JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 5, No. 3, September 2011, pp. 161-166

A Fast Algorithm for Korean Text Extraction and Segmentation from Subway Signboard Images Utilizing Smartphone Sensors

Igor Milevskiy and Jin-Young Ha*
Department of Computer Science and Engineering, Kangwon National University, Chucheon, Korea
io.milewski@kangwon.ac.kr, jyha@kangwon.ac.kr

Abstract
We present a fast algorithm for Korean text extraction and segmentation from subway signboards using smart phone sensors in order to minimize computational time and memory usage. The algorithm can be used as preprocessing steps for optical character recognition (OCR): binarization, text location, and segmentation. An image of a signboard captured by smart phone camera while holding smart phone by an arbitrary angle is rotated by the detected angle, as if the image was taken by holding a smart phone horizontally. Binarization is only performed once on the subset of connected components instead of the whole image area, resulting in a large reduction in computational time. Text location is guided by user's marker-line placed over the region of interest in binarized image via smart phone touch screen. Then, text segmentation utilizes the data of connected components received in the binarization step, and cuts the string into individual images for designated characters. The resulting data could be used as OCR input, hence solving the most difficult part of OCR on text area included in natural scene images. The experimental results showed that the binarization algorithm of our method is 3.5 and 3.7 times faster than Niblack and Sauvola adaptive-thresholding algorithms, respectively. In addition, our method achieved better quality than other methods.

Category: Smart and intelligent computing; Human computing

Keywords: Optical character recognition; Text segmentation; Image binarization; Smart phone

I. INTRODUCTION

Korean text extraction from signboard images is a sophisticated task. Most signboards are made of or covered with plastic, and even text on the signboards is designed to have high contrast with the background, so images captured by smart phone in many cases have glare near the text region [1, 2]. Non-uniform illumination makes global thresholding methods [3] less effective. On the other hand, adaptive thresholding methods are showing much better results; however, these methods are relatively slow from a computational point of view [4].

Unlike Latin characters, Korean characters (Hangul) consist of disjoined letters. Each character consists of two or three letters. This number may be increased up to six, if we count the individual component separately: e.g., "뽥" consists of "ㅂ", "ㅂ", "ㅗ", "ㅏ", "ㄹ", and "ㄱ". Letters inside one character could touch other letters or letters from other characters. Furthermore, text on images of natural scenes taken by smart phone cameras has uneven direction and size.

Recent models of smart phones are equipped with a wide range of sensors including camera, accelerometer, and capacitive touch screen. However, these devices are still limited by processing and memory resources. It is important for an application running on a mobile device to make efficient use of both computing time and memory. Data received from sensors could be used to avoid resource-demanding algorithms in order to reduce computing time and memory usage, thus increasing the user satisfaction.

We used two types of sensors: accelerometer and touch screen. The accelerometer provides information about the smart

Open Access http://dx.doi.org/10.5626/JCSE.2011.5.3.161 http://jcse.kiise.org

Received 25 March 2011, **Accepted** 1 June 2011
*Corresponding Author

phone orientation in space. The angle between axis of smart phone and the directional pull of gravity is used for image rotation. Thus, an image captured by a user while holding a camera at an arbitrary angle is treated as if it is taken by holding the camera ideally horizontally. Next, the Canny edge filter and connected component is applied for character candidate retrieval [5]. The result is subjected to a connected component analysis – pairs of points that define blobs are used during the following text location and segmentation. To determine the text location, the user defines a region of interest by performing a flicking gesture on the touch screen. After the string of characters is reconstructed using connected component information, affine transformation is applied to the region of the string and connected component information, which are used on the final step – text segmentation.

The most difficult part in optical character recognition (OCR) for natural scene images is text extraction including text localizing and text segmentation. The resulting data of our method could be used as OCR input, hence solving the most difficult part of OCR of text area included in natural scene images, with less computation resources.

II. RELATED WORKS

Recently, a number of systems for text extraction from natural scene images have been developed. Image binarization and text localizing are essential in such systems. Popular algorithms for image binarization include k-mean clustering and thresholding. Lee et al. [6, 7] and Jung et al. [8] described method of clustering in HVC (hue, value, chroma) color space; Lai et al. [9] used RBG (red, green, blue) color space. A number of clusters is defined according to the number of the most common colors and is roughly 20. The most popular thresholding algorithms include Otsu [3], Niblack and Sauvola thresholding [4]. The first one is the global thresholding algorithm, which can hardly be used for natural scene images. The last two methods are local adaptive algorithms, which show better quality for image thresholding, and will be described further in Section III.

Text localization algorithms are based on profile analysis. Bae et al. [10] dealt with relatively simple text images, for which text localization could be done using only projection profiles. Park et al. [1] required a text region to be located at the center of outdoor signboard images.

Fragoso et al. [11] described a method that deals with perspectively-distorted text location. Another approach is co-linearity weight calculation [7]; however, these methods handle only joined characters such as Latin and not disjoined ones such as Korean. For instance the string "인천" could be detected as two lines "이처" and "ㄴㄴ".

Previously described methods, that deal with direction ambiguity and disjoined letter-location, are based on Hough transformation [12], which requires intensive computing time and high memory usage, which are not suitable for mobile devices.

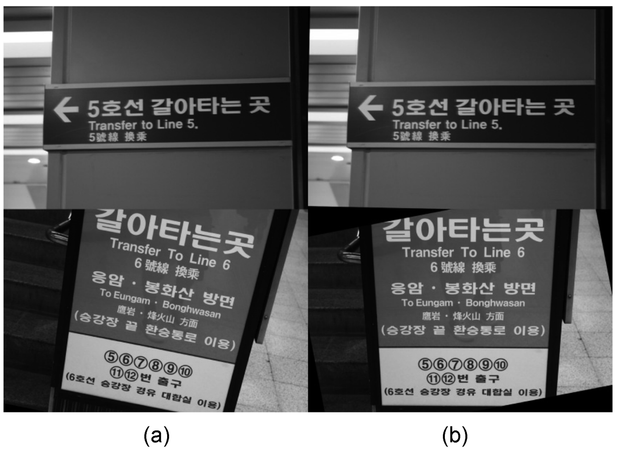

Fig. 1. Accelerometer based image rotation: (a) original and (b) rotated images.

III. PROPOSED METHOD

A. Preprocessing with Accelerometer Employment

To reduce computational time, a captured image is re-sized to 800 × 600 pixels and converted from color to gray scale image.

The goal of preprocessing is to resolve ambiguous text direction. While signboard text is located horizontally, the users can capture images at any arbitrary angle. The accelerometer provides information about the smart phone camera orientation in space. The angle between the axis of smart phone and the direction of gravity is calculated as shown in Eq. (1),

$$angle = \operatorname{atan2}(x, z) \qquad (1)$$

where x and z are the first and third accelerometer values, respectively. The image, rotated by the same angle, has text strings oriented by the horizon. As a result, any image captured by the user is treated as if the camera was held horizontally at the time the image was captured. Examples of color image rotation using accelerometer data are demonstrated in Fig. 1.

B. Image Binarization

Image binarization workflow is presented in Fig. 2.

First, the Canny edge filter is applied to the gray scale image, resulting in an edge map. On the map, every closed loop contour is regarded as a connected component (blob). Each blob could be a character or a part of a character. Connected compo-

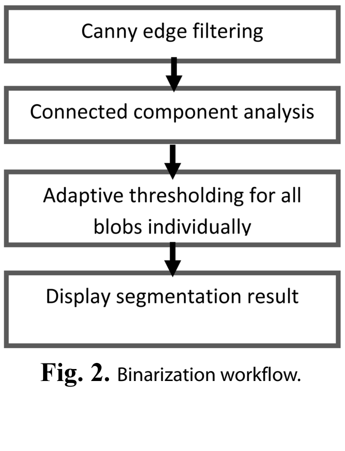
Canny edge filtering
Connected component analysis
Adaptive thresholding for all blobs individually
Display segmentation result

Fig. 2. Binarization workflow.

nent analysis returns a list of boxes that bounds blobs. We use the algorithm from openCV extension library – cvBlobsLib 6. After this step, the blobs that are too small or too big are excluded. Next, the gray-scale image is thresholded on the area defined by a single blob by sliding the window over. The window size depends on the blob size. The threshold level is exclusively calculated for every pixel and defined by Eq. (2),

$$T = m \tag{2}$$

where m is mean gray scale level inside a window around a pixel.

Previous methods [4] described the adaptive Niblack that defines the threshold level by Eq. 3 and Sauvola (Eq. 4) thresholding methods.

$$T = m + k * s \tag{3}$$

$$T = m * [1 - k\,(1 - s/R)] \tag{4}$$

where m is the mean, s is the standard deviation of the gray scale level inside a window around a pixel, R is usually equal to 128, k is a predefined coefficient and is selected according to the text color. If the text color has a darker background, then let k be a negative value in order to get proper foreground text, otherwise it is positive.

The disadvantages of these methods include that they are required to perform thresholding twice, and it is not clear beforehand what coefficient k gives the best results.

The result of thresholding is copied to a blank image and shown to the user for text location (Fig. 3b, 3f). Fig. 3 shows the original images and the results of different binarization methods. The turquoise area of the result of the proposed method was not processed, resulting in much computational time reduction.

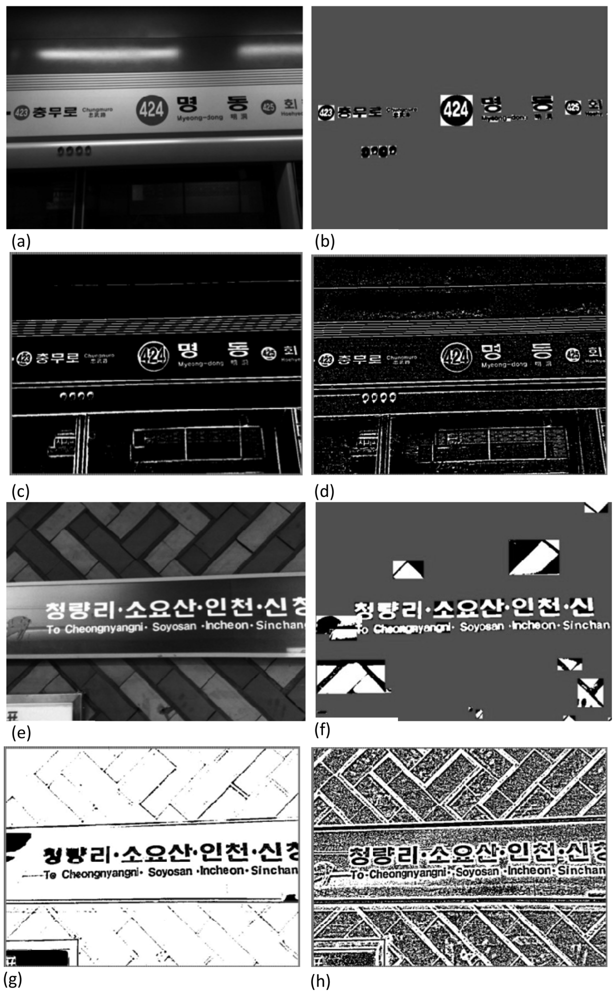

Fig. 3. Original images: (a, e) result of binarization, (b, f) proposed method, (c, g) Niblack, (d, h) Sauvola adaptive thresholding.

C. User Guided Text Location

After the user sees the result of binarization, he can select the area of interest by performing a flicking gesture on the smart phone screen as shown in Fig. 4.

In case the user-defined line is not horizontal, knowing the start and end point of that line horizontal median line is calculated (Fig. 5).

Then, we perform two runs on the set of blobs. Blob information is extracted from the binarization step in Section III.B. First, all blobs that hit the median line are selected (Fig. 6).

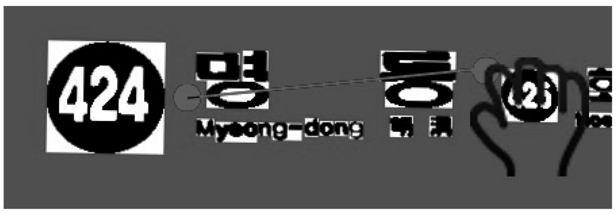

Fig. 4. User defined area of interest. User places marker line over screen area.

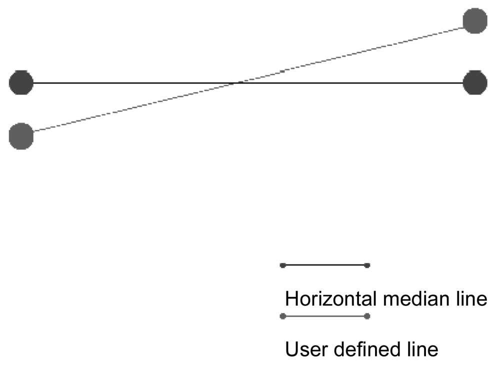

Fig. 5. Horizontal median line and user define line.

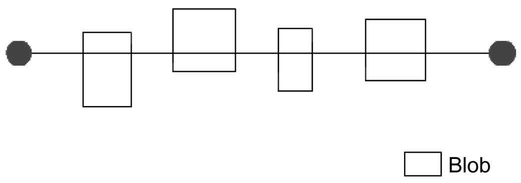

Fig. 6. Text location. First run. Blobs that hit the line are selected.

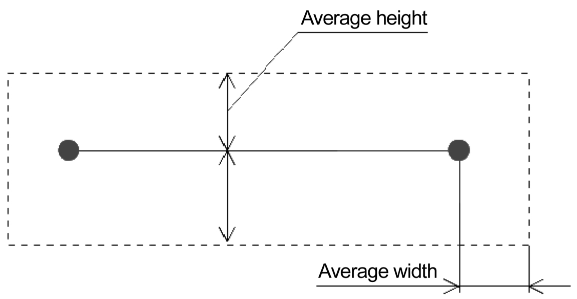

Fig. 7. Text location. Area for the second run.

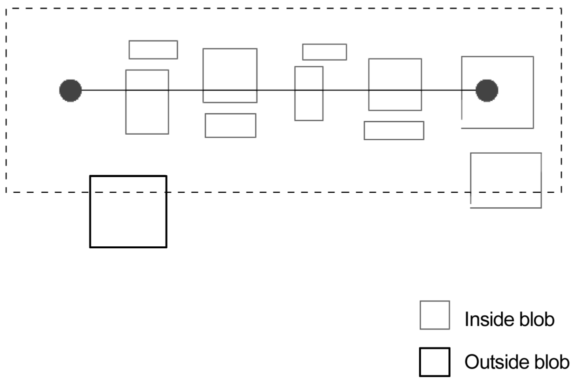

Fig. 8. Inside and outside blobs.

During this step, the average blob height and width is calculated.

On the second run we, select all blobs that are inside the area as shown in Fig.7. Thus, if on the first run only first consonants and vowels were hit during second run final consonants will be selected as well or vice versa.

If more than 50% of the blob is inside the area, then it is regarded as "inside"; otherwise, it is "outside" (Fig. 8).

Blobs selected during the second run are regarded as parts of a string and define four points. Using these four points, affine transformation is performed for both image and blob information.

D. Connected-Components Based Text Segmentation

The process of test segmentation is illustrated in Fig. 9. In the first string "당고개", the last character consists of two disjoined in vertical projection letters: "ㄱ" and "ㅐ". The segmentation algorithm must join them into "개". The second example demonstrates how connected-component analysis due to noise defines two characters "청" and "량" as one solid "청량". The algorithm must handle this problem too.

First, the blobs coordinate information is merged by ordinate axis (Fig. 9b) and line segments are received. Next, the mean height of the line segments is calculated, and, proceeding from right to left, the line segments that are shorter than 70% of the mean height are merged with the line segment on the left (Fig. 9c). Thus, vertical vowels like "ㅏ", "ㅐ" are merged with the first consonants. For example, "ㄱ" and "ㅐ" are united into "개". Finally, proceeding from left to right, we split the line

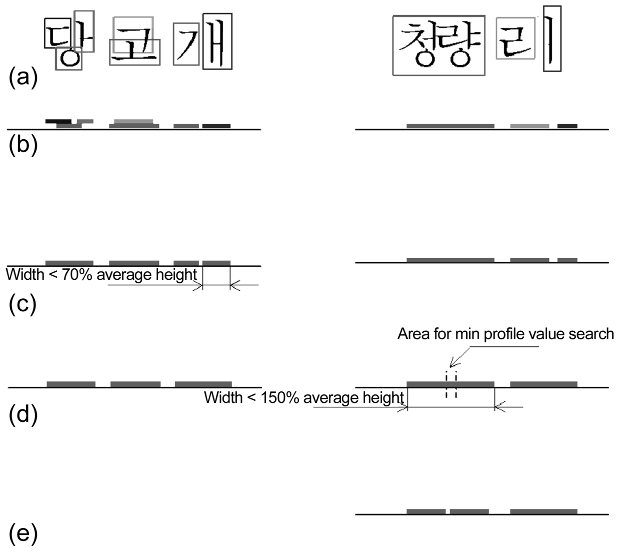
(a)
당 고 개
청량 리
(b)
Width < 70% average height
(c)
Area for min profile value search
Width < 150% average height
(d)
(e)

Fig. 9. Blob based text segmentation. Two difficult cases of character segmentation.

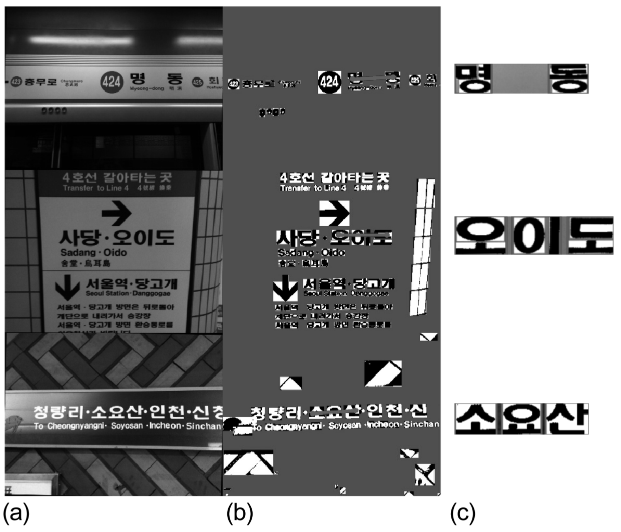

Fig. 10. Text segmentation result: (a) original image, (b) result of binarization, red line is user defined marker-line, (c) segmentation result.

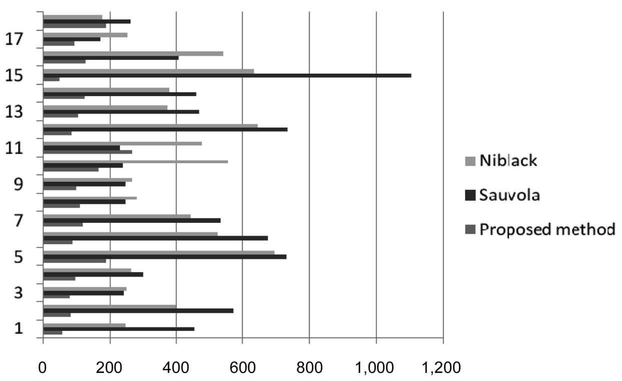

Fig. 11. Binarization runtime.

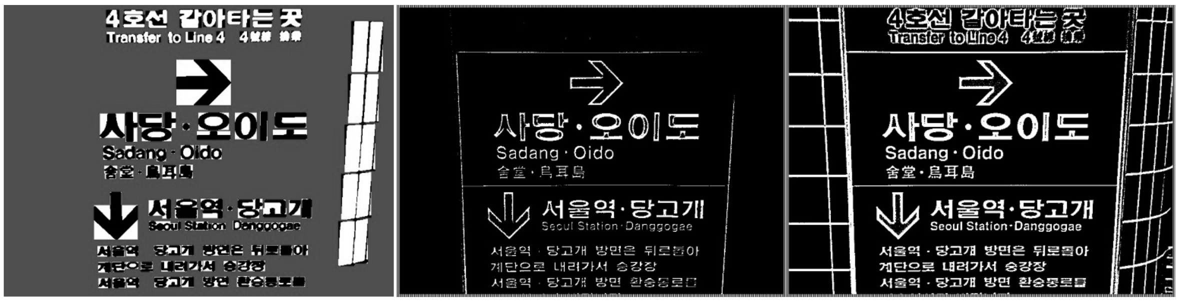

Fig. 12. Quality comparison: (a) proposed method, (b) Sauvola, and (c) Niblack adaptive thresholding.

segments that are longer than 150% of the mean (Fig. 9d), so line segments that define two and more joined letters are split. For example, "청량" is split into "청" and "량" (Fig. 9e). To ensure the split is accurate, we calculate the vertical projection profile for the area between the expected characters. Split is done at the point where the profile has minimum value. At last knowing the start and end points of characters image is cut to give the images for individual characters. Since binarization was performed once, there are two cases: having white text on black background or vice versa. Then, the quantity of white and black pixels is calculated for all blobs. If the number of white pixels is greater than that of black ones, which is the second case (black background, white characters), then color is inverted. The result of segmentation on the real images is demonstrated in Fig. 10.

IV. EXPERIMENTS

A. Implementation Details

The method described above was implemented on iPhone 3Gs running iOS 4.3 and written in C++ STL using precompiled static libraries: OpenCV 2.0 and cvBlobsLib 6. The graphical user interface is written in Objective-C using Quartz 2D graphical library.

B. Runtime

For runtime evaluation, we estimated the time that it would take to perform all steps that our method of binarization includes. To make an accurate comparison, we ran Niblack and Sauvola adaptive thresholdings and connected component analysis and measured the process time for each. We run tests for 18 images three times for each image and calculate the average time. The results are shown in Fig. 11.

On average, runtime for our proposed method is 119 ms; runtime for Niblack and Sauvola adaptive thresholding methods are 411 ms and 449 ms, respectively, which means that the proposed method is 3.5 times faster than adaptive Niblack and 3.7 times faster than the adaptive Sauvola thresholding method. Experiments were executed on a desktop simulator equipped with Intel Core 2 Duo (2.4 GHz) CPU, and 4GB of PC3-8500 RAM.

C. Quality

The adaptive Niblack and Sauvola thresholding result depends on the window size. If window size is not selected correctly, then the result becomes noisy. This effect is illustrated in Figure 12, where characters from "사당", "오이도" strings have black noise inside them on Niblack and Saulova thresholding results. In the proposed adaptive thresholding, the window is recalculated for each blob individually, and thus the result is better. Experiments showed that the window size that is 1.7 times greater than the blob's height is optimal.

V. CONCLUSION AND FURTHER STUDY

We presented a fast algorithm for Korean text extraction and segmentation from subway signboards based on smart phone sensors employment. Smart phone sensors provide valuable information helps simplify algorithms, increase processing speed, and reduce memory usage. Adaptive thresholding shows the best result when the window size is matched to the size of character. In case of natural image processing, the runtime of character candidates search and binarization is much smaller than the time required for a whole-image binarization.

For future research, on text blobs could be filtered better using more sophisticated heuristics, but keeping computing time minimal. The proposed algorithm could be used for the future development of smart phone applications. We plan to deploy it in the subway navigation system, where text from signboard image could be extracted and recognized, and thus provide information about user's current location and directions.

REFERENCES

1. J. Park, G. Lee, A. N. Lai, E. Kim, J. Lim, S. Kim, H. Yang, and S. Oh, "Automatic detection and recognition of shop name in outdoor signboard images," *IEEE International Symposium on Signal Processing and Information Technology 2008*, Sarajebo, Bosnia & Herzegovina, 2008, pp. 111-116.
2. T. N. Dinh, J. H. Park, and G. S. Lee, "Low-complexity text extraction in Korean signboards for mobile applications," *8th IEEE International Conference on Computer and Information Technology 2008*, Sydney, 2008, pp. 333-337.

3. N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on System, Man, and Cybernetics*, vol. 9, no. 1, pp. 62-66, Jan. 1979.
4. J. He, Q. D. M. Do, A. C. Downton, and J. J. Kim, "A comparison of binarization methods for historical archive documents," *Eighth International Conference on Document Analysis and Recognition*, Seoul, Korea, 2005, pp. 538-542.
5. John Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679-698, 1986.
6. S. H. Lee, J. H. Seok, K. M. Min, and J. H. Kim, "Scene Text Extraction using Image Intensity and Color Information," *Chinese Conference on Pattern Recognition*, Nanjing, China, 2009, pp. 1-5.
7. S. H. Lee, M. S. Cho, K. Jung, and J. H. Kim, "Scene text extraction with edge constraint and text collinearity," *20th International Conference on Pattern Recognition*, Istanbul, Turkey, 2010, pp. 3983-3986.
8. J. Jung, E. Kim, S. H. Lee, and J. H. Kim, "Scene text separation using touch screen interface," *Chinese Conference on Pattern Recognition*, Nanjing, China, 2009, pp. 1-5.
9. A. N. Lai, K. N. Park, M. Kumar, and G. S. Lee, "Korean text extraction by local color quantization and k-means clustering in natural scene," *First Asian Conference on Intelligent Information and Database Systems* 2009, Dong Hoi, Vietnam, 2009, pp. 138-143.
10. K. S. Bae, K. K. Kim, Y. G. Chung, and W. P. Yu, "Character recognition system for cellular phone with camera," *29th Annual International Computer Software and Applications Conference 2005*, Edinburgh, UK, 2005, pp. 539-544.
11. V. Fragoso, S. Gauglitz, S. Zamora, J. Kleban, and M. Turk, "TranslatAR: a mobile augmented reality translator," *2011 IEEE Workshop on Applications of Computer Vision*, Kona, HI, 2011, pp. 497-502.
12. O. Shiku, K. Kawasue, and A. Nakamura, "A method for character string extraction using local and global segment crowdedness," *Fourteenth International Conference on Pattern Recognition*, Brisbane, Australia, 1998, pp. 1077-1080.

Igor Milevskiy

Igor Milevskiy is currently a M. Eng. candidate in the Department of Computer and Communications Engineering at the Kangwon National University, Korea. He received his specialist's degree in Computer Science from the Pacific National University, Russia in 2009. His research interests include pattern recognition, image processing, and ubiquitous computing.

Jin-Young Ha

Jin-Young Ha is currently professor at Department of Computer Science and Engineering, Kangwon National University. He was a visiting researcher at IBM T.J. Watson Research Center for 1 year from 2000 to 2001. He graduated from Seoul National University (B.E) in 1987. He got M.S. and Ph.D from KAIST in 1989 and 1994, respectively. His research topics include character/pattern recognition and HCI.