http://dx.doi.org/10.5392/IJoC.2012.8.3.083

Automatic Generation of Training Character Samples for OCR Systems

Ha Le, Soo Hyung Kim, In Seop Na, Yen Do
School of Electronics and Computer Engineering
Chonnam National University, Gwangju, Korea

Sang Cheol Park
Samsung Medison, Seoul, Korea

Sun Hwa Jeong
Electronics and Telecommunications Research Institute

ABSTRACT

In this paper, we propose a novel method that automatically generates real character images to familiarize existing OCR systems with new fonts. At first, we generate synthetic character images using a simple degradation model. The synthetic data is used to train an OCR engine, and the trained OCR is used to recognize and label real character images that are segmented from ideal document images. Since the OCR engine is unable to recognize accurately all real character images, a substring matching method is employed to fix wrongly labeled characters by comparing two strings; one is the string grouped by recognized characters in an ideal document image, and the other is the ordered string of characters which we are considering to train and recognize. Based on our method, we build a system that automatically generates 2350 most common Korean and 117 alphanumeric characters from new fonts. The ideal document images used in the system are postal envelope images with characters printed in ascending order of their codes. The proposed system achieved a labeling accuracy of 99%. Therefore, we believe that our system is effective in facilitating the generation of numerous character samples to enhance the recognition rate of existing OCR systems for fonts that have never been trained.

Keywords*: Character Sample Generation, Optical Character Recognition, Postal Envelope Images, Training Samples, Degradation Model, and Substring Matching.*

1. INTRODUCTION AND PREVIOUS WORK

Optical character recognition (OCR) [1] is the mechanical or electronic translation of scanned images of handwritten, typewritten or printed text into machine-encoded text for the purpose of compact storage, editing, fast retrieval, and other file manipulations through the use of computers. An OCR system enables you to take a book, a magazine article or even specialized documents, such as a check, a receipt, a name card or a postal envelope, and feed it directly into an electronic computer file that can then be edited using a word processor. Because of its wide range of applications, OCR has been a research interest of many researchers working on pictorial pattern recognition.

Over the past years, a lot of studies have been made on the OCR field, which proposed several ways to implement various OCR systems and achieved important results. Sarhan and Al-Zobaidy et al. [2] proposed an OCR system using well-known Neocognitron Artificial Neural Network for its fast processing time and its good performance for pattern recognition problems. They achieved the recognition rate of 95% with 600 Assyrian character sample images for both training and testing stages. H. Guo and J. Zhao et al. [3] proposed a modification of K-Nearest Neighbors for the Chinese minority scripts classification, using wavelet energy distribution and wavelet energy proportions features generated from the discrete wavelet multi-resolution decomposition. The average accuracy of their system is up to 96% with 800 sample images automatically generated by means of random function. Rawat and Kumar et al. [4] described adaptive OCR for Digital Library with human intervention to get feedback for learning. Their system was trained using synthetic data generated from the degradation model of Kanungo [6], and within few iterations of retraining the accuracy of their system was improved to close to 95%. Meshesha and Jawahar et al. [5] used Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) for feature extraction and Decision Directed Acyclic Graph (DDAG) for multi class Support Vector Machine (SVM) classifier. They obtained the accuracy of 96% with the synthetic testing data set and the accuracy of 90% with the real-

* *Corresponding author, Email: ypencil@daum.net*
Manuscript received Jul. 25, 2012; revised Aug 31, 2012; accepted Sep 10, 2012

life testing data set. By combining the entire concept from these papers we know that one of the most important factors that affect the results of OCR systems is training character samples. Advanced OCR systems can recognize a lot of characters from a wide variety of fonts, but they still have difficulty with new fonts, whose characters have never been trained.

New fonts constantly emerge and formulate a myriad of new characters with different typefaces. This, as a consequence, reduces the accuracy of the existing OCR systems. Thus, these existing OCR systems need to be retrained with new character samples, which include labeled character images of new fonts. However, manual generation of these new training character samples is a tedious, time-consuming and costly process, as each single character has to be marked and labeled with the correct character code. Therefore, it is necessary to construct a tool that automatically generates training character samples to avoid the demanding manual tasks.

Several approaches on the generation of training database have been published in literature. One approach is to use synthetic images generated from an electronic document, where the ground truth is available, on which an image degradation model is applied. Different degradation models have been described in literature [9]. However, we can categorize them in two general models. The first models the physics of the apparatus in detail [11], [12]. The completeness of such models can then be justified in part by pointing to the physics. Certainly this can lead to accurate models, but they may be unnecessarily specific and complicated. The second model is more empirical: the simplest model that merely saves the appearances, that is able to generate duplicates of real defective images. Such models cannot be justified by appeals to physics, and must rest on purely statistical measures of completeness [13], [14]. Although, using synthetic data has some advantages over scanning and manual entry, including rapid generation of data at lower cost and continuous control of degradation, synthetic data is generated artificially and may not give as good recognition results as real data. Therefore generating real data is indispensable for training advanced OCR systems. In literature, forced alignment has been used for generating real data for OCR systems. The concept of these methods is to force the transcription of the text line to fit to the image representing the text line using an initially trained recognizer. Zimmermand et al. [15] generated handwriting data using a hidden Markov model recognizer. The path maximizing the probability of finding the word from the transcription gives the optimal cutting points. A similar method is presented by Jaeger et al. [16], and this method was applied to both handwriting and printed text. The disadvantage of this method, apart from being less accurate than manually labeled data, is that both a trained HMM recognizer and the transcription are required. To overcome this problem, Kanungo et al. [7] combined the ease of synthetic data together with real world document image degradation. He used electronic documents to extract the ground truth information (character position, size and code). Furthermore, the document is printed and scanned again. Then the electronic document and the scanned document image are aligned, allowing the computation of the positions of characters in the scanned document image. Since the distortions produced by scanners cannot be described by similarity transformations, a calibration step was introduced to adjust the bounding box of characters. Kim and Kanungo et al. [8] proposed a more robust alignment method, which gives a better result on the UW3 dataset. Recently, Beusekom et al. [10] used Kim and Kanungo's method to estimate the global transformation parameters. Then a local alignment was applied to adapt automatically to scanner distortions. Those methods of generating real data are strictly dependent on the layout of the input documents and the alignment results. The layout information needs to be 100 percent accurate otherwise the systems being evaluated or trained penalized incorrectly [7]. However, the layout of the input documents may vary, so generating layout information is not an easy task.

In this paper, we proposed a novel method that automatically generates real character data without the neccessity of layout information. At first, we generate synthetic character images using a simple degradation model. The synthetic data is used to train an OCR engine, and the trained OCR is used to recognize and label real character images that are segmented from ideal document images. Due to the limitation of using synthetic data in training, the OCR engine is unable to recognize accurately all real character images. Thus, a substring matching method is employed to fix wrongly labeled characters by comparing two strings; one is the string grouped by recognized characters in an ideal document image, the other is the ordered string of characters which we are considering to train and recognize. After substring matching process, recognized characters are classified into three groups, namely correctly recognized group, revised group and wrongly segmented group. The characters of correctly recognized group and revised group are used as training character samples. Based on our method, we build a system that automatically generates 2350 most common Korean and 117 alphanumeric characters from new fonts. The ideal document images used in the system are postal envelope images with characters printed in ascending order of their codes.

The rest of the paper is organized as follows. Section 2 presents a detailed flowchart of the proposed system. Section 3 illustrates an experiment to validate its performance. Discussions and conclusions of this work are given in section 4 and section 5.

2. THE SYSTEM ARCHITECTURE

Manual generation of training character samples is normally done in three steps as shown in Fig. 1. In the first step, a large number of postal envelop images is collected to acquire a complete set of characters with various typefaces, font sizes and font styles. In the second step, these postal envelope images are segmented into character images. Lastly, the most tedious step is to manually label a vast number of character images using visual inspection. Mistakes are inevitable in manual labeling, which consequently affects the accuracy of the recognition results of OCR systems.

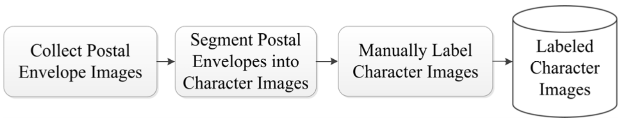

Collect Postal Envelope Images
Segment Postal Envelopes into Character Images
Manually Label Character Images
Labeled Character Images

Fig. 1. Routine Procedure for Manual Generation of Training Character Samples

To overcome the problems associated with manual generation, collecting postal envelop images is replaced with creating ideal postal envelope images. Additionally, an OCR engine is employed to identify unlabeled character images instead of manually labeling them. Here, an ideal postal envelope image is a postal envelope image whose characters are known and printed in ascending order of their codes. In the overview, our proposed system consists of two main stages: synthetic data generation (SDG) and real data generation (RDG) as shown in Fig. 2.

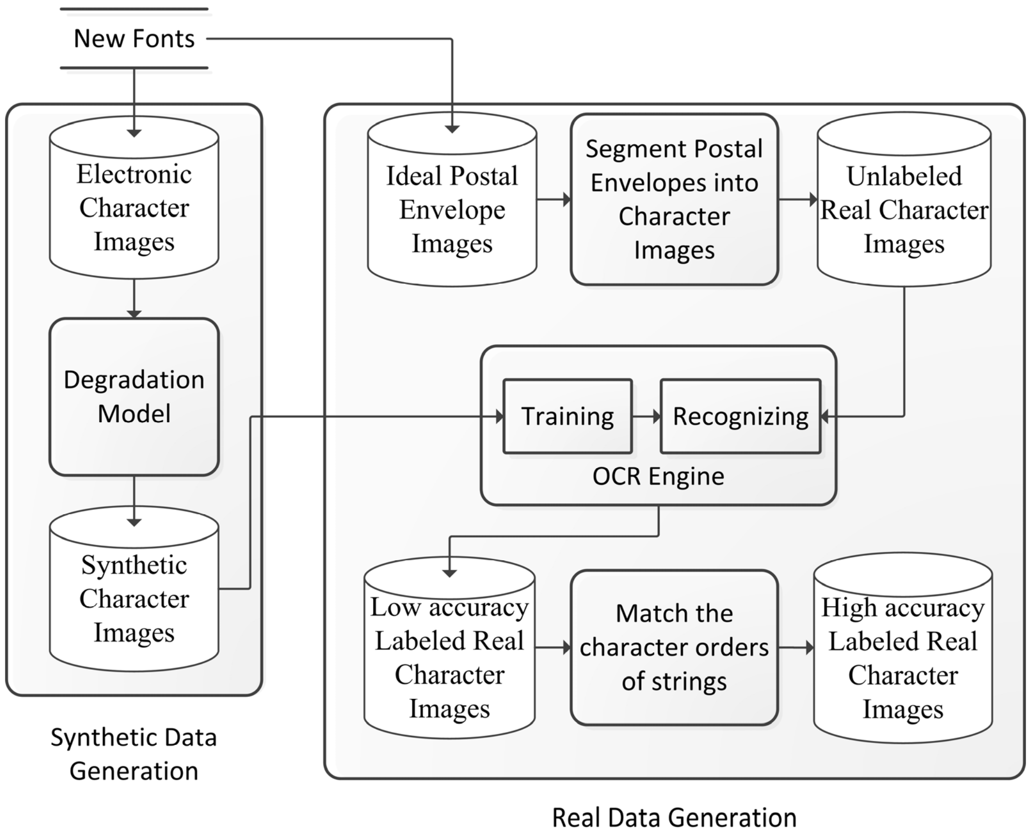

New Fonts
Electronic Character Images
Degradation Model
Synthetic Character Images
Synthetic Data Generation
Ideal Postal Envelope Images
Segment Postal Envelopes into Character Images
Unlabeled Real Character Images
Training
Recognizing
OCR Engine
Low accuracy Labeled Real Character Images
Match the character orders of strings
High accuracy Labeled Real Character Images
Real Data Generation

Fig. 2. Training Database Generation System

2.1 Synthetic Data Generation

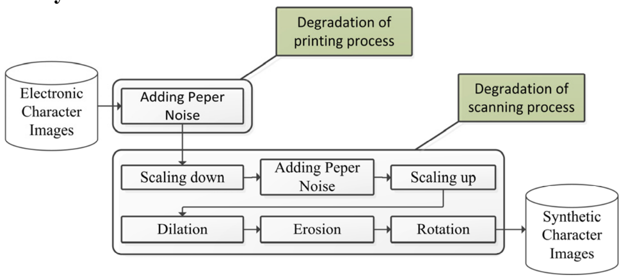

Degradation of printing process
Electronic Character Images
Adding Peper Noise
Degradation of scanning process
Scaling down
Adding Peper Noise
Scaling up
Dilation
Erosion
Rotation
Synthetic Character Images

Fig. 3. Degradation Model

In SDG stage, a database of synthetic character images is created using new fonts. At first, a set of character images with various typefaces, font sizes and font styles is produced using new fonts. As these synthetic character images are directly generated from the specified typefaces, font sizes and styles in the Windows OS, they are definitely clear. In contrast, real character images undergo processes like printing and scanning, which may distort or introduce noises to them. Therefore, to mimic real character images with distortion and noises, we build a degradation model to help synthetic character images resemble real ones. Our degradation model has two steps that mimic the degradation of printing and scanning process as shown in Fig. 3.

2.1.1 Degradation of printing process: We can see that real character images may get noises due to effects of printing such as ink smudges or faint lines running over characters. Thus, pepper noise is added to synthetic character images to resemble the characteristic found in real characters. For example, we used the Korean character “꿨”. This character was generated using font Samsung Gothic with the size of 10 points. This size was modified by our system based on Eq. 1 with DPI = 240, so the actual size of the generated character is 29.5 points as shown in Fig. 4(a). Then, pepper noise is distributed randomly in the synthetic character images as shown in Fig. 4(b). However, noises unassociated with characters are removed as shown in Fig. 4(c). Each real character image has a different quantity of noise, so we add a noise rate option in our system to customize the quantity of noise and diversify the synthetic characters.

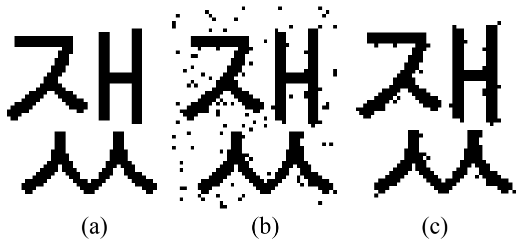


Fig. 4. Degradation of Printing Process, (a) an electronic character image, (b) after adding pepper noise, (c) noises unassociated with character are removed

2.1.2 Degradation of Scanning Process: A real character is affected not only by printing but also scanning. Based on our observation, there are four kinds of degradation on the scanned result: the expansion of the character size caused by the scanning dots per inch (DPI), the noises caused by the specks of dirt on the scanner bed, the spread of the strokes of character caused by the scanner mechanism and the tilt of character when the input documents are slanted in the scanning process by human action.

DPI is a measure of spatial printing or video dot density, in particular the number of individual dots that can be placed in a line within the span of one inch. Sizes of characters on the document image are changed when the document is scanned in different DPIs. The relationship between the size of the hard-copy character and the scanned character is described as,

$$pt' = \frac{0.312 \times DPI \times pt}{25.4} \qquad (1)$$

where *pt* is the size of hard-copy character, *pt'* is the size of character after scanned and *DPI* is the scanning resolution. Thus, the Eq. 1 is used to modify the size parameter in our system, which makes the synthetic character resemble the real scanned character in terms of size.

The noises caused by the specks of dirt on the scanner bed and the low DPI are simulated in three small steps: scaling down the input character image, adding pepper noise to it and finally scaling up the image to its original size. To demonstrate the noises, we scale down the input image with a scale vector *v=(2/3,2/3)* where it gives the most similar effect with scanning process. Then, some pepper noises are added to the image to simulate specks of dirt on the scanner bed. Finally, the input image is scaled up to its original size with a scale vector *v'=1/v*.

Due to the disadvantages of digital scanner, such as reflection, shadow or low contrast, the strokes of character tend to spread out and become bolder. In particular, the characters with bold style are significantly affected by this characteristic. To mimic this characteristic, the morphological dilation operation is performed on the characters with bold style to enlarge their stroke widths. Since the characters with normal style are less affected by this characteristic, instead of dilation operation, we applied morphological closing operation, which associates both dilation and erosion operation, to keep their stroke widths, remove small holes and make the character contours smoother. The structuring element used in our system is the cross element.

Because the input documents may be slanted in the scanning process, the real character images may be askew. For this reason, a rotation operation is employed to imitate this characteristic. Some angles of rotation are applied to diversify the synthetic database.

Fig. 5 shows a synthetic character generated by our degradation model.

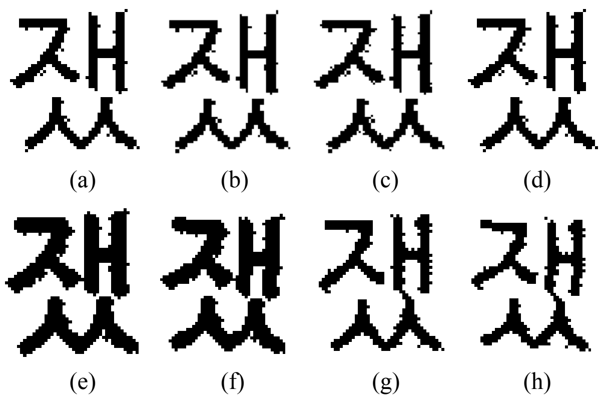
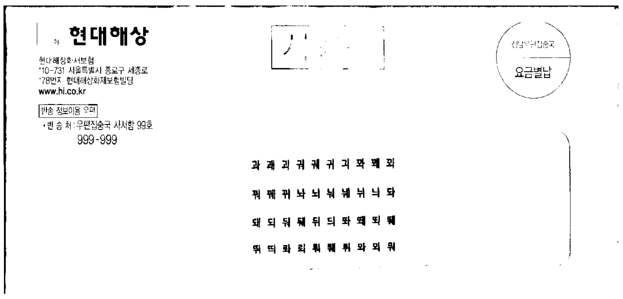


Fig. 5. Degradation of Scanning Process (a) noise image, as described in section 2.1.1, with the noise rate of 5%, (b) scaling down using scale vector *v=(2/3,2/3)*, (c) adding more pepper noises, (d) scaling up using scale vector *v'*, (e) after dilation of character with bold style, (f) after rotation by 1° of (e), (g) after closing of character with normal style and (h) after rotation by 1° of (g)

2.2 Real Data Segmentation

2.2.1 Ideal Postal Envelope Images: Ideal postal envelope images are generated in binary by ETRI (Korea Electronics and Telecommunications Research Institute) using new fonts. Since the focus of this paper is on Korean and English characters, 2467 characters, including 2350 most common Korean characters and 117 alphanumeric characters, are printed in the postal envelope images in ascending order of their code (ideal). As it is not possible to print all characters in one postal envelope, these characters are divided into small groups, each of which will be printed in one postal envelope. Fig. 6 is an example of ideal postal envelope images with characters printed on the right bottom corner of the postal envelope.


현대해상

현다해상호·서보험
110-731 서울특별시 종로구 세종로
178번지 현대해상화재보환빌딩
www.hi.co.kr

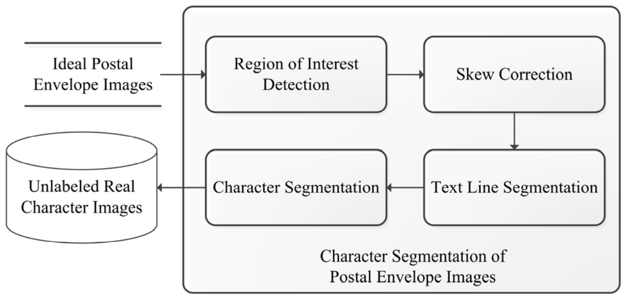
반송 정보이용 우편
• 반 송 처 : 우편집중국 사서함 99호
999-999

서남우편집중국
요금별납

과 괘 괴 궈 궤 귀 긔 꽈 꽤 꾀
꿔 꿰 뀌 놔 뇌 눠 눼 뉘 늬 돠
돼 되 둬 뒈 뒤 듸 똬 뙈 뙤 뛔
뛰 띄 롸 뢰 뤄 뤠 뤼 뫄 뫼 뭐

Fig. 6. An ideal postal envelope image

2.2.2 Character Segmentation of Envelope Images: After generating enough postal envelope images, we apply four image processing steps to extract characters in these images. The image processing steps are shown in Fig. 7.



Ideal Postal Envelope Images
Region of Interest Detection
Skew Correction
Unlabeled Real Character Images
Character Segmentation
Text Line Segmentation
Character Segmentation of Postal Envelope Images

Fig. 7. Character Segmentation in Envelope Images

2.2.2.1 Detection of Region of Interest: The detection of region of interest (ROI) in each envelope image is the foundation of subsequent processing and recognition in the proposed system. In this step, we detect the text region on the right bottom corner of an input postal envelope image, which is considered as the receiver's address, using horizontal and vertical projection profiles. Horizontal projection profile of a binary image is a column vector whose elements indicate the number of black pixels in each row of this image. Vertical projection profile of a binary image is a row vector whose elements indicate the number of black pixels in each column of this image.

The region of interest detection process is conducted as follows:

Step 1: Calculate the horizontal projection profile to y-axis on the input image. The visualization of the horizontal projection values of the input image in Fig. 8(a) is shown in Fig. 8(b). Suppose that H is the height of the input image and the projection value of each row is *projy(i)*, *i=1..H*. We define a horizontal projection threshold T_h as,

$$T_h = \frac{1}{2H}\sum_{i=1}^{H} projy(i) \tag{2}$$

The horizontal projection threshold T_h is used to remove noises and distinguish text lines in the input images. Thus, horizontal projection values are less than or equal to T_h are marked as 0.

$$projy(i) = \begin{cases} 0 & \text{if } projy(i) \le T_h \\ projy(i) & \text{otherwise} \end{cases} \tag{3}$$

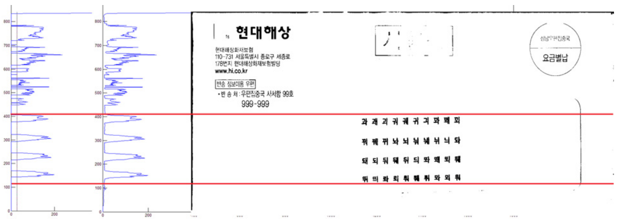


Fig. 8. Horizontal Projection Profile, (a) the input envelope image, (b) horizontal projection profile and (c) the horizontal projection profile after thresholding and the threshold value is the dash line

The visualization of horizontal projection values after thresholding is shown in Fig. 8(c). After thresholding, we remove these text lines whose height is less than a very small value. Based on the constraint that the ROI is located in the right bottom corner of the input envelope, we find the top and bottom coordinates of the text region. The red lines in Fig. 8 present the detected top and bottom coordinates.

Step 2: Extract a text region from the input image using the top and bottom coordinates. The extracted text region is shown in Fig. 9(a).

Step 3: Calculate the vertical projection profile to x-axis on the extracted text region. The visualization of the vertical projection values of the input text region is shown in Fig. 9(b). Suppose that the projection value of each column is *projx(j)*, *j=1..W*. Here, W is the width of the input image. We define a new vertical projection threshold T_v as,

$$T_v = \frac{1}{3W}\sum_{j=1}^{W} projx(j) \tag{4}$$

As similar to T_h, T_v is used to remove noises and distinguish these vertical lines in the extracted text region. Thus, vertical projection values are less than T_v are marked as 0.

$$projx(j) = \begin{cases} 0 & \text{if } projx(j) \le T_v \\ projx(j) & \text{otherwise} \end{cases} \tag{5}$$

The visualization of vertical projection values after thresholding is shown in Fig. 9(c). After thresholding, we can simply remove text lines with small widths and select these vertical text lines close to each other. From these selected vertical text lines, we can find the left and right coordinates of the text region. The red lines in Fig. 9 present the detected left and right coordinates.

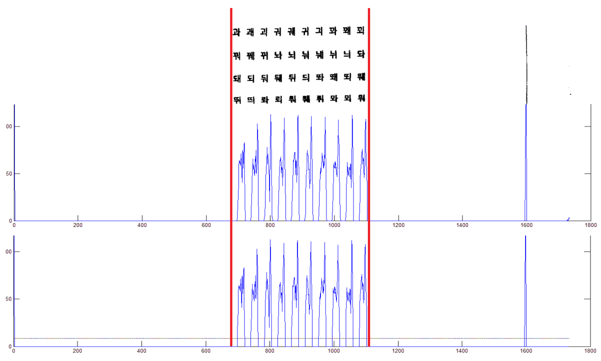
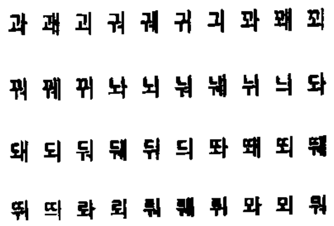


Fig. 9. Vertical Projection Profile, (a) the extracted text region, (b) vertical projection profile and (c) the vertical projection profile after thresholding and the threshold value is the dash line

Step 4: Adjust top, bottom, left, right coordinates of text region to ensure that the text region is completely inside these coordinates. The final detected ROI is shown in Fig. 10.


과 괘 괴 궈 궤 귀 긔 꽈 꽤 꾀
꿔 꿰 뀌 놔 뇌 눠 눼 뉘 늬 돠
돼 되 둬 뒈 뒤 듸 똬 뙈 뙤 뛔
뛰 띄 롸 뢰 뤄 뤠 뤼 뫄 뫼 뭐
Fig. 10. The detected ROI

2.2.2.2 Skew Correction: As seen in the detected ROI, the text region is tilted slightly at an angle. Thus, Hough Transform [17, 18] is applied to detect the skew angle of the text region. The Hough Transform uses the parametric representation of a line,

$$\rho = x\cos\theta + y\sin\theta \tag{6}$$

The variable ρ is the distance from the origin to the line along a vector perpendicular to the line. θ is the angle of the perpendicular projection from the origin to the line measured in degrees clockwise from the positive x-axis, where 0≤θ<180. An illustration of Hough Transform is shown in Fig. 11.

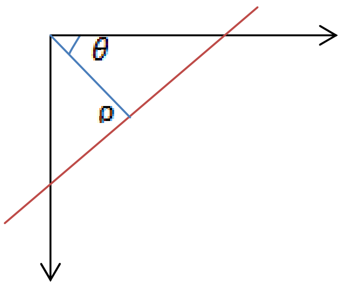
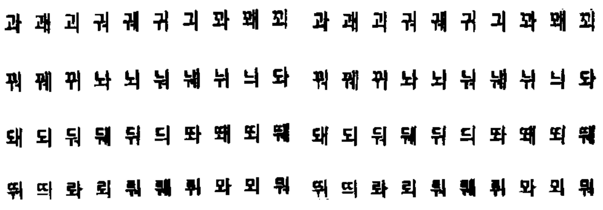
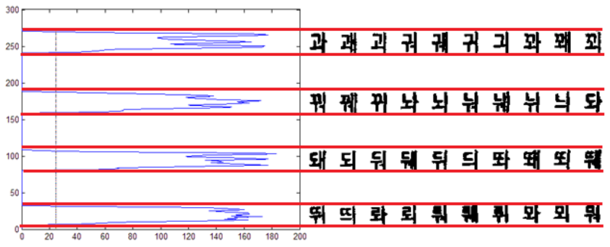


Fig. 11. An illustration of Hough Transform

Since the skew angle of the text lines in the detected ROI is small, a small θ resolution is required for accurate skew angle detection. However, small θ resolution increases the time taken for Hough Transform to process. Thus, in our system, we compute Hough Transform of the text region image twice with different θ resolutions. In the first computation, the range of θ value is (85, 95) and the resolution is 1 degree. Hough Transform checks every angle from 85, 86, …, 95 and chooses the appropriate angle (α) for the input detected ROI. In the second computation, we again apply Hough Transform in the θ range from *α-1* to *α+1*, however, with much smaller theta resolution of 0.2. With the detected skew angle in the second computation, we rotate the text region image to account for skewing. The corrected ROI after skew correction is shown in Fig. 12.


(a) (b)
Fig. 12. Skew Correction
(a) the detected ROI, as shown in Fig. 10, (b) the corrected ROI

2.2.2.3 Text line and character segmentation: After skew correction, to separate the corrected ROI into text lines, we calculate the horizontal projection profile to y-axis on the corrected ROI and find the top and bottom coordinates of each text line. Sample results of text line segmentation are shown in Fig. 13.


(b) (a)
Fig. 13. Text line segmentation
(a) the corrected ROI and
(b) the horizontal projection profile of (a)

With each segmented text line, we calculate the vertical projection profile to x-axis and find the left and right coordinates of each character. After determining four coordinates of one character, we adjust them to ensure that this character fits within the frame established by these coordinates. Finally, we extract these characters and save as unlabeled real character images. Sample results of character segmentation are shown in Fig. 14.

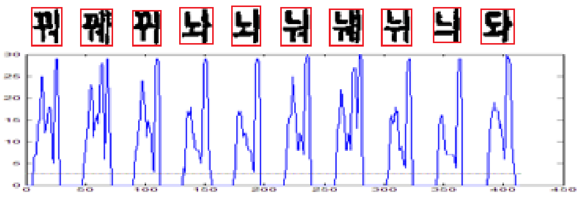


Fig. 14. Character segmentation

2.2.3 Recognition: For character recognition, we use the OCR system developed by ETRI [19]. Character images are divided into 6 by 6 mesh blocks using nonlinear normalization. 10-feature vector is acquired in each block including 8 directional accumulative gradient values and 2 numbers of foreground pixels and background pixels in the block. Then, each character is represented with 360 dimensional features. Before recognizing character images, the OCR system categorizes characters into 7 types; 6 types for Korean characters and 1 type for alphanumeric characters, according to their shape using a multi-layer ANN. Lastly, 7 ANNs are designed for 7 character types, respectively. Here, 7 ANNs consist of 360 nodes for the input layer and 70 nodes for the hidden layer. However, each ANN has a different number of output nodes corresponding to the number of characters in the specified type.

2.2.3 Substring Matching: As the current accuracy of recognizer is not high enough due to the use of synthetic training character samples, some segmented characters may be wrongly recognized. However, we know that these characters are printed in ascending order of their codes. Thus, we present a substring matching method to fix the wrongly recognized characters using Longest Common Subsequence (LCS) [20] and Levenshtein distance [21].

Suppose that 2467 characters printed on postal envelope images are,

$$\begin{cases} C_j,\ j = 1..2467 \\ C_h < C_k \ if\ h < k \end{cases} \qquad (7)$$

We call the string grouped by all 2467 characters C_j the whole string $T=\{C_1, C_2, C_3, \ldots, C_{2467}\}$. In addition, suppose that the recognized characters in an postal envelope image are R_i, $i=1..n$, in which n is the number of recognized characters. With recognized characters R_i, we can form a recognized string $R=\{R_1, R_2, \ldots, R_n\}$. In order to create ideal envelope images, we separate the whole string T into substrings and print them on envelopes. Thus, the recognized string R should be a substring of the whole string T. However, there are probably some segmentation and recognition errors in the previous steps; the recognized character string R is not exactly the same as a substring of the whole string T. Thus, we apply a sophisticated substring matching method to measure the agreement between these two strings. The matching process is conducted in three steps as follows.

Step 1: Apply LCS algorithm to find the LCS between the recognized character string R and the whole string T.

Step 2: Suppose that the positions of the first and the last character determined by LCS are u_1 and u_2 in recognized character string R and v_1 and v_2 in the whole string T. Then, we define the corresponding substring of the recognized character string R is the string extracted from v_1-u_1+1 to v_2-u_2+n of the whole string W. We call the corresponding substring is S, so that $S=\{C_{v1-u1+1}, C_{v1-u1+2}, \ldots, C_{v2-u2+n}\}$.

Step 3: Use Levenshtein distance to measure the agreement between the recognized character string R and the corresponding substring S. The Levenshtein distance between two strings is defined as the minimum number of edits required to transform one string into another, with the allowable edit operations being insertion, deletion, or substitution of a single character. When computing Levenshtein distance to change the recognized character string to the corresponding substring, recognized characters are classified into three groups, namely correctly recognized group, revised group and wrongly segmented group.

Fig. 15 and Fig. 16 show sample results of the proposed character labeling method. The difference between Fig. 15 and Fig. 16 is that there are some wrongly segmented characters marked as blue color in the third line as shown in Fig. 16(b). The codes of these blue characters are suggested by our substring matching method.

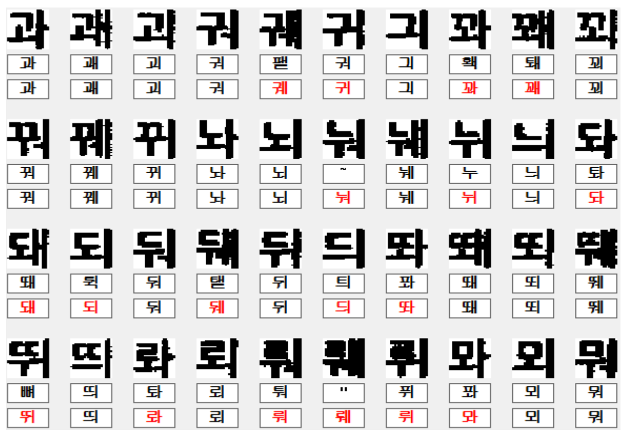
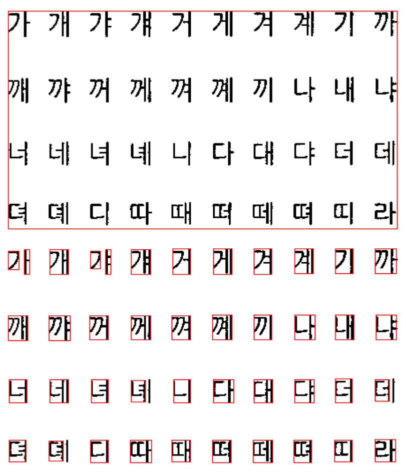


Fig. 15. A sample result of the proposed character labeling method, the first line is the segmented characters, the second line is the recognized character using the OCR system trained by synthetic data and the third line is the labeling result after applying substring matching method. In the third line, the black characters are correctly recognized by the OCR system, and the red characters are revised using Levenshtein distance as presented in step 3 of the substring matching method.


(a)

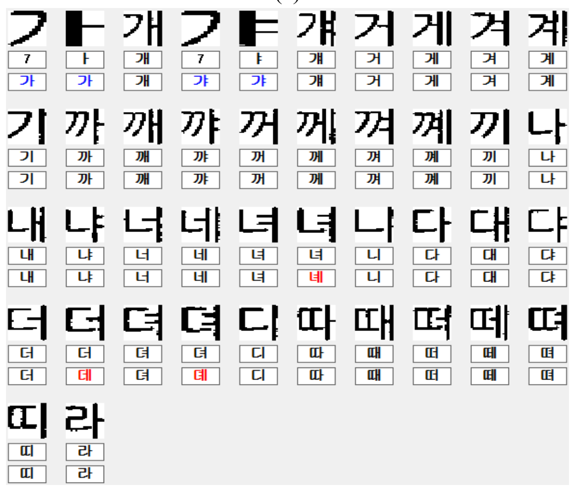

(b)

Fig. 16. A sample result of the character labeling method, (a) the detected ROI and the segmented result and (b) the labeling result

3. EXPERIMENTAL RESULTS

To validate the effectiveness of our system, several experiments are conducted to assess the segmentation and labeling accuracy.

3.1 Data

In our experiments, we test with 9 Korean fonts, and each font is used to generate 144 synthetic data sets for training and 36 real data sets for testing. One data set consists of 2467 characters. Synthetic data sets are generated by changing the following options: DPI (200 and 240), font styles (Bold, Regular), font sizes (9pt, 10pt and 13pt), noise rates (1%, 2%, 5% and 7%) and rotation angles (-1◦, 0◦ and 1◦). Real data sets are created by changing font style and font size options. However, each real data set is separated into 51 small groups with different number of characters. Each group is printed into 2 different kinds of postal envelopes, one is the regular envelope without plastic window and the other is the envelope with plastic window. In addition, each postal envelope is scanned 3 times in 200 DPI to enlarge the real character database. Thus, theoretically, in each font, we have 355,248 synthetic characters and 88,812 real characters in 1,800 postal envelope images. However, due to the manual scanning, some images are scanned more than 3 times while some images may not be placed in the scanner. Therefore, we practically got a different number of real character images and postal envelope images for each font. Table 1 and 2 show the options for generating synthetic and real data sets, and table 3 shows the number of generated synthetic characters and the number of envelope images for each font.

Table 1. Options for generating synthetic data sets

Synthetic Data	Options				
	DPI	Font Styles	Font Sizes (pt)	Noise Rates (%)	Rotation Angles (^{o})
	200, 240	Bold, Regular	9, 10, 13	1, 2, 5, 7	-1, 0, 1

Table 2. Options for generating real data sets

Real Data	Options			
	Number of scanning	Font Styles	Font Sizes (pt)	Kinds of Envelopes
	3	Bold, Regular	9, 10, 13	Regular Envelope and Envelope with Plastic Window

Table 3. Experimental data

Fonts	Characters		Envelope Images
	Synthetic	Real	
Nanum Gothic	355,248	86,735	1,811
UnDinaru	355,248	86,496	1,808
Malgun Gothic	355,248	87,815	1,840
Samsung Gothic	355,248	87,783	1,837
Seoul Namsan	355,248	86,741	1,813
Seoul Hangang	355,248	87,109	1,819
Sun Mion	355,248	85,695	1,796
Un Gothic	355,248	86,501	1,810
The Gothic	355,248	86,920	1,830

3.1 Results

Our proposed system automatically segmented all characters from each postal envelop image and then classified the characters into three groups, correctly recognized group using the OCR system, revised group using substring matching and wrongly segmented group. To verify the accuracy, we manually checked if each character in these groups was correctly segmented and recognized.

Because of segmentation errors, the summation of the number of incorrectly and correctly segmented characters may be different from the total number of characters. In our observation, there are three common segmentation errors: some characters are not detected, some characters are separated into two or three parts, and some noises are considered as characters. The segmentation accuracy is computed by the proportion of the number of correctly segmented characters to the summation of the number of incorrectly and correctly segmented characters. Our proposed system achieved the maximum accuracy of 99.5 and the minimum accuracy of 99.26% for Seoul Hangang and Sun Mion, respectively. Finally, the proposed system achieved the average segmentation accuracy of 99.66% as shown in the Table 4.

Table 4. Results of Character Segmentation

Fonts	No. Char	Incorrect	Correct	Accuracy (%)
Nanum Gothic	86,735	129	86,673	99.85
Un Dinaru	86,496	159	86,328	99.82
Malgun Gothic	87,815	132	87,598	99.85
Samsung Gothic	87,783	181	87,230	99.79
Seoul Namsan	86,741	519	85,773	99.40
Seoul Hangang	87,109	106	85,880	99.88
Sun Mion	85,695	628	84,774	99.26
Un Gothic	86,501	312	86,143	99.64
The Gothic	86,920	479	86,702	99.45
AVERAGE	86,866	294	86,345	99.66

The recognition accuracy of the OCR system trained with synthetic characters generated by the proposed degradation model is computed by the proportion of the number of correctly recognized characters to the total number of characters. The accuracy of the substring matching is computed by the proportion of the correctly labeled characters to the total number of characters. The performance of the OCR system and the substring matching is shown in Table 5 and Fig. 17. Average recognition of the OCR system trained with synthetic training samples was 68% while the average labeling accuracy was increased to 99% after applying substring matching.

Table 5. Performance of the OCR system and the substring matching

Fonts	No. Char	OCR system trained with synthetic data	Substring matching

		Correct	Accuracy	Correct	Accuracy
Nanum Gothic	86,735	68,208	78.64	86,644	99.90
Un Dinaru	86,496	61,415	71.00	86,120	99.57
Malgun Gothic	87,815	58,775	66.93	87,233	99.34
Samsung Gothic	87,783	72,625	82.73	87,192	99.33
Seoul Namsan	86,741	66,347	76.49	85,436	98.50
Seoul Hangang	87,109	70,034	80.40	85,840	98.54
Sun Mion	85,695	39,689	46.31	83,950	97.96
Un Gothic	86,501	47,198	54.56	85,354	98.67
The Gothic	86,920	49,247	56.66	85,760	98.67
AVERAGE	86,866	59,282	68.00	85,948	99.00

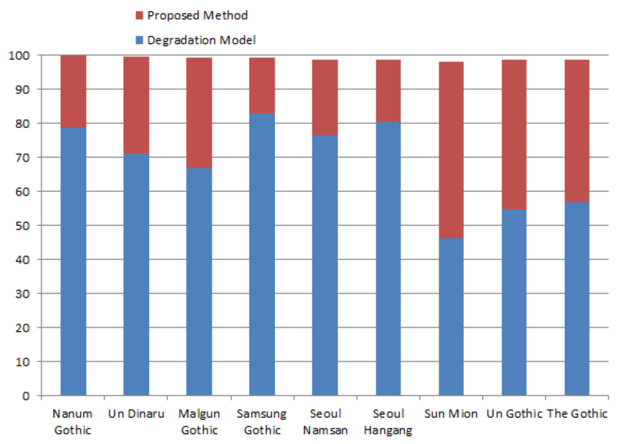


Fig. 17. The performance of proposed method, the accuracy of the OCR system trained with synthetic data generated by the proposed degradation model is shown in blue bar, and the accuracy of the labeling increases after applying substring matching is shown in red bar.

4. DISCUSSION

Our proposed method achieves a high accuracy of 99%, but there are two factors which can affect this accuracy: the accuracy of the segmentation result, and the performance of the degradation model. If the performance of the degradation model is too low, the substring matching method will fail to find the appropriate corresponding substring in step 2 of substring matching method. However, the performance of the degradation model is sufficient to avoid this problem. Even when the performance of the degradation falls to 46.31% in the case of the Sun Mion font, our proposed method still provide a high labeling accuracy of 97.96%. More critical than the performance of the degradation model is the accuracy of the segmentation result. One incorrect segmented character not only affects the labeling result of itself but also the labeling result of the characters around it. To increase the accuracy of the segmentation result, we can put some constraints in the ideal postal envelope images so that we can easily segment characters on each postal envelope images. However, we want to keep the ideal postal envelope images as closely similar to the real postal envelopes as possible. Thus, in our system no constraint is used. The accuracy of the segmentation result in our system is 99.66%, which is sufficient for a high labeling accuracy.

5. CONCLUSIONS

In this paper, we propose a new system for automatic generation of a training character samples for OCR systems. This system is based on our novel method using a degradation model, an OCR engine and a substring matching method. Our method does not need to use the layout information of the ideal documents, which may ruin the labeling result if the layout information is not 100 percent correct. Besides that, the proposed system yields promising labeling results. Therefore, we believe that the proposed system is able to reduce the need for manpower in generating huge training samples. The system can also be applied to several other applications to improve the recognition performance of existing OCR systems.

ACKNOWLEDGMENT

This research was supported by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2012-H0301-12-3005).

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2012-0002883)

REFERENCES

[1] Nagy, G., Twenty years of document image analysis in PAMI, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.22, no.1, pp.38-62, 2000.

[2] N. S. Sarhan and L. Al-Zobaidy, Recognition of Printed Assyrian Character Based on Neocognitron Artificial Neural Network, *The International Arab Journal of Information Technology*, vol4, no.1, 2007.

[3] H. Guo and J. Zhao, A Chinese Minority Script Recognition Method Based on Wavelet Feature and Modified KNN, *Journal of Software*, vol.5, no.2, 2010.

[4] Sachin Rawat, A Semi-automatic Adaptive OCR for Digital Libraries, *Centre for Visual Information Technology*, 2006.

[5] M. Meshesha and C. V. Jawahar, Optical Character Recognition of Amharic Documents, *Center for Visual Information Technology*, 2007.

[6] Tapas Kanungo, Robert M. Haralick, Henry S. Baird, Werner Stuezle and David Madigan, A Statistical, Nonparametric Methodology for Document Degradation Model Validation, *IEEE Transaction on Pattern Analysis and Machine Intelligence 22*, 2000.

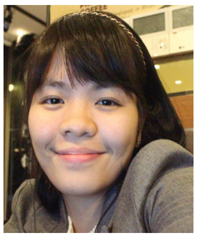
[7] T. Kanungo and R. M. Haralick, An automatic closedloop methodology for generating character groundtruth for scanned documents, *IEEE Trans. Pattern Anal. Mach. Intell.,* pp.179–183, 1999.
[8] D.-W. Kim and T. Kanungo, Attributed point matching for automatic ground truth generation, *Int. Journal on Document Analysis and Recognition*, pp.47–66, 2002.
[9] H. S. Baird, The state of the art of document image degradation modeling, *IAPR Workshop on Document Analysis Systems*, 2000.
[10] J. van Beusekom, F. Shafait, and T. M. Breuel, Automated OCR Ground Truth Generation, *In 8th IAPR Workshop on Document Analysis Systems*, pp.111–117, 2008.
[11] H. S. Baird, Document image defect models. *In Document image analysis*, pp.315-325, 1995.
[12] H. S. Baird, Calibration of Document Image Defect Models, *2nd UNLV Symp. on Document Analysis & Information Retrieval*, pp.26-28, 1993.
[13] T. Kanungo, Global and Local Document Degradation Models, *Document Analysis and Recognition*, pp.730-734, 1993.
[14] T. Kanungo, Document Degradation Models and Methodology for Degradation Model Validation, *Ph.D. Dissertation*, 1996.
[15] M. Zimmermann and H. Bunke. Automatic segmentation of the IAM off-line database or handwritten english text. *Proc Int. Conf. on Pattern Recognition*, 2002.
[16] S. Jaeger, S. Manke, J. Reichert, and A. Waibel. Online handwriting recognition: the npen++ recognizer. *Int. Journal on Document Analysis and Recognition*, pp.1433–2833, 2001.
[17] P. V. C. Hough, Method and means for recognizing complex patterns, *U.S. Patent 3069654*, 1962.
[18] R. O. Duda and P. E. Hart, Use of The Hough Transform to Detect Lines and Curves in Pictures, *Commun. ACM*, vol.15, no.1, pp.11–15, 1972.
[19] Seung Ick Jang and Youn Seok Nam, A Method of Machine-Printed Hangul Recognition using Grapheme Recognizer, *Proc. of Korea Information Processing Society Spring Conference*, vol.11, no.1, pp.351 - 354, 2004.
[20] L. Bergroth, A Survey of Longest Common Subsequence Algorithms, *Seventh International Symposium on String Processing and Information Retrieval*, 2000.
[21] V. I. Levenshtein, Binary codes capable of correcting deletions, insertions, and reversals, *Cybemetics and Control Theory*, vol.10, no.8, pp.707-710, 1966.

Ha Le
He received the B.S in Computer Science from Hanoi University of Science and Technology, Vietnam in 2010. Since 2011, he has been a master student in the Department of Computer Science, Chonnam National University, Korea. His main research interests include pattern recognition, image processing, text recognition, object segmentation and object tracking.

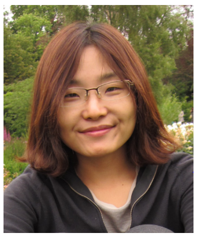
Soo Hyung Kim
He received his B.S. degree in Computer Engineering from Seoul National University in 1986, and his M.S. and Ph.D degrees in Computer Science from Korea Advanced Institute of Science and Technology in 1988 and 1993, respectively. From 1990 to 1996, he was a senior member of research staff in Multimedia Research Center of Samsung Electronics Co., Korea. Since 1997, he has been a professor in the Department of Computer Science, Chonnam National University, Korea. His research interests are pattern recognition, document image processing, medical image processing, and ubiquitous computing.

In Seop Na
He received his B.S., M.S. and Ph.D. degree in Computer Science from Chonnam National University, Korea in 1997, 1999 and 2008, respectively. Since 2012, he has been a research professor in Department of Computer Science, Chonnam National University, Korea. His research interests are image processing, pattern recognition, character recognition and digital library.

Sang Cheol Park
He received his B.S. and M.S degree in Computer Science from Chosun University, Korea in 1999 and 2001, respectively and his Ph.D. degree in Computer Science from Chonnam National University, Korea in 2006. From 2006 to 2010, he was a member of research staff in Medical Imaging Center, Department of Radiology, University of Pittsburgh. From 2010, he has been a research professor in the Department of Computer Science, Chonnam National University, Korea. His research interests are medical image processing, pattern recognition, content-based image retrieval, image matching and 3D image reconstruction.



Yen Do
She received her B.S. degree in Information Technology from Hanoi University of Science and Technology, Vietnam in 2009. Since 2011, she has been a master process student in the Department of Computer Science, Chonnam National University, Korea. Her main research interests include pattern recognition, document image processing and ubiquitous computing.



Seon Hwa Jeong
She received her B.S. and M.S. degrees in Statistics from Chonnam National University in 1996 and 1998 respectively and her Ph.D degree in Computer Science of the same university in 2001. Since 2001, she has been a senior researcher in Electronics and Telecommunications Research Institute, Korea. Her research interests are pattern recognition, postal image processing, postal automation, address and postcode system.