Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Automatic character detection and segmentation in natural scene images*

ZHU Kai-hua†, QI Fei-hu, JIANG Ren-jie, XU Li

(*Department of Computer Science and Technology, Shanghai Jiao Tong University, Shanghai 200030, China*)

†E-mail: godzkh@gmail.com

Received Oct. 18, 2005; revision accepted Feb. 22, 2006

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, all these CCs are fed into a cascade of classifiers trained by Adaboost algorithm. Each classifier in the cascade responds to one feature of the CC. Proposed here are 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 rapidly discarded while more computation is spent on promising text-like CCs. The CCs passing through the cascade are considered as text components and are used to form the segmentation result. A prototype system was built, with experimental results proving the effectiveness and efficiency of the proposed method.

Key words: Text detection and segmentation, Adaboost, NLNiblack decomposition method, Attentional cascade
doi:10.1631/jzus.2007.A0063 **Document code:** A **CLC number:** TP391.41

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 scene text recognition is rapidly expanding. 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. Automatic recognition, translation or enunciation of these texts will be of great help for foreign travelers, visually impaired people and computer programs performing video indexing or meeting processing, etc. (Doermann *et al*., 2003).

Fully automatic text extraction from images, especially from scene images, has always been a challenging problem. The difficulties arise from 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, rapid processing is usually desired.

There is growing research work focusing on real scene text detection these years. Current text detection approaches can be classified into two categories.

Fig.1 A difficult natural scene image

* Project supported by OMRON under PVS project

The first category is the texture-based methods. Shin *et al*.(2000) used a star-like pixel mask to expose the intrinsic features of text occurrences. Clark and Mirmehdi (2000) proposed 5 localized measures and used a combination of these measures to get candidate text regions. Frequency domain techniques are also used to detect text-like texture, such as: Fourier transform on short scanning line (Chen *et al*., 1999), discrete cosine transform (Zhong *et al*., 2000), Gabor transform (Ferreira *et al*., 2003), wavelet decomposition (Li *et al*., 2000), multi-resolution edge detector (Gao and Yang, 2001). 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 have 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.

The second category is the connected-component (CC) based methods. Color quantization (Wang and Kangas, 2003), morphological operation (Hasan and Karam, 2000) and symmetric neighborhood filters (Haritaoglu, 2001) 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 (Hasan and Karam, 2000; Haritaoglu, 2001; Wang and Kangas, 2003), aligning-and-merging analysis (Wang and Kangas, 2003), layout analysis (Gao and Yang, 2001), hierarchical connected components analysis (Haritaoglu, 2001). These rules are often instable and cannot guarantee robust detection result.

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

The framework of our proposed algorithm is shown in Fig.2. The method is composed of three stages. In the first stage, a novel Non-Linear Niblack (NLNiblack) method is used, 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.

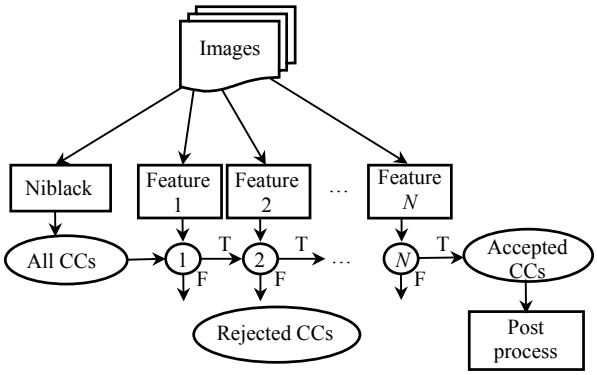


Fig.2 The text detection algorithm

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

The first point is that a classifier cascade is used, which can easily discard most non-text CCs and rapidly focus on more promising text CCs. This idea is inspired by face detection technique (Viola and Jones, 2001) and can process images rapidly while achieving high detection rates. However, due to the essential difference between text detection and face detection, a specific learning scheme was originally proposed for the text detection problem.

The second point is that a series of novel features is proposed, each of which has specific contribution to the text detection task. As will be shown later, some of the features can take the advantage of an image's texture characteristic, some of them can exploit the spatial coherent information and some of them can efficiently accelerate the whole algorithm, etc. By using these features, our algorithm can have the advantages of both texture-based methods and CC-based methods, while suppressing their drawbacks. This work is an innovative attempt to formulate a series of features for text detection.

A prototype system was developed using a mobile phone, Sony Ericsson S700c, attached with a

120 M pixel sensor. This system exhibited in the Shanghai International Industry Fair 2004. Its ability to automatically detect, segment and translate English and Japanese signs into Chinese proved the effectiveness and efficiency of our algorithm.

NLNIBLACK DECOMPOSITION

As we know, decomposing an image into a set of CCs is a very crucial step in CC-based methods. If the decomposition step yields poor results, the performance of the whole algorithm will be in question. There are several existing methods (Hasan and Karam, 2000; Haritaoglu, 2001; Wang and Kangas, 2003) aiming at effective and robust decomposition. Besides this concern, the computation efficiency and simplicity of implementation also concern us. Therefore, we propose a very efficient NLNiblack thresholding method inspired by Winger *et al*.(2000):

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

$$T_{\pm}(x,y)=\hat{\mu}_{p1}(x,y,W_{\mathrm{B}})\pm k\cdot\hat{\sigma}_{p2}(x,y,W_{\mathrm{F}}),$$

$$\hat{\mu}_{p1}(x,y,W_{\mathrm{B}})=Order[Mean(f(x,y),W_{\mathrm{B}}),p1,W_{\mathrm{B}}],$$

$$\hat{\sigma}_{p2}(x,y,W_{\mathrm{F}})=Order[Deviation(f(x,y),W_{\mathrm{F}}),p2,W_{\mathrm{F}}],$$

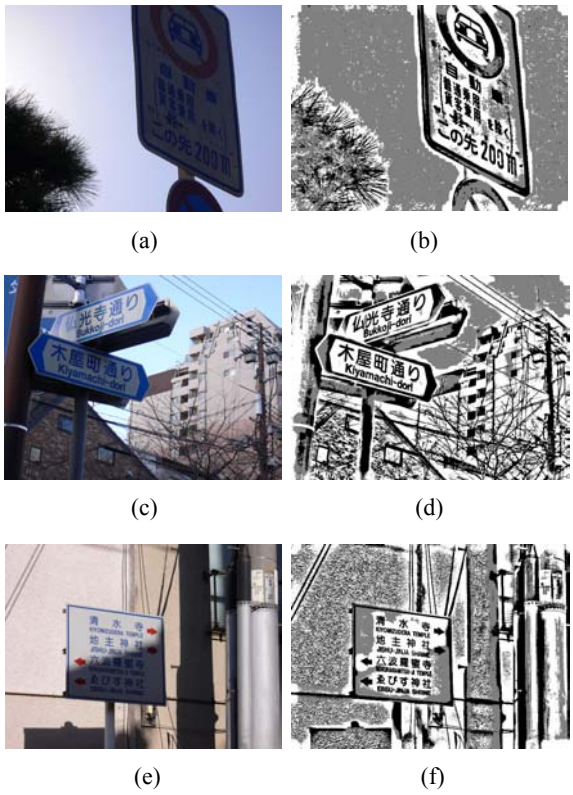
where k is set to be 0.18 as standard Niblack method; $f(x,y)$ is the input pixel intensity at position (x,y); $Mean(.,W)$ is the mean value filter with width W; $Deviation(.,W)$ is the standard deviation filter with width W; $Order[.,p,W]$ is the ordered statistics filter with p percentile and width W.

The difference between the NLNiblack and the original Niblack is that two ordered statistics filters, $Order[.,p,W]$, are added to the background filter $\hat{\mu}_{p1}(x,y,W_{\mathrm{B}})$ and foreground filter $\hat{\sigma}_{p2}(x,y,W_{\mathrm{F}})$.

In the background filter $\hat{\mu}_{p1}(x,y,W_{\mathrm{B}})$, the filter width, W_{B}, is equal to 1/16 of image width and $p1$ is set to be 50%. Because the large median filter can extract background objects while not excluding their high frequency components. This background filter can handle the uneven lighting in natural scenes.

In the foreground filter $\hat{\sigma}_{p2}(x,y,W_{\mathrm{F}})$, 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 the neighboring regions and can effectively increase the local noise suppression.

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



(a) (b)

(c) (d)

(e) (f)

Fig.3 NLNiblack results: black 1, white −1. (a) (c) (e) are original pictures and (b) (d) (f) are the corresponding NLNiblack results

FEATURES TO DETECT CHARACTERS

After decomposing the image into a set of CCs, the segmentation problem is converted into a classification problem—what has to be done is to classify all candidate CCs into 2 categories, text or non-text. Then 12 novel features are proposed to expose the intrinsic characteristics of text CCs.

Geometric features

The first three features are geometric features.

They are just some common features but can effectively discard a large proportion of apparently non-text CCs at 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:

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

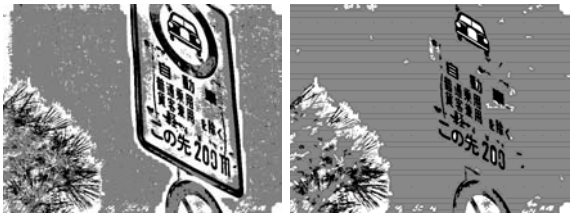
Length Ratio is used to discard too long or too short CCs:

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

Aspect Ratio is used to discard too thin CC:

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

From 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 method for training the classifiers will be discussed in Section 4.



(a) (b)

Fig.4 Effect of geometric classifiers. (a) Input: NLNiblack result; (b) After geometric classifiers

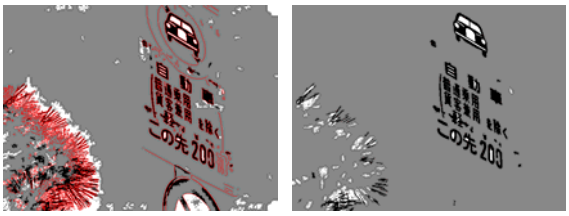
Edge contrast feature

The edge contrast feature plays the most important role in the whole algorithm. Proposing of this feature is based on a very common observation, i.e., regardless of 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 advantage of the texture-based detection methods and also has a very strong response to large characters.

$$\begin{cases} Feature\_EdgeContrast \\ \qquad = \dfrac{Border(CC) \cap Edge(Picture)}{Border(CC)}, \\ Edge(Picture) = Canny(Picture) \cup Sobel(Picture), \end{cases} \quad (5)$$

where *Canny*(*Picture*) and *Sobel*(*Picture*) mean the normalized Canny and the Sobel response of the image, respectively. *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. In Fig.5a the grey mask is the *Edge*(*Picture*) response and we can find CCs with small edge closure degree are discarded.



(a) (b)

Fig.5 Effect of edge contrast classifiers. (a) Input: geometric result; (b) After edge contrast classifiers

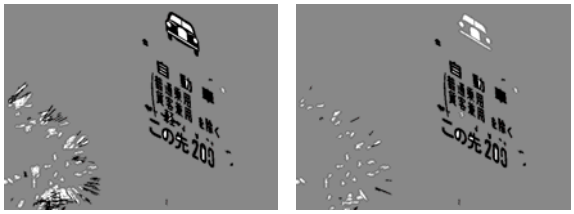
Shape regularity feature

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 way. These features are used to suppress noise with irregular shape but have strong texture response.

$$\begin{cases} Feature\_ContourRoughness \\ \qquad = \dfrac{|CC - open(imfill(CC), 2\times 2)|}{|CC|}, \\ Feature\_CCHoles = |imholes(CC)|, \\ Feature\_Compact = \dfrac{Area(CC)}{|Border(CC)|^2}, \\ Feature\_OccupyRatio = \dfrac{Area(CC)}{Area(BoundingBox(CC))}, \end{cases} \quad (6)$$

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 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) (b)

Fig.6 Effect of shape regularity classifiers. (a) Input: edge contrast result; (b) After shape regularity classifiers

Stroke statistics feature

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

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

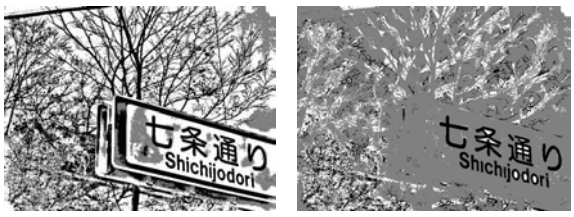
$$Feature\_Stroke\_Mean = Mean(strokeWidth(skeleton(CC))). \quad (7)$$

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 very likely to be noise:

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

In Eqs.(7) and (8), *skeleton*(.) stands for the morphological skeleton operation and *strokeWidth*(.) stands for the shortest distance from the pixel on the CC skeleton to the outside pixels.

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



(a) (b)

Fig.7 Effect of stroke statistic classifiers. (a) Input: NLNiblack result; (b) After stroke statistic classifiers

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

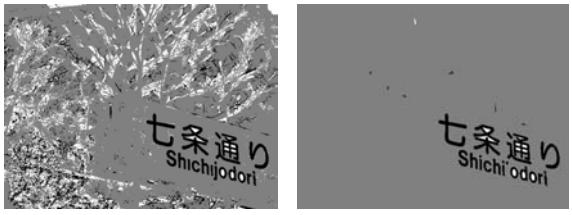
$$Feature\_AreaRatio\_S = \frac{Area(imdilate(CC, 5\times 5))}{Area(Picture)}. \quad (9)$$

Spatial coherence boundary touching

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

In Eqs.(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 (Fig.8).



(a) (b)

Fig.8 Effect of spatial coherence classifiers. (a) Input: stroke result; (b) After spatial coherence classifiers

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 computationally 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 using less computation. This requirement reminds us of the Adaboost scheme and the attentional cascade architecture used in face detection (Viola and Jones, 2001).

Although we use Adaboost to train all the classifiers and also build an attentional cascade, our method substantially differs from the techniques used in (Viola and Jones, 2001) because text detection and face detection are two essentially different tasks. Table 1 compares these two tasks.

Table 1 Text detection vs face detection

	Text detection	Face detection
Basic unit	Connected component	24×24 detection window
Feature number	12/CC	45396/Window
Feature quality	High	Vary violently
Negative sample	Easy to find	Need careful consideration
Performance information	Not known in advance	Known after feature selection

Notation

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

Important assumption

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

$$F=\prod_{i=1}^{M} f_i, \quad D=\prod_{i=1}^{M} d_i.$$

Table 2 Notations in the training produce

Parameter	Meaning
f	False positive rate: *area*(*error*)/*area*(*negative*)
d	Detection rate: *area*(*hit*)/*area*(*positive*)
FR	False rejection rate: 1−*f*=[*area*(*negative*)−*area*(*error*)]/*area*(*negative*)
P	Positive training set
N_i	The *i*th negative training set
f_i	Maximum false positive rate of the *i*th layer
d_i	Minimum detection rate of the *i*th layer
F	Overall false positive rate
D	Overall detection rate
M	Number of classifier in the cascade
h_i	The *i*th weak classifier in the cascade
w_i	Weight of the *i*th classifier in Adaboost learning scheme

or

$$\log F=\sum_{i=1}^{M}\log f_i, \quad \log D=\sum_{i=1}^{M}\log d_i. \tag{11}$$

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:

$$d_i=(D_{\text{dispatch}})^{\gamma}, \tag{12}$$

where D_{dispatch} is the detection rate that can be dispatched and γ stands for the 'quality' portion of the *i*th classifier. We find that this formula has close relationship to the idea of indifference curve proposed by Jie *et al*.(2004).

Cascade building process

First, we will use the standard Adaboost training scheme (Viola and Jones, 2001) to train a "strong classifier", a linear combination of 12 "weak classifiers". Every weak classifier only responds to one single feature of CC and decides whether or not the CC is text.

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 dis-

patching the entire expected detection rate into 12 classifiers.

- User selects overall minimum detection rate D_{target}.
- Random select 200 pictures from total 368 pictures
  - 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};
  For each $feature_j$ 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 the biggest $f_k(D_i)$;
  $\gamma=FR_k(D_i,1-D_i)/SUM_j(FR_j(D_i,1-D_i))$;
  $d_i=(D_{target}/D_i)^{\wedge}\gamma$;
  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 $D_i=D_i\times d_i$;
  $Feature=Feature-feature_k$;
End

Fig.9 Cascade training algorithm

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

Last but not least, the CCs in the black layer and white layer are combined to form the finial result. The adjacent CCs' confidence margin obtained by the 13th classifier is compared, and then the CCs with smaller margin are omitted. The remaining CCs are considered as final result.

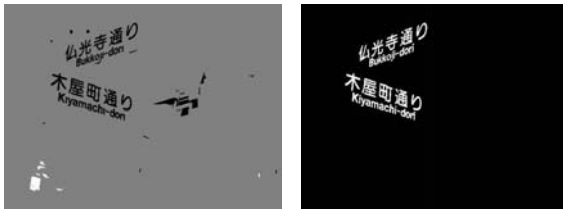
EXPERIMENTAL RESULTS

System architecture

The authors implemented a prototype system and exhibited it in the Shanghai International Industry Fair 2004. The used Sony Ericsson S700c attached with a 120 M pixel sensor, takes a photo of the natural sign. Then this image is transferred through Bluetooth OBEX protocol to a processing server, 1.6 GHz CPU and 256 M RAM. After seeing the image arrives, 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 resulting image is sent back to the mobile phone. The whole process is done in less than 1 s, and this demo showed that the authors' segmentation algorithm is very robust and fast.

System evaluation

To better evaluate the algorithm, a database containing 368 difficult scene images (640×480) was built and all the ground truth was labelled manually (like Fig.10b).



(a) (b)

Fig.10 Evaluation of the result. (a) Resulting picture; (b) Ground truth picture

A very strict pixel-wise evaluation criterion was used to measure performance, shown as follows:

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

The evaluation score is given in Table 3, which shows the proposed algorithm is very robust.

Table 3 Overall performance of our algorithm

	Precision (%)	Recall (%)
Training set	92.3	98.0
Testing set	88.9	97.5

Besides the standard evaluation, experiments were also conducted to prove the effect of every feature (Fig.11). One feature was omitted in the whole cascade and then the final precision without this fea-

ture was evaluated. It was found that without the edge contrast feature, the overall performance dropped sharply, which indicated the edge contrast feature contributes most to the performance. On the contrary, the geometric features contribute almost nothing in the precision.

The algorithm's average running time for processing one picture was 0.34 s. In a more detailed experiment, the features will be omitted one by one to see their contribution to the average running time. Fig.12 shows that without the geometric features, the running time increases to 1.72 s, indicating that the geometric features can effectively discard many non-text CCs at very small computational cost. On the contrary, stroke features are the most computationally 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.

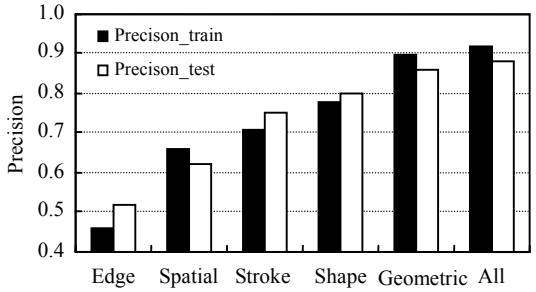


Fig.11 Effect proof for every feature

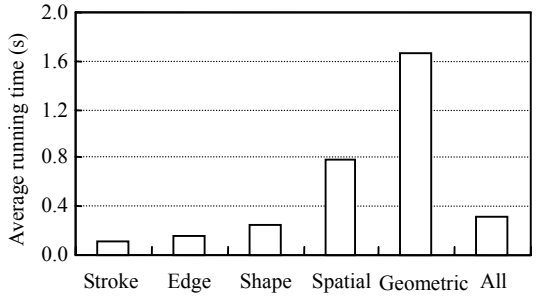


Fig.12 Efficiency proof for every feature

CONCLUSION

This paper presents a novel detection algorithm for scene text. In sum, this paper:

(1) Proposes a fast and robust decomposition method called NLNiblack;

(2) Proposes 12 novel features for connected component based detection method;

(3) Proposes an Adaboost modification to train the cascade on text detection problem;

(4) Proposes implementation of a fast and robust prototype system.

References

Chen, B.T., Bae, Y., Kim, T.Y., 1999. Automatic Text Extraction in Digital Videos Using FFT and Neural Network. Proceedings of the IEEE International Fuzzy Systems Conference. Seoul, Korea.

Clark, P., Mirmehdi, M., 2000. Finding Text Regions Using Localised Measures. Proceedings of the 11th British Machine Vision Conference, p.675-684.

Doermann, D., Liang, J., Li, H., 2003. Progress in Camera-based Document Image Analysis. Proceedings of the 7th International Conference on Document Analysis and Recognition. Edinburgh, Scotland, **1**:606-616.

Ferreira, S., Thillou, C., Gosselin, B., 2003. From Picture to Speech: An Innovative OCR Application for Embedded Environment. ProRISC 2003. Veldhoven, Netherland.

Gao, J., Yang, J., 2001. An Adaptive Algorithm for Text Detection from Natural Scenes. CVPR 2001, p.84-89.

Haritaoglu, I., 2001. Scene Text Extraction and Translation for Handheld Devices. CVPR 2001, p.408-413.

Hasan, Y.M.Y., Karam, L.J., 2000. Morphological text extraction from images. *IEEE Transactions on Image Processing*, **9**(11): 1978-1983. [doi:10.1109/83.877220]

Jie, S., Rehg, J.M., Bobick, A., 2004. Automatic Cascade Training with Perturbation Bias. CVPR 2004, **2**:276-283.

Li, H., Doermann, D., Kia, O., 2000. Automatic text detection and tracking in digital video. *IEEE Transactions on Image Processing*, **9**(1):147-156. [doi:10.1109/83.817607]

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

Viola, P., Jones, M., 2001. Robust Real-time Face Detection. ICCV01, **2**:747.

Wang, K., Kangas, J., 2003. Character location in scene images from digital camera. *Pattern Recognition*, **36**(10): 2287-2299. [doi:10.1016/S0031-3203(03)00082-7]

Winger, L., Robinson, J.A., Jernigan, M.E., 2000. Low-complexity character extraction in low-contrast scene images. *International Journal of Pattern Recognition and Artificial Intelligence*, **14**(2):113-135. [doi:10.1142/S0218001400000106]

Zhong, Y., Zhang, H.J., Jain, A.K., 2000. Automatical caption localization in compressed video. *IEEE Transactions on PAMI*, **22**(4):385-392.

(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) are original pictures, (f)~(j), (p)~(t), (A)~(E), (K)~(O) are result pictures