ICPR2002 in Quebec City, Canada, Vol.3, pp.473-476 (2002)

Text Scanner with Text Detection Technology on Image Sequences

Keechul Jung[1], Kwang In Kim[2], Takeshi Kurata[3], Masakastu Kourogi[3], JungHyun Han[4]
[1]: Pattern Recognition and Image Processing Lab., Michigan State Univerisity, US.
[2]:Artificial Intelligence Lab., KAIST, Korea.
[3]: National Institute of Advanced Industrial Science and Technology, Japan.
[4]: School of Electrical and Computer Engineering, Sungkyunkwan University, Korea.

Abstract

We propose a text scanner, which detects wide text strings in a sequence of scene images. For scene text detection, we use a multiple-CAMShift algorithm on a text probability image produced by a multi-layer perceptron. To provide enhanced resolution of the extracted text images, we perform the text detection process after generating a mosaic image in a fast and robust image registration method.

1. Introduction

Based on the fact that texts within an image are very useful for describing the contents of the image and can be easily extracted comparing with other *semantic contents*, researchers have recently attempted *text-based image indexing* using various image processing techniques [1-6].

The texts in digital videos can be classified into *scene texts* that exist naturally in the video and *caption texts* that artificially overlaid on the video. Scene texts are more difficult situations [5] and very little works have been done comparing with caption texts.

Although very effective in text detection in real scene images, texture-based text detection method has several shortcomings: (1) difficulties in manually designing a texture classifier for various text conditions, (2) expensive computation in the texture classification stage, and (3) moreover the traditional text detection methods are usually for the images captured by high-resolution cameras or scanners. However in the real applications, it would be better for us to use a low-resolution camera thanks to its economical merit.

In this paper, we propose a texture-based ***text scanner*** method. In order to tackle previously mentioned shortcomings of traditional text detection methods, we use a multiple continuously adaptive mean shift (*MultiCAMShift*) algorithm on the text probability image (*TPI*), produced by a multi-layer perceptron (*MLP*) receiving the mosaic image as an input image. To support wide text strings captured by a low-resolution camera, we create a mosaic image from input image sequences in a fast and robust image registration method. This approach efficiently detects texts in real scene images, captured by a low-resolution camera, as we use a MLP for texture analysis and MultiCAMShift algorithm for enhancing speed on the mosaic image.

2. Scene Text Detection

Our overall text detection system consists of 2 steps: image registration and text detection.

2.1 Image Registration

Our image registration method estimates frame-to-frame alignment parameters by the following computationally efficient steps [7]. Its real-time processing is of great significance because text detection is a pre-processing stage of OCR.

Repeat steps 1-4 on the successive frames until either motion-compensation-error is below a threshold or a fixed number of iterations has been completed.

1. Compute pseudo motion vectors that are rough estimates of the optical flows at each pixel.
2. Verify the computed motion vectors by pixel-wise matching
3. Estimate the motion parameters from the verified motion vectors.
4. Compensate the global motion between successive frames
5. Update the motion parameters by using M-estimator.

To create a new panoramic image from image sequences, we estimate frame-to-frame alignment parameters between captured frames, and merge each frame to a mapped plane by warping it with the estimated parameters. We adapt affine transform to a cylindrical panorama and projective transform to a planar panorama.

2.2 MLP for Texture Classifier

In this section we describe MLP-based texture classification for scene text detection in brief. Readers are referred to [6, 8] for more detail description. We use a 2-hidden-layer-structure MLP, which is fully connected between adjacent layers, to make a texture classifier that

discriminates between text pixels and non-text ones. An input image is scanned by the MLP, which receives the gray values of a given pixel and its neighbors within an input window. We call the resulting image (produced by the MLP) a text probability image (TPI), where each pixel's value is in the range [0,1] and represents the probability that the corresponding input pixel is a part of text. The MLP can adapt itself to the various styles and sizes of texts and complex backgrounds. To handle practically infinite non-text images in the training stage of MLP, we use the *bootstrap method* recommended by Sung and Poggio [9]. This bootstrap method is used to get non-text training samples more efficiently and to force the MLP to learn a precise boundary between text and non-text classes.

2.3 MultiCAMShift Algorithm

We use a MultiCAMShift algorithm on the TPI for text detection, which is a modified version of the CAMShift algorithm [10, 11]. To avoid the full scanning of an input image with the MLP, which we mentioned as a drawback of the texture-based methods in Section 1, CAMShift algorithms are invoked at several seed positions on the TPI. This leads to great computational savings when text regions do not dominate the image. Fig. 1 is the MultiCAMShift algorithm for text detection.

At the initial stage of the CAMShift algorithm on each search window, we decide whether the search window contain texts or not. During consecutive iterations, we estimate the size, position, skew angle of the text region using 2D moments [10], change the parameters of a search window depending on the estimated values, and then perform a window-merge operation to eliminate overlapping search windows. If the mean shift is larger than either of the threshold values ε_x (along x-axis) and ε_y (along y-axis), the iteration continues.

We convolve every pixel *(x, y)* in a search window. If a pixel *(x,y)* has been convolved by MLP in previous iterations, re-use *TPI(x,y)*. Otherwise we perform convolution using MLP at pixel *(x,y)*. The size, position, and skew angle of text regions are estimated by 2 dimensional moment calculations [10]. At each iteration, we have to merge some overlapping search windows to save processing time of the MultiCAMShift algorithm. When overlapping windows exist, we examine whether they are originally a single text or separate texts. This is done by checking the degree of overlap between two regions as follows. Let D_α and D_β be the areas covered by two text regions α and β. Then the degree of overlap between α and β is defined as

$$\Lambda(\alpha,\beta)=\max\left(\mathrm{s}\left(D_\alpha\cap D_\beta\right)/\mathrm{s}\left(D_\alpha\right),\ \mathrm{s}\left(D_\alpha\cap D_\beta\right)/\mathrm{s}\left(D_\beta\right)\right), \quad (1)$$

where $\mathrm{s}(\lambda)$ counts the number of pixels within λ. Then, α and β are determined to be a single text if $T_O \le \Lambda(\alpha,\beta)$ where T_O is a threshold set to be 0.9. Otherwise they are separate texts.

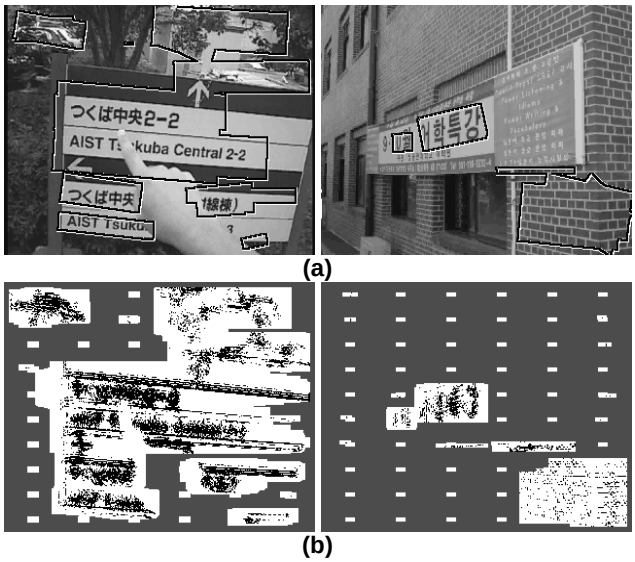

Fig. 2. Results of text detection: (a) detected text regions, (b) corresponding TPIs.

Overlapping windows identified as a single text are merged, and the merging process is iterated until no more

- Set up the initial locations ($mean_x(i)_0$, $mean_y(i)_0$) and sizes $\lambda_x(i)_0 \times \lambda_y(i)_0$ of search windows *W*s
- **Do**
  - For each search window *W(i)*
    - Generate the TPI within *W(i)* using MLP.
    - Based on the mean shift vector, derive the new position and size of a text region.
    - Modify *W(i)* according to the derived values.
  - Merge overlapping windows.
  - Increment the iteration number *t*.
  - **While** ($\| mean_x(i)_t - mean_x(i)_{t+1} \| > \varepsilon_x$ **or** $\| mean_y(i)_t - mean_y(i)_{t+1} \| > \varepsilon_y$)
- Filter localized text regions using area and aspect ratio of the region-bounding rectangles.

Fig. 1. MultiCAMShift algorithm for text detection.

regions are mergeable.

Fig. 2 shows the intermediate detection results using MultiCAMShift: (a) shows examples of detected text regions and (b) shows TPIs corresponding to the detected regions. White regions in Fig. 2(b) are convolved regions with MLP. Black pixels in the white regions denote text pixels. Regardless of the character types ((a) Japanese/English/numerals, (b) Korean) the MLP shows a great performance in text detection. From the experimental results we can conclude that our MLP-based texture classifier is enough for the text detection in a various conditions. Moreover its learning capability is suggesting great adaptability for the scene text detection.

Nonetheless it is still not enough to use the results of the MLP without performing any post-processings such as filtering out the false alarms. As one can see in the second images of Fig. 2, the MLP texture filter fails to detect the excessively small characters. However these extremely small characters are not important for the purpose of indexing. After convergence we filter out non-text regions using the size and the aspect ratio of the text-bounding boxes. We have used the following heuristics to filter out non-text boxes:

(1) Text region must be larger than MinArea=100 pixels.
(2) The aspect ratio of the text region must be smaller than MaxAspect=20.

3. Experimental Results

The proposed text detection approach has been tested on several scene images. First of all, we test the proposed method to several still images including license plate images, road sign images, etc. And then, we apply it to the wide scene images containing nameplates, signboard, shop-signs, advertisements, etc with the help of image mosacing.

For parameters for image registration, refer to [7]. In a training stage of the MLP, we use a constant learning rate η=0.02, momentum m=0.5 and the sigmoidal activation function $f(net) = a\tanh(b\ net)$ with a= 1.7 and b=0.75. After several experiments, we set the input window size of the MLP to 13×13.

For MultiCAMShift algorithm, the initial locations of the search windows should be dependent on applications. They should also be determined to be dense enough not to miss texts located between them, and should not be too dense to obtain moderate processing speed. We have IW/52 × IH/24 search windows for the image of size (IW, IH), located at the regular interval. For the stopping criterion of MultiCAMShift algorithm, we set ε_x=2 and ε_y=1. The average number of MultiCAMShift iterations for an image is 5. It shows the efficiency of the MultiCAMShift algorithm. One of the extra benefits is that processing time of our MultiCAMShift-based text detection method depends on the size of text regions in the input image. The less text regions, the less processing time.

In this section, we show the experimental result for the sample data. Fig. 3 is the results of text detection for a frame: MultiCAMShift algorithm marks the detected text regions with black lines. One can see the example of mis-classification for very small characters and low contrast characters. However it seems to be also difficult for human to detect the text region in this circumstance. One can see the example of false alarm in Fig. 3. However these could be easily discarded by OCR routine.

Fig. 3. Examples of text detection for a frame.

Fig. 4 and 5 are the case of wide banner in outdoor environment. Input images (Fig. 4-5(a)) are a sequence of left-top to right-bottom. We use a fast and robust method for estimating frame-to-frame registration. Also as we generate the text probabilities only for pixels within the search window, the average number of pixels convolved by MLP is reduced to about a tenth of exhaustive search's [6]. So is the processing time. Moreover, the system has given a processing time shorter than 0.05 second per a frame with shifting (interval size of 3 pixels for each column and row).

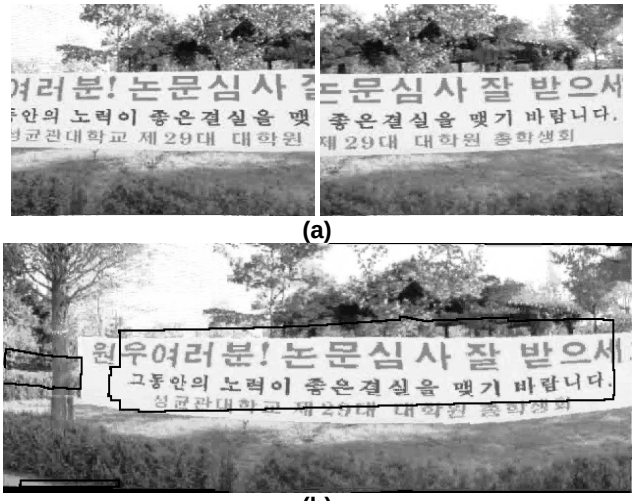

Fig. 4. Examples of text detection for an image sequence: (a) input frames and (b) mosaic image with detection result.

Fig. 5. Another examples of text detection for an image sequence: (a) input frames and (b) mosaic image with detection result.

4. Conclusions

In this paper, a novel and efficient text detection technique using MLP and MultiCAMShift algorithm is presented. It can be distinguished from other approaches by the following features: (1) texture classifier for generating TPIs can be automatically constructed using MLP; (2) by adopting MultiCAMShift we do not need to analyze the texture properties of an entire image; (3) the proposed method works particularly well in extracting texts from the complex and textured background; and (4) it can detect the wide text regions with the help of mosaic technique.

For the indexing purpose, text tracking is extremely important to relieve burden of recognizing every texts in every frames. Accordingly future works should include the utilization of the proposed method on text tracking.

References

[1] Huiping Li, David Doerman, and Omid Kia, "Automatic Text Detection and Tracking in Digital Video," *IEEE Trans. on Image Processing*, Vol. 9, No. 1, pp.147-156, 2000.

[2] Yu Zhong, Hongjiang Zhang, and Anil K. Jain, "Automatic Caption Localization in Compressed Video," *IEEE Trans. on PAMI*, Vol. 22, No. 4, 2000.

[3] Anil. K. Jain, and Bin Yu, "Automatic Text Location in Images and Video Frames," *Pattern Recognition*, Vol. 31, No. 12, pp.2055-2076, 1998.

[4] Axel Wernicle and Rainer Lienhart, "On the Segmentation of Text in Videos," *IEEE International Conference on Multimedia and Expo*, Vol. 3, pp. 1511-1514, 2000.

[5] J. Ohya, A. Shio, and S. Akamatsu, "Recognizing Characters in Scene Images," *IEEE Trans. on PAMI*, Vol. 16, pp. 214-224, 1994.

[6] K. Y. Jeong, K. Jung, E. Y. Kim, and H. J. Kim, "Neural Network-based Text Location for News Video Indexing," *International Conference of Image Processing*, 1999.

[7] Masakatsu Kourogi, Takeshi Kurata, Junichi Hoshino, and Yoichi Muraoka, "Real-time Image Mosaicing from a Video Sequence," *Proceedings of International Conference of Image processing*, Vol.4, pp.133-137, 1999.

[8] Anil K. Jain and Kalle Karu, "Learning Texture Discrimination Masks," *IEEE Trans. on PAMI*, Vol. 18, No. 2, pp. 195-205, 1996.

[9] K. K. Sung and T. Poggio, Example-based Learning for View-based Human Face Detection, *IEEE Trans. on PAMI*, Vol. 20, No. 1, pp. 39-51, 1998.

[10] Gary R. Bradski and Vadim Pisarevsky, "Intel's Computer Vision Library: Application in Calibration, Stereo, Segmentation, Tracking, Gesture, Face and Object Recognition," *IEEE Conference of Computer Vision and Pattern Recognition*, Vol. 2, pp. 796 –797, 2000.

[11] Dorin Comaniciu and Visvanathan Ramesh, "Robust Detection and Tracking of Human Faces with an Active Camera," *International Workshop on Visual Surveillance*, pp.11-18, 2000.

Acknowledgement

This work was supported by grant No. 1999-2-515-001-5 from the Basic Research Program of the Korea Science & Engineering Foundation and Pattern Recognition and Image Processing Lab., Michigan State University.