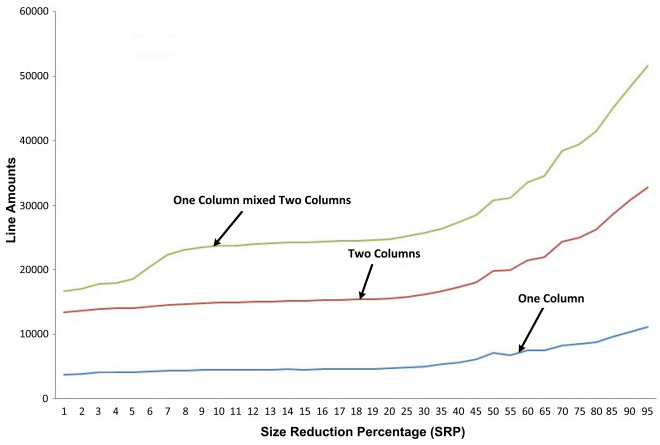


Fig. 12. Number of extracted lines against different sizes of image reduction for three types of J-images.

lines and its classification results. Most objects in the images are well localized and identified.

To further understand the classification accuracy of proposed approach, each extracted line in the 417 J-images is investigated to verify its correctness. Table II lists the number of objects, number of detected objects, and number of detected but mis-classified objects for the four categories of objects. The table also shows the classification accuracy of each category, which is defined as one minus the number of detected but mis-classified objects over the number of objects in that category. From this table we can see that for ordinary text which occupies a very large portion of detected lines, the accuracy approaches 99%, while for the other three categories of objects, the accuracies are about 90%. The accuracy considering all four types of objects can achieve to nearly 98%. This demonstrates the potential of proposed approach to be an effective technique for analyzing scanned journal pages without employing any OCR technology.

TABLE II
ACCURACY OF OBJECT DETECTION USING PROPOSED APPROACH. HERE N_1, N_2, AND N_3 REPRESENT NUMBER OF OBJECTS, DETECTED OBJECTS, AS WELL AS DETECTED BUT MISS-CLASSIFIED OBJECTS, RESPECTIVELY.

	$\mathbf{O}_{text}$	$\mathbf{O}_{fig}$	$\mathbf{O}_{iMath}$	$\mathbf{O}_{eMath}$	All objects
N_1	25686	509	1682	1588	29465
N_2	25610	483	1757	1615	29465
N_3	218	46	164	159	587
Accuracy	99.2%	91%	90.2%	90%	98%

In order to further improve the accuracy of proposed system in the future, some lines that are falsely classified are examined. Four cases of mis-classified objects are displayed in Fig. 14 and discussed in the following.

1) Some embedded mathematical expressions are located at the final line of a paragraph and hence the lines have smaller width. These lines are classified as isolated mathematical expressions as shown in Fig. 14(a).

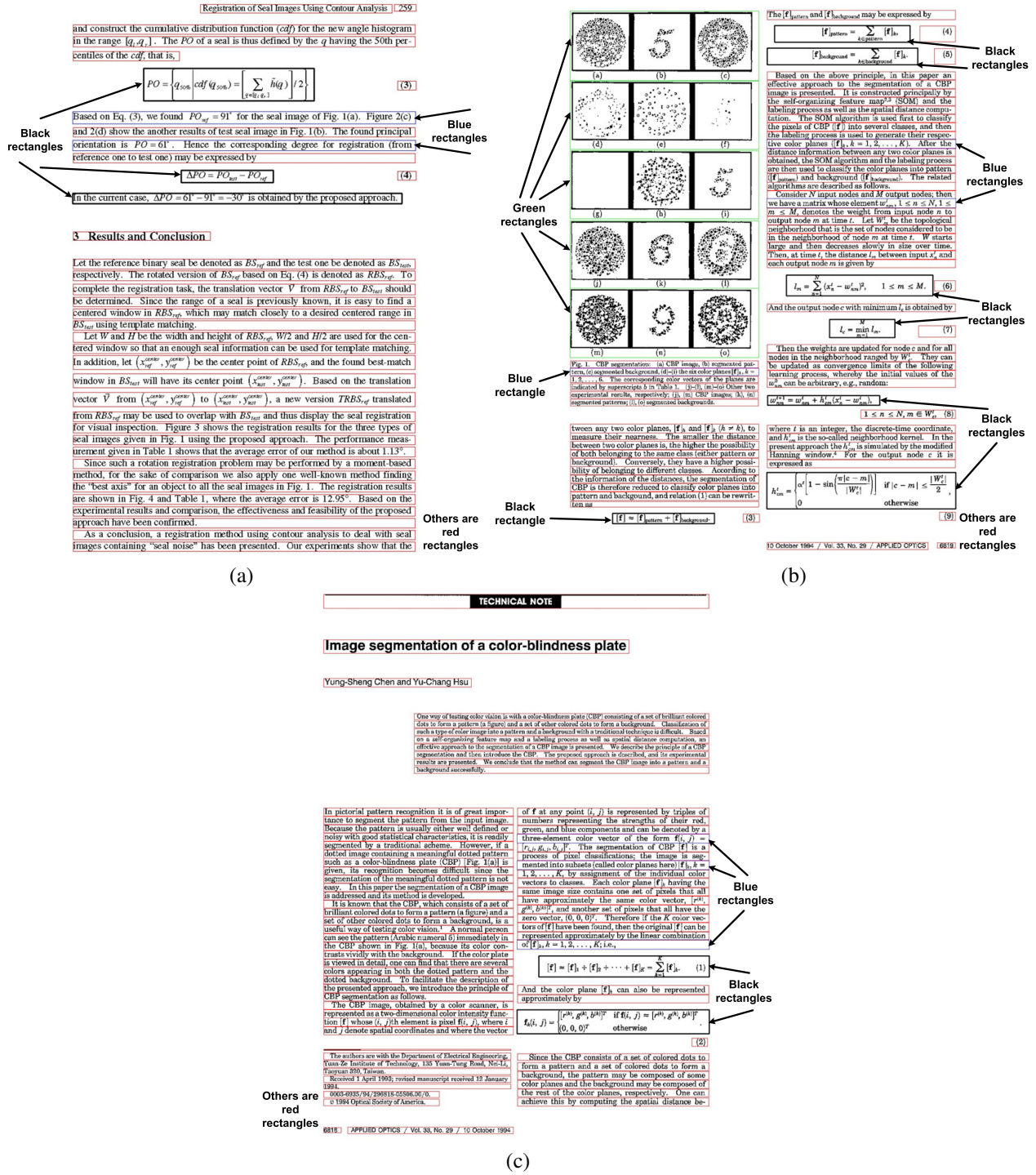


Fig. 13. Some J-images with detected lines for the cases of (a) one-column, (b) two-column, and (c) two-column mixed with one-column. The red, green, black, and blue rectangles indicate the extracted lines are ordinary texts, figure objects, isolated mathematical expressions, and embedded mathematical expressions, respectively.

2) Some isolated mathematical expressions have a conjunction such as "and" between them. Because its position (and density of black pixels) is very similar to that of a superscript or subscript of isolated mathematical expressions, it is possibly detected as a superscript or subscript as shown in Fig. 14(b).
3) Some headers and footers in a J-image consisting of complex logos or images are possibly classified as an isolated mathematical expression as the example shown in Fig. 14(c).
4) Some text lines have large embedded mathematical expressions in them. These lines are not easily to decompose by simple horizontal scanning of white pixels along a row. Hence, these lines may merge to a large block which may be classified as a figure object as show in Fig. 14(d).

Although some lines were not correctly classified, our approach still performed well for most objects. Moreover, our approach has strong capability in J-image skew restoration, page orientation recovering, and page inversion correction. We believe that it could be a useful preprocessing tool for further journal page understanding.

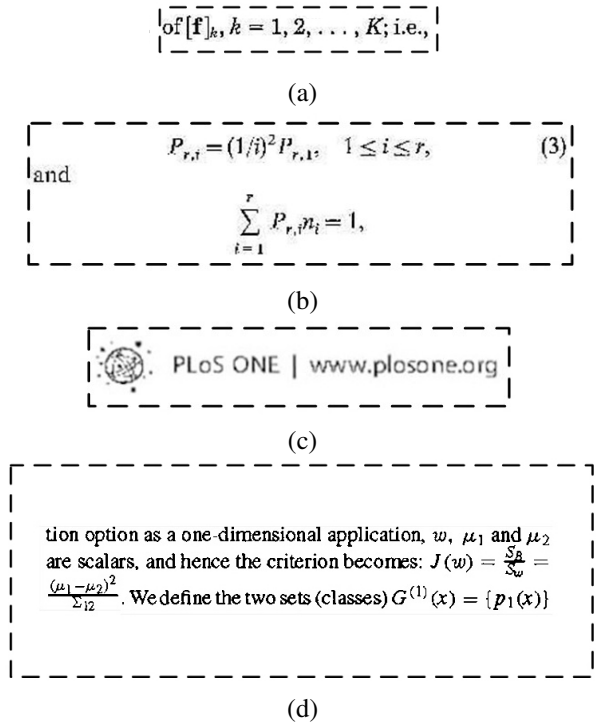


Fig. 14. Four cases of false line detection.

VIII. Conclusion and Future Works

In this paper, we have presented an effective journal page analysis system which can extract the lines in the page and classify them into four categories including ordinary text lines, figures or tables, isolated mathematical expressions, and embedded mathematical expression lines. Image skew, incorrect image orientation, and page inversion that might be produced from scanning imperfections were also considered in the system. The proposed system can perfectly detect such imperfections and correct an image to an appropriate orientation. The system design is based on very fundamental image processing techniques such as vertical and horizontal image scanning, image dilation, and size reduction. Most lines in a journal image were correctly extracted by these techniques. The classification of extracted lines is also achieved by simple techniques such as variance of lower bounding curve of a line, width or height of the extracted line, and equal mark checking. No OCR technique is applied in our approach which makes the system easy to be implemented. Although the design of the system is simple, experimental results on 417 journal pages showed that the system can achieve to 90% above detection accuracy for extracted lines. This confirms the feasibility of the proposed approach.

In addition to improving detection accuracy, along this research the mathematical expression analysis is worthy of studying in the near future, which may include three main processes. They are to recognize the symbols and operators in a mathematical expression line, construct the relationships among the recognized objects, and translate the mathematical expression into a Tex (LaTex) format for further application. It is expected that such a future work may benefit the transformation of a J-image into a well-structured document as well as the application of document retrieval.

References

[1] T.-Y. Chang, Y. Takiguchi, and M. Okada, "Physical structure segmentation with projection profile for mathematic formulae and graphics in academic paper images," in *International Conference on Document Analysis and Recognition*, vol. 2, pp. 1193–1197, 2007.

[2] B. Chaudhuri and U. Garain, "An approach for recognition and interpretation of mathematical expressions in printed document," *Pattern Analysis and Applications*, vol. 3, pp. 120–131, 2000.

[3] Y.-S. Chen, "Registration of seal images using contour analysis," in *Proc. 13th Scandinavian Conference on Image Analysis*, vol. LNCS 2749, pp. 255–261, 2003.

[4] Y.-S. Chen and P.-H. Li, "Skew detection using contour analysis for a scanned journal page," in *Proc. The 13th IAPR International Conference on Machine Vision Applications*, pp. 81–84, 2013.

[5] Y.-S. Chen, F.-S. Chen and C.-H. Teng, "An optical music recognition system for skew or inverted musical scores," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 27, no. 7, 1353005 (23 pages), 2013.

[6] C.-H. Chou, S.-Y. Chu, and F. Chang, "Estimation of skew angles for scanned documents based on piecewise covering by parallelograms," *Pattern Recognition*, vol. 40, no. 2, pp. 443–455, 2007.

[7] S. Chowdhury, S. Mandal, A. Das, and B. Chanda, "Automated segmentation of math-zones from document images," in *Proc. International Conference on Document Analysis and Recognition*, pp. 755–759, 2003.

[8] A. Das and B. Chanda, "A fast algorithm for skew detection of document images using morphology," *International Journal on Document Analysis and Recognition*, vol. 4, no. 2, pp. 109–114, 2001.

[9] P. Dey and S. Noushath, "e-PCP: A robust skew detection method for scanned document images," *Pattern Recognition*, vol. 43, no. 3, pp. 937–948, 2010.

[10] B. V. Dhandra, V. S. Malemath, H. Mallikarjun, and R. Hegadi, "Skew detection in binary image documents based on image dilation and region labeling approach," in *Proc. International Conference on Pattern Recognition*, vol. 2, pp. 954–957, 2006.

[11] H. Fan, L. Zhu, and Y. Tang, "Skew detection in document images based on rectangular active contour," *International Journal on Document Analysis and Recognition*, vol. 13, no. 4, pp. 261–269, 2010.

[12] U. Garain, B. Chaudhuri, and A. Chaudhuri, "Identification of embedded mathematical expressions in scanned documents," in *Proc. International Conference on Pattern Recognition*, vol. 1, pp. 384–387, 2004.

[13] U. Garain and B. Chaudhuri, "Segmentation of touching symbols for OCR of printed mathematical expressions: an approach based on multifactorial analysis," in *Proc. International Conference on Document Analysis and Recognition*, pp. 177–181, 2005.

[14] U. Garain, "Identification of mathematical expressions in document images," in *Proc. International Conference on Document Analysis and Recognition*, pp. 1340–1344, 2009.

[15] Y. S. Guo, L. Huang, C. P. Liu, and X. Jiang, "An automatic mathematical expression understanding systems," in *Proc. International Conference on Document Analysis and Recognition*, vol. 2, pp. 719–723, 2007.

[16] J. Jin, X. Han, and Q. Wang, "Mathematical formulas extraction," in *Proc. International Conference on Document Analysis and Recognition*, pp. 1138-1141, 2003.

[17] A. Kacem, A. Belaid, and M. Ben Ahmed, "EXTRAFOR: automatic extraction of mathematical formulas," in *Proc. International Conference on Document Analysis and Recognition*, pp. 527–530, 1999.

[18] A. Kacem, A. Belaid, and M. Ahmed, "Embedded formulas extraction," in *Proc. International Conference on Pattern Recognition*, vol. 1, pp. 676–680, 2000.

[19] H. J. Lee and J. S. Wang, "Design of a mathematical expression understanding system," *Pattern Recognition Letters*, vol. 18, no. 3, pp. 289–298, 1997.

[20] J.-H. Lee, M.-Y. Wu and T.-H. Tseng, "A framework of video news system using image segmentation and augmented reality," *Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2011*, WCE 2011, 6-8 July, 2011, London, U.K., pp. 1663–1668.

[21] S. Li, Q. Shen, and J. Sun, "Skew detection using wavelet decomposition and projection profile analysis," *Pattern Recognition Letters*, vol. 28, no. 5, pp. 555–562, 2007.

[22] Y. Li, K. Wang, W. Guan, and L. Tang, "The research of mathematical formula recognition method base on baseline structure analysis," in *Proc. International Conference on Internet Computing in Science and Engineering*, pp. 53–59, 2008.

[23] H. Liu, Q. Wu, H. Zha, and X. Liu, "Skew detection for complex document images using robust borderlines in both text and non-text regions," *Pattern Recognition Letters*, vol. 29, no. 13, pp. 1893–1900, 2008.

[24] V. N. Manjunath Aradhya, G. Hemantha Kumar and P. Shivakumara, "Skew estimation technique for binary document images based on thinning and moments," *Engineering Letters*, vol. 14, no. 1, pp. 127–134, 2007.

[25] U. Pal and B. B. Chaudhuri, "An improved document skew angle estimation technique," *Pattern Recognition Letters*, vol. 17, no. 8, pp. 899–904, 1996.

[26] S. B. Rezaei, A. Sarrafzadeh and J. Shanbehzadeh, "Skew detection of scanned document images," *Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2013*, IMECS 2013, 13-15 March, 2013, Hong Kong, pp. 451–456.

[27] P. Shivakumara and G. Hemantha Kumar, "A novel boundary growing approach for accurate skew estimation of binary document images," *Pattern Recognition Letters*, vol. 27, no. 7, pp. 791–801, 2006.

[28] X. D. Tian, H. Y. Li, X. F. Li, and L. P. Zhang, "Research on symbol recognition for mathematical expressions," in *Proc. International Conference on Innovative Computing, Information and Control*, vol. 3, pp. 357–360, 2006.

[29] J. Y. Toumit, S. Garcia-Salicetti, and H. Emptoz, "A hierarchical and recursive model of mathematical expressions for automatic reading of mathematical documents," in *Proc. International Conference on Document Analysis and Recognition*, pp. 119–122, 1999.

[30] S. Tsujimoto and H. Asada, "Major components of a complete text reading system," *Proceedings of the IEEE*, vol. 80, no. 7, pp. 1133–1149, 1992.

[31] S. Yadav and S. Sawarkar, "Retrieval of information in document image databases using partial word image matching technique," *Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2009*, IMECS 2009, 18-20 March, 2009, Hong Kong, pp. 902–907.