2009 IEEE/ASME International Conference on Advanced Intelligent Mechatronics
Suntec Convention and Exhibition Center
Singapore, July 14-17, 2009

Text Extraction from Images Captured via Mobile and Digital Devices

Jian Yuan, Yi Zhang, Kok Kiong Tan, Tong Heng Lee
Department of Electrical and Computer Engineering
National University of Singapore
Email: {g0801794, zhangyi, kktan, eleleeth}@nus.edu.sg

***Abstract*—This paper presents the development of a human-machine interactive software application, named 'Textract', for text extraction and recognition in natural scene images captured via mobile and digital devices. The texts are subsequently translated into another language (simplified Chinese in this project) so that a device such as a mobile phone serves as a portable language translator. In considering of the resource constraint nature of mobile devices, the proposed solution makes best possible choices to balance between recognition accuracy and processing speed.**

***Index Terms*—edge detection, labeling, segmentation, OCR.**

I. Introduction

MOBILE devices (mobile phones, digital cameras, portable gaming devices, etc) are rampantly available nowadays. These small and inexpensive devices facilitate new ways of interaction with the physical world. Over the years, mobile devices gain increasing computational power and storage capabilities. For mobile phones only, beyond sending text messages and making voice calls, recent mobile phones also offer various features like camera, music and video playback, games, web browsing, even GPS, etc.

Meanwhile, people have more opportunities to travel around globally nowadays. Language barrier is usually a big problem. Information on signboards is particularly important. The need for a fast mobile translation device for simple texts on natural scenes especially signboards (e.g.,Fig. 1) is obvious. It is a natural to firstly think of mobile phones as the platform since most mobile phones are equipped with cameras and people carry them almost everywhere they go.

Fig. 1. Sample signboard images captured with a mobile phone

Automatic text extraction from natural scene images is a hot yet tough research topic. Primary challenges lie in the variety of texts: font, size, skew angle, distortion by slant and tilt, shape of the object which texts is on, etc. Environmental factors like uneven illumination and reflection, poor lighting conditions as well as complex backgrounds add more complications.

In developing a mobile application, one must always take into account the resource constraint nature of mobile devices (limited processor speed, memory and battery). Take, for example, a Nokia N81(8GB) which is a typical high-end mobile phone as of Year 2009. It has a CPU clock rate of 369 MHz, which is roughly 1/5 of a typical computer CPU (1.8GHz). A program which takes 4 seconds in computer may take 20 seconds in a mobile phone. In order to make the program considerably fast, techniques adopted in this paper may not deliver the best result among all techniques available but instead deliver relatively good results in short time.

Thus, a mobile solution for automatic text extraction and translation must be able to not only take into account the complexity posed by the problem but also deliver results within a tolerable time. In developing Textract, we relax the modeling of the problem by making following assumptions:

1) The application will only recognize a few commonly used non-italic font types which are usually the case in natural scene images. It works best for Sans-serif and considerably well for Serif (see Fig.2). However, it is not supposed to work on less common fonts like Blackletter, Script, etc.

Fig. 2. Sans-serif font (left) & Serif font with serifs in red (right)

2) Texts on images should be sufficiently large and thick relative to images frames as compared to small texts in a paper document. This assumption is usually valid because texts on signboards are usually large in order to get attentions.
3) Image should not be taken under poor lighting conditions or with uneven illumination.
4) Texts should be roughly horizontal. A significant skew angle is undesirable. The reason for not considering skew angles is that skew angles can only be derived if there are sufficient texts while the number of letters on signboards can usually be just two or three.

The application comprises two main stages: processing stage

and recognition stage. This paper focus more on processing (text extraction) stage.

In the processing stage, the original color image captured by mobile phone is firstly transformed to a grayscale image. Next, the sharp transitions are detected using a revised Prewitt edge detection algorithm. Then the image will be segmented into several regions. Each region can be regarded as an object. Finally, an elimination rule is specially developed to rule out abnormal objects (area too large or too small, width is far longer than the height, etc).

In the recognition stage, characters are first grouped into words, and words are grouped into a sentence. The OCR (Optical Character Recognition) exploits two main features of the character objects: crosses and white distances which will be explained in detail in later sections. Finally, recognized words are checked against a simple dictionary which includes most commonly used words when travelling. The translated meanings can be presented immediately.

II. Processing Stage

The processing stage can be subdivided into 3 sub-stages: pre-processing, region segmentation and post-processing. A flow chart below (Fig.3) depicts the steps involved in processing stage.

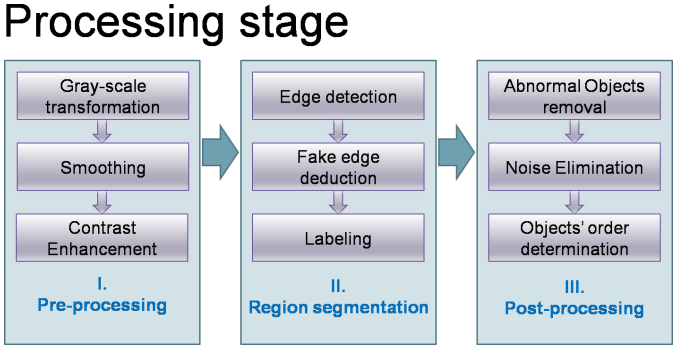


Fig. 3. Flow chart depicting the processing stage

A. Pre-processing

1) Gray Scale Transformation: In this paper, grayscale transformation means transforming a color image to a grayscale image. It will reduce the computational complexity as well as memory requirements significantly. For a specific pixel with RGB values R (red), G (green) and B (blue) respectively, the gray scale representation of this pixel can be derived from:

$$Y = 0.3R + 0.59G + 0.11B \tag{1}$$

2) Smoothing: The predominant use of smoothing is to suppress image noise. It is done by using a median value model (median filtering) which makes use of redundancy in the image data: a pixel will be replaced with the median value of itself and the 8 neighbors. Median value model preserves sharp transitions in the image, so that the image will not be blurred too much as a result of smoothing as compared to other smoothing models like 2D Gaussian smoothing.

3) Contrast Enhancement: Contrast enhancement will be applied to an image captured with the background color similar to the text color or under poor lighting conditions. A simplified enhancement algorithm is given by:

$$G(x,y) = \frac{(S(x,y) - min\_S) \times 255}{max\_S - min\_S} \tag{2}$$

where $S(x,y)$ denotes the pixel value, min_S and max_S are the minimum and maximum grayscale values of $S(x,y)$ respectively. If max_S and min_S are close to each other, e.g., $max\_S - min\_S \leq 80$, the gray scale image will be monotonous with a low contrast. Gray scale extension will increase contrast via extrapolation to achieve $max\_S - min\_S = 255$.

B. Region segmentation

The main objective of region segmentation is to segregate the gray image into several regions, so as to separate the background from the objects. Two of the most common techniques in segmentation are thresholding and edge detection. Thresholding is good for images with bimodal histograms which imply clear cut of background and objects. However, if the background illumination is uneven or the histogram is multimodal which means more than 3 gray levels exist in the image, thresholding will fail. Besides, adaptive algorithms like Ostu's method (Ostu, 1979) must be used to get an optimal threshold which is time consuming. Edge detection works well as long as the uniform illumination assumption holds.

1) Edge Detection: The goal of edge detection is to mark the points in the photo where the luminous intensity changes sharply. Edge detection algorithms generally compute a derivative of this intensity change. Several edge detection algorithms have been developed. Here, a revised first-order Prewitt (Prewitt) edge detector is applied to detect all possible edges. A threshold is also calculated so that edge values beyond the threshold turns to black while all other pixels turn to white.

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & -1 \\ 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & -1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & 1 \end{bmatrix}$$
$$\begin{bmatrix} -1 & -1 & -1 \\ 1 & -2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} -1 & -1 & 1 \\ -1 & -2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -1 & 1 \end{bmatrix}$$

Fig. 4. Convolution kernels

First, through the convolution operations, $D(x,y)$ can be obtained by the following convolution computation:

$$D(x,y) = max(F_i \otimes S(x,y)) \tag{3}$$

where $D(x,y)$ denotes the results after edge detection at (x, y), and F_i is the i^{th} kernel listed above. The following decision making process is proposed to differentiate the edges:

1) Calculate the average value of D(x, y).

$$aver\_E = \frac{\sum_{x=1}^{M}\sum_{y=1}^{N} D(x,y)}{M \times N} \quad (4)$$

2) If $D(x,y) > 2.4 \times aver\_E$, then the pixel at (x,y) will be converted to black. Otherwise, it will be kept white.

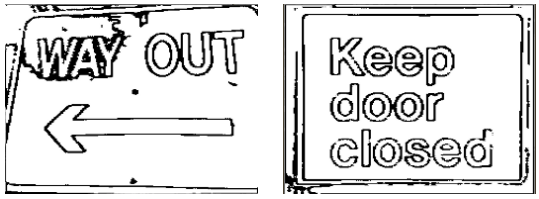


Fig. 5. Images after edge detection

The reason why Prewitt is used instead of a Canny edge detector (Canny, 1986) which is usually considered to give better results is that Canny detector usually applies a 5x5 Gaussian smoothing filter and a rather complicated edge determining algorithm, which is not a viable option in a resource constrained device.

2) Fake Edge Reduction: Through edge detection, many fake edges may exist in the image due to the non-uniform texture in the original image. A reduction function is needed to remove these fake edges.

Let $Aver(x,y)$ denotes the average value of the 8-neighbour of pixel located at (x,y), and $D(x,y)$ denotes the value f the pixel. Then the following are defined:

$$if\ |Aver(x,y) - D(x,y)| > 127.5$$
$$D(x,y) = 255 - D(x,y)$$
$$Otherwise,\ D(x,y) unchanged \quad (5)$$

After this step, some isolated fake edges will be removed. Although this step can not completely remove all fake edges, it reduces computation for the following steps.

3) Labeling: Connected component labeling seeks to assign a unique label to each subset of objects which are connected. A two-pass algorithm is developed to label the connected components in the image. It consists of assigning provisional labels and analyzing final labels using a union-find method.

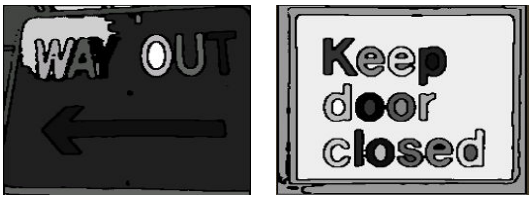


Fig. 6. Images after labeling, with different gray-level denotes different regions

C. Post-processing

1) Abnormal Objects Removal: (Integrated with noise elimination for computational simplicity)

After the previous stage, each object is labeled and could be considered as an object. The objective in this stage is to eliminate invalid objects and retain only wanted objects, i.e., meaningful objects representing letters.

Abnormal objects usually include boundary, signs, rifts etc. This part involves a sequence of carefully designed rules for intelligently selecting the 'right' objects.

For convenience, a term 'enclosed' is defined as: if object A is enclosed by object B, it means the smallest rectangle that encloses B encloses the smallest rectangle that encloses A, which is pictorially depicted as in Fig. 7. The letter 'd' encloses the solid object because the red rectangle is enclosed by the blue rectangle.

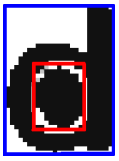


Fig. 7. pictorial definition of "enclose"

Four steps are devised to filter out unwanted objects. (Note: objects in each step are referring to those left after previous step only.)

Step I
- Any object touching the frame is eliminated. (This is based on the assumption that when a snapshot is taken, the text will not touch the frame).
- Any object satisfying any one of the following criterion is eliminated:
  - $height > 0.8 \times height\ of\ the\_image$;
  - $width > 0.4 \times width\ of\ the\ image$;
  - $height < 15\ and\ area < 150$;
  - $height < 0.06 \times\ height\ of\ the\ image$;
  - $height/width > 16$;
  - $width/height > 2$;
  - $area > 0.08 \times total\ area\ of\ the\ image$.

Step II
- Any object satisfying any one of the following criterion gets eliminated:
  - $area > 8 \times average\ area$;
  - $area < 0.15 \times average\ area$;
  - $area < max\ area/20\ (noise\ elimination)$.

Step III
- Any object satisfying any one of the following criterion gets eliminated:
  - $height > 1.8 \times averageheight$;
  - $width > 1.8 \times averagewidth$;

Step IV
- Any object which is enclosed by another one is eliminated.

Step I filters out abnormal objects using their absolute features (i.e. not utilizing the relative features like average area and average height which include information of other objects). The reason to do this is to ensure that no obvious abnormal objects corrupt the relative features).

Step II filters out abnormal objects using relative features like average area and maximum area. Step I ensures that objects at this step include as many valid objects as possible.

Step III filters out abnormal objects also using relative features average height and average width. Step II also ensures that this step includes as many valid objects as possible.

Step IV filters out those which are enclosed by another one.

Fig.8 shows the results after abnormal objects removal.

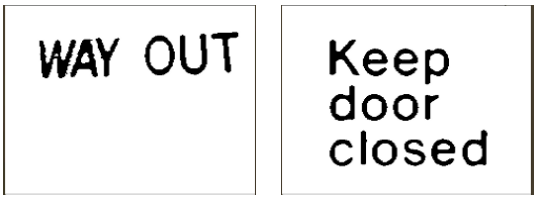


Fig. 8. Images after abnormal objects removal

2) Objects' Order Determination: This step is basically to determine the order of the letter objects when they are viewed as a part of a word and a sentence. The order of letters should correspond to the order when they are arranged as a word. i.e. 'W' is assigned the order 1; 'A' is assigned the order 2, etc.

III. Recognition Stage

This stage is about implementing a simple yet efficient OCR (Optical Character Recognition) algorithm. OCR is the translation of images of handwritten, typewritten or printed texts (usually captured by a scanner) into machine-editable text. Artificial Neural Network (ANN) has gained a lot of popularities in pattern recognition in recent researches. ANN need a large number of training samples to train the network properly. There are a lot of standard database for handwritten texts but not for printed texts especially when they undergo previous processing steps.

In this paper, since texts to be recognized are all printed texts, template matching will yield considerably good results. A template matching method is developed to recognize the characters. It utilizes the two main features of a character: crosses and white distances.

Definitions of crosses and white distances:

Think of a character 'A' strictly confined in a square box as shown in Fig.9. The white pixels are 0 and blue pixels (i.e. body of the text 'A') are 1.

1) A cross (either vertical or horizontal) is defined as the number of zero_one crosses as it is traversed from one side to another (either vertically or horizontally) as shown pictorially in Fig.9.
2) White distances are defined as the distance as it is traversed from one direction (left, right, downwards, upwards) until the first 1 is met, as shown pictorially in Fig.9.

Using information of vertical crosses, horizontal crosses, left white distances, right white distances, top white distances and bottom white distances and comparing with a template of these 6 traits of all alphabets, characters will be differentiated with very high accuracy.

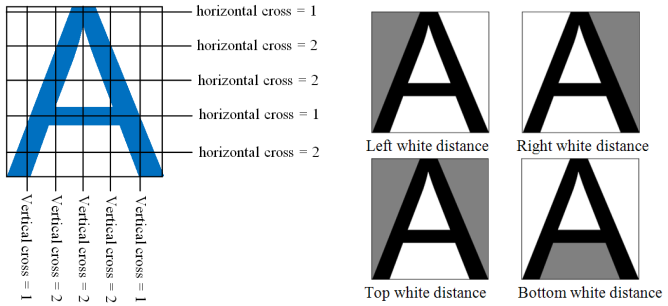


Fig. 9. A pictorial definition of crosses & white distances (shaded area)

For example, the six attributes of letter 'A' is as follows in the application. (White distances are obtained by normalizing the object into a 15x10 size object)

- $Vertical\_crosses = \{1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2\}$;
- $Horizontal\_crosses = \{1, 1, 1, 2, 2, 2, 2, 1, 1, 1\}$;
- $Left\ white\ distances = \{3, 3, 3, 3, 3, 2, 2, 2, 2, 1, 1, 1, 0, 0, 0, 0\}$;
- $Right\ white\ distances = \{4, 3, 3, 3, 3, 2, 2, 2, 2, 1, 1, 1, 0, 0, 0, 0\}$;
- $Top\ white\ distances = \{13, 9, 6, 2, 0, 0, 2, 5, 9, 12\}$;
- $Bottom\ white\ distances = \{0, 0, 2, 3, 3, 3, 3, 3, 0, 0\}$.

A group of templates of all information about the 52 alphabetical letters (both capital and small letters) is prepared; detected attributes of each letter are compared against the templates using Least Squares Method.

Finally, letters identified from the image and converted to ASCII values are subsequently filled into a linguistic model to be grouped into words and checked against a dictionary. The meanings or translations of the texts will be retrieved immediately.

Certainly, there are situations when a letter is misrecognized and the translation could not be found, for example capital letter 'I' and small letter 'l' are very much alike. Error correcting models are built to tackle these situations.

The translation includes both words and phrases. Common phrases included in the dictionary like "car park", "fire hose reel", "keep door closed" and etc could be directly translated, while cases like "slippery when wet" would be translated separately.

IV. Platform & Implementation

The whole application is developed on J2ME (Java 2 Micro Edition) platform which is a specification of a subset of the Java platform for small and resource-constrained devices such

as mobiles phones and set-top boxes. Textract can be installed in any Java supported camera phone, which is widely available in the market nowadays. A sample emulation result in J2ME is shown in Fig.10.

Fig. 10. J2ME emulator UI: Image before preprocessing (left), recognition result with the image after processing stage(right).

V. Results

There is no standard database available for natural scene images captured by mobile devices. 10 photos were captured manually at various places in National University of Singapore. The results are shown in Table I.

It can be seen that Textract offers good results in recognizing large texts. It gains high accuracy if photos are taken well according previous assumption. However, it should be noted that Textract will fail when texts are too thin as shown in the last image where certain parts of the texts fade out.

A solution proposed to remedy this problem as implemented in this application is to capture images in larger size and allow user to manually zoom in and crop the area of interest. Through this way, texts will be relatively larger and thicker compared to the cropped image. With user interactions, recognition accuracy increases significantly.

VI. In a real phone

Fig.11 demonstrates Textract in a Nokia N81 recognizing and translating an natural scene image containing texts “DANGER ZONE”.

VII. Conclusion

In this paper, an approach for text extraction from images captured from sign boards with mobile phones is proposed. The program has been tested successfully on a J2ME emulator and a NOKIA N81 phone. The average processing time is about 18

TABLE I
Test Results

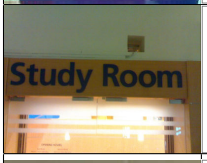
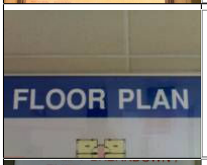
Photos (format: PNG, image size: 640x480)	Processing results by Textract	Results by Textract (with translation)
	FIRE HOSE REEL	翻译如下: ql: 无结果 fore: 无结果 hose reel: 软管绞盘
	Block Adm	翻译如下: block: 块，妨碍，街区 adm: 无结果
	SECURITY POST	翻译如下: security post: 安全局
	CAUTION SLIPPERY WHEN WET	翻译如下: cautioh: 无结果 slippery: 滑 when: 在...时候 wet: 湿
	EXIT	翻译如下: exit: 出口
	Keep out	翻译如下: keep: 保留，保持 out: 出去，外面的
	Study Room	翻译如下: study room: 自习室
	FLOOR PLAN	翻译如下: floor: 地面，地板，层 plan: 计划，设计 d: 无结果 o: 无结果
	IN CASE OF FIRE DO NOT USE LIFT	翻译如下: in case of fire do not use lift: 火灾时请勿使用电梯
	WATER RISER	翻译如下: vjateq: 无结果 r: 无结果 ser: 无结果

Fig. 11. Captured image (left) and translation result (right) in a real phone

seconds. In the software, user interactions can be incorporated in the way that users are able to select an area of interest. As the mobile phone processor speed and memory size increase, the processing time will be greatly reduced. Also, there is still space for improvement on robustness against font types and font thickness, as well as translations as sentences instead of individual words using machine translation techniques.

REFERENCES

[1] R. Brinkmann, Median filter, *The art and science of digital compositing*. , pp. 51 - 52, Morgan Kaufmann, 1999.

[2] J. C. Russ, Correcting Imaging Defects, *The image processing handbook, 5th ed.* , pp. 206 - 214, CRC Press, 2006.

[3] M. Sonka, V. Hlavac and R. Boyle, Thresholding, *Image Processing, Analysis, and Machine Vision, 3rd ed.* , pp. 176 - 183, Thomson, 2008.

[4] S. V. Rice, G. Nagy and T. A. Nartker, *Optical character recognition: an illustrated guide to the frontier*. , Springer, 1999.

[5] N. Otsu, A threshold selection method from gray-level histograms, *IEEE Transactions on systems, Man, And cybernetics, Vol. SMC-9, No.1.* , January 2008.

[6] M. S. Nixon and A. S. Aguado, Prewitt edge detection operator, *Feature extraction and image processing*. , pp. 121 - 123, Academic Press, 2008.

[7] R. R. Rakesh, P. Chaudhuri and C. A. Murthy, Thresholding in Edge Detection: A Statistical Approach, *IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 13, NO.7.* , JULY 2004.

[8] S. Mori, H. Nishida and H. Yamada, *Optical Character Recognition*. ,pp. 2-39, John Wiley & Sons, NIC. 1999.

[9] J. Canny, A Computational Approach to Edge Detection, *IEEE Transactions of Pattern Analysis and Machine Intelligence, Vol. PAMI-8, No. 6.* , November 1986.

[10] V. A. Shapiro and P. K. Veleva, An Adaptive Method for Image Thresholding, *IEEE Comput. Soc. Conf. on Pattern Recognition and Image Process, 1992*.

[11] M. J. Atallah, Connected Components, *Algorithms and Theory of Computation Handbook*. , pp. 22-2 - 22-4, CRC Press, 1998.

[12] Y. Zhang and K. K. Tan, Text Extraction from Images Captured via Mobile and Digital Devices, *5th Int. Conf. on Industrial Automation, TITS-07-P03*. , Montreal Canada, 11-13 June 2007

[13] J. Yang, J. Gao, Y. Zhang, X. Chen and A. Waibel, An Automatic Sign Recognition and Translation System, *PUI 2001, November 15-16, 2001, Orlando, FL, USA.*.