Using Adaboost to Detect and Segment Characters from Natural Scenes

Kaihua Zhu Feihu Qi
Renjie Jiang Li Xu
Shanghai Jiao Tong University

Masatoshi Kimachi Yue Wu
Tomoyoshi Aizawa
Omron Corporation, JAPAN

Abstract

We present a robust connected-component (CC) based method for automatic detection and segmentation of text in real-scene images. This technique can be applied in robot vision, sign recognition, meeting processing and video indexing. First, a non-linear Niblack method (NLNiblack) is proposed to decompose the image into candidate CCs. Then, we feed all these CCs into a cascade of classifiers trained by Adaboost algorithm. Each classifier in the cascade responds to one feature of the CC. We propose 12 novel features which are insensitive to noise, scale, text orientation and text language. The classifier cascade allows non-text CCs of the image to be quickly discarded while spending more computation on promising text-like CCs. The CCs passing through the cascade are considered as text components and are used to form the segmentation result. We have built a prototype system and the experimental results prove the effectiveness and efficiency of the proposed method.

1. Introduction

Text detection and segmentation from a natural scene is very useful in many applications. With the increasing availability of high performance, low priced, portable digital imaging devices, the application of the scene text recognition is rapidly expanding [1]. By using cameras attached to cellular phones, PDAs, or standalone digital cameras, we can easily capture the text occurrences around us, such as, street signs, advertisements, traffic warnings or restaurant menus. Automatically recognition, translation or enunciation of these texts will be of great help for foreign travelers, visually impaired people and computer programs which perform the video indexing or meeting processing, etc. [1]

Fully automatic text extraction from images, especially from scene images, has always been a challenging problem. The difficulties underlie in variations of scene text in terms of character font, size, orientation, texture, language and color, as well as complex background, uneven illumination, shadows and noise of images (Fig. 1 shows one example). In addition, a high speed of processing is usually desired.

There are growing works focusing on the real scene text detection these years. Current text detection approaches can be classified into two categories.

The first category is the texture based methods. Shin et al. [5] use a star-like pixel mask to expose the intrinsic features of text occurrences. In [6], P. Clark et al. carefully propose 5 localized measures and use a combination of these measures to get candidate text regions. The frequency domain techniques are also used to detect text-like texture, such as: Fourier Transform on short scanning line [8], discrete cosine transform [13], Gabor Transform [4], Wavelet decomposition [2], Multi-resolution edge detector [10]. We find these methods perform quite well on relatively small characters such as text lines on a menu or a document, because smaller texts often possess stronger texture responses. However, for big characters such as road signs or shop names (like Fig. 1), the strong texture response of complex background will mislead these algorithms and leave the big characters undiscovered.

Fig. 1. A difficult natural scene image

The second category is the connected component (CC) based methods. Color quantization [14], Morphological operation [7] and Symmetric Neighborhood Filters [9] are often used to form the candidate CCs. We find these methods can effectively deal with the big characters as well as the small ones, but to choose the exact text CC from the candidate ones often relies on heuristic rules, such as: aspect ratio [7][12][14], aligning-and-merging analysis [14], layout analysis [10], Hierarchical Connected Components Analysis[9]. These rules are often instable and can not guarantee robust detection result.

In this paper, we propose a more stable and more robust CC-based algorithm. This algorithm can enable us to integrate the heuristic rules and features in a more regularized and effective way. Therefore, our algorithm can effectively tackle various difficulties in the natural scene: such as complex background, complex text layout, different text language, uneven illumination, wild variation of text size and orientation.

The framework of our proposed algorithm is showed in Fig. 2. The method is composed of three stages. In the first stage, we employ a novel Non-linear Niblack (NLNiblack) method, which can efficiently and effectively decompose the gray image into candidate CCs. In the second stage, every candidate CC is fed into a series of classifiers and each classifier will test one feature of this CC. If one CC is rejected by any of the classifiers in the cascade, then it is considered as a non-text CC and need no further judgment. In the last stage, the CCs passing through the whole classifier cascade will be processed by a post processing procedure and form the final segmentation result.

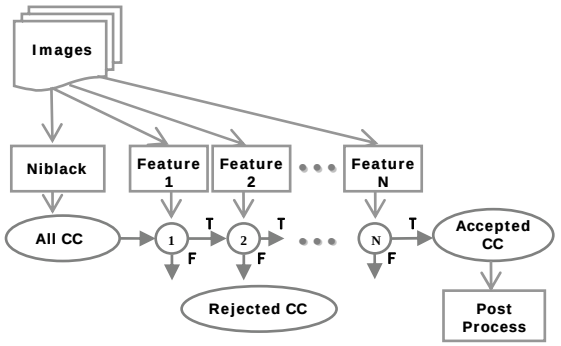


Fig. 2. The text detection algorithm

This framework substantially differs from the existing text detection algorithms in two key points.

The first point is that we utilize a classifier cascade, which can easily discard the majority of the non-text CCs and quickly focus on more promising text CCs. This idea is inspired by face detection technique [15] and is capable of processing images rapidly while achieving high detection rates. However, due to the essential difference between text detection and face detection, we originally propose a specific learning scheme for text detection problem.

The second point is that we propose a series of novel features, each of which has specific contribution to the text detection task. As we will show later, some of the features can take the advantage of texture characteristic of the image, some of them can exploit the spatial coherent information and some of them can efficiently speed up the whole algorithm, etc, By using these features, our algorithm can possess the advantages of both texture based methods and CC based methods, while suppressing their drawbacks. Our work is an innovative attempt to formulate a series of features for text detection.

We developed a prototype system using a mobile phone, Sony Ericsson S700c, attached with a 120 Mega pixel sensor and exhibited this system in Shanghai International Industry Fair 2004. It can automatically detect, segment and translate the English and Japanese signs into Chinese and prove the effectiveness and the efficiency of our algorithm.

The paper is organized as follows. In section 2, we present the non-linear Niblack decomposition method. Section 3 gives twelve features for effectively discriminating the text CCs from non-text ones. Then we describe how to train the classifier cascade using these features in section 4 and we also describe the post process in this section. Issues in system development and the experimental result are discussed in section 0. Section 6 gives the conclusion.

2. Non-linear Niblack decomposition

As we know, decomposing the image into a set of CCs is a very crucial step in CC-based methods. If the decomposition step gets poor results, the performance of the whole algorithm will drop dramatically. There are several existing methods [7][14][9] aiming at effective and robust decomposition. Beside this concern, the efficiency of computation and the low complexity of implementation also concern us. Therefore, we propose a very efficient non-linear Niblack (NLNiblack) thresholding method inspired by [16]:

$$NLNiblack(x,y)=\begin{cases}1 & f(x,y)>T_{+}(x,y)\\ -1 & f(x,y)<T_{-}(x,y)\\ 0 & others\end{cases} \quad (1)$$

$$T_{\pm}(x,y)=\hat{\mu}_{p1}(x,y,W_B)\pm k\cdot\hat{\sigma}_{p2}(x,y,W_F)$$

$$\hat{\mu}_{p1}(x,y,W_B)=Order\left[Mean\left(f(x,y),W_B\right),p1,W_B\right]$$

$$\hat{\sigma}_{p2}(x,y,W_F)=Order\left[Deviation\left(f(x,y),W_F\right),p2,W_F\right]$$

where

k is set to be 0.18 as standard Niblack method.

f(x,y) is the input pixel intensity at position (x,y).

Mean(· , W) is the mean value filter with W width.

Deviation(· , W) is the standard deviation filter with W width.

Order [· , p,W] is the ordered statistics filter with p percentile and W width.

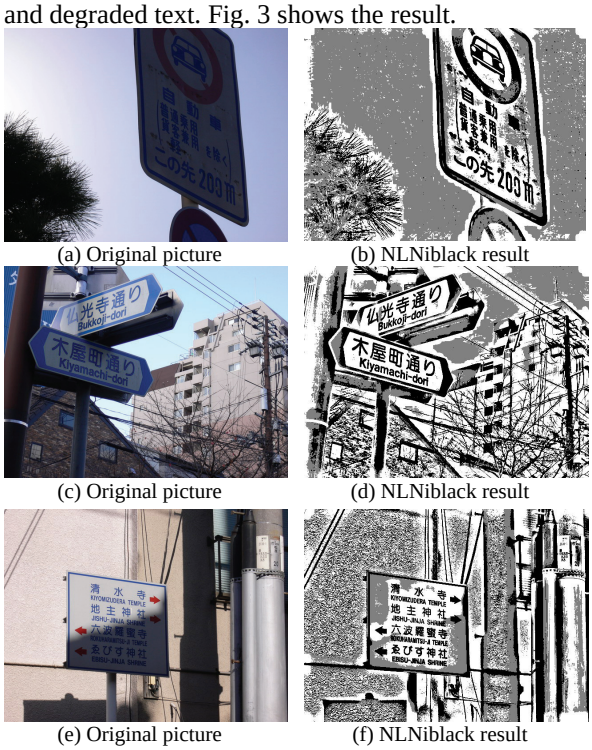
The difference between the NLNiblack and the original Niblack is that we just add two ordered statistics filter Order[. , p,W] to the background filter $\hat{\mu}_{p1}(x,y,W_B)$ and foreground filter $\hat{\sigma}_{p2}(x,y,W_F)$.

In the background filter $\hat{\mu}_{p1}(x,y,W_B)$, the filter width, W_B, is equal to 1/16 of image width and p1 is set to be 50%. It is because the large median filter can extract the background objects while not

excluding their high frequency components. This background filter can handle the uneven lighting in natural scenes.

In the foreground filter $\hat{\sigma}_{p2}(x, y, W_F)$, the filter width W_F, is 1/5 of W_B and p2 is set to be 80%. This high percentile filter can effectively 'spread' the influence of small areas with high variance to neighboring regions and can effectively increase local noise suppression.

Then we label the CCs in two thresholded layers, 1 and -1, respectively. The proposed NLNiblack decomposition can effectively handle the difficult conditions, such as low contrast, uneven illumination and degraded text. Fig. 3 shows the result.



(a) Original picture (b) NLNiblack result

(c) Original picture (d) NLNiblack result

(e) Original picture (f) NLNiblack result

Fig. 3. NLNiblack result: black 1, white -1

3. Features to detect characters

After decomposing the image into a set of CCs, we convert the segmentation problem into a classification problem – all we need to do is to classify all candidate CCs into 2 categories, text or non-text. Then we propose 12 novel features to expose the intrinsic characteristics of text CCs.

3.1. Geometric Features

The first three features are *geometric feature*. They are just some common features but can effectively discard a large proportion of apparently non-text CCs with very small computational expense. So they can dramatically decrease the execution time of the whole algorithm.

Area Ratio is used to discard too big or too small CCs:

$$Feature\_AreaRatio = \frac{Area(\mathrm{CC})}{Area(\mathrm{Picture})} \quad (2)$$

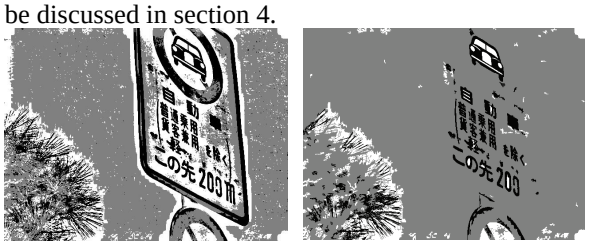
Length Ratio is used to discard too long or too short CCs:

$$Feature\_LengthRatio = \frac{\max\{w, h\}}{\max\{\mathrm{PicW}, \mathrm{PicH}\}} \quad (3)$$

Aspect Ratio is used to discard too thin CC:

$$Feature\_AspectRatio = \max\{w/h, h/w\} \quad (4)$$

According to these three features, we can build three classifiers, each of which will test one feature. The effect of these geometric features can be viewed in Fig. 4.—after the filtering process of the geometric classifiers, apparently non-text CCs are filtered out. The way of training the classifiers will be discussed in section 4.



(a) Input: NLNiblack Result (b) after geometric classifiers

Fig. 4. Effect of geometric classifiers

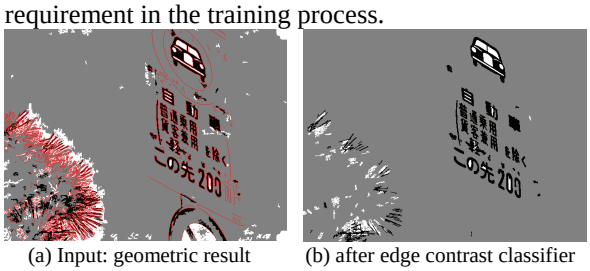
3.2. Edge Contrast Feature

The Edge Contrast Feature plays the most important role in the whole algorithm. Proposing this feature is based on a very common observation – regardless the complex background and the uneven lighting, text CCs are often 'highly closed' by edge response. Therefore, we use Eq.(5) to measure the edge closure degree of a CC. This feature fully takes the advantages of the texture based detection methods and moreover it also has a very strong response to large characters.

$$Feature\_EdgeContrast = \frac{Border(CC) \cap Edge(Picture)}{Border(CC)} \quad (5)$$

$$Edge(Picture) = Canny(Picture) \cup Sobel(Picture)$$

where Canny(Picture) and Sobel(Picture) mean the normalized Canny and the Sobel response of the image, respectively. And Border(CC) means the border pixels of the CC. This feature provides an image independent measurement of every CC's edge contrast. This kind of independency is a key requirement in the training process.



(a) Input: geometric result (b) after edge contrast classifier

Fig. 5. Effect of edge contrast classifiers

In Fig. 5.(a) the red mask is the *Edge(Picture)* response and we can find CCs with small edge closure degree are discarded.

3.3. Shape Regularity Feature

Text CCs often possess more regular shape than arbitrary noise CCs in the natural scene. Based on this observation, we propose 4 features: *Holes*, *Contour Roughness*, *Compactness* and *Occupy Ratio* (Eq.(6)). We can find text CCs often have smaller value in *Holes* and *Contour Roughness*, but larger value in *Compactness* and *Occupy Ratio*, while non-text CCs behave just the opposite. These features are used to suppress the noise which have irregular shape but have strong texture response.

$$Feature\_ContourRoughness = \frac{\left|CC - open\left(imfill\left(CC\right), 2\times 2\right)\right|}{\left|CC\right|} \quad (6)$$

$$Feature\_CCHoles = \left|imholes\left(CC\right)\right|$$

$$Feature\_Compact = \frac{Area\left(CC\right)}{\left|Border\left(CC\right)\right|^2}$$

$$Feature\_OccupyRatio = \frac{Area\left(CC\right)}{Area\left(BoundingBox\left(CC\right)\right)}$$

where

imfill(·) fills the holes in the CC.

imholes(·) count the holes in the CC.

BoundingBox(·) is the bounding box of the CC.

In Fig. 6, we can see the irregular noises with high texture and contrast responses are effectively reduced. For instance, it is very difficult to discard the small 'CAR' symbol on the board without using the shape regularity features.

(a) Input: edge contrast result (b) after shape reg classifier

Fig. 6. Effect of shape regularity classifiers

3.4. Stroke Statistics Feature

Character is composed of strokes, so we proposed 2 computational demanding features which expose the stroke statistics about CC. These two features check other aspects of 'irregularity' in the term of character stroke.

The first feature is *Mean Stroke Width* based on the observation that character stroke width is often relatively small:

$$Feature\_Stroke\_Mean = Mean\left(strokeWidth\left(skeleton\left(CC\right)\right)\right) \quad (7)$$

The second feature is *Normalized stroke deviation* based on the observation that strokes of character often have similar width and the CC with big stroke variance is more likely to be noise:

$$Feature\_Stroke\_std = \frac{Deviation\left(strokeWidth\left(skeleton\left(CC\right)\right)\right)}{Mean\left(strokeWidth\left(skeleton\left(CC\right)\right)\right)} \quad (8)$$

In Eq. (7) and (8), skeleton(·) stands for the morphological skeleton operation and strokeWidth(·) stands for the shortest distance between the pixel on the CC skeleton to the outside pixels.

(a) Input: NLNiblack Result (b) after stroke classifiers

Fig. 7. Effect of stroke statistic classifiers

In Fig. 7, we can find that the big characters survive after these classifiers while noises are effectively reduced.

3.5. Spatial Coherence Features

The last two spatial coherence features exploit the spatial coherence information to filter out the non-text CCs. Noises will have less spatial regularity and coherence, so we propose these two features:

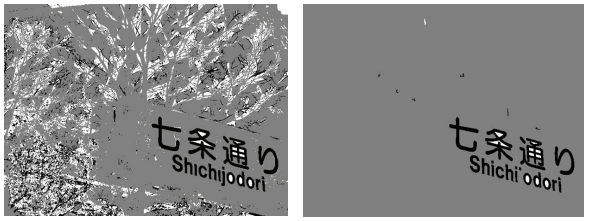
Spatial coherence area ratio

$$Feature\_AreaRatio\_S = \frac{Area\left(imdilate\left(CC, 5\times 5\right)\right)}{Area(\text{Picture})} \quad (9)$$

Spatial coherence boundary touching

$$Feature\_Boundary\_S = Bound\left(imdilate\left(CC, 5\times 5\right)\right) \quad (10)$$

In Eq.(9) and (10), imdilate(·, strel) stands for the morphological dilation operation with structural element, strel. In this stage, the apparently non-text CCs have already been discarded. Then in every layer, if some CC expands significantly after being dilated with a small structural element, it is more likely to be spatially correlated random noise. On the contrary, the text CCs will not act like this because of the structural nature of characters. By using the spatial coherence features, we can efficiently reduce the noises (In Fig. 8).



(a) Input: stroke result (b) after spatial classifiers

Fig. 8. Effect of spatial coherence classifiers

4. Classifier cascade training

Since we have already had a set of CCs and for every CC we have 12 features which can effectively separate text CC from non-text CC, the remaining

problem is how to use these features. The easiest solution is heuristically setting all thresholds manually. This is a very unstable approach. Therefore, a better solution may be the machine learning method.

Then the further problem is which machine learning method to use. Since some of the features we use are computational demanding, such as, *stroke statistics* features and *edge contrast* feature, it is unwise to calculate all of the 12 features together during classification. We need a mechanism to discard most of the non-text CCs by less computation. This requirement reminds us of the Adaboost scheme and the attentional cascade architecture used in face detection [15].

Although we use Adaboost to train all the classifiers and also build an attentional cascade, our method substantially differs from the techniques used in [15] because text detection and face detection are two essentially different tasks. Table 1 gives a comparison between these two tasks.

Table 1. Text detection vs. face detection

	Text detection	**Face detection**
Basic unit	Connected component	24x24 detect window
Feature num	12 / CC	45,396 / Window
Feature Quality	High	Vary violently
Negative sample	Easy to find	Need careful consideration
Performance Information	Not known in advance	Known after feature selection

4.1. Notation

Before going into training scheme details, we will clarify the notation we use at first. See Table 2.

Table 2. Notations

f	False positive rate: $\frac{area(error)}{area(negative)}$
d	Detection rate: $\frac{area(hit)}{area(positive)}$
FR	False rejection rate: $1-f=\frac{area(negative)-area(error)}{area(negative)}$
P	positive training set
N_i	ith negative training set
f_i	maximum false positive rate of ith layer
d_i	minimum detection rate of ith layer
F	overall false positive rate
D	overall detection rate
M	number of classifier in the cascade
h_i	ith weak classifier in the cascade
w_i	weight of ith classifier in Adaboost learning scheme

4.2. Important Assumption

We will feed all the CCs into the classifier cascade. If one CC is rejected by any of classifier, it is regarded as non-text CC. Therefore, it is easy to know that we have the following relationship:

$$F=\prod_{i}^{M} f_i \qquad D=\prod_{i}^{M} d_i \tag{11}$$

$$\log(F)=\sum_{i=1}^{M}\log(f_i) \quad \log(D)=\sum_{i=1}^{M}\log(d_i)$$

In the logarithm conversion form of the basic relationship, we can find that the overall detection rate is linearly dispatched to all the classifiers. Then we can assume the logarithm form of minimum detection rate is linearly dispatched according to the 'quality' of each classifier. Therefore, we will have the dispatching formulation as follows:

$$d_i=\left(D_{dispatch}\right)^{\gamma} \tag{12}$$

where $D_{dispatch}$ is the detection rate can be dispatched and γ stands for the 'quality' portion of the ith classifier. We find that this formula has close relationship to the idea of indifference curve proposed by J. Sun. et.al [17].

4.3. Cascade Building Process

First, we will use the standard Adaboost training scheme [15] to train a *strong classifier*, a linear combination of 12 weak classifiers. Every weak classifier only responds to one single feature of CC and makes the decision whether the CC is text or not.

- User selects overall minimum detection rate D_{target}.
- Random Select 200 pictures from total 368 pics
  - *P = set of positive examples*
  - *N = set of negative examples*
- $F_0 = 1.0$; $D_0 = 1.0$; $i = 0$
- *Feature = { feature$_j$ | j = 1 to M }*

for i = 1: M
  $D_i = D_{i-1}$
  foreach feature$_j$ in *Feature*
    get distribution of feature$_j$ based on {P,N}
    calculate d_j (D_i), f_j (D_i) FR_j (D_i,1- D_i)
  end
  choose the feature k with Biggest f_k (D_i)
  γ = FR_k (D_i,1- D_i)/SUM$_j$ (FR_j (D_i,1- D_i));
  d_i = (D_{target}/ D_i)^γ ;
  training: d_i = h_i (d_i, P, N)
  $N = \varnothing$
  evaluate the current cascaded detector h_i on the set of non-text CCs and put any false detections into the set *N. and D_i = D_i * d_i;*
  Feature=Feature- feature$_k$;
end

Fig. 9. Cascade training algorithm

Second, based on the combination weight we get from Adaboost, we use the following algorithm to train the attentional cascade (Fig. 9). Our method differs from the existing methods in adaptively

dispatching the entire expected detection rate into 12 classifiers.

Third, we will add a post process part after the cascade. The *strong classifier* we train in the first step will be used as the 13th classifier in the cascade. All 12 features of the CC passing through the previous cascade have been calculated, so only a linear combination operation is needed for the *strong classifier*, which can further improve the accuracy.

The last but not least, we will combine the CCs in the black layer and white layer together to form the finial result. We will compare the adjacent CCs' confidence margin which is obtained by the 13th classifier, and then omit the CCs with smaller margin. The remaining CCs are considered as final result.

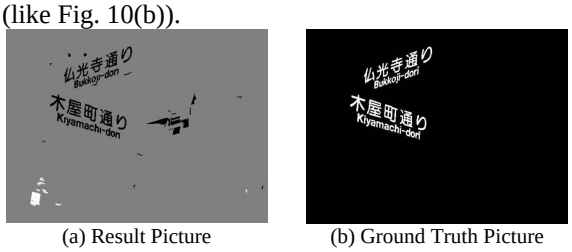
5. Experimental Results

5.1. System architecture

We implemented a prototype system and exhibited it in Shanghai International Industry Fair 2004. We use Sony Ericsson S700c, which is attached with a 120 Mega pixel sensor, to take a photo of the natural sign. Then this image is transferred through Bluetooth OBEX protocol to a processing server, 1.6GHz CPU and 256M RAM. After seeing the image arrive, the server will do the detection and segmentation of the image. The segmented regions are regularized and then sent to the recognition and translation module. Finally, the result image is sent back to the mobile. The whole process is done in less than 1 s, and this demo shows that our segmentation algorithm is very robust and fast.

5.2. System evaluation

To better evaluate our algorithm, we built a database containing 368 difficult scene images (640x480) and labeled all the ground truth manually (like Fig. 10(b)).



(a) Result Picture (b) Ground Truth Picture

Fig. 10. Evaluation the result

We employ a very strict pixel-wise evaluation criterion to measure performance, showed as follows:

$$\begin{aligned} hit &= area\left(Result \,\&\, GroundTruth\right) \\ error &= area\left(Result \,\&\, \overline{GroundTruth}\right) \\ miss &= area\left(\overline{Result} \,\&\, GroundTruth\right) \\ precision &= \frac{hit}{hit + error}, \quad recall = \frac{hit}{hit + miss} \end{aligned} \tag{13}$$

The evaluation score is showed below (Table 3) and we can find that our algorithm is very robust:

Table 3. Overall performance

	Precision	**Recall**
Training set	92.3%	98 %
Testing set	88.9%	97.5 %

Besides the standard evaluation, we also establish experiments to prove the effect of every feature (see Fig. 11):

Fig. 11. Effect proof for every feature

We omit one feature in the whole cascade and then evaluate the final precision without this feature. We can find that without the *edge contrast* feature, the overall performance drops sharply, which indicates the *edge contrast* feature contributes most on the performance. On the contrary, the *geometric* features almost contribute nothing in the precision.

The average running time of the algorithm processing one picture is 0.34s. In a more detailed experiment, we also omit the features one by one to see their contribution to the average running time. In Fig. 12, we can find that without the *geometric* features, the running time will increase to 1.72s. It is saying that the *geometric* features can effectively discard a lot of non-text CCs in very small computational cost. On the contrary, stroke features are the most computational demanding features, but thanks to the previous classifiers, it will just exam the most promising text CCs, so it will not impose great burden on the algorithm efficiency.

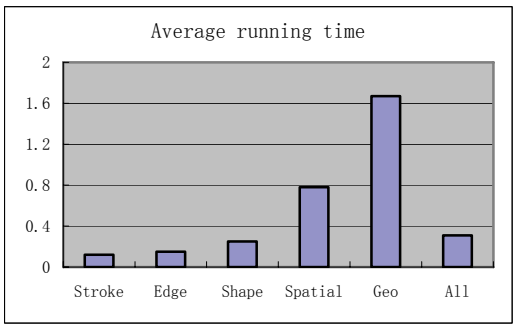


Fig. 12. Efficiency proof for every feature

6. Conclusions

In this paper, we present a novel detection algorithm for scene text. In sum, our contributions are:

- Propose a fast and robust decomposition method called NLNiblack.
- Propose 12 novel features for connected component based detection method.
- Propose an Adaboost modification to train the cascade on text detection problem.
- Implement a fast and robust prototype system.

Acknowledgements

The authors would like to thank reviewers for their comments on this paper. This work was performed at Computer Vision Laboratory, SJTU, and was supported by OMRON under PVS project.

References

[1] Doermann, Progress in camera-based document image analysis, 7th ICDAR Conference, 2003

[2] Li, D. Doermann and O. Kia. Automatic Text Detection and Tracking in Digital Video. IEEE Transactions on Image Processing. Vol. 9, No. 1, pp. 147-156, Jan. 2000.

[3] Rainer Lienhart. Automatic Text Recognition for Video Indexing. Proc. ACM Multimedia 96, Boston, MA, pp. 11-20, Nov. 1996.

[4] MulS. Ferreira, C. Thillou, B. Gosselin, 2003, From Picture to speech: an innovative OCR application for embedded environment, 17th ProRISC 2003, Veldhoven

[5] C.S. Shin, K.I. Kim, M.H. Park, H.J. Kim. Support Vector Machine-based Text Detection in Digital Video. Proceedings of the IEEE Signal Processing Society Workshop on Neural Networks for Signal Processing X, Vol. 2, pp. 634-641, 2000.

[6] P. Clark and M. Mirmehdi. Finding Text Regions Using Localised Measures. Proceedings of the 11th British Machine Vision Conference, pp. 675-684, September 2000.

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

[8] Byung Tae Chen, Younglae Bae, Tai-Yun Kim, Automatic Text Extraction in Digital Videos using FFT and Neural Network, IEEE International Fuzzy Systems Conference Proceedings, August 22-25, 1999, seoul, Korea

[9] Ismail Haritaoglu, Scene text extraction and translation for handheld devices, 2001 IEEE

[10] J. Gao and J. Yang, An Adaptive Algorithm for Text Detection from Natural Scenes, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2001

[11] Takuma Yamaguchi, Minoru Maruyama, Character Extraction from Natural Scene Images by Hierarchical Classifiers. ICPR 04, 687-690, 2004

[12] Nobuo Ezaki, Marius Bulacu, Lambert Schomaker, Text Detection from Natural Scene Images: Towards a System for Visually Impaired Persons. ICPR, 683-686, 2004

[13] Yu Zhang, et al., Automatical caption localization in compressed video, IEEE Trans. Pattern Anal. Mach. Intell. 22 (4) (2000) 385–392.

[14] Kongqiao Wanga, Jari A. Kangasb, Character location in scene images from digital camera, Pattern Recognition 36 (2003) 2287 – 2299

[15] Paul A. Viola, Michael J. Jones: Robust Real-Time Face Detection. ICCV 2001: 747

[16] L. Winger, J.A. Robinson, M. ED Jernigan, Low-Complexity Character Extraction In Low-Contrast Scene Images, International Journal of Pattern Recognition and Artificial Intelligence, Vol. 14, No. 2 (2000) 113-135

[17] Jie Sun Rehg, J.M. Bobick, A. , Automatic cascade training with perturbation bias, CVPR 2004, II-276- II-283 Vol.2, July 2004

(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

(k) (l) (m) (n) (o)

(p) (q) (r) (s) (t)

(u) (v) (w) (x) (y)

(A) (B) (C) (D) (E)

(F) (G) (H) (I) (J)

(K) (L) (M) (N) (O)

Fig. 13. More experiment results: (a)~(e),(k)~(o),(u)~(y),(F)~(J)original pictures, (f)~(j),(p)~(t),(A)~(E),(K)~(O) result pictures