Chapter 5

Preprocessing for Images Captured by Cameras

Chih-Chang Yu, Ming-Gang Wen, Kuo-Chin Fan and Hsin-Te Lue

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/52110

1. Introduction

Due to the rapid development of mobile devices equipped with cameras, the realization of what you get is what you see is not a dream anymore. In general, texts in images often draw people's attention due to the following reasons: semantic meanings to objects in the image (e.g., the name of the book), information about the environment (e.g., a traffic sign), or commercial purpose (e.g., an advertisement). The mass development of mobile device with low cost cameras boosts the demand of recognizing characters in nature scenes via mobile devices such as smartphones. Employing text detection algorithms along with character recognition techniques on mobile devices assists users in understanding or gathering useful information around them. A useful mobile application is the translation tool. Using handwriting as the input is widely used in current translation tools on smartphones. However, capturing images and recognizing texts directly is more intuitive and convenient for users. A translation tool with character recognition techniques recognizes texts on the road signs or restaurant menus. Such application greatly helps travelers and blinds.

The mobility advantage inspires users to capture text images using mobile devices rather than scanners, especially in outdoors. Optical character recognition (OCR) is a very mature technique accomplished by many previous researchers. However, camera-based OCR is a more difficult task than traditional OCR using scanners. Scanned images are captured with high resolution, even illumination, simple background, high contrast, and no perspective distortion. These properties ensure that high recognition rates can be achieved when employing OCR. Conversely, images captured by cameras on mobile devices include many external or unwanted environmental effects which deeply affect the performance of OCR. These images are often captured with low resolution and fluctuations such as noises, uneven illuminations or perspective distortions, etc. In that case, low quality images cause the camera-based OCR more challenging than traditional OCR. The reason is that the extracted character blobs are usually broken or stuck together (also called as "ligature") in low quality images. It is a prerequisite to clearly detect foreground texts before proceeding

INTECH
open science | open minds

to later recognition task. To facilitate the processing of camera-based OCR, a good preprocessing is highly required.

This chapter discusses how to segment text images into individual single characters to facilitate later OCR kernel processing. Before the character segmentation procedure, several works such as text region detection and text-line construction need to be done in advance. First, regions in images are classified into text and non-text region (e.g. graphics, trademarks, etc.). Second, the text components are grouped to form text-lines via a bottom-up approach. After text-line construction, typographical structure is analyzed to distinguish inverted (upside-down) texts. Finally, a character segmentation method is introduced to segment ligatures, which often appear on text images captured by cameras. In the following sections, these processes will be described in detail.

2. Related works

Instead of discussing the character recognition techniques, this chapter focuses on the new challenges imposed by the imperfect capturing conditions mentioned in the first section. More specifically, some methods are proposed to detect foreground texts and segment each character from an image correctly. In the proposed preprocessing system, there are three main procedures: text detection, text-line construction and character segmentation. Before that, a brief review of several works done by previous researchers is described in the following subsections.

2.1. Text detection

The current text detection researches are roughly divided into rule-based and classifier-based approaches. Rule-based methods [1-5] formulate rules with prior-knowledge to distinguish text and non-text blocks. Conditional constraints are often adopted in these rules, such as the sizes of connected components, edge information, color information and texture information. Adopting edge information is inspired by the observations that texts often cluster together and have high contrast to backgrounds. Regions with large enough variances and sufficient amount of edge pixels are regarded as the text candidates. Color information is utilized with region growing and clustering methods [6, 7]. The rules formulated by experienced experts filter texts efficiently but may not be robust. Texts themselves can be regarded as textures [8]. In this type of approach, images are transformed to frequency domains by using filters such as DCT [9], FFT [10], Wavelet [11], Gabor filter [12], etc. to reveal distinct textural properties so that text regions can be separated from background regions.

The classifier-based methods [13-16] utilize the extracted features as the input of specific classifiers, such as neural networks or Support Vector Machines (SVM) to classify text and non-text components. The classifiers usually need enough samples to be trained well. Moreover, the parameters of these classifiers often have to be tuned case by case to get the best classification results.

2.2. Text-line construction

After finding text components, these components are linked one another to form meaningful text-lines (i.e. words and sentences). Text-lines are constructed based on the distance between two text blocks with the observation that the row spacing is often larger than the character spacing in most documents. In traditional page segmentation, top-down approaches such as X-Y Cut [18, 19] and the run-length smearing algorithm (RLSA) [20] are widely used to find paragraphs, text-lines, and characters, and then segment them by horizontal and vertical projections. However, both methods are infeasible to segment the document when the image is skewed.

From another point of view, when document images are with unknown structures, the bottom-up methods are more practical than the top-down methods to construct text-lines. Hough transform is a well-known algorithm to find potential alignments in images. However, Hough transform is a computationally expensive method. The minimum spanning tree methods [21, 22] are employed according to the properties of text clustering and typesetting. The extracted minimum spanning trees are not considered the text-line structures yet; some criteria are further adopted to delete redundant or add additional edges to form complete text-lines. Basu et al. [23] propose a water flow method to construct text-lines. Hypothetical water flows from both left and right image margins to opposite image margins. Areas are wetted after the flood. In their approach, text regions are obstructions which block water flows so that the un-wetted areas can be linked to form text-lines. The disadvantage of water flow algorithm is that the threshold of the flow filter is empirically determined.

2.3. Character segmentation

Traditional character segmentation techniques are categorized into projection methods, minimal segmentation cost path method, recognition-feedback-based methods, and classifier-based methods. Projection methods [3, 24] project the image along the horizontal and vertical directions. The locations with no projection values are believed to be the locations of spacing. The projection methods are efficient but have difficulties in resolving ligatures and broken characters. More specifically, when ligatures occur, the amount of spacing is often less than the number of characters. Conversely, the amount of spacing is often more than the number of characters when one character breaks into several blobs. Hence, confirming segmentation locations using the projection method only is risky in camera-based OCR because ligatures and broken characters are very likely to occur. If characters are stuck together severely, the segmentation results will be wrong. Another situation is the emergence of broken characters. Broken characters result in over-segmentation due to the occurrence of many locations with no projection values. It is infeasible to segment characters by using projection method if images are skewed or contain italic fonts. Hence, the projection methods often collaborate with other methods for correct segmentation result. The minimal segmentation cost method is to find a segmentation path with minimal cost in images. The weights of foregrounds and backgrounds are pre-specified. To reduce the complexity of finding the optimal segmentation path, certain constraints such as path movement range and divided zones are integrated with dynamic programming [25, 26].

The recognition-feedback-based methods, [27, 29] provide a recovery mechanism for wrong segmentations. These methods seek some segmentation points to divide ligatures into several segmented blocks. The segmented blocks are then fed into the recognition kernel. If the recognition rate of the segmented block is above a certain threshold, the segmentation point is considered as legal. Otherwise, the segmented block is illegal and the corresponding segmentation point is abandoned. This method is more reliable than the projection methods, but the computation cost is also more expensive. Classifier-based methods [30, 31] select segmentation points using classifiers trained by correct segmentation samples. The drawback of classifier-based method is that classifiers require enough training samples to obtain better segmentation results.

3. Preprocessing

The main challenge for the preprocessing system is that the captured images are often with low resolution. Although cameras on mobile devices are capable of taking higher resolution images, the computation cost is still an issue nowadays. The preprocessing system consists of three modules: text detection, text-line construction, and character segmentation to provide acceptable inputs (i.e. individual character images) for OCR.

3.1. System flowchart

The flowchart of the preprocessing system is illustrated in Figure 1. In the text detection module, foreground blobs are separated from backgrounds. These foreground blobs are classified into text connected components (CC) and non-text CCs using the text-noise filter. In the text-line construction module, the text CCs are used to construct rough text-lines first. Then the text-line completeness and reading order confirmation are achieved via the features of employed typographical structure. In the character segmentation module, each text CC is classified as a single character or a ligature. If the text CC is classified as a ligature, it is segmented via the proposed segmentation mechanism.

3.2. Text detection

The first work of the preprocessing system is to find the locations of texts. Text images include texts, graphic, backgrounds, and tables. In general, texts are with high contrast to the backgrounds. Based on this observation, foregrounds can be separated from backgrounds by image binarization. The segmented foregrounds are labeled as connected components (CCs) by 8-ways connected component labeling method. However, binarizing all images using a fixed threshold is improper because the external lighting conditions of

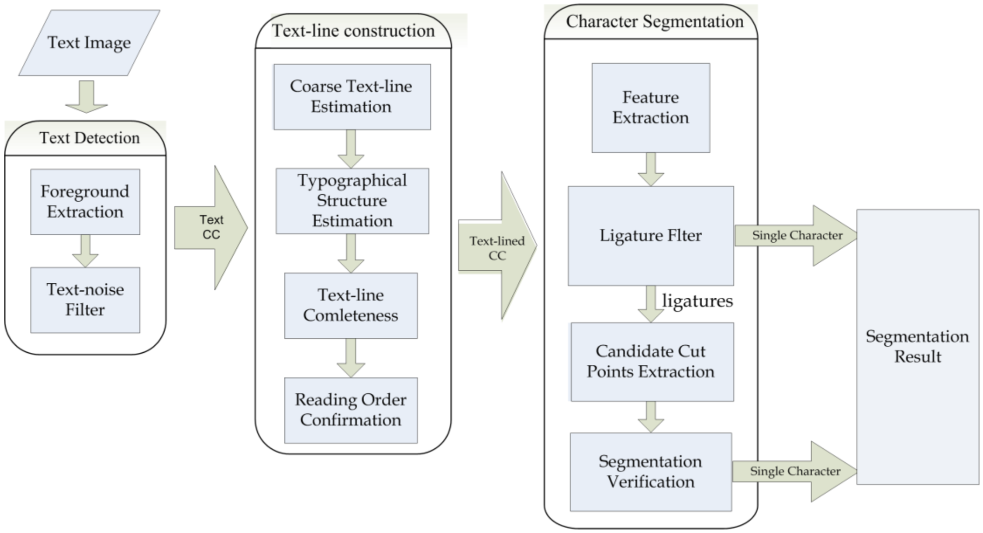
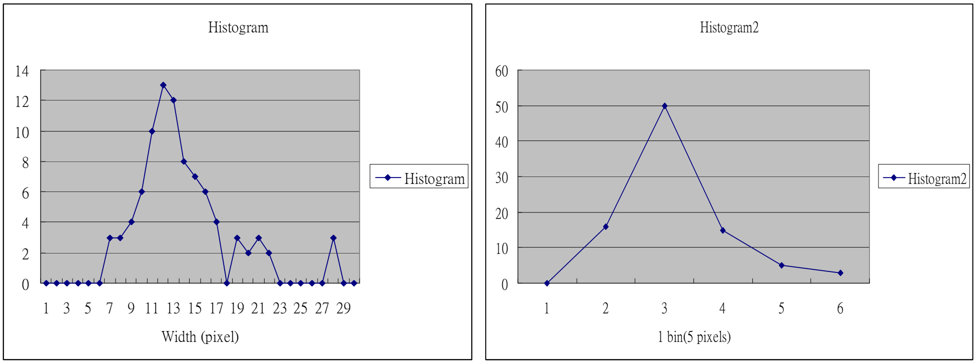


Figure 1. Flowchart of the preprocessing system.

text images are usually not the same. A two-stage binarization mechanism which adopts the well-known Otsu's method [32] is proposed. In the first stage, foreground blobs are extracted using a global threshold which is automatically found by the Otsu's method. The found foreground blobs contain noises, pictures, and texts. To reduce the computational cost of the text-line construction module, these blobs are classified into text and non-text CCs using a text-noise filter. Only the text CCs are used to construct a rough text-line in the text-line construction module. Afterwards, Otsu's method is performed again in a small region of each individual text-line area to complete the text CCs. It is helpful for the character segmentation module when the contours of text CCs are clearer after the binarization in the second stage.

A statistical approach is adopted to distinguish text CCs from non-text CCs. The widths and heights of CCs form two histograms. Figure 2 (a) is an example of the width histogram. Every 5 bins of the histogram in Figure 2 (a) are summed up to form the second histogram (see Figure 2 (b)). The majority of the second histogram can be acquired and the average width is calculated by the width values belong to the major bin. As shown in Figure 2 (b), the majority bin is bin #3, which corresponds to the 11th -15th bin of the histogram in Figure 2(a). Hence, the average width of CCs is 13 in this case. Same procedure can be applied to the height histogram to obtain the average height of CCs. CCs sizes of which are larger than the ratio of product of average width and average height are labeled as non-text CCs.



(a) (b)

Figure 2. Example of histogram used for finding the average width of text CCs.(a) Histogram of the widths of CCs and (b) histogram which sums up every 5 bins of (a).

The CCs are normalized to a fixed size before passing the text-noise filter. Then, auto-regressive (AR) features [42] are extracted from the CCs as the inputs of neural network for text-noise classification. The misclassified text CCs in this procedure are recovered using the properties of text clusters and text-lines during the text-line construction procedure, which will be described in the following subsection.

3.3. Text-line construction

The goal of the text-line construction is to find the reading order of a text and construct a linked-list of characters. A distance-based method is designed herein to construct text-lines according to the following characteristic: the row spacing is often greater than the word spacing in most document layouts. Instead of calculating the distance between the central points of two text CCs, the distance between two CCs is estimated by the "out-length". The out-length is defined as the length of the segment between the bounding boxes of two text CCs (see Figure 3). The advantage of using the out-length measurement is that the out-length values remain small even the widths of text CCs are large (this usually happens on ligatures) as long as they are on the same text-line. Figure 4 illustrate the consideration of neighboring CCs for each CC by using the out-length. If we consider the distances between the central points of CCs, CC1 and CC2 will be considered as close CCs and the text-line will thereby be constructed in a wrong direction. Instead, CC3 is closer to CC2 than CC1 using the out-length measurement. Hence, a correct text-line can be constructed.

A two-stage statistical method is proposed herein to find the reading order of text-lines. In the first stage, for each text CC, a neighboring candidate CC which has the smallest out-length to it is chosen. Then, the angle θ between the horizontal line and the line linking the central points of these two neighboring CCs is computed (see Figure 4). A histogram is constructed and the angle θ_m with the majority votes in the histogram is utilized to determine the coarse reading order (that is, the orientation) of the document. The coarse reading order estimated in the first stage is temporally assumed as the correct reading order to construct the initial text-lines. For each CC, only the smallest and second smallest out-length values are considered according to the fact that a character in text-lines has two neighbors at most. The text-line construction algorithm is stated as follows:

Step 1. For an unvisited CC_i and its neighboring CC_j, angle θ_{ij} which is the angle between CC_i and CC_j are evaluated by the following equation

$$\theta_m - \varepsilon < \theta_{ij} < \theta_m + \varepsilon \tag{1}$$

where θ_m is the temporary reading order orientation, and ε is a tolerance threshold. The purpose of Eq. (1) is to link several CCs into a text-line along a straight direction. If θ_{ij} satisfies the inequality in Eq. (1), go to Step 2. Otherwise, select another neighboring CC_k with the second smallest out-length and check the inequality again using angle θ_{ik}. If θ_{ik} satisfies the inequality in Eq. (1) is satisfied for θ_{ik}, go to Step 3. If both θ_{ij} and θ_{ik} cannot satisfy Eq. (1), go to step 4.

Step 2. Link CC_i to CC_j. Go to step 1 and check the next text candidate.

Step 3. Link CC_i to CC_k. Go to step 1 and check the next text candidate.

Step 4. CC_i cannot be connected with any CC at this stage. Find another unvisited CC_p and go to step 1. If all CCs have been visited, terminate the algorithm.

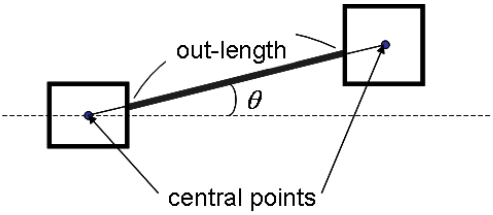


Figure 3. Illustration of out-length and slant angle between two CCs.

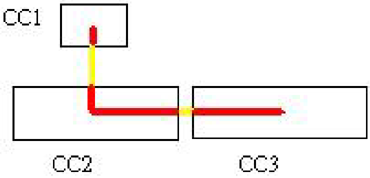
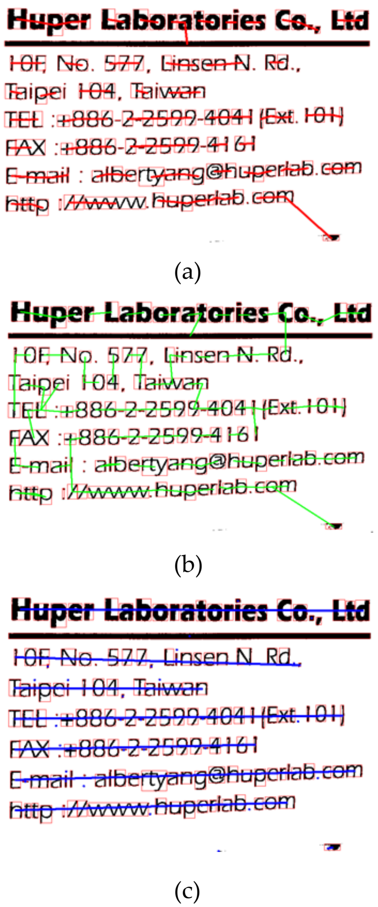


Figure 4. Illustration that CC3 is closer to CC2 than CC1 by using out-length, but CC1 is closer than CC3 estimated by using the distance between the central points of the CCs.

Figure 5 (a) depicts the link between all CCs and their corresponding nearest CCs using the out-length measurement. Figure 5 (b) illustrates the link of the second nearest CCs. The coarse orientation θ_m of text-lines in Figure 5 is horizontal. After performing the algorithm, most CCs are linked to form some text-lines, as shown in Figure 5 (c). Some estimated text-lines in Figure 5 (c) are not accurate enough. These inaccurate text-lines will be refined in the next stage.


Huper Laboratories Co., Ltd
10F, No. 577, Linsen N. Rd.,
Taipei 104, Taiwan
TEL :+886-2-2599-4041 (Ext. 101)
FAX :+886-2-2599-4161
E-mail : albertyang@huperlab.com
http ://www.huperlab.com

(a)

(b)

(c)

Figure 5. Illustration of the coarse text-line construction in the first stage. (a) Links of the nearest neighbors (b) links of the second nearest neighbor (c) results of performing the text-line construction algorithm.

In the second stage, the extracted text-lines are further refined using typographical structures and the geometry information of CCs in text-lines. Typographical structures [34] have been designed since the era of typewriter and are still preserved in the printed fonts today. Figure 6 illustrates the four lines (called Typo-lines) which bound all printed English characters in three areas. The four lines are named as the *top line*, *upper line*, *baseline*, and *bottom line*. The three areas within the four lines are called the upper, central, and lower zones. The printed alphanumeric characters and punctuation marks locate in particular positions according to the design of typographical structure. For instance, capital letters only appear in the upper zone and central zone. The printed alphanumeric characters and punctuation marks are classified into seven categories, called Typo-classes, according to their locations in the Typo-lines. The seven Typo-classes are listed below:

1. *Full*: the character occupies three zones, such as j, left parenthesis, right parenthesis, and so on.
2. *High*: the character is located in both upper and central zones, such as capital letters, numerals, b, d, and so on.
3. *Short*: the character is only located in the central zone, such as a, c, e, and so on.
4. *Deep*: the character appears in central zone and lower zone. Only the four lowercase letters g, p, q, and y belong to this Typo-class.
5. *Subscript*: the punctuation mark is closer to the baseline, such as comma, period, and so on.
6. *Superscript*: the punctuation is closer to the upper line, such as quotation marks, double quotation marks, and so on.
7. *Unknown*: the class is given when the Typo-class cannot be confirmed due to the lack of certain Typo-lines.

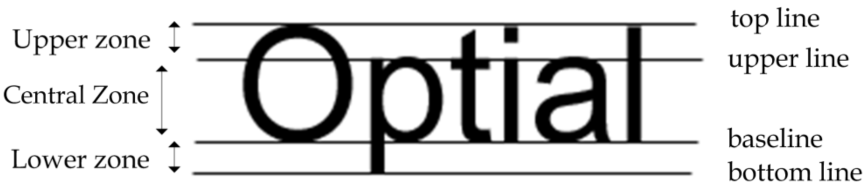


Figure 6. Typographical structure

To determine the typographical structure, a Typo-line extraction algorithm which integrates k-means and least mean square error (LMSE) algorithm is proposed. First, the y coordinates of the top edges of all CCs which belong to the same text-line are classified into two clusters using the k-means algorithm. Second, the LMSE algorithm is applied to each cluster to determine the corresponding Typo-lines (top line and upper line). The baseline and bottom line can also be extracted using the same procedure. Figure 7 depicts the extracted text-line in the first stage and the Typo-lines which are obtained in the second stage.

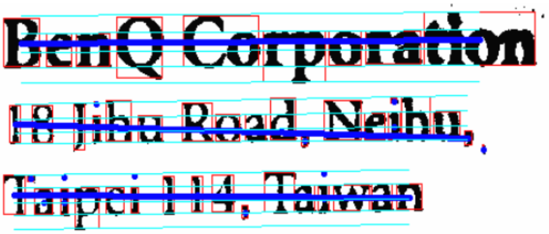


Figure 7. Initial text-line and extracted Typo-lines

The LMSE algorithm for finding Typo-lines is described as follows. The line formulation to represent a Typo-line is

$$y = f(x) = a + bx \tag{2}$$

Then the least square error E can be formulated as

$$E = \sum_i \left[y_i - f(x_i) \right]^2 = \sum_i \left[y_i - \left(a + bx_i \right) \right]^2 \tag{3}$$

The least square error is minimal when E is zero. The first derivative is applied on E:

$$\begin{aligned} \frac{\partial E}{\partial a} &= 2\sum_{i=1}^{n} \left[y_i - \left(a + bx_i \right) \right] = 0, \\ \frac{\partial E}{\partial b} &= 2\sum_{i=1}^{n} \left[y_i - \left(a + bx_i \right) \right] x_i = 0 \end{aligned} \tag{4}$$

Equation (4) can be extended as follow:

$$\begin{aligned} \sum_{i=1}^{n} y_i &= \sum_{i=1}^{n} a + \sum_{i=1}^{n} bx_i = a\sum_{i=1}^{n} 1 + b\sum_{i=1}^{n} x_i, \\ \sum_{i=1}^{n} y_i x_i &= \sum_{i=1}^{n} ax_i + \sum_{i=1}^{n} bx_i^2 = a\sum_{i=1}^{n} x_i + b\sum_{i=1}^{n} x_i^2 \end{aligned} \tag{5}$$

Finally, the two unknowns a and b can be solved by

$$\begin{aligned} a &= \frac{\left(\sum_{i=1}^{n} y_i \right)\left(\sum_{i=1}^{n} x_i^2 \right) - \left(\sum_{i=1}^{n} x_i \right)\left(\sum_{i=1}^{n} y_i x_i \right)}{n\left(\sum_{i=1}^{n} x_i^2 \right) - \left(\sum_{i=1}^{n} x_i \right)^2}, \\ b &= \frac{n\left(\sum_{i=1}^{n} y_i x_i \right) - \left(\sum_{i=1}^{n} y_i \right)\left(\sum_{i=1}^{n} x_i \right)}{n\left(\sum_{i=1}^{n} x_i^2 \right) - \left(\sum_{i=1}^{n} x_i \right)^2} \end{aligned} \tag{6}$$

The orientation of texts is refined by taking the mean of the upper line and baseline. However, both the correct text CCs or upside-down text CCs generate a horizontal text-line. To solve this problem, the coarse reading order is also further confirmed in this stage. The confirmation is accomplished by analyzing the Typo-classes of the characters. The characters of *Full* and *Short* types remain the same when the image is rotated 180 degrees, but the *High* and *Deep* types do not. An observation is that all lowercase letters consist of 13 *Short* types, 8 *High* types, 4 *Deep* types and 1 *Full* type. The reading order can be confirmed by a cue that the appearance rates of the *High* and *Deep* type characters are significantly different when the texts are upside-down. Baker and Piper [35] calculated the appearance rates of 100362 lowercase letters in newspapers and novels. The appearance rates of the Typo-classes are listed in Table 1. The reading order is correct if the appearance rate of the *High* type is significantly larger than that of the *Deep* type. Hence, if the documents are captured with a slanted angle, the images can be de-skewed according to the slope of Typo-lines.

Typo-Class	Appearance rate
Full	0.2
High	35
Short	59.1
Deep	6

Table 1. Appearance rate of Typo-class

The details of reading order confirmation process is summarized as follows:

1. If the extracted text-line is not horizontal, rotate the image to horizontal according to the orientation of the estimated text-line.
2. Extract Typo-lines and verify whether the number of the High type characters is larger than that of the Deep type characters or not. If the number of the High type characters is greater than that of the Deep type characters, the reading order orientation is correct. Otherwise, rotate the image by 180 degree and inverse the order of text CCs in the text-line.

In the aforementioned text-noise filter, the text CCs may be wrongly classified as noises due to the low quality of images. These mis-classified text CCs are often located around or inside the text-lines (e.g. the dots or commas). Sometimes these missing text CCs result in breaking the text-lines (see Figure 15). To solve this problem, the bounding boxes of all estimated text-lines are slightly extended to seek possible merge. If two text-lines are overlapped after an extension, they are merged into a single text-line. Moreover, if the mis-classified text CCs fall in the bounding box of the text-lines, they are reconsidered as the text CCs and linked to the existed text CCs in the text-lines. The bounding boxes of the text-lines are extended by twice of the average width of characters to recover the mis-classified CCs nearby. By utilizing the characteristics of the typographical structure, the text CCs that are mis-classified as noises by the text-noise filter can be recovered.

3.4. Character segmentation

In traditional character segmentation, the ligatures often result from the specific character sequences with the specific font. For example, the character sequences “ti” with the font “Times new roman” are usually considered as the character “d”. In terms of the images captured by cameras, the characters are touched severely due to the blurred character boundaries. In this section, a character segmentation mechanism with the ligature filter is introduced. The text CCs are classified as a single character or ligature using the devised filter. The proposed filter consists of two stages. In the first stage, seven intrinsic features of CCs are obtained after using the projection method on text CCs. The vertical/horizontal projection is obtained by calculating the amount of foreground pixels in the vertical/horizontal direction respectively. Denote that the vertical projection and horizontal projection are P_v and P_h respectively. The intrinsic features are described as follows:

- c_1: the height-width ratio of the CC.
- c_2: the index of the maximum value of P_v with respect to the left boundary of the CC.
- c_3: the index of the maximum value of P_h with respect to the upper boundary of the CC.
- c_4: maximum value of P_v divided by the height of CC.
- c_5: maximum value of P_h divided by the width of CC.
- c_6: find the maximum value of P_v' divided by the height of CC where P_v' is the histogram which averages P_v every 5 bins.
- c_7: find the maximum value of P_h' divided by the height of CC where P_h' is the histogram which averages P_h every 5 bins.

The feature set $C=\{c_1,c_2,c_3,c_4,c_5,c_6,c_7\}$ is trained by two SVMs to classify CCs as a single character or a ligature. The feature set $\{c_1, c_2, c_3, c_4, c_5\}$ is used as the input for the first SVM, and $\{c_1, c_6, c_7\}$ is used for the second one. Some *High* type characters such as "ti" and "fl" are usually misclassified as "d" and "H" respectively. To cope with this problem, if the CC is considered as a single character by the first SVM and the Typo-class of the CC is *High* type as well, the CC is further verified by the second SVM. The positive and negative image samples for SVM training include 7 common types of font (Arial, Arial Narrow, Courier New, Time New Roman, Mingliu, PMingliu, and KaiU) and 4 different font sizes (32, 48, 52, and 72). The positive samples consist of single alphanumerical characters and punctuations. The negative samples are composed of two connected alphanumerical characters. The illustration of negative image samples is shown in Figure 8.

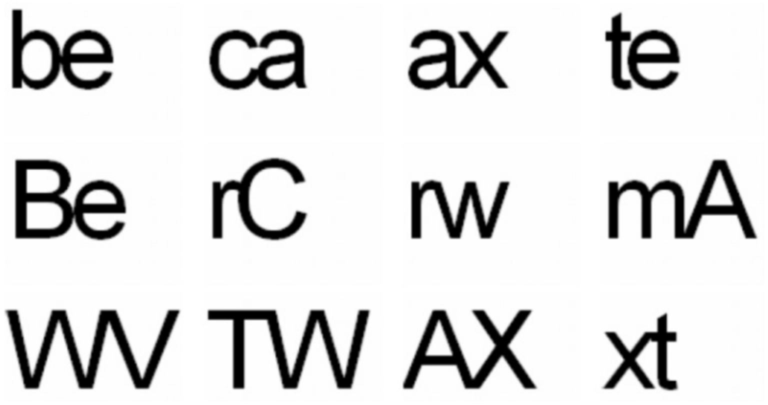


Figure 8. Illustration of negative samples of SVM database.

Text CCs which cannot pass both SVM classifiers are considered to be possible ligatures. These CCs will enter the second stage. In the second stage, the periphery features are extracted from the CCs. The periphery features are composed of 32 character contour values f_i, where $i = 1, 2,\ldots, 32$ as shown in Figure 9. In Figure 10, the closer the peripheral feature to the central position, the larger weight it is assigned. f_i is defined as follow:

$$f_i = W_{i \bmod 8} \frac{p_i}{l_i} \tag{7}$$

where the weight $W_{i\text{mod}8}$ can be obtained by referring to Figure 10. If $0 < i < 9$ or $16 < i < 25$, l_i is the character width. Otherwise, l_i is the character height. P_i is the distance between the boundary to the contour, i.e. the length of the blue band in Figure 9, where $0 < i < 9$ is the length of the boundary to the left contour, and so on. The 32 periphery features and an additional feature, the height-width ratio of CC, are concatenated to form a feature vector ***F***={ $f_1, f_2, \ldots, f_{33}$}.

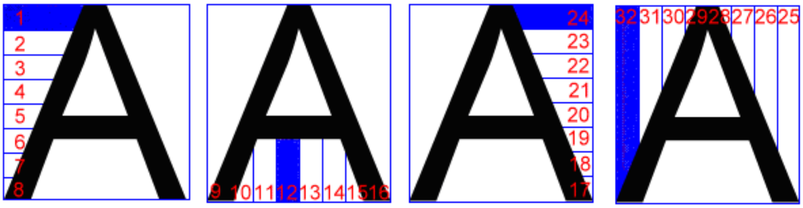


Figure 9. Illustration of 32 periphery features

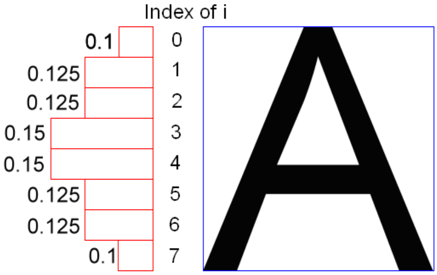


Figure 10. Illustration of weights of periphery features

The feature vector F is compared with the feature vector T, which is obtained from the templates. Suppose there are n templates need to be compared. For each periphery feature f_i, the score d_{ij} is defined as follow:

$$d_{ij} = \begin{cases} 1 \quad if \quad \left| f_i - T_i^j \right| < th_1 \quad \& \quad \left| f_{33} - T_{33}^j \right| < th_3 \\ -1 \quad if \quad \left| f_i - T_i^j \right| > th_2 \quad \& \quad \left| f_{33} - T_{33}^j \right| < th_3 , i = 1, \ldots, 32 \quad j = 1, \ldots, n \\ 0 \quad otherwise \end{cases} \tag{8}$$

two scores PV_j and NV_j are obtained by

$$PV_j = \sum_{i=1}^{32} d_{ij}, \forall d_{ij} > 0, NV_j = \sum_{i=1}^{32} d_{ij}, \forall d_{ij} < 0, j = 1, \ldots, n \tag{9}$$

Then, the final similarity PV_{max} and NV_{min} are obtained by finding the maximum value of PV_j and the minimum value of NV_j for j=1,…,n respectively. If PV_{max} is larger than a threshold and NV_{min} is smaller than another threshold as well, the CC is considered as a single character. Otherwise, the CC is considered as a ligature.

If the CCs are regarded as ligatures by the ligature filter, the CCs will enter to the character segmentation mechanism. The character segmentation mechanism consists of three steps:

1. Search the cut point candidates.
2. Segment ligatures using dynamic programming.
3. Verify the segmentation result.

Three features are utilized in searching possible cut points in a ligature: the vertical projection, the vertical profile, and the gray level vertical projection. Figure 11 (c) shows the vertical projection obtained from the image in Figure 11 (b). The vertical profile, also called the Caliper distance [31], is the distance between the top contour pixel and the bottom contour pixel in each bin. For example, shown in Figure 11 (e) is the vertical profile obtained from the image in Figure 11 (d).

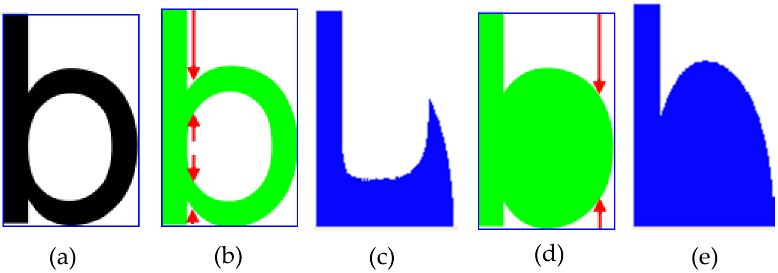


Figure 11. Illustration of vertical projection and vertical profile (a) Original character image, (b) accumulation of pixels in obtaining vertical projection, (c) the vertical projection of (b), (d) accumulation of pixels in vertical profile, and (e) the vertical profile of (d).

Define G as the set of the gray level projection of CC. That is, $G = \{g(0), g(1),\ldots,g(w\text{-}1)\}$ where w is the width of CC. The gray level projection $g(x)$ is formulated as follows:

$$g(x) = \sum_{y=0}^{h} \left| I(x,y) - 255 \right| \tag{10}$$

where $I(x,y)$ is the gray level value at pixel (x,y) and h is the height of the image. Figure 12 illustrates the process in obtaining the gray level projection in a gray level image. Figure 12 (b) depicts the projection result using Eq. (10). Figure 12 (c) is the final result after normalizing the gray level projection $g(x)$.

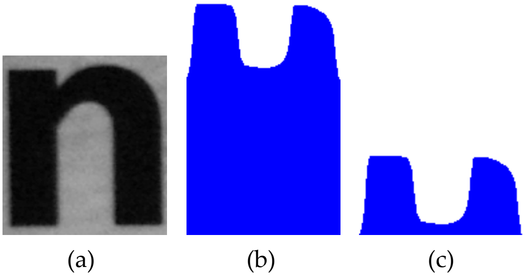


Figure 12. Illustration of gray level projection (a) Original image, (b) the gray level projection, and (c) the normalized gray level projection.

Denote the histograms of the three features mentioned above are *V*. The following equation is used to evaluate the validity of being a cut point at location *x*:

$$p(x) = \frac{V(lp) - 2V(x) + V(rp)}{V(x) + 1}, \quad x = 1, \ldots, w \tag{11}$$

where *V*(*lp*) is the first peak in the left of *x*, *V*(*rp*) is the first peak in the right of *x*, and *V(x)* is the value of *x*. The larger value of *p(x)*, the higher possibility *x* is a cut point. A selection rule is designed according to the following two criteria. The first criterion is that the number of cut points increases when the width-height ratio of CC increases. Hence, more points with larger values of *p(x)* have a higher tendency to be chosen as cut points. The second criterion is that the cut points near an already selected cut point should be ignored to reduce computation cost due to the restriction of minimum stroke width of a character. Figure 13 depicts the selection of cut point candidates. Given a ligature image shown in Figure 13(a), the cut point between 'n' and 'o' cannot be found by using the vertical projection only (see Figure 13 (f)). However, it can be successfully found by utilizing the vertical profile or Gray level projection as shown in Figure 13 (g) and (f).

If there are *n* cut point candidates, there will be 2^n combinations of selecting cut points, and only one combination of all possibilities segments the ligature image correctly. It is too difficult to find the correct combination without an efficient pruning mechanism. The periphery features of a character image are utilized again as the inputs of SVM to output a confidence value for evaluating the quality of the segmentation result. A combination of the cut points which has the highest confidence value is considered as the final segmentation result. The maximum confidence value can be efficiently computed using Dynamic Programming (DP). Suppose the number of cut point candidates plus the left and right boundary of the ligature image is *n*, $0 \le i \le i + k \le j \le n$, where *i, j, k, n, a, b* are integers, and *i*, *i*+*k*, *j* are the indices of cut points. The boundary conditions of DP are described as

$$m(i,j) = \begin{cases} 0, \quad if \quad i = j \\ Max\{(m(i,i+k) \times a + m(i+k,j) \times b) / (a+b), m(i,j)\}, \quad if \quad i < j \end{cases} \tag{12}$$

where $m(i\,,\,j)$ is the confident value of the image between cut points i and j. a and b are the number of segmented characters in the image between i and $i+k$, $i+k$ and j, respectively. Figure 14 is an example of explaining the character segmentation procedure. Figure 14 (a) shows the image of a business card. The personal information in the business card is erased to protect personal privacy. Figure 14 (b) is the text-detection result. Each red rectangle in Figure 14 (b) indicates one CC. CCs identified as ligatures are further segmented by the character segmentation process. Take the ligature CC, "Support", as an example (see Figure 14 (c)). In this example, $m(i,j)$ is the confident value ranged from 0 to 4 given by SVM. There are 2 values in each block of the DP table in Figure 14 (d). The upper value is the confident value of the character image between row i and column j, whereas the lower value indicates the selected cut point index in the character image between row i and column j. If the confident value of the whole image between row i and column j is larger than the average confident value of the image divided into 2 parts between (i, $i+k$) and ($i+k$, j), then the cut point index will be set to -1. The final segmentation combination of the CC (0, 1) (1, 3) (3, 4) (4, 5) (5, 6) (6, 8) (8, 9) derived by DP is obtained (see the upper left corner of Figure 14 (c)). As shown in Figure 14 (e), the final segmentation result can then be obtained with each blue rectangle indicating one segmented character.

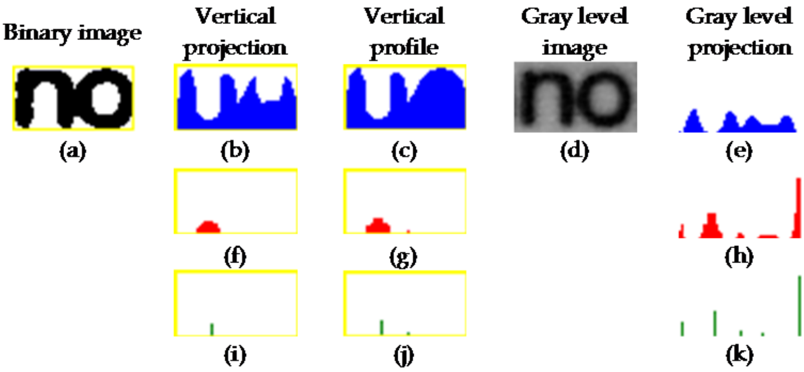
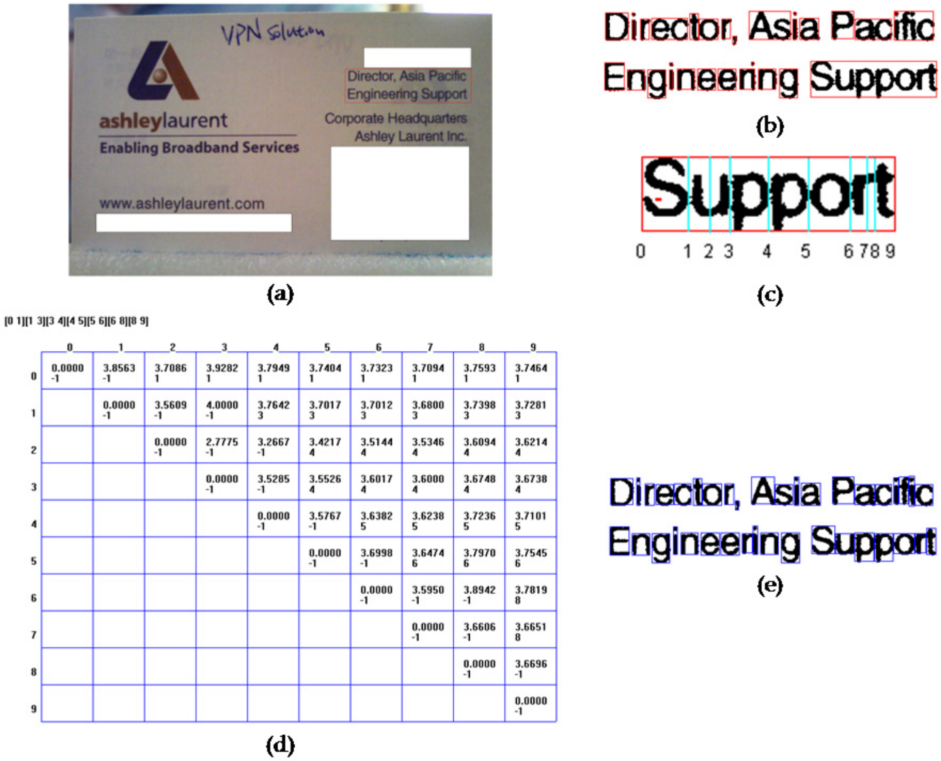


Figure 13. Illustration of cut point candidate searching (a) Binary image, (b) vertical projection, (c) vertical profile, (d) gray level image, (e) gray level projection, (f)-(h) results are obtained by performing Eq. (8) on (b)(c)(e), and (i)-(k) results after cut points selection.

In the character segmentation procedure, it is inevitable to encounter the over-segmentation problem. To remedy this, the procedure verifies the segmentation result by merely using the Typo-class information. For example, character 'm' is usually segmented into two parts, recognized as 'r n' or 'n 7', which is unreasonable because the typo class of 'm' is *Short* and the typo classes of 'n 7' are *Short* and *High*. Table 2 tabulates the designed check table for verification with each element representing one unreasonable situation. If a character is segmented into the specific combination as listed in Table 2, the segmentation of the character is ignored to preserve the original character by not performing the segmentation task on it.



[0 1][1 3][3 4][4 5][5 6][6 8][8 9]

	0	1	2	3	4	5	6	7	8	9
0	0.0000 -1	3.8563 -1	3.7086 1	3.9282 1	3.7949 1	3.7404 1	3.7323 1	3.7094 1	3.7593 1	3.7464 1
1		0.0000 -1	3.5609 -1	4.0000 -1	3.7642 3	3.7017 3	3.7012 3	3.6800 3	3.7398 3	3.7281 3
2			0.0000 -1	2.7775 -1	3.2667 -1	3.4217 4	3.5144 4	3.5346 4	3.6094 4	3.6214 4
3				0.0000 -1	3.5285 -1	3.5526 4	3.6017 4	3.6000 4	3.6748 4	3.6738 4
4					0.0000 -1	3.5767 -1	3.6382 5	3.6238 5	3.7236 5	3.7101 5
5						0.0000 -1	3.6998 -1	3.6474 6	3.7970 6	3.7545 6
6							0.0000 -1	3.5950 -1	3.8942 -1	3.7819 8
7								0.0000 -1	3.6606 -1	3.6651 8
8									0.0000 -1	3.6696 -1
9										0.0000 -1

Figure 14. Example of cut points selection using DP. (a)Origin image, (b)result of text detection and connected component labeling (c)possible cut points in a CC “Support” (d) the corresponding DP table of (c) (e)final character segmentation result.

Text	Segmentation combination	Type	Text	Segmentation combination	Type
M	r,n、n,7、n,1	3	C	C,:	1
N	r,1、r,7	3	c	c,:	3
W	v,v	3	B	I,3	1
W	V,V	1	D	I,3	1
H	I,7、t,1、t,7	1			

Table 2. Check table for verifying the segmentation result.

4. Experiments

In the experiments, text images captured from fifty business cards by a two- million-pixel webcam with resolution 1600×1200 are collected as testing images. Testing images includes the business cards with simple binary backgrounds and complex color backgrounds. There are 9,550 characters and 419 touched characters (1,050 single characters in the touched characters) for a total of 10,600 characters in the testing images. The experiments demonstrate the visual results of reading order confirmation, ligature filter, and character segmentation.

Figure 15 illustrates the experimental result on the process of correcting the reading order. The image is captured in an incorrect reading order (see Figure 15(a)). Text CCs are extracted using binarization and connected component labeling (see Figure 14(b)). Figure 14(c) shows the result of text-line construction. Texts in the left side of Figure 14(d) show the estimated orientation of text-lines. The major angle is the θ_m which is described in section 3.3. The right side of Figure 14(d) is the result using the introduced reading order confirmation algorithm.

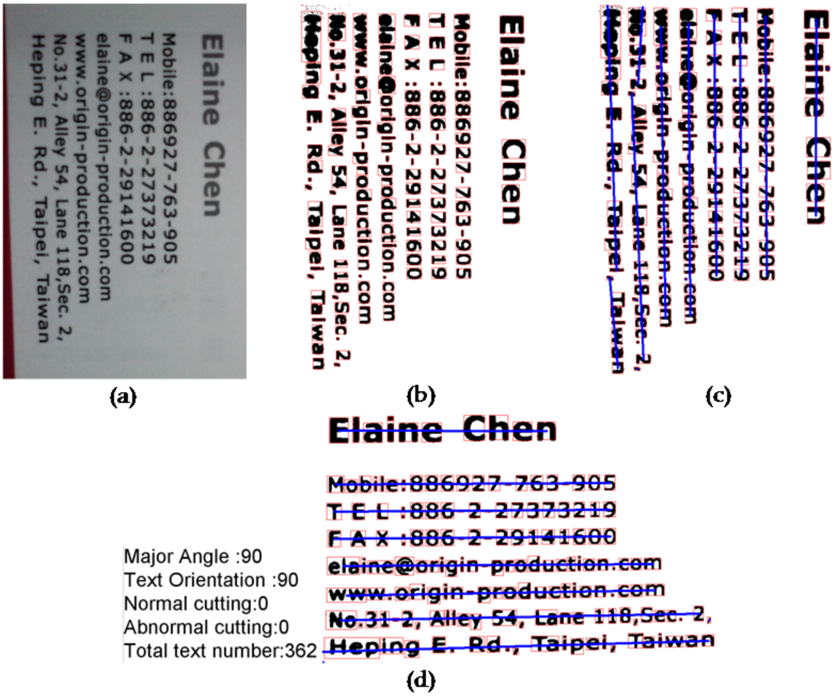
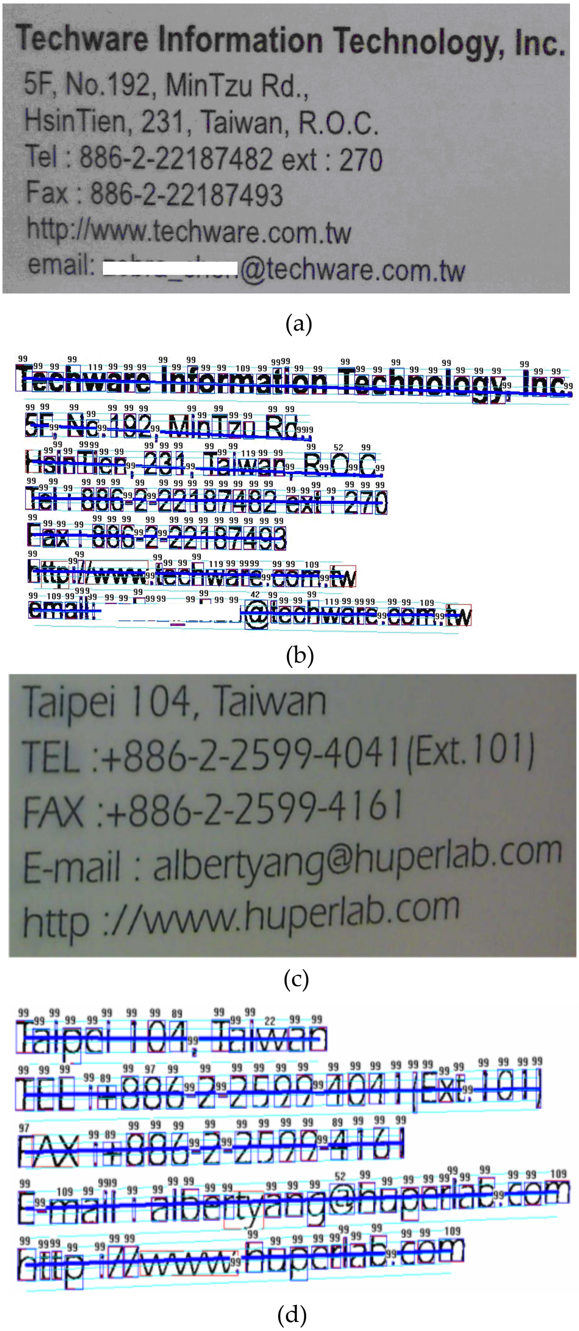


Figure 15. Illustration that reading order confirmation. (a) Source image, (b) performing connected component method in binary image (a), (c) connected component linking, (d) reading order estimation, and (e) image rotation result.

In the second experiment, fifty images of business card are considered as testing images for evaluating the accuracy rate of the ligature filter. The accuracy rate is defined as the number of correct filtered CCs divided by number of total CCs. The average accuracy rate of the proposed ligature filter is 92.14%. Figure 16 shows two examples of the results of ligature filter. CCs with numbers above indicate that they are not ligatures.



Techware Information Technology, Inc.
5F, No.192, MinTzu Rd.,
HsinTien, 231, Taiwan, R.O.C.
Tel : 886-2-22187482 ext : 270
Fax : 886-2-22187493
http://www.techware.com.tw
email: [redacted]@techware.com.tw

(a)

(b)

Taipei 104, Taiwan
TEL :+886-2-2599-4041(Ext.101)
FAX :+886-2-2599-4161
E-mail : albertyang@huperlab.com
http ://www.huperlab.com

(c)

(d)

Figure 16. Results of ligature filter (a),(c) cut source image and (b),(d) the corresponding filtering result.

In the third experiment, same 50 images are used for analyzing the performance of the character segmentation procedure. The accuracy rate of character segmentation is defined as the number of correct segmented ligatures divided by the number of all ligatures. In our experiments, the overall accuracy rate of character segmentation is 98.57%. Figure 17 is a worse case of the character segmentation. The uneven illumination and blur result in severe ligatures after text detection module. It is difficult to find good cut points to segment these ligatures precisely.

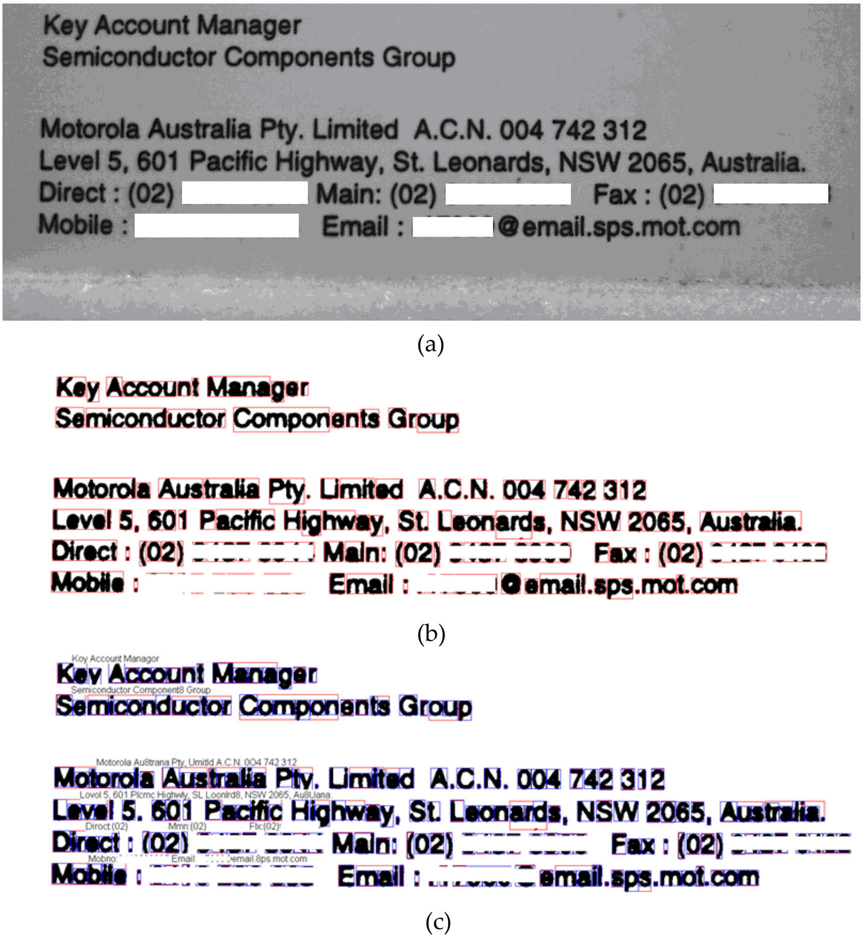

Key Account Manager
Semiconductor Components Group
Motorola Australia Pty. Limited A.C.N. 004 742 312
Level 5, 601 Pacific Highway, St. Leonards, NSW 2065, Australia.
Direct : (02) [redacted] Main: (02) [redacted] Fax : (02) [redacted]
Mobile : [redacted] Email : [redacted]@email.sps.mot.com
(a)
(b)
(c)

Figure 17. Bad result of character segmentation. Uneven illumination and blur appear severely in image. (a) Original image, (b) binarized image, and (c) character segmentation and recognition result.

The character recognition method proposed in [36] is implemented to evaluate the overall performance of the preprocessing system. The recognition rate of characters is 94.90%. Recognizing blurred and ligatures caused by illumination variation and out of focus is challenging. However, the proposed preprocessing system can overcome these difficulties and achieve a high recognition rate.

5. Conclusions

A preprocessing system dedicated to text images captured by cameras is introduced in this chapter. The preprocessing plays a crucial role in dominating the success of later character recognition because text images captured by cameras are usually accompanied with severe uneven illuminations. Three modules in the preprocessing system are introduced in detail: A text detection module, a text-line construction module, and a character segmentation module. Experimental results demonstrate the feasibility and validity of each module of the preprocessing system. The characteristics of the preprocessing system are summarized as follows:

1. *A text-noise filter which filters out non-text CCs efficiently.* A two-stage binarization is used for detecting texts in images and sharpening the contour of CCs. Text and non-text CCs are classified by the devised text-noise filter.
2. *Reading order determination by typographical structures.* When text-lines are constructed, the reading order of the text-lines is still unknown because there are two possible reading orientations of a text-line. A reading order confirmation scheme is proposed by analyzing the typographical structures.
3. *A ligature filter with character segmentation mechanism for improving the efficiency of character segmentation*. The intrinsic features and periphery features are used for classifying ligatures and individual characters. The character segmentation mechanism is only used for ligatures so that the efficiency of the character segmentation module can be improved.

Built upon this work, some works can be accomplished in the future:

1. *Detect texts in the complex background*. The proposed text detection method is appropriate for document images but has defects on complex background. To induce color information of text images and clustering method to the text detection module may be a good try because texts in the same text-line usually have similar colors.
2. *Detect and recognize texts on irregular surface.* The introduced modules are effective for recognizing texts on document images. However, texts often locate on non-plane surface such as a cylinder. It will be helpful to recognizing these texts correctly.
3. *Merge broken characters.* Both broken characters and ligatures cannot be recognized well by OCR. The introduced method solves the ligature problem but do not coping with the broken character problem. A preprocessing system is more complete than that of this work by involving some mechanisms to merge broken characters.
4. *Correct Perspective distortion.* Document images without the margin are hard to correct perspective distortion. Other information needs to be considered for performing affine transformation in a distorted document image.

Author details

Chih-Chang Yu[*]
Department of Computer Science and Information Engineering,
Vanunug University, Zhongli, Taiwan (R.O.C.)

[*] Corresponding Author

Ming-Gang Wen
Department of Computer Science and Information Engineering,
National United University, Miaoli, Taiwan (R.O.C.)

Kuo-Chin Fan and Hsin-Te Lue
Department of Computer Science and Information Engineering,
National Central University, Zhongli, Taiwan (R.O.C.)

Acknowledgement

The authors would like to thank the National Science Council of Taiwan for financially supporting this research under Contract No. 101-2221-E-238-012-.

6. References

[1] Chen X, Yang J, Zhang J & Waibel A (2004) Automatic detection and recognition of signs from natural scenes. IEEE Transaction on Image Processing, vol. 13, (January 2004), pp. 87-99, ISSN 1057-7149

[2] Ezaki N, Bulacu M & Schomaker L (2004) Text detection from natural scene images: towards a system for visually impaired persons. Proceedings of the 17th International Conference on Pattern Recognition, vol. 2, pp. 683-686, ISBN 0-7695-2128-2, Cambridge, UK, August, 2004

[3] Lienhart R & Wernicke A (2002) Localizing and segmenting text in images and videos. IEEE Transaction on Circuits System and Video Technology, vol. 12, (April 2002), pp. 256-268, ISSN 1051-8215

[4] Lyu M R; Song J & Cai M. (2005) A comprehensive method for multilingual video text detection, localization, and extraction. IEEE Transaction on Circuits System and Video Technology, vol. 15, (February 2005), pp. 243-255, ISSN 1051-8215

[5] Wu W, Chen X & Yang J (2005) Detection of text on road signs from video. IEEE Transaction on Intelligent Transportation Systems, vol. 6, (Dec. 2005), pp. 378-390, ISSN 1524-9050

[6] Zhong T, Karu K & Jain A K (1995) Locating text in complex color images," Pattern Recognition, vol. 28, (Oct. 1995), pp. 1523-1535 ISSN 0031-3203

[7] Kim K C, Byun H R, Song Y J, Choi Y W, Chi S Y, Kim K K & Chung Y K (2004) Scene text extraction in natural scene images using hierarchical feature combining and verification. Proceedings of the 17th International Conference on Pattern Recognition, vol. 2, pp. 679-682, ISBN 0-7695-2128-2, Cambridge, UK, August, 2004

[8] Kim, K. I.; Jung, K. & H. Kim (2003). Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm. IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 25, (Dec. 2003), pp. 1631-1639, ISSN 0162-8828

[9] Lim, Y. K.; Choi, S. H. & Lee, S. W. (2000). Text extraction in MPEG compressed video for content-based indexing. Proceedings of the 15th International Conference on Pattern Recognition, pp. 409-412, ISBN 0-7695-0750-6, Barcelona, Spain, September 3-7, 2000

[10] Chun, B. T.; Bae, Y. & Kim, T. Y. (1999). Automatic text extraction in digital videos using FFT and neural network. Proceedings of the IEEE International Conference on Fuzzy Systems, vol. 2, pp. 1112-1115, Seoul, South Korea, August 22-25, 1999, ISBN 0-7803-5406-0

[11] Gllavata, J.; Ewerth, R. & Freisleben, B. (2004). Text detection in images based on unsupervised classification of high-frequency wavelet coefficients. Proceedings of the 17th International Conference on Pattern Recognition, vol. 1, pp. 425-428, ISBN 0-7695-2128-2, Cambridge, UK, August, 2004

[12] Thillou, C.; Ferreira, S. & Gosselin B. (2005). An embedded application for degraded text recognition. EURASIP Journal on Advances in Signal Processing, vol. 2005, pp. 2127-2135, August 2005, ISSN 1687-6180

[13] Hu, S. & Chen, M. (2005). Adaptive Fréchet kernel based support vector machine for text detection. Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 5, pp. 365-368, ISBN 0-7803-8874-7, 18-23 March, 2005

[14] Yamguchi T. & Maruyama, M. (2004). Character extraction from natural scene images by hierarchical classifiers. Proceedings of the 17th International Conference on Pattern Recognition, vol. 2, pp. 687-690, ISBN 0-7695-2128-2, Cambridge, UK, August, 2004

[15] Bargeron, D.; Viola, P. & Simard, P. (2005). Boosting-based transductive learning for text detection. Proceedings of the 8th International Conference on Document Analysis and Recognition, vol. 2, pp. 1166-1177, ISBN 0-7695-2420-6, Seoul, Korea, August 29 – September 1, 2005

[16] Jung, K. (2001). Neural network-based text location in color images. Pattern Recognition Letters, vol. 22, Issue 14, (December 2001), pp. 1503-1515, ISSN: 0167-8655

[17] Jung, K.; Kim, K. I. & Jain, A. K. (2004). Text information extraction in images and video: a survey. Pattern Recognition, vol. 37, (May 2004), pp. 977-997, ISSN 0031-3203

[18] Fan, K. C. & Wang, L. S. (1998). Classification of machine-printed and handwritten texts using character block layout variance. Pattern Recognition, vol. 31, (September 1998), pp. 1275-1284, ISSN 0031-3203

[19] Meunier, J. L. (2005). Optimized XY-cut for determining a page reading order. Proceedings of the 8th International Conference on Document Analysis and Recognition, vol. 1, pp. 347- 351., ISBN 0-7695-2420-6, Seoul, Korea, 29 Aug.-1 Sept. 2005

[20] Gatos, B.; Antonacopoulos, A. & Stamatopoulos, N. (2007). Handwriting segmentation contest," Proceedings of the 9th International Conference on Document Analysis and Recognition, pp. 1284-1288, ISBN 978-0-7695-2822-9, Curitiba, Brazil, September 23-26, 2007

[21] Yin, F. & Liu, C. L. (2009). Handwritten Chinese text line segmentation by clustering with distance metric learning. Pattern Recognition, vol. 42, (Dec. 2009), pp. 3146-3157, ISSN 0031-3203

[22] Simon, A.; Pret, J. C. & Johnson, A. P. (1997). A fast algorithm for bottom-up document layout analysis. IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 19, (March 1997), pp. 273-277, ISSN 0162-8828

[23] Basu, S.; Chaudhuri, C.; Kundu, M.; Nasipuri, M. & Basu, D. K. (2007). Text line extraction from multi-skewed handwritten document. Pattern Recognition, vol. 40, (June 2007), pp. 1825-1839, ISSN 0031-3203

[24] Thillou, C. M.; Ferreira, S.; Demeyer, J.; Minetti, C. & Gosselin, B. (2007). A multifunctional reading assistant for the visually impaired. EURASIP Journal on Image and Video Processing, vol. 3, (November 2007), pp. 1-11, ISSN: 1687-5281

[25] Chen, Y. K. and Wang, J. F. (2000). Segmentation of single- or multiple touching handwritten numeral string using background and foreground analysis. IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 22, (November 2000), pp. 1304-1317, ISSN 0162-8828

[26] Tse, J.; Curtis, D.; Jones, C. & Yfantis, E. (2007). An OCR-independent character segmentation using shortest-path in grayscale document images. Proceedings of the 6th International Conference on Machine Learning and Applications, pp. 142-147, ISBN 0-7695-3069-9, Cincinnati, USA, December 13-15, 2007.

[27] Liu, C. L.; Sako, H. & Fujisawa, H. (2004). Effects of classifier structures and training regimes on integrated segmentation and recognition of handwritten numeral strings. IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 26, (November 2004), pp. 1395-1407, ISSN 0162-8828

[28] Casey R. G. & Lecolinet, E. (1996). A survey of methods and strategies in character segmentation. IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 18, (July 1996), pp. 690-706, ISSN 0162-8828

[29] Vellasques, E.; Oliveira, L. S.; Britto, A. S.; Koerich, A. L. & Sabourin, R. (2008). Filtering segmentation cuts for digit string recognition. Pattern Recognition, vol. 41, (October 2008), pp. 3044-3053, ISSN 0031-3203

[30] Marinai, S.; Gori, M. & Soda, G. (2005). Artificial neural networks for document analysis and recognition. IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 27, (January 2005), pp. 23-35, ISSN 0162-8828

[31] Thillou, C. M.; Mancas, M. & Gosselin, B. (2005). Camera-based degraded character segmentation into individual components. Proceedings of the 8th International Conference on Document Analysis and Recognition, vol. 2, pp. 755-759, ISBN 0-7695-2420-6, Seoul, Korea, 29 August-1 September, 2005.

[32] Otsu, N. (1979). A threshold selection method from gray level histograms. IEEE Transactions on Systems, Man, and Cybernetics, vol. 9, (January 1979), pp. 62-66, ISSN 0018-9472

[33] Kim, K. I.; Jung, K.; Park, S. H. & Kim, H. J. (2002). Support vector machines for texture classification. IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 24, (November 2002), pp. 1542-1550, ISSN 0162-8828

[34] Zramdini, A. & Ingold, R. (1998). Optical font recognition using typographical features. IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 20, (August 1998), pp. 877-882, ISSN 0162-8828

[35] Beker H. & Piper, F. (1983). Cipher systems: The protection of communication, John Wiley & Sons, ISBN 978-0471891925

[36] Chang, F.; Chou, C. H.; Lin, C. C. & Chen, C. J. (2004). A prototype classification method and its application to handwritten character recognition. Proceedings of IEEE International Conference on Systems, Man and Cybernetics, vol. 5, pp. 4738-4743, ISBN: 0-7803-8566-7, The Hague, Netherlands, October 10-13, 2004.