Pattern Recognition Letters 30 (2009) 114–122

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

A stroke filter and its application to text localization

Cheolkon Jung [a,*], Qifeng Liu [b], Joongkyu Kim [a,*]

[a] School of Information and Communication Engineering, Sungkyunkwan University, 300 Cheoncheon-dong, Suwon, Kyunggido 440-746, Republic of Korea
[b] Samsung Advanced Institute of Technology, Yongin, Kyunggido 446-712, Republic of Korea

ARTICLE INFO

Article history:
Available online 3 June 2008

Keywords:
A stroke filter
Text localization
Text information extraction

ABSTRACT

Most researchers have used edge, intensity, corner, and texture features for text localization in video images. However, these features do not fully coincide with the features of the text, and can not fulfill all the necessary conditions of the text. Therefore, it is very difficult to localize text robustly in video images which have complex backgrounds with strong edge or texture clutter using these features. In this paper, we propose a stroke filter which can detect strokes of texts for robust text localization. By using this stroke filter, we can remove text candidates which have strong edges but are not text. Furthermore, we apply the stroke filter to our text localization system and localize text more robustly in the video images. The effectiveness and efficiency of the proposed method is verified by extensive experiments on a challenging database containing 480 video images.

1. Introduction

Text in videos carries rich and useful information, which can help a machine to understand their content. Therefore, text information is used in the fields of automatic annotation, indexing, summarization, and searching of videos (Dimitrova et al., 2002; Wang et al., 2000; Lyu et al., 2005; Chen et al., 2004; Ye et al., 2005; Kim et al., 2001). The extraction of text information is very important, because text contains high-level semantic information. In general, the extraction of text information from videos involves three major steps: text localization, text segmentation, and text recognition (Jung et al., 2004). Text localization is used to localize the text lines using rectangular boxes. Text segmentation is performed to compute the foreground of the text. The text recognition is conducted to convert the segmented text image into plain text. In this paper, we focus on text localization in images.

Up to the present, many great achievements in text localization have been made by researchers. The existing methods of text localization can be briefly classified into four categories: (1) intensity–similarity connected component analysis (CCA)-based method (Jain and Yu, 1998), (2) edge-based method (Lyu et al., 2005), (3) corner-based method (Hua et al., 2001), and (4) texture-based method (Wu et al., 1997). More recently, many researchers have focused on the application of pattern classification to text localization based on elaborately selected features (Zheng et al., 2004; Chen et al., 2004; Kim et al., 2001, 2003; Chen and Yuille, 2004; Ye et al., 2005).

However, there are still some problems with the existing methods of text localization. Video images often have complex backgrounds with strong edge or texture clutter, and it is very difficult to detect the graphic or scene text with high accuracy. For example, Fig. 1b shows the Canny edge image corresponding to the image in Fig. 1a. The text is located within the red dotted circle in Fig. 1a. It is difficult even for the human eye to find the text in Fig. 1b, because there is a lot of edge clutter around it. We believe that the fundamental reason for this difficulty is that these conventional features do not fully coincide with the distinctive features of the text. That is to say, the intensity–similarity CCA, edge, corner or texture features constitute only a small part of characteristics of the text in Fig. 1c.

Although recently some researchers have used Adaboost to select the optimal features (Chen and Yuille, 2004), the selected features are still a combination of the traditional features. Therefore, we attempt to take the intrinsic characteristics of the text into account on the basis of observation of text. As shown in Fig. 2, the text in video images consists of one or several strokes. We define that a sub-image is text, if and only if:

(1) local constraint – there are many stroke-like structures in the sub-image,
(2) global constraint – these stroke-like structures have specific spatial distributions

where a stroke is defined as a straight line or arc used as a segment of a character.

According to the local constraint in the text definition, we design a stroke filter which is based on local region analysis, and according to the global constraint, we use spatial-similarity CCA to localize the text candidates.

* Corresponding authors. Tel.: +82 31 290 7199; fax: +82 31 290 7687.
E-mail addresses: ckjung@ece.skku.ac.kr (C. Jung), qfliu1976@yahoo.com.cn (Q. Liu), jkkim@skku.edu (J.K. Kim).

0167-8655/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2008.05.014

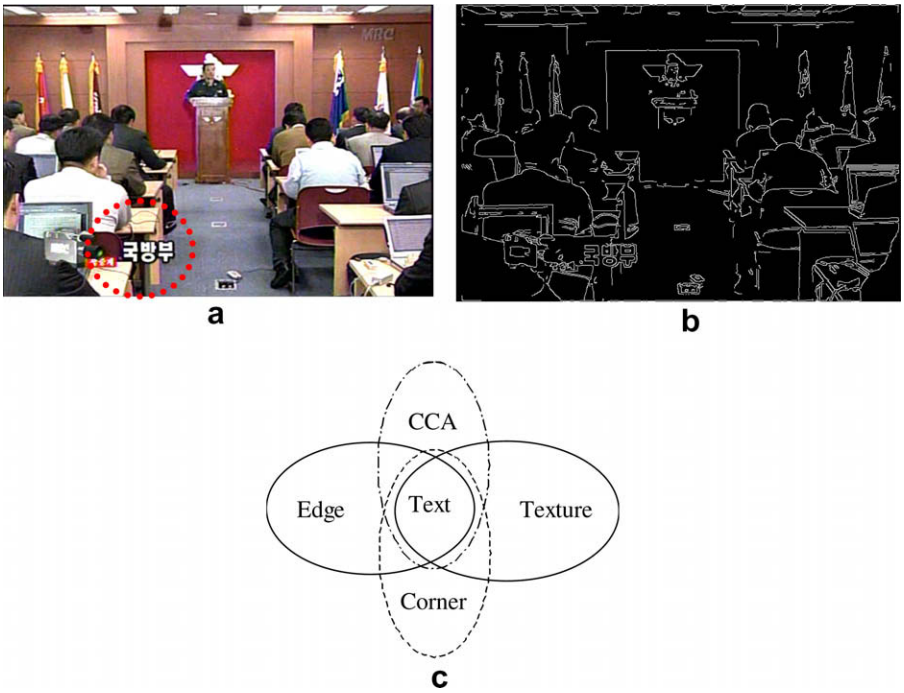

Fig. 1. Illustration of problems in text localization. (a) Original image. (b) Canny edge image. (c) Relationship of features.

Fig. 2. Text image.

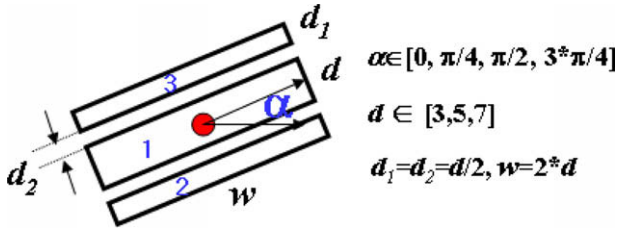

Fig. 3. A stroke filter.

The remainder of this paper is organized as follows. In Section 2, we describe the stroke filter in detail. In Section 3, we compare the stroke filter with the other related filters. Our text localization method using the stroke filter is described in Section 4. Finally, we describe our experimental results in Section 5 and conclude this paper in Section 6.

2. A stroke filter

Strokes of a pen or brush are the movements or marks that we make with it when writing or painting. The stroke filter which can detect the strokes in video images is designed as follows. First, we define a local image region as a stroke-like structure, if and only if, in terms of its intensities,

(1) it is different from its lateral regions, and
(2) its lateral regions are similar to each other, and
(3) it is nearly homogenous.

For each pixel in the source image, we compute its stroke filter response. As shown in Fig. 3, the central point denotes an image pixel (x,y), around which there are three rectangular regions. Let index 1 denote the central region and index 2 and 3 both lateral regions. The orientation and scale of these local regions are determined by α and d, where d, the width of the rectangular region is determined with prior knowledge of text obtained by conducting experiments on text images. There is also the interval, d_2, between the central region and lateral regions in the stroke filter, because dark or bright lines are often embedded around some texts to convey their meanings efficiently, and blurred edges appear around texts as a result of their compression, as shown in Fig. 2. According to the definition of a stroke-like structure, we define the stroke filter response of the pixel (x,y) as

$$R_{\alpha,d}(x,y) = \frac{|\mu_1 - \mu_2| + |\mu_1 - \mu_3| - |\mu_2 - \mu_3|}{\sigma}. \quad (1)$$

The terms on the right-hand side of (1) have clear physical meanings corresponding to the constraints of the definition of a stroke-like structure, where μ_i is the estimated mean of the intensities in the region i. The response is in proportion to $|\mu_1 - \mu_2| + |\mu_1 - \mu_3|$ and inverse proportion to $|\mu_2 - \mu_3|$. σ is the standard deviation of the intensities in the region 1 and a measure of how spread out the intensities of the region are. Therefore, this measure can reflect how homogeneous the stroke is, and be in inverse proportion to the response. The more likely it is that the pixel (x,y) belongs to a stroke-like structure, the higher its response is. To derive the profiles of the stroke filter, Eq. (1) is rewritten in the form of the bright stroke response R^B:

$$\begin{aligned} R^B_{\alpha,d}(x,y) &= \begin{cases} \frac{2\mu_1 - 2\mu_2}{\sigma} & \text{if } (\mu_2 > \mu_3), \\ \frac{2\mu_1 - 2\mu_3}{\sigma} & \text{if } (\mu_2 > \mu_3), \end{cases} \quad \text{if} (\mu_1 > \mu_2 \ \& \ \mu_1 > \mu_3) \\ &= \frac{2\mu_1 - 2\max(\mu_2, \mu_3)}{\sigma} \quad \text{if } (\mu_1 > \mu_2 \ \& \ \mu_1 > \mu_3) \end{aligned} \quad (2)$$

and the dark stroke response R^{D}:

$$R^{D}_{\alpha,d}(x,y) = \begin{cases} \frac{2\mu_3-2\mu_1}{\sigma} & \text{if } (\mu_2 > \mu_3), \\ \frac{2\mu_2-2\mu_1}{\sigma} & \text{if } (\mu_2 > \mu_3), \end{cases} \quad if(\mu_1 > \mu_2 \ \& \ \mu_1 > \mu_3) = \frac{2\min(\mu_2,\mu_3)-2\mu_1}{\sigma} \quad \text{if } (\mu_1 > \mu_2 \ \& \ \mu_1 > \mu_3). \tag{3}$$

Note that: (a) $R_{\alpha,d}(x,y) = 0$ if $\mu_2 \leqslant \mu_1 \leqslant \mu_3$ or $\mu_3 \leqslant \mu_1 \leqslant \mu_2$; (b) $R^{B}_{\alpha,d}(x,y)$ ($R^{D}_{\alpha,d}(x,y)$) is high when the pixel (x,y) is one of the strokes with a bright (dark) color. Therefore, the profiles of the bright and dark stroke filters are shown in Fig. 4a and b, respectively. In these figures, the number is the index of each region in the stroke filter (Fig. 3) and the plus/minus signs indicate positive/negative effects on the response.

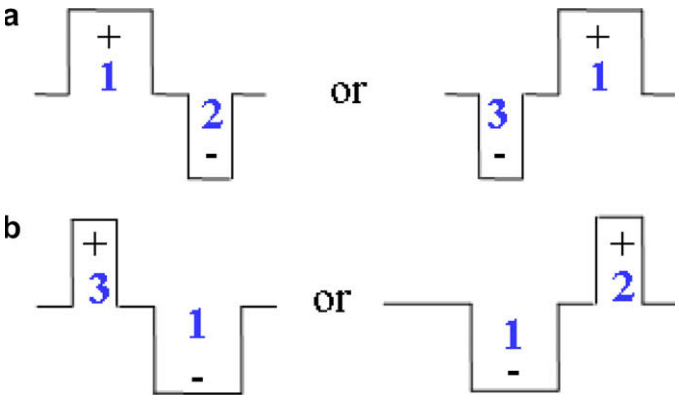

Fig. 4. The profiles of stroke filters. (a) A bright stroke filter. (b) A dark stroke filter.

By stroke filtering, we extract the stoke features (R^{B}, O^{B}, S^{B}, R^{D}, O^{D}, S^{D}) of any pixel (x,y) as follows (note that (R^{D}, O^{D}, S^{D}) have similar expressions):

$$\begin{cases} R^{B}(x,y) = \max\limits_{(\alpha,d)} R^{B}_{\alpha,d}(x,y), \\ O^{B}(x,y) = \operatorname{argmax}\limits_{(\alpha)} R^{B}_{\alpha,d}(x,y), \\ S^{B}(x,y) = \operatorname{argmax}\limits_{(d)} R^{B}_{\alpha,d}(x,y), \end{cases} \tag{4}$$

where R, O and S are the response, orientation and scale of the stroke filter, and B and D are the bright and dark stroke filter, respectively.

Another very important factor for a practical system is the processing speed. One of the advantages of the stroke filter is that it enables us to design a smart scheme of fast implementation without loss of precision. We use three strategies:

(1) *E*dge mask – since text must have rich edges, we only perform stroke filtering on edge pixels and its neighbors. The filter mask is the dilation of Canny edge map of source image.
(2) *Q*uick filling – once we detect a pixel with strong response, it and its neighbors are set as the same response, orientation and scale. This is based on the local spatial consistency of text features.
(3) *I*ntegral image – this has been used for quick Haar-like filtering by (Lienhart and Maydt, 2002).

On one typical video image (720 × 480) as shown in Fig. 5a, we tested the stroke filter with different combination of above three strategies to compare their computation costs, and the results are shown in Table 1, where "*E*", "*Q*" and "*I*" denote the above three strategies, respectively. We find that the final computation cost of a quick stroke filter is only 2.8% of the original one.

Fig. 5. Features extracted by a stroke filter in a real text image. (a) Original image. (b) Text sub-image. (c) Stroke response map. (d) Binarization of (c). (e) Stroke width map. (f) Stroke orientation angle map.

Table 1
Comparison of computation costs (one 3.0 GHz CPU)

Module	Time (ms)			
	Original	E	$E+Q$	$E+Q+I$
Filtering	1884	324	302	34
Canny	0	7	7	7
Dilation	0	2	2	2
Integral	0	0	0	10
Total	1884	333	311	**53**

Table 2
Comparison between related filters

Filter	$\mathbf{I}_{cl}$	$\mathbf{S}_{cl}$	$\mathbf{S}_{ll}$	$\mathbf{H}_c$	$\mathbf{R}_s$
Canny	×	√	×	×	5
Gabor	√	√	×	×	1
Haar	×	√	×	×	4
Ratio	√	√	×	×	3
Stroke	√	√	√	√	4

The √ (×) mark means the corresponding factor is (is not) considered.

Some intermediate results in a real text image are shown in Fig. 5. Fig. 5b shows the text sub-image corresponding to the original image in Fig. 5a, where the text located within the red dotted circle in Fig. 5a and c shows the stroke response map which is made by combining the bright and dark stroke responses, and Fig. 5d shows the binarized map of Fig. 5c. Fig. 5e and f show the stroke width and angle map, respectively. The bright regions in Fig. 5e and f indicate the width and orientation angle with large scales, respectively. It can be observed that a stroke filter can remove most step-like edges, and at the same time, the text regions are enhanced well, as shown in Fig. 5d. Some non-text regions between two strokes are also detected, because all dark and bright strokes are filtered. However, they do not affect the detection of the text candidates, because they will increase the response of the stroke filter.

3. Comparison with other filters

There have been some related filters which could be used for text localization, such as Canny edge filter (Canny, 1986), Gabor filter (Chen et al., 2001), Haar-like line filter (Lienhart and Maydt, 2002) and ratio edge filter (Tupin et al., 1998). In some respects, the stroke filter may be similar to these filters, since all of them are based on local image difference between the central and lateral regions. However, the stroke filter is distinctively different from them, as summarized in Table 2. Note that

(1) I_{cl} denotes the interval between central and lateral regions, which is marked as d_2 in the stroke filter. As mentioned above, the interval is needed because of the embedded dark or bright lines and blurred edges placed around the texts.
(2) S_{cl} denotes the difference between the central and lateral regions, which is evaluated by $|\mu_1 - \mu_2| + |\mu_1 - \mu_3|$ in a stroke filter. For text strokes, S_{cl} should be large.
(3) S_{ll} denotes the similarity between two lateral regions, which is evaluated by $|\mu_2 - \mu_3|$ in a stroke filter. For text strokes, S_{ll} should be small.
(4) H_c denotes the homogeneity of the central region, which is measured by σ in a stroke filter. For text strokes, H_c should be small.
(5) R_s denotes the speed, where 1–5 denote very slow, slow, medium, fast and very fast, respectively.

In addition, we compare the five filters on a synthetic image containing various kinds of edge, such as step edges, strokes (i.e., the left four vertical bright and dark lines), bright edges and dark edges, as shown in Fig. 5a. The responses of the canny edge, Garbor, Haar-like line, ratio edge and stroke filters on the synthetic image are shown in Fig. 6b–f, respectively. From Fig. 6f, we can see that the stroke response of the bright and dark edges on the right side is weaker than that of the other filters, because its lateral regions are not similar. However, the stroke response of the four vertical lines on the left side is more prominent than that of the other filters. This means that the stroke filter is more sensitive to strokes

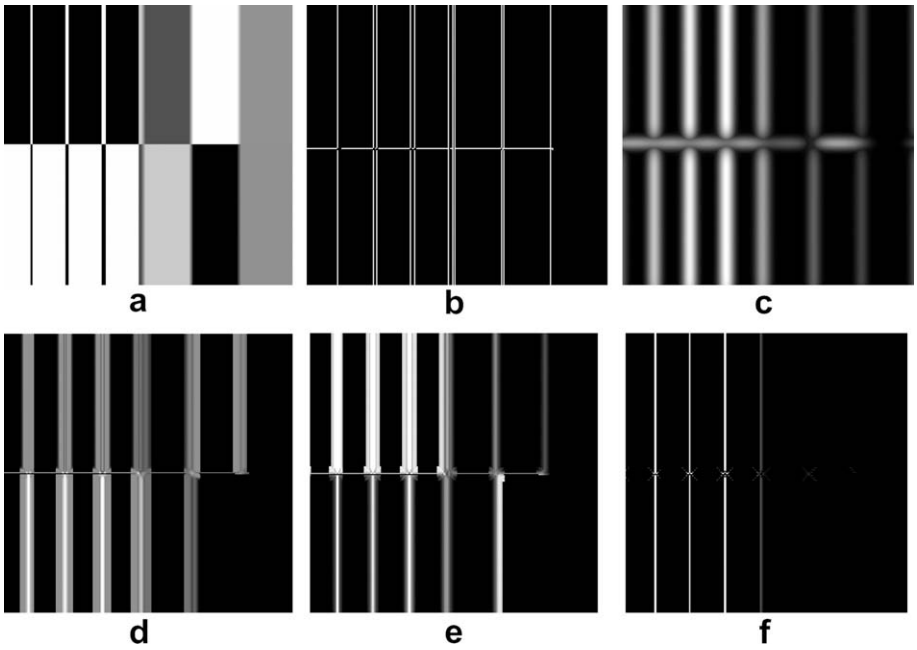

Fig. 6. Comparison of responses of some related filters. (a) Source image. (b) Canny edge filter. (c) Gabor filter. (d) Haar-like line filter. (e) Ratio edge filter. (f) Stroke filter.

than the other filters. Also, as shown in Table 2, the four factors, I_{cl}, S_{cl}, S_{ll} and H_c, are considered in the stroke filter, which means that it outperforms the other filters in stroke detection. In summary, compared with the other filters, the proposed stroke filter is the most suitable especially for stroke detection.

4. The stroke filter-based text localization system

The overall flowchart of our text localization method using the proposed stroke filter is shown in Fig. 7. We use a cascade scheme for the robust and accurate localization of text. Our method consists of three modules: text candidate detection, text verification, and text line refinement. The main difference between our method and the other text localization methods using pattern recognition (Kim et al., 2001, 2003; Chen et al., 2004; Ye et al., 2005) lies in the fact that we use this stroke filter in the text candidate detection module, and add a much more elaborate module of text line refinement based on the SVM output score.

4.1. Text candidate detection by a stroke filter

In the text candidate detection module, we first compute the strokes by the proposed stroke filter, and connect them into clusters by means of a morphologic filter. The morphologic operation is performed to generate the binary stroke map. Then, we compute the connected components (CCs) in the stroke map resulting from the morphologic operation. We analyze each CC by its *Y*-(horizontal) and *X*-(vertical) axis projection profiles. In this CCA, the connectivity of the CC value is specified as 4, but not 8, because we try to avoid over-connection. In this module, our previous method (Jung et al., submitted for publication) uses a Canny filter, but the method described in this paper uses the proposed stroke filter. By using the stroke filter, the edge clutter of Fig. 1b can be removed. Representative results are shown in Fig. 8. Fig. 8a and b are the dilation maps produced by the Canny filter and the stroke filter, respectively. Most of the edge clutter in Fig. 8a is removed by the stroke filter, as shown in Fig. 8b. Fig. 8c and d are the localization results by the Canny filter and the stroke filter, respectively. The text is located at the red dotted circle, and localized in a blue box. The text is not localized by the Canny filter, as shown in Fig. 8c, but is localized robustly in Fig. 8d, because the edge clutter is removed by the stroke filter in this text candidate detection module.

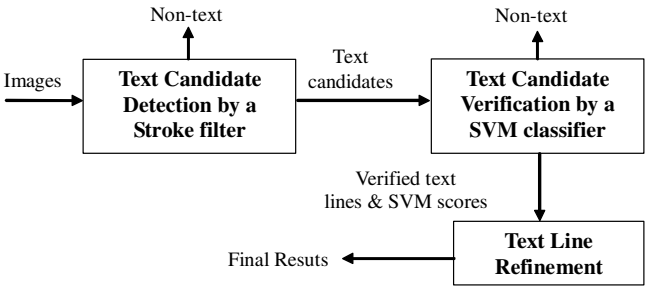

Fig. 7. Cascade framework of the proposed text localization method.

4.2. Text candidate verification by an SVM classifier

Compared with other classifiers such as neural network and decision tree, support vector machine (SVM) is easier to train and has better generalization ability (Chen et al., 2004; Ye et al., 2005). Therefore, we use the SVM classifier and choose radial basis function (RBF) as kernel function of this SVM classifier. The basic idea of SVM is to implicitly project the input space onto a higher dimensional space where the two classes are more linearly separable. An extensive discussion of SVM can be found in Vapnick's work (1995). Given the text candidate produced by the text candidate detection module, we first normalize it to a height of 15 pixels height and corresponding resized weight. Then, we use a sliding window with 15×15 pixels to generate several samples. We use the trained SVM classifiers to classify each of these samples and fuse the results. The SVM output scores of theses samples are averaged. If the average is larger that a predefined threshold, then the whole text line is regarded as a text line, otherwise it is regarded as a false alarm. We set the threshold value as -0.3 based on the trade-off between the false acceptance rate (FAR) and false rejection rate (FRR).

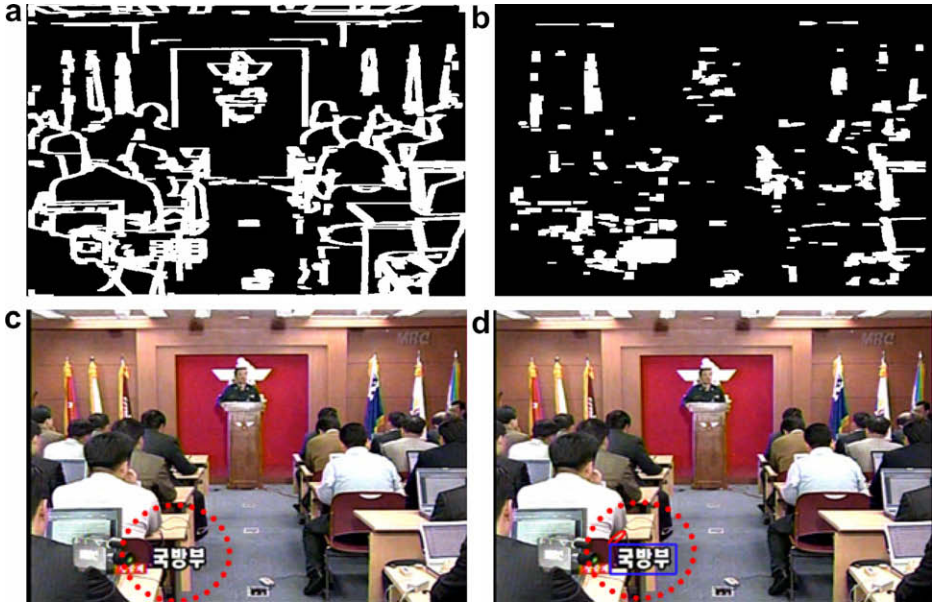

Fig. 8. Some representative results. (a) The dilation map produced by the Canny filter. (b) The dilation map produced by the stroke filter. (c) Text localization result produced by the Canny filter. (d) Text localization result produced by the stroke filter.

We use normalized the gray intensity (Gray) and constant gradient vector (CGV) as the features for text candidate verification. The gray intensity is shown in Kim et al.'s work (2003) to be the best feature compared with the wavelet and histogram features. To deal with cases with low contrast, we use the normalized gray intensity (N-Gray), which is defined as

$$Nf(s) = \frac{f(s) - V_{\min}}{V_{\max} - V_{\min}} \times 255, \quad (5)$$

where $f(s)$ is the original gray intensity of pixel s, which is normalized into $Nf(s)$ from range $[V_{\min}, V_{\max}]$ to $[0,255]$. For classifier fusion, we choose the CGV as the second feature, since it was reported in Chen et al.'s work to be superior to gray spatial derivatives, distmap, discrete cosine transformation (DCT) coefficients, etc. In order to improve the classification performance, we fuse the N-Gray and CGV features together. Since the SVM output score is the sum of weighted kernel distance between the test samples and support vectors, it is reasonable to fuse these two features in the following way:

$$f_0 = P_1 \cdot f_1 + P_2 \cdot f_2, \quad (6)$$

where f_0 is the final SVM score, P_1 and P_2 are the prediction accuracies, and f_1 and f_2 are the SVM scores of the two features, respectively.

To handle large variation (from 10 to 150 pixels) in the text size, we use the method of pyramid decomposition and composition, in which the source image is decomposed into $N(=3)$ levels with decreasing size. For each pyramid level, we perform text candidate detection and verification with the same parameters. Then, these results from the different levels are grouped together. The main problem occurs when the same text line is detected together in two or three levels as shown in Fig. 9. To accomplish this, we perform pyramid composition based on the SVM output score. The result with the highest SVM output score is selected. This method is reasonable because the highest SVM output score means the best localization of text.

4.3. Text line refinement

However, there are some problems in the initial text localization results obtained by the text candidate verification module. Most of the results are good, but some of the text line boundaries are not accurate. For example: (a) some text candidates contain too much background, (b) some of the text is missed by the text candidate detection module, and (c) some characters are intersected by the edges of two text line boxes, as shown in Fig. 10. These problems will have a detrimental effect on the recognition results. Therefore, the initial localization text lines should be refined in order to solve these three problems. We make use of the SVM output score and color distribution for the text line refinement. This output score of the SVM can provide a quantitative measure of the closeness to the text. By using this score, we can refine the initial

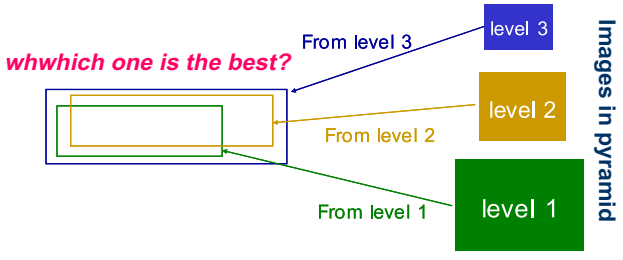

Fig. 9. Problem of pyramid decomposition.

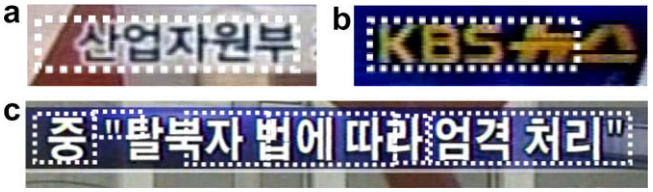

Fig. 10. Problems of initial localization results. (a) Included background. (b) A missed text. (c) Text box fractions.

localized text lines. The text line refinement module consists of text boundary shrinking, combination and extension functions.

In the text boundary shrinking function, we get rid of the non-text parts on the right and left sides of the initial text line based on the SVM output score. In the text candidate verification module, the SVM output scores have been computed in advance. The initial text line shrinks to the center position of the text region if the score is smaller than the threshold.

The initial text localization results contain many text fractions, which should be combined together as shown in Fig. 10c. For the purpose of boundary combination of these text fractions, we can avoid accepting non-text image parts falsely by using the SVM output scores. We choose two of the initial localized text lines as a pair, which will be combine together if: (1) they are near each other or (2) they are not near each other, but the image part between them is accepted by the SVM. We check for any pairs of the initial text lines.

In order to the missed text parts, we perform the text boundary extension function based on the SVM output score and image similarity measurement, which are combined together to improve the performance. We estimate the color probability density functions (PDFs) of the initial localized text lines and the outer sub-image. If the similarity between two PDFs is high and the sub-image is accepted by the SVM, the initial text line is extended to include the outer sub-image. The detailed description of the text line refinement module is provided in our previous work (Jung et al., submitted for publication). In addition, some representative results of these three functions are shown in Fig. 11.

5. Experiments

We conducted experiments with a PC having one CPU (Intel Pentium 3.0 GHz) with Window XP using VC++6.0. In our ground truth database, there are news images from South Korea TV channels (KBS, MBC and SBS), whose sizes are 720×480 pixels (the text height varies from 10 to 150 pixels). In these images, 435 images are used for testing and 357 for training. There are 3965 positive and 3453 negative samples for training and 7437 positive and 8000 negative samples for testing. We also use Microsoft common database containing 45 images. All of the video frames in this database have a size of 352×288 pixels. These database consist of a variety of cases, including text in different font-size, font-color, and languages, text on textured background, text of poor quality, etc.

The user interface of the proposed text localization method is illustrated in Fig. 12. The top-left window in this figure shows the results of the text localization process. The red numbers over the blue boxes are the order of each text localization result. The bottom-left window in this figure is an intermediate result which shows the SVM output score of each red box in the text localization results. The right window of this figure shows the results of the performance evaluation on the test database.

Fig. 13 illustrates some representative results of the proposed stroke filter-based text localization method. In Fig. 13a–l, the black boxes indicate the localized text lines produced by the proposed

Fig. 11. Representative results of the text line refinement. Left: initial localized text line. Right: refined text line. (a) Text boundary shrinking function. (b) Text boundary combination function. (c) Text boundary extension function.

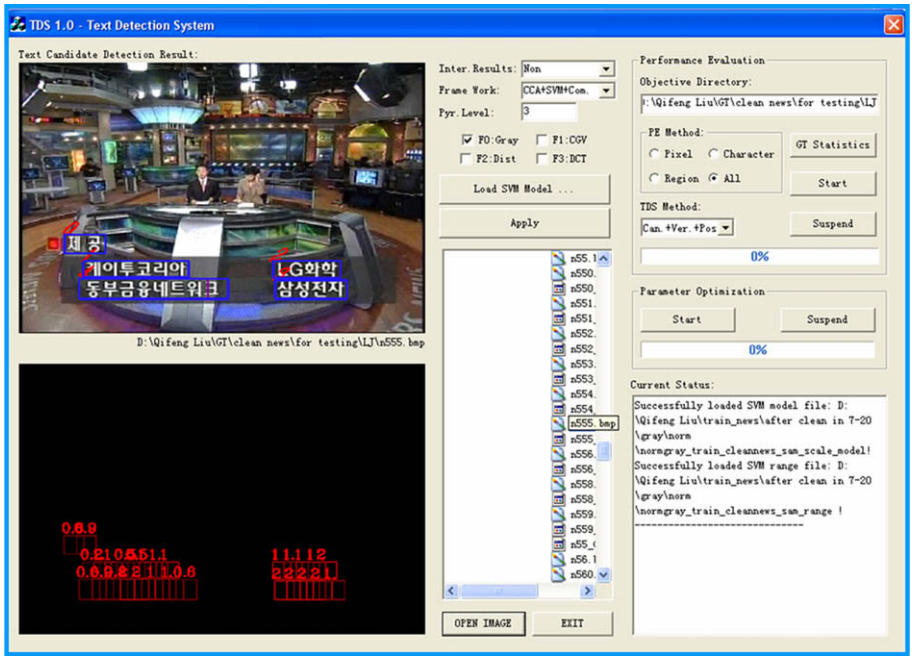

Fig. 12. User interface of the proposed text localization method.

method. Most of the texts are localized robustly and accurately despite the presence of large and small font-sizes, colors, and languages. However, there are also some incorrect results. Some of the text lines are extended excessively beyond the text regions in Fig. 13i and j by the text line refinement module, because the extended regions are similar to the text regions. In Fig. 13k, the windows of the building are prone to be falsely detected as text lines. We predict that, in a future work, these falsely detected results will be able to be removed by making use of the feedback from the OCR (optical character recognition) results.

We compared the proposed method, "Stroke filter + CCA", with the previous method, "Canny filter + CCA" (Jung et al., submitted for publication), and the results are shown in Table 3. Note that for detecting large text with a height of 150 pixels, we perform the same algorithm on 3 pyramid levels of the source image.

The performance is measured on the text line level. We define that a detected text line is correct on the condition that the intersection of the detected text line (DTL) and the ground truth text line (GTL) covers more than 90% of the DTL and 90% of the GTL (Lyu et al., 2005). We use a tool named making database for text detection and segmentation (MDTDS), implemented by ourselves to semi-automatically collect the GTLs in the test images (Jung et al., submitted for publication). Thus, the recall rate, precision rate, and time cost are defined as

(1) Recall rate = (number of correct DTLs)/(number of GTLs);
(2) Precision rate = (number of correct DTLs)/(number of all DTLs);
(3) Time cost (s/frame) = (total processing time)/(number of frames).

The recall and precision rates of the stroke filter-based method for text candidate detection are about 2.7% and 1.7% higher than those of the Canny filter-based method, respectively. Thus, 2.7% of the text candidates that are missed are recalled by the proposed stroke filter. The reason why these text candidates are recalled is because the edge clutter around the texts in video images is removed by the stroke filter. However, the processing time is about 0.04 s higher per frame in the case of the method with the stroke filter.

Therefore, the stroke filter-based method is more accurate, but more computationally expensive than the Canny filter-based method in terms of text candidate detection. However, the advantages of the stroke filter are not limited to its higher accuracy.

Fig. 13. Some experimental results of the stroke filter-based text localization method.

Table 3
Comparison of two methods of text candidate detection

Methods	Recall rate (%)	Precision rate (%)	Time cost (s/frame)
Canny filter + CCA	94.3	40.7	0.065
Stroke filter + CCA	97.0	42.4	0.111

Table 4
Performance comparison between the Canny and stroke filter-based text localization methods

Methods	Recall rate (%)	Precision rate (%)	Time cost (s/frame)
Canny filter-based method	97.2	97.6	0.640
Stroke filter-based method	97.7	97.6	0.686

Actually, because the stroke filter can discover the intrinsic features of text including its response, orientation and scale, as mentioned in Section 2, it is potentially very powerful in subsequent processes, such as SVM verification and even text segmentation. Once the image has been filtered in the text candidate detection module, only a small additional computation cost will be needed in subsequent processes.

Table 4 shows the performance comparison between the Canny filter and stroke filter-based text localization methods. We make use of SVM as a classifier, and N-Gray and CGV as features for text candidate verification. From Table 4, it can be observed that the recall rate of the stroke filter-based method is only 0.5% higher than that of the canny filter-based method. Although the stroke filter-based method enables 2.7% more of the text candidates in the text candidate detection procedure as compared to the Canny filter-based method, there is little difference in the final performance of these two methods. Also, the recall rates of the two methods are slightly higher than those of the text candidate detection module including the stroke filter. This means that the text line refinement module has a significant effect on the final performance of the text localization method. The refinement module in our localization method enables 2.9% more of the texts that are missed by the text candidate detection module to be recalled. Therefore, there is little difference in the final performance of the two methods.

In order to provide an idea about the quality of these results, experiments are performed on Microsoft common database to compare the proposed method with three representative methods of Ye et al. (2005), Li et al. (2000) and Lienhart and Wernicke (2002). From Table 5, it can be observed that our method is faster than Li et al.'s (2000); and Lienhart and Wernicke's (2002) methods, while our method is slower that Ye et al.'s (2005) method.

Table 5
Performance comparison of four methods

Methods	Recall rate (%)	Precision rate (%)	Time cost (s/frame)
The proposed method	94.9	98.8	0.196
Ye et al.'s (2005) method	94.2	97.6	0.064
Li et al.'s (2000) method	91.4	94.4	0.356
Lienhart et al.'s (2002) method	94.2	91.9	0.242

However, it can be observed that both recall and precision rates of our method perform better than those of any other methods. This shows that our text localization method-based on the stroke filter achieves considerably better accuracy and robustness as compared to the conventional methods.

6. Conclusions

In this paper, we describe the proposed stroke filter and its application to text localization in video images. The benefit of the stroke filter is that it discovers the intrinsic characteristics of the text by making use of the relationship between the local regions. Therefore, it is more suitable for text localization than the other filters. The proposed stroke filter is applied to a real text localization system. The experiments demonstrate the good performance of the stroke filter for text candidate detection. The proposed stroke filter allows 2.7% more of the text candidates that are missed to be recalled in the text candidate detection. The reason why these missed candidates are recalled is because the edge clutter around the texts in video images is removed by the stroke filter. However, the stroke filter has little effect on the final performance of the text localization operation. This is due to the fact that the performance is improved by the text line refinement module based on the SVM output score. The refinement module in our localization method allows 2.9% more of the texts that are missed by the text candidate detection module to be recalled. Thus, if the best performance of text localization is needed at the cost of the computation time, the stroke filter can be used for text localization. The experimental results and performance comparisons with other methods are reported in detail, thereby confirming that our method is capable of robustly localizing texts in video images by using the stroke filter.

In addition, we believe that the stroke filter will play an important role in fields other than text localization. If the additional effort were made, the stroke filter could be applied to text verification using the features of the filter, including the response, orientation and scale. Moreover, it could also be applied to text segmentation, or extended to the fields of line-like structure detection, such as road detection from remote images.

Acknowledgements

An initial version of this paper appeared in the IEEE International Conference on Image Processing (Liu et al., 2006). The authors would like to thank the anonymous IEEE ICIP and Pattern Recognition Letters reviewers for their valuable comments and suggestions. The partial work reported in this paper was conducted while the authors were with the Samsung Advanced Institute of Technology (SAIT).

References

Canny, J.F., 1986. A computational approach to edge detection. IEEE Trans. Pattern Anal. Machine Intell. 8, 679–698.
Chen, X., Yuille, A.L., 2004. Detecting and reading text in natural scenes. In: Proc. IEEE Conf. on CVPR, pp. 366–373.
Chen, D., Shearer, K., Bourlard, H., 2001. Text enhancement with asymmetric filter for video OCR. In: Proc. Internat. Conf. on Image Analysis and Processing, pp. 192–197.
Chen, D., Jean-Marc, O., Herve, B., 2004. Text detection and recognition in images and video frames. Pattern Recognition, 595–608.
Dimitrova, N., Zhang, H.J., Shahraray, B., Sezan, I., Zakhor, A., Huang, T., 2002. Applications of video content analysis and retrieval. IEEE Multimedia 9 (3), 43–55.
Hua, X., Chen, X.R., Liu, W., Zhang, H.J., 2001. Automatic location of text in video frames. In: Proc. ACM Multimedia 2001 Workshop: Multimedia Information Retrieval.
Jain, A.K., Yu, B., 1998. Automatic text location in images and video frames. Pattern Recognition 31 (12), 2055–2076.
Jung, C., Liu, Q., Kim, J.K., submitted for publication. Accurate text localization in images based on SVM output score. Image and Vision Comput.
Jung, K., Kim, K.I., Jain, A.K., 2004. Text information extraction in images and video: A survey. Pattern Recognition 37 (5), 977–997.
Kim, K.I., Jung, K., Park, S.H., Kim, H.J., 2001. Support vector machine-based text detection in digital video. Patten Recognition 34, 527–529.
Kim, K., Jung, K., Kim, J.H., 2003. Texture-based approach for text detection in image using support vector machines and continuously adaptive mean shift algorithm. IEEE Trans. Pattern Anal. Machine Intell. 25 (12), 1631–1639.
Lienhart, R., Maydt, J., 2002. An extended set of Haar-like features for rapid object detection. In: Proc. IEEE Conf. Image Processing, pp. 900–903.
Lienhart, R., Wernicke, A., 2002. Localizing and segmenting text in images and videos. IEEE Trans. CSVT 12, 1224–1229.
Li, H., Doermann, D., Kia, O., 2000. Automatic text detection and tracking in digital video. IEEE Trans. Image Process. 9, 147–156.
Liu, Q., Jung, C., Kim, S., Moon, Y., Kim, J., 2006. Stroke filter for text localization in video images. In: Proc. IEEE Conf. Image Processing, pp. 1473–1476.
Lyu, M., Song, J., Cai, M., 2005. A comprehensive method for multilingual video text detection, localization, and extraction. IEEE Trans. CSVT 15 (2), 243–255.
Tupin, F., Maitre, H., Mangin, J.F., Nicolas, J.M., Pechersky, E., 1998. Detection of linear features in SAR images: Application to road network extraction. IEEE Trans. GeoScience Remote Sensing 36, 434–453.
Vapnick, V., 1995. The Nature of Statistical Learning Theory. Springer.
Wang, Y., Liu, Y., Huang, J.C., 2000. Multimedia content analysis using both audio and visual clues. IEEE Signal Process. Mag. 17 (6), 12–36.
Wu, V., Manmatha, R., Riseman, E.M., 1997. TextFinder: An automatic system to detect and recognize text in images. IEEE Trans. PAMI 21 (11), 1224–1229.
Ye, Q., Huang, Q., Gao, W., Zhao, D., 2005. Fast and robust text detection in images and video frames. Image Vision Comput. 23, 565–576.
Zheng, Y., Li, H., Doermann, D., 2004. Machine printed text and handwriting identification in noisy document images. IEEE Trans. PAMI 26 (3), 337–353.