IMAGE TEXT TRANSLATION

Vikas Bhatnagar, Gargi Dhyani, Himanshu Yadav, Harshita Gupta
Department of Computer Science and Engineering
Moradabad Institute of Technology Moradabad, India

Abstract – Influenced by the well-known translation application 'Google Lens' that is able to recognize the text captured by a mobile phone camera, translate the text, and display the translation result back onto the screen of the mobile phone. Our text extraction and recognition algorithm has a correct-recognition rate that is greater than 95% on character level. In this report, we demonstrate the system flow, the text detection algorithm and detailed experiment result.

Keywords- Android, Image Translation, OCR, Natural Language Processing, Tesseract.

I. INTRODUCTION

The inspiration of a real time text translation mobile application provide help to the tourist to remove the language barrier. The application we developed enable the users to navigate in a native language environment. Our mobile application helps the user to get text translate as faster and easier as a button click Smartphone camera captures the text and returns the translated results as per user request in the real world.

The application system we developed generally include automatic text detection, OCR text language detection, correction and text translation. Presently, the current version of our application is used to translation English to Hindi, Urdu, Bengali, Telugu and Tamil and vice versa, but it has some future scope that can be easily be extended into a much wider range of language sets

A. Prior and Related Work

1) Text Extraction: Text extraction techniques are used because text embedded in images and videos provides important information. Many characters of text regions have been summarized and characterized effectively by several features, e.g. text pixels have homogeneous color, character strikes form different texture, etc. Y. Hasan and J. Karam developed a text extraction algorithm that works morphological edge/gradient detection Algorithm by Epshtein et-al tackled the problem from another approach using text stroke transform.

2) OpenCV: OpenCV stands for Open Source Computer Vision. It is a library of programming functions for real time computer vision. The library has more than 2000 optimized algorithms and has been widely used around the world. More than this, android programmers are able to implement many digital image processing algorithms in Android phone platform.

3) Optical Character Recognition: OCR, Optical Character Recognition, is developed to translate scanned images of handwritten, or printed text into machine-encoded text. OCR software have been developed to accomplish this technique. Tesseract, originally developed as a software at Hewlett Packard between 1985 and 1995, now sponsored by Google, is considered to be one of the most accurate open source OCR Engine currently available. It is capable of recognizing text in variety of languages in a binary image format.

4) Text Correction: The text correction is a necessary step after OCR text recognition, since the result returned by the OCR engine is not always correct due to noise in it. This type of errors can be categorized into a non-word error - which means that the text string returned by OCR does not correspond to any valid word in a given word set. Such text correction systems include A spell, and also the well-known Peter Norvig's text correction algorithm.

II. SYSTEM FLOW

In this paper, we propose a text translation algorithm that consists of following steps:

1) Morphological edge detection
2) Text feature filtering
3) Text region binarization

4) Optical character recognition
5) Text correction
6) Text translation
7) Display of the translation

A. Step 1 - Canny Edge Detection

Canny edge detection is a technique to extract useful structural information from different vision objects and dramatically reduce the amount of data to be processed. It has been widely applied in various computer vision systems. Canny has found that the requirements for the application of edge detection on diverse vision systems are relatively similar. Thus, an edge detection solution to address these requirements can be implemented in a wide range of situations. The general criteria for edge detection include:

1. Detection of edge with low error rate, which means that the detection should accurately catch as many edges shown in the image as possible
2. The edge point detected from the operator should accurately localize on the center of the edge.
3. A given edge in the image should only be marked once, and where possible, image noise should not create false edges.

To satisfy these requirements Canny used the calculus of variations – a technique which finds the function which optimizes a given functional.

The optimal function in Canny's detector is described by the sum of four exponential terms, but it can be approximated by the first derivative of a Gaussian.

Among the edge detection methods developed so far, canny edge detection algorithm is one of the most strictly defined methods that provides good and reliable detection. Owing to its optimality to meet with the three criteria for edge detection and the simplicity of process for implementation, it became one of the most popular algorithms for edge detection.

Process of Canny Edge Detection Algorithm:

The Process of Canny edge detection algorithm can be broken down to 5 different steps:

1. Apply Gaussian filter to smooth the image in order to remove the noise
2. Find the intensity gradients of the image
3. Apply non-maximum suppression to get rid of spurious response to edge detection
4. Apply double threshold to determine potential edges
5. Track edge by hysteresis: Finalize the detection of edges by suppressing all the other edges that are weak and not connected to strong edges.

B Step 2 - Text Feature Filtering

In order to reduce the number of connected components that have to be analyzed, a close operation with a 5 by 5 structuring element is performed to the binary edge image obtained from Step 1. After the close operation, all connected components of the edge image are screened with their position, size, and area information. A candidate of letter should meet a set of constraints in size and shape. In our algorithm, we select connected components as letter candidates if the following requirements are met: 1) Width of the bounding box < 0.5 image width 2) Height of the bounding box < 0.3 image height 3) 0.1
< center width of the bounding box < 0.9 4) 0.3 < center height of the bounding box < 0.7 5) Width vs. height ratio < 10 6) Width of the bounding box > 10 pixels 7) 0.1 < Connected component filled area over (width height of the bounding box)
< 0.95 8) Width of the bounding box > 10 pixels 9) 0.1 < Connected component filled area over (width height of the bounding box) < 0.95 After the first round filtering, it is expected that most of the non-letter components would be removed. So, the majority of the remaining candidates should be letters with the same font and size. Based on this condition, we calculate the mean height hm of the bounding box of the remaining components, and remove any connected component with its height smaller than 0.6hm or greater than 1.8hm.

C. Step 3 - Text Region Binarization

Each remaining boundary box is used as a mask to the original gray-scale image. Since each bounding box is relatively small compared to the size of the entire image, no further adaptive thresholding method is implemented. Theoretically after this step, only stroked letters are left as the foreground, 1, and the rest of the image would go to background 0.

D. Step 4-6 - Text Recognition, Correction, and Translation

Since the project is focused on implementing text extraction on a mobile phone, we implemented the following three steps - text recognition, correction and translation on a server with open source software for simplicity's sake. Google's open source OCR - Tesseract is used as the optical text recognition engine. Peter Norvig's algorithm is added to the routine to perform text correction. Then Google translator is used to translate the text into Chinese.

E. Step 7- Display of the Translation

The translated text string from Step 6 is sent back to the mobile device (Android phone) from the server, and then displayed at the top center region of the screen. A sample text extraction process flow is shown in Figure 1 below. The final result frame display after step Figure 4-7 is shown in Figure 2.

III. TEST AND RESULTS

The performance of our system is evaluated by the rate of successful recognition. We decide to use recognition rate rather than successful translation rate as the criterion of performance, because the recognition rate more directly measures the successfulness of the text identification algorithm, whereas the measure of translation rate can be influenced by the Google translation engine, over which we have no control. The recognition rate is defined as, the ratio between the number of successfully recognized letters and the total number of letters in a test image.

We conducted experiments to evaluate the performance of our system under different scenarios. The font size, font and means of display are varied in order to test their effect on the performance of the system.
Two phrases," Digital Image Processing" and "Visual Information Plays an Important Role" are used for the test. The two phrases contain 44 and 72 letters respectively. We counted the number of letters recognized from the output of the OCR engine and calculated the recognition rate.

Fig.1

A. Font

The test phrases were displayed in Arial, Calibri, Times New Roman and Arial Bold, and their recognition rates were measured. We did not observe significant difference of the recognition rate among the four fonts.

B. Font Size:

We used large, medium and small sizes of letters for testing. The font size of large letters was 40, the size of medium letters was between 20 and 25, and the size of small letters was between 10 and 15. We fixed the camera lens approximately 30 cm away from the letters, so that small letters would appear small in the camera frame. We found significant effect of the font size on the recognition rate. As shown in the table above, large text

The Recognition Rate for Text with Large, Medium and Small Font sizes achieve much higher recognition rate than medium and small size texts do.

TABLE I:

Font Sizes	Large	Medium	Small
Recg.Rate (letter)	0.94	0.83	0.88
Recg.Rate (word)	0.83	0.84	0.81

C. Means of Display

The text was displayed on a computer screen and on a piece of paper. We tested the recognition performance with both display methods. As we had expected, the recognition rate of the text displayed on a computer screen was slightly higher than the result of the text printed on a piece of paper. This is because computer screen has higher contrast than paper.

TABLE II:

The Recognition Rate for Text on a Computer Screen and on Paper

Display	Computer Screen	Paper
Recg.Rate (letter)	0.90	0.87
Recg.Rate (word)	0.82	0.85

We repeated the above experiment after text correction is applied. The text correction improves the performance by 5% if we measure how many words are successfully recognized. We summarized our experimental results in Fig 2..

Fig.2

IV. CONCLUSION

We have achieved an Android based application for real-time text extraction, recognition and translation. The average correct character-recognition rate is above 95%. From the performance evaluation of our system, we concluded that our application is very robust for text written in different languages. Following work needs to be done in order to drive our application into a defense purpose application.

Further use of camera pen or touch pen extracts the characters from the opposite side of the person and then translate it into the user defined language and recognize what the text is in addition of this more language translation selections for the user.

REFERENCES

[1] Yassin M.Y.Hasan and Lina J.Karam, Morphological Text Extraction from Images. IEEE Transaction on Image Processing Vol.9 No.11, Nov 2000

[2] Nobuyuki Otsu, A threshold selection method from gray-level histograms. IEEE Trans.Sys.,Man., Cyber 9

[3] http://code.google.com/p/tesseract-ocr/

[4] http://norvig.com/spell-correct.html

[5] http://austingulati.com/2009/07/google-translate-php-api/

[6] Farshad Ghazizadeh, Optical Character Recognition. US Patent: 5,007,809.

[7] Huiping Li, David Doermann and Omid Kia, Automatic Text Detection and Tracking in Digital Video. IEEE Transaction on Image Processing Vol. 9 No. 1, Jan 2000