A Robust Color-independent Text Detection Method from Complex Videos

Yan Zhao[1], Tong Lu[1,2]*, Wujun Liao[1]
[1]*State Key Laboratory of Software Novel Technology, Nanjing University, Nanjing 210093*
[2]*Jiangyin Institute of Information Technology of Nanjing University, China*
E-mail: zhiyesashou@163.com, lutong@nju.edu.cn

***Abstract*—Video text carries meaningful contextual information and semantic clues for visual content understanding. In this paper, we propose a novel hybrid algorithm to fast detect video texts even under complex backgrounds. We first use an SVM classifier trained by our new StrOke unIt Connection (SOIC) operator to identify seed stroke units. Stroke shape distributions, instead of color or texture features, are extracted and trained in our method. Then the stroke units are tracked and extended into their surroundings to form text lines, obeying seed stroke geometric constraints. Experimental results show that our approach is color and language independent, and robust to video illuminations.**

Keywords-video text detection*; *stroke distribution*; *SOIC

I. INTRODUCTION

Text detection plays an important role in a wide variety of video tasks, such as video retrieval, abstract extraction, and video content understanding, due to the fact that video texts can provide meaningful contextual and semantic clues. For example, subtitles and headlines in news videos summarize the reports and score texts describe game result in sport videos. Considering the characteristics of regularity and usage, video texts can be classified into overlay video text and natural scene text. The former is superimposed on a video scene for better human understanding, while the latter is naturally part of a natural scene. Video texts are in general highly compact and structured [1, 3]; however, it is still a challenging problem to design an algorithm that performs well in all situations, such as language and color independent, scale and rotation invariant, and especially, robust to distinguish natural scene texts from their complex surroundings.

Shape, color, and texture are the most commonly adopted entrances to detect video texts. Shape tracking approach [2, 4-6] separates text from other elements in a scene by tracking its nearly constant stroke width. In these methods, stroke width is first computed for each pixel with geometric reasoning (e.g., the Stroke Width Transform in [2] and the Stroke Filter in [5]). Neighboring pixels with approximately similar stroke width are then merged to form candidate letters and finally similar letters are grouped into text strings. This approach is color independent; however, considering width is not robust enough to well distinguish text and non-text on pixel level, they face difficulties of deciding proper stroke width to track, especially when dealing with texts having different sizes in the same video frame, and problems of selecting suitable neighboring pixels for merging.

Color-based approach [7-8] assumes video text is composed of a uniform color, distinguished from other elements in a video frame. After color transformation in RGB or HSV space, the known connected component (CC) methods can perform satisfactorily results with known text color and simple background. This approach is attractive for many video applications due to the fact that its computational cost is relatively low. However, it is rarely true that video text consists of a uniform color or only has tiny color variation within it, especially considering the degradations resulted from compression coding and the low contrast between a natural scene text and its complex background [1].

Unlike shape tracking and color-based approaches, texture analysis algorithm uses distributions of wavelet [9], DCT coefficients [10] or intensity variance [11] to detect texts. However, considering the great computational cost of scanning different scales for each video frame, it is difficult to apply in real-time video text detection. Moreover, these algorithms are typically unable to detect sufficient slanted texts [1] and lack adaptability.

Recently, machine learning-based approaches have also been proposed for the detection of text areas with great success. Anthimopoulos et. al. [12] use an SVM classifier to train the features obtained by the Local Binary Pattern-based operator (eLBP) to detect video texts. Li et. al. [13] suggest the use of the mean, second order (variance) and third-order central moments of the LH, HL, and HH component to train a three-layer neural network. However, most of these methods are color or language dependent, and can only take a binary decision of an initial input is text or not, but without the capability to refine it. For a more comprehensive survey of methods for text detection, see [14].

Based on our previous text detection from drawing images [6], we propose a novel hybrid algorithm to fast detect both overlay and scene texts from video frames in this paper. Our method has two stages. In the first stage, each frame is over-segmented into a group of $N \times N$ patches and candidate seed text-like patches are searched using an SVM classifier trained on the features obtained by a new StrOke unIt Connection (SOIC) operator, describing the stroke distributions inside each over-segmented patch. A stroke unit here is defined as a text-like segment inside a patch. Then in the second stage, the results are further refined by tracking and extending the seed stroke units to their surroundings to identify text lines. This stage is relatively efficient due to the fact that text color of each stroke unit has been well extracted inside each seed patch using the SOIC operator in the first stage.

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.83

Our two-stage hybrid method has the following advantages: 1) our SVM classifier result is color independent without knowing a predefined text color, due to that our SOIC operator is extracted from stroke distributions instead of color or texture features; and 2) our method is efficient for complex scene text detection by first identifying seed text-like patches using an SVM classifier and then tracking into their surroundings under seed stroke geometric constraints. Moreover, our method is robust to illuminations and efficient for real-time video analysis or content-oriented video retrieval.

The following sections are organized as follows. Section 2 gives the definition of our SOIC operator. Then we introduce our seed text-like patch identification method in Section 3. Seed patches extending is illustrated in Section 4. Next, Section 5 gives our experimental results and discussions. Finally, Section 6 concludes the method.

II. SOIC OPERATOR DEFINITION

In this section, we introduce our SOIC operator in details. We first convert each video frame f_i into a 256 gray image $f_i^{'}$ and over-segment $f_i^{'}$ into a number of patches with the size of $N \times N$ pixels (N is a scale factor related with video frame resolution). Then we partition each patch into five sub-regions of $r_{f_{ij}^{'}}^1$, $r_{f_{ij}^{'}}^2$, $r_{f_{ij}^{'}}^3$, $r_{f_{ij}^{'}}^4$ and $r_{f_{ij}^{'}}^5$ (see Fig. 1, where N=11, j is the patch index of $f_i^{'}$). $r_{f_{ij}^{'}}^1$, $r_{f_{ij}^{'}}^2$, $r_{f_{ij}^{'}}^3$ and $r_{f_{ij}^{'}}^4$ are used to extract the corner stroke unit distributions inside a patch, while $r_{f_{ij}^{'}}^5$ indicates the internal stroke unit distribution.

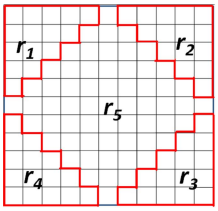


Figure 1. Patch partition with N=11

Next, we use the partitioned sub-regions to extract a 25-dimensional SOIC feature. Suppose given a stroke unit u_x inside a patch $f_{ij}^{'}$ (see Fig. 2), SOIC is used to describe the shape distributions of u_x with two considerations: unit width and sub-regions connectivity. We calculate the unit width distribution vector $\mathbf{v_1}(f_{ij}^{'}, u_x)$ as follows:

$$\mathbf{v_1}(f_{ij}^{'}, u_x)_k = \begin{cases} \sum(p_k \in u_x)/(10 \times N - 50), & k = 1,2,3,4 \\ \sum(p_k \in u_x)/(N^2 - 10 \times N + 50), & k = 5 \end{cases} \quad (1)$$

where p_k is a pixel on u_x in the k th sub-region.

Similarly, the sub-regions connectivity vector $\mathbf{v_2}(f_{ij}^{'})$ can be calculated by

$$\mathbf{v_2}(f_{ij}^{'}, u_x)_{k_1 k_2} = \begin{cases} 0, & r_{f_{ij}^{'}}^{k_1} \text{ is not connected with } r_{f_{ij}^{'}}^{k_2} \text{ by } u_x \\ 1, & r_{f_{ij}^{'}}^{k_1} \text{ is connected with } r_{f_{ij}^{'}}^{k_2} \text{ by } u_x \end{cases} \quad 1 \le k_1, k_2 \le 5 \quad (2)$$

As a result, our SOIC operator is defined by

$$\mathbf{v_{SOIC}}(f_{ij}^{'}, u_x) = \mathbf{v_1}(f_{ij}^{'}, u_x) \cup \mathbf{v_2}(f_{ij}^{'}, u_x) \quad (3)$$

Fig. 2 gives two patch groups, where Fig. 2(a), (b) and (c) are text-like patches and the rests are non-text ones. It can be seen that the SOIC operator are helpful to distinguish text-like patches from non-text ones by calculating their stroke unit distributions among the 5 sub-regions.

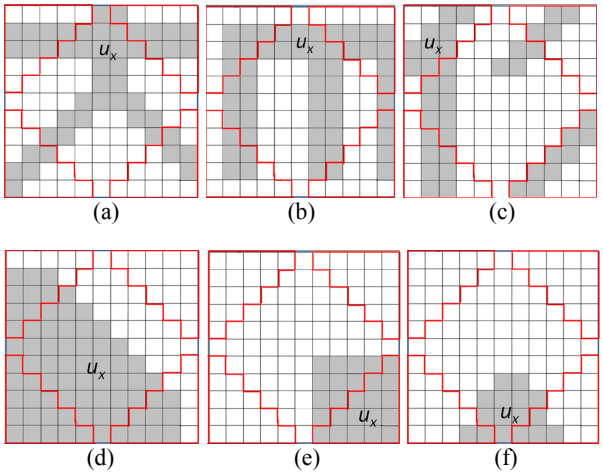


Figure 2. Text-like and non-text patch examples. (a)-(c): text-like patches; (d)-(f): non-text patches.

III. SOIC FEATURE EXTRACTION FOR TRAINING VIDEO FRAMES

In the learning step, our SVM classifier is trained from a group of real-life video frames. The result SVM classifier will be used to distinguish seed text-like patches from their surroundings in Section IV.

To train the SVM classifier, we first label each training patch with "+1" (indicating a text patch) or "-1" (a non-text patch). Instead of simply using pixel color as the existing methods (e.g., [15] and [16]), we hope extract stroke distributions, described by the SOIC operator, as our training feature for color and language independent detection. However, difficulties come from that given a training patch $f_{ij}^{'}$, the label "+1" or "-1" cannot provide enough details to directly distinguish stroke unit gray intervals from their backgrounds. Fig. 3 illustrates an example, where Fig. 3(a) and Fig. 3(b) show a non-text patch and a text patch with their corresponding gray histograms, respectively. It can be

seen that only a proper gray interval has been identified from the gray histogram, can its corresponding SOIC feature be calculated. This task is accomplished as follows.

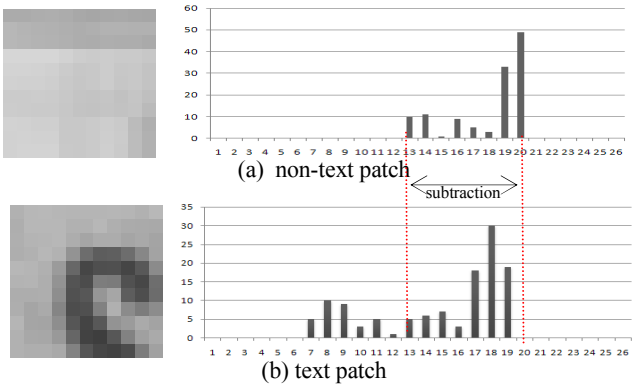


Figure 3. Pixel gray histogram inside a patch

We first identify background gray intervals for each non-text patch $f_{ij}^{'(-1)}$. A non-text patch can be further classified into two categories: background-non-text and foreground-non-text patches. The former has a small gray range and can be directly discarded before SOIC feature extraction, while the latter contains useful information in distinguishing text and non-text patches to train our SVM parameters. The latter can be dealt with by the following steps:

a) Calculate a gray histogram $h(f_{ij}^{'(-1)})$ with M bins (M adopts 26 in our experiments), and find the highest three bins $h_1(f_{ij}^{'(-1)})$, $h_2(f_{ij}^{'(-1)})$ and $h_3(f_{ij}^{'(-1)})$;

b) Create a new stroke unit candidate u_x^{-1} with the three gray intervals from $h_1(f_{ij}^{'(-1)})$, $h_2(f_{ij}^{'(-1)})$ and $h_3(f_{ij}^{'(-1)})$;

c) Calculate $\mathbf{v}_{\mathbf{SOIC}}(f_{ij}^{'(-1)}, u_x^{-1})$ according to (3).

Next, we calculate the stroke unit candidate u_x^{+1} for each "+1" patch $f_{ij}^{'(+1)}$ using the extracted surrounding gray intervals from each $f_{ij}^{'(-1)}$ patch as follows:

a) Visit the 8-neibouring patches of $f_{ij}^{'(+1)}$ to obtain its non-text patch set $S_{f_{ij}^{'(+1)}}^{-1}$;

b) Obtain all the background gray intervals $r'^{s^{-1}} = \sum_{S^{-1}} r'_{f_{ij}^{'(-1)}}$ in $S_{f_{ij}^{'(+1)}}^{-1}$;

c) Remove $r'^{s^{-1}}$ from $S_{f_{ij}^{'(+1)}}^{+1}$, create a new interval $r'^{s^{+1}}$ and get the stroke unit candidate $u_x^{+1} = h_1(r'^{s^{+1}}) \cup h_2(r'^{s^{+1}}) \cup h_3(r'^{s^{+1}})$, where $h_1(r'^{s^{+1}})$, $h_2(r'^{s^{+1}})$ *and* $h_3(r'^{s^{+1}})$ are the highest three bins of $h(r'^{s^{+1}})$;

d) Calculate the corresponding SOIC feature $\mathbf{v}_{\mathbf{SOIC}}(f_{ij}^{'(+1)}, u_x^{+1})$ for u_x^{+1} according to (3).

Finally, we add the calculated $\mathbf{v}_{\mathbf{SOIC}}(f_{ij}^{'(-1)}, u_x^{-1})$ and $\mathbf{v}_{\mathbf{SOIC}}(f_{ij}^{'(+1)}, u_x^{+1})$ into the SVM space to train the classifier parameters. We select the radial basis function as the kernel in our SVM classifier.

IV. TWO-STAGE VIDEO TEXT DETECTION FOR TESTING VIDEO FRAMES

To detect texts from a testing video frame f_i', we adopt a two-stage method as mentioned in Section I: 1) directly identify seed text-like patches from f_i' using the trained SVM classifier parameters, and 2) extend the seed patches into their surroundings using a constrained stroke unit tracking algorithm.

A. Stage 1: Seed text patch identifcation

In this stage, our task is to search seed text-like patches directly from the over-segmented patches of a testing frame. Similarly, how to extract a proper stroke unit is difficult since we have no color priors. Moreover, no "+1" or "-1" labels exist in a testing video frame, increasing the complexities.

Considering there always exists background patches within a small gray range in f_i', we can start from such "pure" background patches to calculate proper gray intervals, due to the fact that most texts have a specific color contrast against their surroundings for human perception. Obeying this idea, we first identify the gray intervals surrounding each "pure" background patch as follows:

a) Search all the patches of f_i' and find "pure" background patches having a gray range less than th_{test} (th_{test} adopts 50 in our experiments);

b) Suppose such a "pure" background patch f_{ij}' having a gray interval r_{ij}' = [$g_{f_{ij}'}^{low}$, $g_{f_{ij}'}^{high}$], remove r_{ij}' from every 8-neibouring patch $f_{(ij),k(0\le k\le 7)}'$ of f_{ij}', that means, let $r_{(ij),k-}' = r_{(ij),k}' - r_{ij}'$;

c) Create a new stroke unit candidate $u_{(ij),k}'$ for $f_{(ij),k}'$ as:

i. $u_{(ij),k}' = h_1(r_{(ij),k-}') \cup h_2(r_{(ij),k-}') \cup h_3(r_{(ij),k-}')$, if $max(r_{(ij),k-}') \le h(f_{(ij),k}')/3$ and $min(r_{ij}') \ge 2 * h(f_{ij}')/3$, or $min(r_{(ij),k-}') \ge 2 * h(f_{(ij),k}')/3$ and $max(r_{ij}') \le h(f_{ij}')/3$;

ii. $u'_{(ij),k} = h_1(r'_{(ij),k-}) \cup h_2(r'_{(ij),k-}) \cup h_3(r'_{(ij),k-})$, if

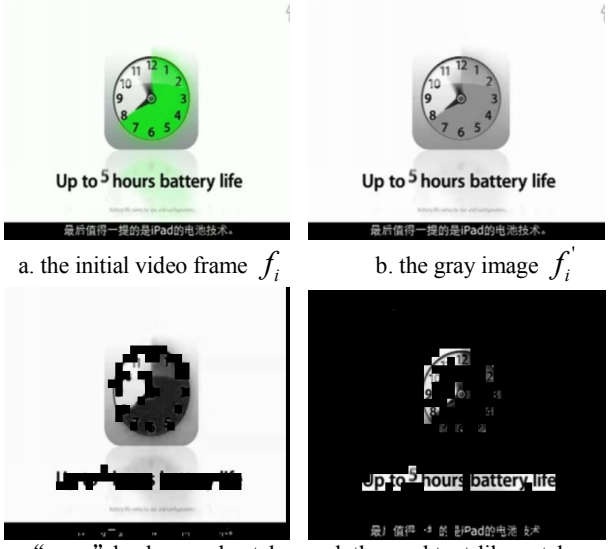
$max(r'_{ij}) \le 2 * h(f'_{ij})/3$ and $min(r'_{ij}) \ge h(f'_{ij})/3$

and $min(r'_{ij}) - max(r'_{(ij),k-}) \ge M/6.5$, or

$max(r'_{ij}) \le 2 * h(f'_{ij})/3$ and $min(r'_{ij}) \ge h(f'_{ij})/3$

and $min(r'_{(ij),k-}) - max(r'_{ij}) \ge M/6.5$.

After identifying the gray intervals inside each $f'_{(ij),k}$, we extract its SOIC feature and use the trained SVM parameters in Section III to further classify $f'_{(ij),k}$ as either a seed text-like candidate or a non-text patch. Fig. 4 shows some examples from real-life testing video frames.



a. the initial video frame f_i b. the gray image f'_i

c. the "*pure*" background patches d. the seed text-like patches

Figure 4. Seed text patches identification using the SVM classifier.

B. *Stage 2: Seed patches extending*

Stage 1 identifies seed text-like patches using the trained SVM classifier. However, these seeds may include either real text patches or background patches. Moreover, some text patches without salient stroke unit distributions as SOIC defined may also be missed during Stage 1.

In stage 2, we use a stroke unit tracking algorithm to further refine the seed patches by extending the units to their surroundings, obeying geometric constraints to form text strokes. In this stage, our method is similar to SWT [2], however, SWT is a global algorithm by calculating and comparing all possible parallel lines for each pixel, while our method is a patch-based method, having the prior seed stroke unit gray intervals obtained in Stage 1. As a result, our method is faster and more robust for complex video frames. Our stroke unit tracking algorithm is as follows:

a) For a seed patch f'_{ij}, suppose a pixel $pix_{u'_{(ij),k}}$ on $u'_{(ij),k}$, calculate the average stroke unit width by

$$W_{f'_{ij}} = \sum_{k=1}^{N} SWT_{pix_{u'_{(ij),k}}} / N \quad (4)$$

where N is the pixel number on $u'_{(ij),k}$;

b) Visit all the 8-neibouring patches $f'_{(ij),k}$ of f'_{ij}, merge them if

$$|W_{f'_{(ij),k}} - W_{f'_{ij}}| < (W_{f'_{(ij),k}} + W_{f'_{ij}})/4 \quad (5)$$

and $u'_{(ij),k}$ is connected with u'_{ij} (see Fig. 5), and other patches are removed as noise patches.

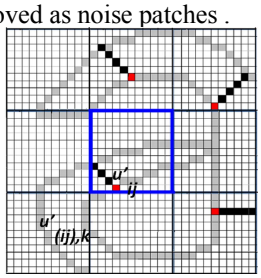


Figure 5 Seed text-like patches extending.

Figure 6 shows an example, where Fig. 6(a) shows a group of seed text-like patches, while Fig. 6(b) gives the extended and refined result.

a) Seed patch examples b) the extended and refined result

Figure 6 Seed text-like patch extending example.

V. EXPERIMENTS AND DISCUSSIONS

In our experiments, we label 100 frames from movie, advertisement and sport videos as our training data, each averagely composed of 15061 non-text patches and 1203 text patches. We then use another 100 frames and the ICDAR database [17] as our testing data. Table 1 shows the results.

TABLE I. TEXT DETECT RESULTS ON ICDAR DATABASE

	Artificial texts	**Scene texts**
Average patches	3047	7391
Average text patches	540	773
Average non-text patches	2507	6618
Correct text patches	440	618
Correct non-text patches	2067	4196
Detection rate (%)	81.4	80.0

We compared our algorithm with [2]. Table. 2 gives the results. It can be seen that the average precision and recall is higher than SWT. Fig. 7 shows two examples for comparison. In [2], text detection is disturbed by illustration variations and no texts can be identified. However, our algorithm

obtains acceptable results, verifying the robustness to illustrations in our method.

TABLE II. COMPARISION OF ALGORITHEMS

Algorithm	Precision	Recall
SWT	0.73	0.6
Our algorithm	0.8	0.62

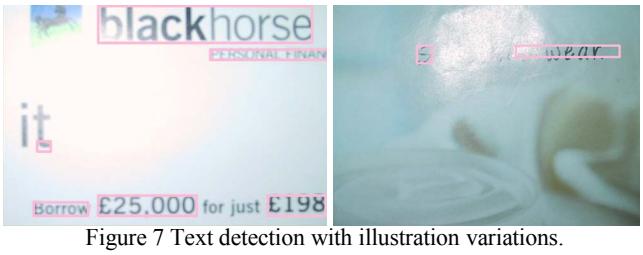


Figure 7 Text detection with illustration variations.

Finally, Fig. 8 gives more text detection results from real-life videos using our method.

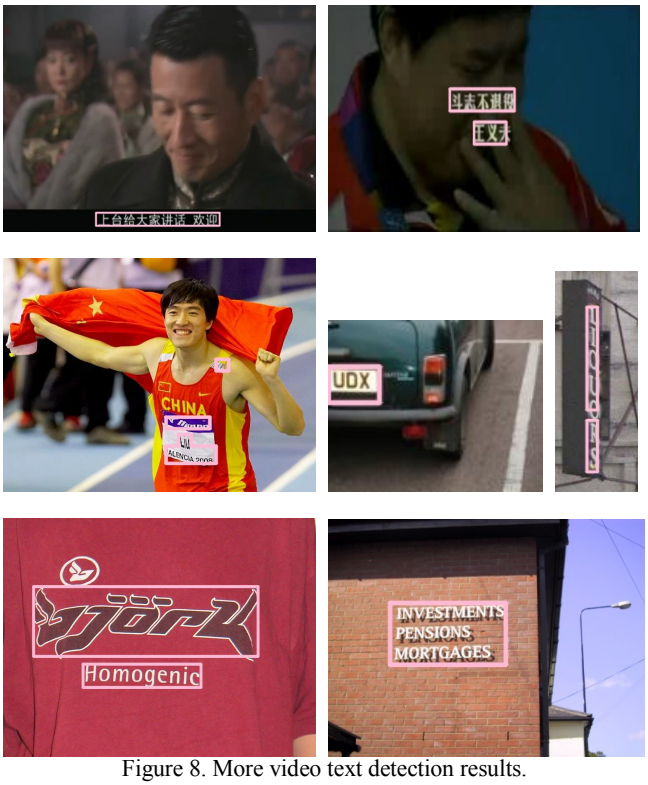


Figure 8. More video text detection results.

VI. CONCLUSION

In this paper, we propose a novel hybrid algorithm to fast detect video texts even under complex background without color or texture priors. We propose a new SOIC operator in over-segmented frame patches to extract their local stroke unit shape distributions. Then we use an SVM classifier to identify seed text-like patches, and further refine the candidates by tracking into their surroundings, obeying geometric constraints to form text strokes. Finally, text lines are identified by merging neighboring patches. Our future work includes further improve the stroke unit extending algorithm, to more accurately discard non-text patches and simultaneously add those patches wrongly discarded by the SVM classifier. Text morphs and statistical text features may be helpful for this target.

Acknowledgement

The work described in this paper was supported by the Natural Science Foundation of China under Grant No. 60723003 and 61003113 and 61021062, the 973 Program of China under Grant No. 2010CB327903, and the Natural Science Foundation of JiangSu under Grant No BK2009082.

References

[1] W. Kim, C. Kim, "A new approach for overlay text detection and extraction from complex video scene," Proc. IEEE Symp. IEEE Transactions on Image Processing, Feb. 2009,18(2), pp.401-411.

[2] B. Epstein, E. Ofek, and Y. Wexler, "Detecting Text in Natural Scenes with Stroke Width Transform," Computer Vision and Pattern Recognition (CVPR'10), Jun. 2010, pp.2963-2970.

[3] M. R.Lyu, J.Q. Song, M. Cai, "A comprehensive Method for Multilingual Video Text Detection, Localization, and Extraction," IEEE Transactions on Circuits and Systems for Video Technology, vol. 15, Feb. 2005, pp.243-255.

[4] K. Subramanian, P. Natarajan, M.Decerbo, D.Castanon, "Character-Stroke Detection for Text-Localization and Extraction," International Conference on Document Analysis and Recognition (ICDAR'07), Sep. 2007, pp. 33-37.

[5] C. Jung, Q. Liu, J. Kim, "A stroke filter and its application for text localization," Pattern Recgontion Letters, Jan. 2009, 30(2), pp. 114-122.

[6] Z.Y. Zhang, T. Lu, F. Su, R.Y. Yang, "A new text detection algorithm for content-oriented line drawing image retrieval," the 2010 Pacific-Rim Conference on Multimedia (PCM'2010), Leture Notes in Computer Science, vol.6297/2010, pp.338-347.

[7] E. Kim, S. H. Lee, J. H. Y Kim, "Scene Text Extraction using Focus of Mobile Camera," International Conference on Document Analysis and Recognition (ICDAR'09), Jul. 2009, pp.166-170.

[8] P. Shivakumara, Q. P. Trung, L. T. Chew, "New Wavelet and Color Features for Text Detection in Vedio," Pattern Recognition(ICPR'10), 2010,pp.3996-3999.

[9] P. Shivakumara, T. Q. Phan, L. T. Chew, "A Robust Wavelet Transform Based Technique for Video Text Detection," International Conference on Document Analysis and Recognition (ICDAR'09), Jul. 2009, pp. 1285-1289.

[10] S. Lu, K.E. Barner, , "Weighted DCT coefficient based text detection," Acoustics, Speech and Signal Processing(ICASSP'08), Mar. 2008, pp.1341-1344.

[11] O. Hori, Res. &Dev.Center, Toshiba Corp., Kawasaki, "A video text extraction method for character recognition," International Conference on Document Analysis and Recognition (ICDAR'99), Sep. 1999,pp.25-28.

[12] M. Anthimopoulos, B. Gatos, I. Pratikakis. "A two-stage scheme for text detection in video images," Image and Vision Computing, vol:28(9), Sep. 2010, pp. 1413-1426.

[13] C. Jung, Q.F. Liu, J. Kim, "Accurate text locatization in images based on SVM output scores," Image and Vision Computing, vol:27, 2009, pp. 1295-1301.

[14] K. Jung, K. W. In Kim, A. K, Jain, "Text information extraction in image and video:a survey," Pattern Recognition, vol: 37(5), May. 2004, pp. 977-996.

[15] Z.G Cheng, Y.C.i Liu, "Caption location and extraction in digital video based on SVM," Machine Learning and Cybernetics, Aug. 2004, pp.3515-3519.

[16] Q.X. Ye, Q.M. Huang, W. Gao, D.B. Zhao, "Fast and Robust text detection in images and video frames," Image and Vision Computing, vol: 23(6), Jun. 2005, pp. 565-576.

[17] http://algoval.essex.ac.uk/icdar/Datasets.html