ELSEVIER
Available online at www.sciencedirect.com
SCIENCE DIRECT®
Signal Processing: *Image Communication* 19 (2004) 205–217
SIGNAL PROCESSING: IMAGE COMMUNICATION
www.elsevier.com/locate/image

A localization/verification scheme for finding text in images and video frames based on contrast independent features and machine learning methods

Datong Chen[a,*], Jean-Marc Odobez[a] , Jean-Philippe Thiran[b]

[a] *Dalle Molle Institute for Perceptual Artificial Intelligence (IDIAP), Rue du Simplon 4, 1920 Martigny, Switzerland*
[b] *Swiss Federal Institute of Technology Lausanne (EPFL), Signal Processing Institute (ITS), 1015 Lausanne, Switzerland*

Received 16 December 2002; received in revised form 6 May 2003; accepted 12 June 2003

Abstract

Automatic character detection in video sequences is a complex task, due to the variety of sizes and colors as well as to the complexity of the background. In this paper we address this problem by proposing a localization/verification scheme. Candidate text regions are first localized by using a fast algorithm with a very low rejection rate, which enables the character size normalization. Contrast independent features are then proposed for training machine learning tools in order to verify the text regions. Two kinds of machine learning tools, multilayer perceptrons and support vector machines, are compared based on four different features in the verification task. This scheme provides fast text detection in images and videos with a low computation cost, comparing with traditional methods.

Keywords: Image OCR; Video OCR; Content-based Indexing; Text detection; Support vector machine

1. Introduction

Content-based multimedia database indexing and retrieval tasks require automatic extraction of descriptive features that are relevant to the subject materials (images, video, etc.). The typical low level features that are extracted in images or video include measures of color [25], texture [13], or shape [14]. Although these features can easily be extracted, they do not give a clear idea of the image content. Extracting more descriptive features and higher level entities, for example text [3] or human faces [24], has attracted more and more research interest recently. Text embedded in images and video, especially captions, provide brief and important content information, such as the name of players or speakers, the title, location and date of an event, etc. These text can be considered as a powerful feature (keyword) resource as are the information provided by speech recognizers for example. Besides, text-based search has been successfully applied in many applications while the robustness and computation cost of the feature matching algorithms based on other high level features are not efficient enough to be applied to large databases.

*Corresponding author.
E-mail addresses: chen@idiap.ch (D. Chen), odobez@idiap.ch (J.-M. Odobez), jp.thiran@epfl.ch (J.-P. Thiran).

0923-5965/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S0923-5965(03)00075-4

Text detection and recognition in images and video frames, which aims at integrating advanced optical character recognition (OCR) and text-based searching technologies, is now recognized as a key component in the development of advanced image and video annotation and retrieval systems. Unfortunately, contrary to most of the OCR problems, text characters contained in images and videos can be any grayscale value (not always black characters on a white background), low resolution, variable size, and embedded in complex backgrounds. Experiments show that applying conventional OCR technology directly on the video frames leads to poor recognition rates. Therefore, efficient detection and segmentation of text characters from background is necessary to fill the gap between image and video documents and the input of a state-of-the-art OCR system.

Previously proposed methods show that characters can be detected by exploiting the characteristics on vertical edge, texture and edge orientations. One system for localizing text in covers of Journals or CDs [30] recognized that text were contained in regions with high horizontal variance, and satisfied certain spatial properties. The spatial properties could be exploited in a connected component analysis process. Smith et al. [22] localized text by first detecting vertical edges with a predefined template, then grouping vertical edges into text regions using a smoothing process. These two methods are fast but produce many false alarms because many background regions may also have strong horizontal contrast.

Wu et al. [29] described a text localization method based on texture segmentation. Texture feature was computed at each pixel from the derivatives of the image at different scales. Using a K-means algorithm, pixels were classified into three classes in the feature space. The class with highest energy in the feature space indicated text candidates while the two others indicated non-text and uncertainty. The high complexity of texture segmentation is the main drawback of this method and the segmentation quality is quite sensitive to background noises.

In a more recent work, Garcia et al. [8] proposed a feature, called variance of edge orientation, for text localization which exploited the fact that text string contained edges in other orientations. Variation of edge orientations was computed in local area from image gradient, and combined with the edge features for locating text blocks. However, the method does not take care of characteristics coming from parallel character edges.

Besides the properties of individual characters, Sobottka et al. [23] suggested that baseline detection could be used for text string localization. More precisely, text strings are characterized by specific top and bottom baselines, which thus should be detected in order to assess the presence of a text string in an image block.

Temporal information of a text string in consecutive video frames can be used for extracting some video text. There are two kinds of text in video captions, which are graphic text planned into the scene of video frames, and scene text, which are originally contained in the scene. Temporal information is usually helpful for detecting captions since they are mostly stationary but not very helpful for the scene text, which may have various motion and transforms. Sato [18] and Lienhart [12] computed the maximum or minimum value at each pixel position over frames. The values of the background pixels that are assumed to have more variance through video sequence will be pushed to black or white while the values of the text pixels are kept. However, this method can only be applied on black or white characters. Li [11] proposed a multiframe enhancement for unknown grayscale text which computes the average of pre-located text regions in multiple frames for further segmentation and recognition. The average image has a smaller noise variance but may propagate blurred characters in frames. These temporal methods are helpful in extracting motion constrained text, for examples static captions. More temporal information can be exploited in combining text segmentation and recognition results based on a tracking system [4]. However, these temporal techniques are out of the scope of this paper and will not be discussed further.

The manually designed heuristic features-based methods outlined above usually provide fast detection systems but are not very robust when the texture of the text and the background are very complex. As an alternative, some text detection systems use trained detectors based on machine learning methods [11,12]. These systems extract wavelet [11] or derivative [12] features from fix-size blocks of pixels and classify the feature vectors into text or non-text using neural networks. Because the neural network based classification is performed in the whole image, the detection system is not very efficient in terms of computation cost. Moreover, the training of classifiers becomes difficult when the ratio of the sizes of the biggest character and smallest is varying more than 2 times and also when both the grayscale values of character and background are unknown.

Therefore, there are two problems in obtaining an efficient and robust text detection system using machine learning methods. One is how to avoid performing computational intensive classification on the whole image, the other is how to reduce the variances of character size and grayscale in feature space before training. In this paper, we address these problems by proposing a localization/verification scheme. In this scheme, text blocks are quickly extracted in images with a low rejection rate. This localization process allows us to further extract individual text lines and normalize the size of the text. We then perform precise verification in a set of feature spaces that are invariant to text grayscale changes.

2. Text localization

The direct application of common classifiers, such as k-nearest neighbors (k-NN), multilayer perceptrons (MLP) or support vector machine (SVM) for text detection in complex backgrounds produces a high false alarm rate and a rejection rate [12]. The difficulty of the classification mainly comes from the large variability of the text sizes, for example the biggest character is 5 times as the smallest one in video sequences. A solution is to normalize the text size before performing the classification task. However, this normalization requires the prior localization of the text. To avoid this chicken-and-egg problem, approximate locations of text can be firstly provided by using the following localization procedure with a low CPU cost and, most importantly, a low rejection rate.

2.1. Candidate text regions extraction

Let S denote the set of sites (pixels) in an input image. The task of extracting text-like regions, without recognizing individual characters, can be addressed by estimating at each pixel (x,y) $((x,y)\in S)$ in an image I the probability $P(T|(x,y),I)$ of belonging to a text block and then grouping the pixels with high probabilities into regions. In order to have a fast algorithm, we exploit the fact that text regions contain short edges in vertical and horizontal orientations, and that these edges are connected to each other due to the connections between character strokes.

First, vertical and horizontal edges are detected individually by finding directional zero crossing maps C_v and C_h produced by a Canny filter [2]. Let $C_\mathrm{v}(x,y)=1$ and $C_\mathrm{h}(x,y)=1$, respectively, indicate the vertical and horizontal zero crossing points, and let $C_\mathrm{v}(x,y)=0$ and $C_\mathrm{h}(x,y)=0$ indicate all the non-zero crossing points. Then, according to the type of edge (vertical or horizontal), different mathematical morphology operators [20], namely dilation, are used so that the vertical edges are connected in horizontal direction while horizontal edges are connected in vertical direction:

$$D_\mathrm{v}(x,y)=C_\mathrm{v}(x,y)\oplus \mathrm{Rect}_\mathrm{v} \quad \text{and}$$
$$D_\mathrm{h}(x,y)=C_\mathrm{h}(x,y)\oplus \mathrm{Rect}_\mathrm{h}. \tag{1}$$

The dilation operators Rect_v and Rect_h are defined to have the rectangle shapes 5×1 and 3×6, as illustrated in Fig. 1. Fig. 2(b and c) displays the vertical and horizontal edges resulting of this process for the video frame showed in Fig. 2(a).

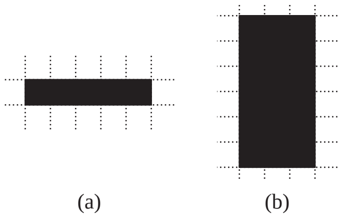

Fig. 1. (a) 5 × 1 vertical edge dilation operator and (b) 3 × 6 horizontal edge dilation operator

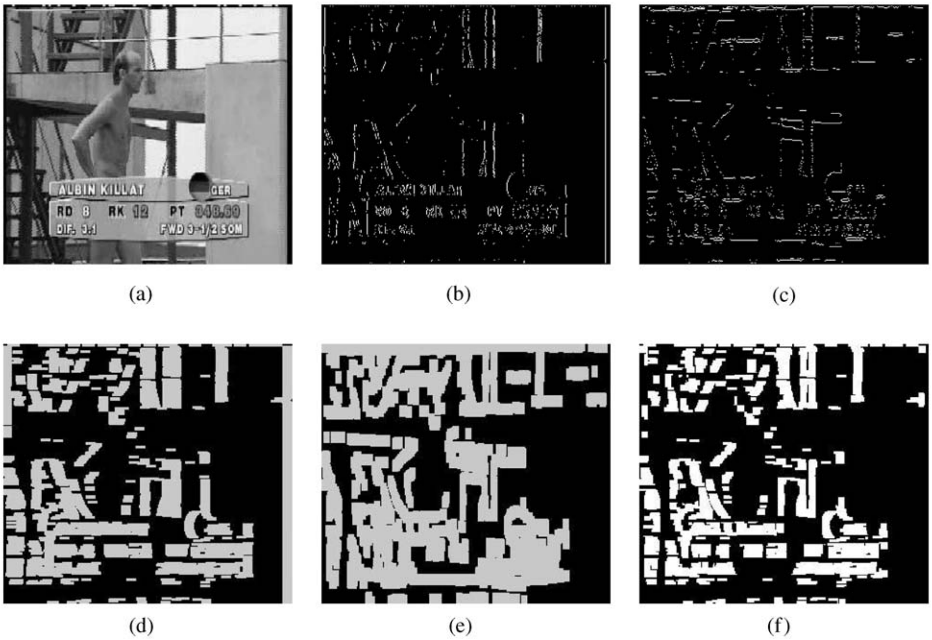

Fig. 2. Candidate text region extraction: (a) original image; (b) vertical edges detected in image (a); (c) horizontal edges detected in image (a); (d) dilation result of vertical edges using 5 × 1 vertical operator; (e) dilation result of horizontal edges using 3 × 6 horizontakl operator; (f) candidate text regions.

The vertical and horizontal edge dilation results are shown in Fig. 2(d and e). We consider only the regions that are covered by both the vertical and horizontal edge dilation results as candidate text regions. Thus, the probability $P(T|(x,y),I)$ can be estimated as

$$P(T|(x,y),I) = D_{\mathrm{v}}(x,y)D_{\mathrm{h}}(x,y). \tag{2}$$

Fig. 2(f) illustrates the result of this step.

The above text detection procedure is fast and invariant to text intensity changes. It works for reliablely detecting text with a height between 8 and 35 pixels, and even more (see character size distribution Fig. 6) on some images (for images with a 360 × 288 resolution). This size range is wide enough to account for the large majority of text encountered in video frames and other documents. However, there exist texts with even larger size, especially in journal covers. An extension of the proposed localization algorithm for these large characters consists of applying the algorithm on scaled image pyramid, as described in [29].

Also, another important point is that the threshold of the edge detection step can be set such that no true text regions will be rejected.

2.2. Text size normalization

In order to normalize text sizes, we need to extract individual text lines from paragraphs in candidate text regions. This task can be carried on by detecting the top and bottom baselines of horizontally aligned text strings. The baseline detection has also two additional purposes: on one hand, to eliminate false alarms, such as slant stripes, which do not contain any well defined baselines; and on the other hand, to refine the location of text strings in candidate regions that contain text connected with some background objects.

Baseline detection starts by computing the Y-axis projection $h(y)$, where $h(y)$ denotes the number of text pixels in line y and the fill-factor F defined as the density of text pixels inside the bounding box of the candidate text region. If the

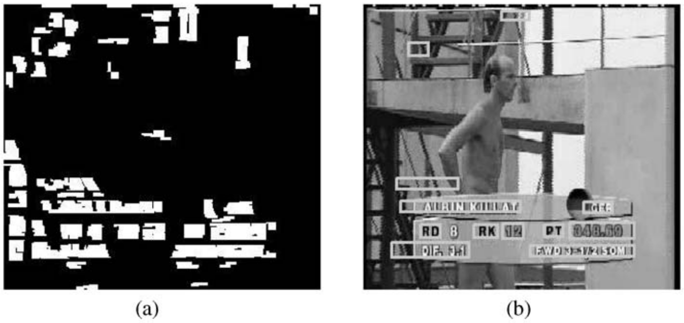

Fig. 3. Text line localization: (a) candidate text region with located baselines (top and bottom boundaries) and (b) the rectangle boundaries of candidate text line.

fill-factor is too low, we iteratively split the region using horizontal lines until the fill-factor of all the regions is above a given threshold TF, in our case TF = 70%. The splitting lines are located using the three following algorithms, applied sequentially:

1. *Varying length splitting*: This algorithm aims at splitting region containing text strings of different lengths or text strings connected with background objects whose length is usually shorter than that of text strings. We select the Y-coordinate y_0 which has the maximum absolute derivative of $h(y)$. If this maximum derivative is above a given threshold t_g and $h(y_0)$ is below 50% of the length of the longest line in the region, we split the region at line y_0.
2. *Equal length splitting*: When a region consists of two text lines of similar lengths, it may be split using Otsu's thresholding method [16]. Considering $h(y)$ as a one-dimension histogram, Otsu's method finds the threshold (line number y_0) that minimizes the intra-class variance of the two text lines. Then, if $h(y_0)$ is less than 50% the longest line in this region, we split the region at line y_0.
3. *Baseline refinement*: If a region cannot be split by the above two algorithms, we assume that it may contain only one text line, and we refine the top and bottom boundaries (baselines) of the region to yield more precise location. To this end, we search for the greatest region (in height) whose fill-factor is above the given threshold TF (see [3] for details).

Fig. 3(a) demonstrates the result of performing the above text line extraction step in Fig. 2(f). Typical characteristics of text string are then employed to select the resulting regions and the final candidate text line should satisfy the following constraints: it must contain between 75 and 9000 pixels; the horizontal–vertical aspect ratio must be more than 1.2; the height of the region must be above 8 pixles. Fig. 3(b) shows the rectangle boundaries of the candidate text lines.

3. Text verification

As in many other works, the text localization procedure described in the previous subsection is rather empirical and may therefore produce false alarms (i.e. non-text regions). To remove these false alarms in the candidate text lines, we use trained verifiers using machine learning methods on both positive (text) and negative (false alarms) examples resulting from the localization step. There are two kinds of machine learning methods based on either empirical risk minimization, or structural risk minimization. The empirical risk minimization based methods, e.g. MLP [1], minimize the error over the data set. On the contrary, structural risk minimization based methods, such as SVM [27], aim at minimizing a bound on the generalization error of a model in a high dimensional space rather than minimizing error over data set. The training examples that are far away from the decision hyperplanes will not change the support vectors, which may indicate a potentially better generalization on unseen backgrounds. In this section, both MLP and SVM are tested in the task of text verification.

3.1. Feature extraction

After the text localization step, each candidate text line is normalized using bilinear interpolation into an image I having a 16 pixels height (see Fig. 4 for some examples). A feature image I_f of the same size is then computed from I. The fixed size input feature vectors z_i to the MLP or SVM are directly extracted from I_f on 16×16 windows W sliding over the image grid S of these normalized images. Since the grayscale values of text and background are supposed to be unknown, we tested four different kinds of features invariant to the absolute grayscale values to produce the feature image. These four alternatives are now described.

A1. *Spatial derivatives*: In order to measure the contribution of the contrast in the text verification process, we employ the spatial derivatives of the image as the first feature. For each pixel in sliding window, we compute its derivatives in x and y directions. Therefore, a feature vector of a sliding window (256 dimensions) has 512 dimensions.

A2. *Distance map*: Since the grayscale values of both characters and backgrounds are varying, the derivatives give out different values. In fact, contrast of text character is background dependent, which implies that the contrast may not be a stable feature for text verification. Thus, we considered as a second feature image the distance map DM, which only relies on the position of strong edges in the image. DM is defined by Toriwaki and Yokoi [26]

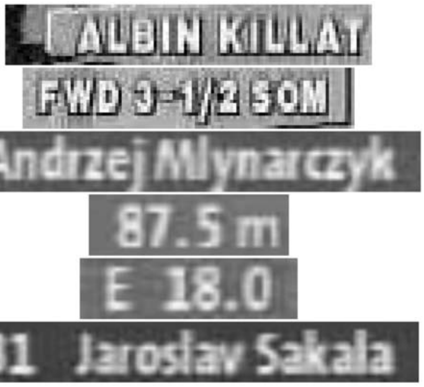

Fig. 4. Normalized text lines.

$$\forall (x,y) \in S, \quad \mathrm{DM}(x,y) = \min_{(x_i,y_i) \in E} d((x,y),(x_i,y_i)) \tag{3}$$

where $E \subseteq S$ is a set of edge points, and d is a distance function, in our case the Euclidean distance. Although the distance map is independent of the grayscale value of characters, the edge set E still relies on the contrast between text and background and the threshold employed in edge detection.

A3. *Constant gradient variance features*: To avoid the need for setting any threshold, we propose a feature called constant gradient variance (CGV). The variance normalization technique is usually used to enhance grayscale images or to preprocess input data in speech [15,19]. Here, we apply this technique on gradient image to normalize the contrast at a given point using the local contrast variance computed in a neighborhood of this point. More formally, let $g(x,y)$ denote the gradient magnitude at pixel (x,y), and let $\mathrm{LM}(x,y)$ (resp. $\mathrm{LV}(x,y)$) denote the local mean (reps. the local variance) of the gradient defined by

$$\mathrm{LM}(x,y) = \frac{1}{|\mathcal{G}_{x,y}|} \sum_{(x_i,y_i) \in \mathcal{G}_{x,y}} g(x_i,y_i) \tag{4}$$

and

$$\mathrm{LV}(x,y) = \frac{1}{|\mathcal{G}_{x,y}|} \sum_{(x_i,y_i) \in \mathcal{G}_{x,y}} (g(x_i,y_i) - \mathrm{LM}(x,y))^2, \tag{5}$$

where $\mathcal{G}_{x,y}$ is a 9×9 neighborhood around (x,y). Then, the CGV value at pixel x,y is defined as

$$\mathrm{CGV}(x,y) = (g(x,y) - \mathrm{LM}(x,y)) \sqrt{\frac{\mathrm{GV}}{\mathrm{LV}(x,y)}}, \tag{6}$$

where GV denotes the global gradient variance computed over the whole image grid S. Assuming that $g(x,y) \sim \mathcal{N}(\mathrm{LM}(x,y), \mathrm{LV}(x,y))$, i.e. follows a normal law with $\mathrm{LM}(x,y)$ mean and $\mathrm{LV}(x,y)$ variance, it is easy to show that

$$\mathbb{E}[\mathrm{CGV}(x,y)] = 0 \tag{7}$$

and

$$\mathbb{E}[(\mathrm{CGV}(x,y))^2] = \mathrm{GV}, \tag{8}$$

where $\mathbb{E}$ denotes the expectation operator. Statistically, each local region in the CGV image thus has the same contrast variance. Note, however, that a site with a high CGV value still corresponds to an edge with a high local brightness contrast. In general, this method will also enhance the noise in regions with a uniform grayscale value. However such regions will be very rare in our case since the localization step only provides candidate text images that contain many vertical and horizontal edges.

A4. *DCT coefficients*: The last feature vector we used is composed of discrete cosine transform (DCT) coefficients mainly for comparison with the above features. This feature is widely used in JPEG and MPEG compression scheme and is a representative feature in the frequency domain. The DCT coefficients can be computed using a fast DCT algorithm presented by Feig [7].

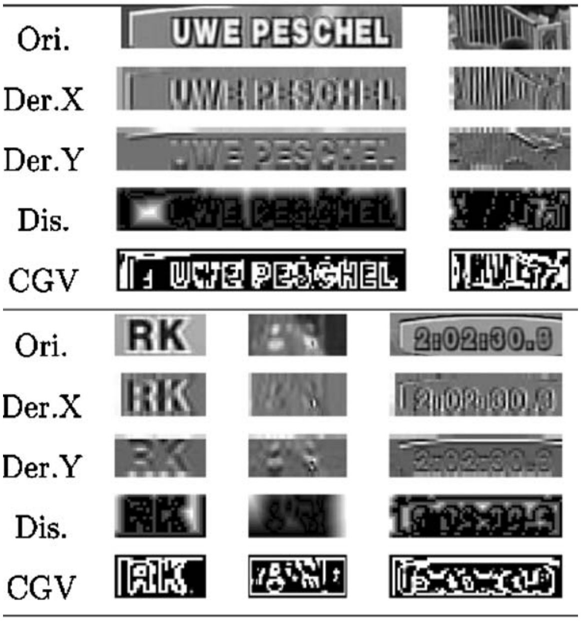

Fig. 5. Examples of three training features. The grayscale values shown in the images are scaled into the range of 0–255 for display reasons. DCT feature images are not shown in this figure because of they are not very visually meaningful.

Fig. 5 illustrates some examples of the derivative features, the distance map feature and the constant variance feature. DCT feature images are not shown in this figure because of they are not very visually meaningful. It can be seen that the constant gradient variance features provide similar values around the characters for text of different grayscale values, such as images "UWE PESCHEL" and "RK".

3.2. Multilayer perceptrons (MLP)

MLP is a widely used neural network that consists of multiple layers (an input layer, hidden layers and an output layer) of neurons. Each neuron computes a weighted sum of the output of the neurons in the previous layer, followed by a activity function. In our experiments, the MLP contains only one hidden layer and the activity function is selected as a Sigmoid function:

$$f(t) = \frac{1}{1+\exp(-t)}. \tag{9}$$

Therefore, for each input vector $\mathbf{z} = (z_1, z_2, \ldots, z_m)$, the output O_i^{h} of the neuron i in the hidden layer is defined as

$$O_i^{\mathrm{h}} = f\left(\sum_{j=1}^{m} w_{ij}^{\mathrm{h}} \cdot z_j + b_i^{\mathrm{h}}\right) \tag{10}$$

and the output O_k^{o} of a neuron k in the output layer is defined as

$$O_k^{\mathrm{o}} = \sum_{i=1}^{n_{\mathrm{h}}} w_{ki}^{\mathrm{o}} \cdot O_i^{\mathrm{h}} + b_k^{\mathrm{o}}. \tag{11}$$

The number of hidden neurons n_{h} is decided by using cross-validation during the training process, which will be described below.

3.3. Support vector machine (SVM)

SVM is a technique motivated by statistical learning theory and has been successfully applied to numerous classification tasks [6,10]. The key idea of SVM is to implicitly project the input space into a space of higher dimension (possibly infinite) where the classes are more linearly separable. This projection is implicit because the learning and decision processes only involve inner dot product in the feature space that can be computed using a kernel. An extensive discussion of SVMs can be found in [27]. In this paper, we will only consider a binary classification task with n_t labeled training examples: $(\mathbf{z}_1, y_1), (\mathbf{z}_2, y_2), \ldots, (\mathbf{z}_{n_t}, y_{n_t})$, where $\mathbf{z}_i$ denote an input vector and $y_i = \pm 1$ indicates its class label.

If the data set is linearly separable, there exists at least one hyperplane $\mathbf{w} \cdot \mathbf{z} + b = 0$ (decision surface) that separates all the training examples:

$$y_i(\mathbf{z}_i \cdot \mathbf{w} + b) \geqslant 1 \quad \forall i, \tag{12}$$

where $\mathbf{w}$ and b are normalized by the following constraint:

$$\min_{i=1,\ldots,n_t} |\mathbf{w} \cdot \mathbf{z}_i + b| = 1. \tag{13}$$

The margin of such a hyperplane $\mathbf{w} \cdot \mathbf{z} + b = 0$ is defined by the sum of the shortest distance from the hyperplane to the closest positive example and the shortest distance from the hyperplane to the closed negative example. This margin is simply $2/\|\mathbf{w}\|$, where $\|\mathbf{w}\|$ is the Euclidean norm of $\mathbf{w}$. Thus, the maximum margin can be obtained by minimizing $\|\mathbf{w}\|^2$ subject to the constraints Eq. (12). The optimal $\mathbf{w}^*$ can be defined by using a set of weight λ_j on the training examples as

$$\mathbf{w}^* = \sum_{j=1}^{n_t} \lambda_j y_j \mathbf{z}_j. \tag{14}$$

Then, the classification of an unknown example $\mathbf{z}$ is based on the sign of the function:

$$G(\mathbf{z}) = \sum_{j=1}^{n_t} \lambda_j y_j \mathbf{z} \cdot \mathbf{z}_j + b^*, \tag{15}$$

where $b^* = y_j - \mathbf{w}^* \cdot \mathbf{z}_j,\ \lambda_j \neq 0$.

This method can be generalized to the nonlinear case through the mapping of the training examples $\mathbf{z}_i$ into an alternative high dimensional feature space $\phi(\mathbf{z}_i)$ implicitly achieved by choosing a kernel: $K(\mathbf{z}_i, \mathbf{z}_j) = \phi(\mathbf{z}_i) \cdot \phi(\mathbf{z}_j)$. The decision task is then simply transformed into:

$$G(\mathbf{z}) = \sum_{j=1}^{n_t} \lambda_j y_j K(\mathbf{z}, \mathbf{z}_j) + b^*. \tag{16}$$

3.4. Training

The data base we used consists of text and non-text examples resulting from the localization step. We divided the database equally into a training set and a test set. Training and testing were performed by using a MLP or a SVM classifier.

In the case of MLP, the network consists of an input layer, a hidden layer and an output indicating classes of text and non-text. The MLP is trained by the Back-propagation algorithm. The number of hidden neurons, which indicate the capacity of the MLP, is chosen by performing a K-fold cross validation on the training set. It is described below.

In the case of SVM, the training is performed by using a quadratic programming algorithm. As a choice for the kernel, we use typical Radial basis function (RBF), defined by

$$K(\mathbf{z}_1, \mathbf{z}_2) = \mathrm{e}^{(-\|\mathbf{z}_1 - \mathbf{z}_2\|^2)/2\sigma^2}, \tag{17}$$

where the kernel bandwidth σ is a parameter that is decided by using K-fold cross-validation. The K-fold cross-validation using a SVM as an example can be described like that:

1. Partition the training data set into K parts of equal size, refer to as "folds". Then, assign each fold a possible value of σ.
2. For $i = 1$–K, train the SVM using all the folds except the ith as the training set and the ith σ as parameter. Evaluate the error of the output hypothesis using the ith fold as the test set.
3. Take the value of σ corresponding to the lowest error rate computed in (3) as the optimal parameter and train the SVM using all the training data to get a good support vector set.

3.5. Text-line verification

In the text verification step, the feature vectors discussed in Section 3.1 and provided to the classifier are extracted from the normalized candidate text line on 16 × 16 sliding windows with a slide step of 4 pixels. Thus, for each candidate text line $\mathbf{r}$, we obtained a set of feature vectors $Z_r = (\mathbf{z}_1^r, \ldots, \mathbf{z}_l^r)$. Let $G(\mathbf{z}_i^r)$ denotes the output of the MLP or the magnitude of the SVM, which indicates the confidence that the vector $\mathbf{z}_i^r$ belongs to a text line. The confidence of the whole candidate text line $\mathbf{r}$ is then defined as

$$\mathrm{Conf}(\mathbf{r}) = \sum_{\mathbf{z}_i^r \in Z_r} G(\mathbf{z}_i^r).\frac{1}{\sqrt{2\pi\sigma_0}} \mathrm{e}^{-d_i^2/2\sigma_0^2}, \tag{18}$$

where d_i is the distance from the geometric center of the ith sliding window to the geometric center of the text line $\mathbf{r}$, and σ_0 is a scale factor depending on the text line length. Finally, the candidate text line $\mathbf{r}$ is classified as a real text region if $\mathrm{Conf}(\mathbf{r}) \geqslant 0$.

4. Experiments

In this section, we first report experiments for evaluating the performance of the raw text verification algorithm (i.e. based on a single input vector, not using Eq. (18)) to find out which classifier/feature combination is the most powerful. Then we evaluate the performance of the whole localization/verification scheme using the optimal feature and classifier combination.

4.1. Feature and classifier evaluation

The text verification algorithms using a combination of the proposed features and classifiers were designed and trained on a database consisting of still images and videos recorded from TV. The videos include advertisements, sports, interviews, news, movies. The still images include covers of Journals, maps, and flyers. Each video frame or image has 352 × 288 or 720 × 576 resolution in JPEG or MPEG-1,2 formats. Only the grayscale information is used in the experiments.

The feature extraction, training and testing procedures described in Section 3 were applied on the database. More precisely, 2400 candidate text regions containing both true text lines and false alarms were randomly selected from the output of the text localization step applied on this database. From these regions the feature extraction step produced 76,470 vectors for each of the four kinds of features. It was ensured that the test set and the training set contained vectors extracted from the same windows (i.e. same image and location) in all the experiments, where one experiment is characterized by a couple (classifier, feature).

Table 1
Error rates of the SVM and MLP classifiers for the text verification task

Training tools	DIS(%)	DERI(%)	CGV(%)	DCT(%)
MLP	5.28	4.88	4.40	4.72
SVM	2.56	3.99	1.07	2.0

DIS denotes the distance map feature. DERI denotes the grayscale spatial derivative feature. CGV denotes the constant gradient variance feature. DCT denotes the DCT coefficients.

Table 1 lists the error rate measured on the test set for each of the four features and for each classifier. First, the error rate in these results are very low. The proposed scheme is also about 5 times faster than the typical MLP detection methods. Second, whatever the considered feature, the SVM classifier gives better results than the MLP classifier. This might be explained by the nature of the SVM classifier, which optimizes a bound on the generalization error rather than the empirical risk. Finally, the proposed constant gradient variance feature provides the best result. It is likely due to its better invariance to text/background contrast.

We have done other text verification experiments using the concatenation of the four features as input vector to the classifier. We gained a 0.3% error reduction. However, due to the additional complexity (feature extraction, score computation), we considered that using the best feature/classifier combination (i.e. SVM with CVG features) was a more sensible choice.

4.2. System evaluation

The evaluation of the text detection performance is closely related to the ratio of text strings and backgrounds in the evaluation data base. In one extreme case, applying a text detection algorithm on a data base that contains no text will only produce false alarms. In the other extreme case, a data base with images and video frames all containing text lines will more exhibit good presicion performance in comparison. Considering this issue, both the text localization step

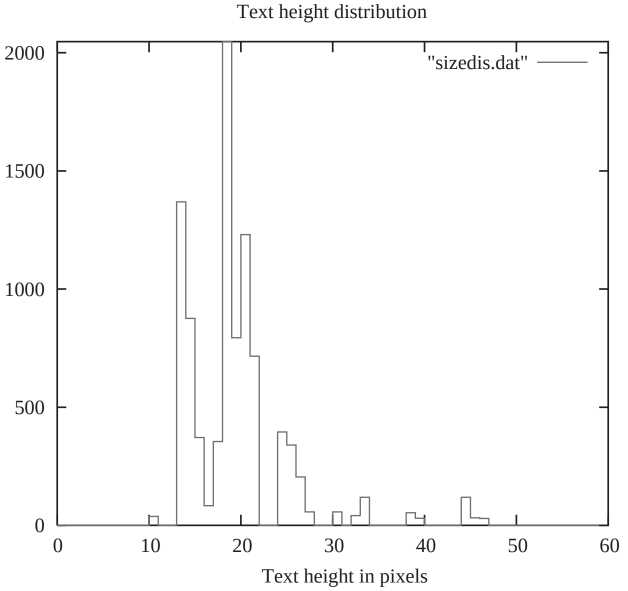

Fig. 6. The histogram of text string heights in the considered News program (height unit is pixel).

and the verification step (in the next subsection) are evaluated on a half an hour video containing a Belgian news program[1] in French provided by Canal+ in the context of the CIMWOS[2] European project. This video contains captions and some scene text (about 7% of all text strings) as well as many frames that have no text. It is believed that the video is representative of general video frames (sports and documentary) and more other documents. The sizes of text strings in this video is plotted in Fig. 6 in terms of the height of text strings in pixels. To make sure that the proposed detection scheme can be also applied in images or key frame, we are not using any temporal filtering.

4.2.1. Evaluation of text localization

The performance of the text localization step is first measured in terms of pixel recall rate (PRR), pixel false alarm rate (PFR) and CPU cost. The recall rate is defined as

$$\mathrm{PRR} = \frac{\mathrm{RP}}{\mathrm{TP}}, \tag{19}$$

where RP denotes the total number of text pixels recalled by the algorithm, and TP is defined as the total number of text pixels in the ground truth. The false pixel alarm rate is defined as

$$\mathrm{PFR} = \frac{\mathrm{PF}}{\mathrm{PI}}, \tag{20}$$

where PF denotes the number of false alarm pixels and PI denotes the total number of pixels in the images. The computational cost is measured by numbers of addition and multiply operation per pixel in average.

At the lower (pixel) level, we compare the performance of the proposed algorithm with two other algorithms, namely the derivative texture algorithm [29] and the vertical edge-based algorithm [23] in Table 2, which can provide pixel-level localization and are often adopted in recent text detection systems [17,28]. These two algorithms are implemented by ourselves according to the referenced papers. It can be observed that the proposed feature, which is based on short and connecting vertical and horizontal edges, yields the highest recall rate. The computation cost of the proposed method is lower than the derivative texture algorithm and similar to the vertical edge-based method.

[1] From the Radio-Télévision Belge d'expression Française (RTBF).

[2] Combined image and word spotting.

Table 2
Performances and running costs of different text detection techniques

iX-based	PRR(%)	PFR(%)	CPU costs
Derivative texture	90.54	3.48	225(×), 325(+)
Vertical edge	86.42	16.17	34(×), 35(+)
Proposed	94.51	1.73	36(×), 44(+)

RPR denotes the recall pixel rate and FPR denotes the false pixel alarm rate. The computational cost is measured by numbers of addition and multiply operation per pixel in average.

At the higher level, the baseline detection leads to regions (text lines). The further results at this level show that although the use of the horizontal edges increases the computation load, it also saves time by producing less false alarm regions, thus reducing the cost of the connected component analysis and baseline detection steps.

For additional evaluation, we counted the text strings that were correctly located. A set of criterions is designed based on region level, which makes more sense for text recognition and content-based retrieval tasks. Region recall rate (RRR) is defined as

$$\mathrm{RRR} = \frac{\mathrm{RR}}{\mathrm{RT}}, \tag{21}$$

where RR denotes the total number of correct text regions located by the algorithm, and RT is defined as the total number of text regions in the ground truth.

Region precision rate (RPR) is defined as

$$\mathrm{RPR} = \frac{\mathrm{RR}}{\mathrm{RE}}, \tag{22}$$

where RE denotes the total number of text regions extracted by the text localization algorithm. A ground-truth text region is considered to be correctly located if it has an 80% overlap with one of the detected string regions. With the proposed method, we extracted 9369 text lines and 7537 false alarms in the CIMWOS data base. There were no rejected regions. The precision of this localization step on this database is $9369/(9369 + 7537) = 55.4\%$ as shown in Table 3.

Table 3
Performance of the proposed text localization method alone on the CIMWOS database

	RRR(%)	RPR(%)
Ideal result	100	100
Proposed method	100	55.4

RRR denotes the region recall rate and RPR denotes the region extraction precision rate.

Table 4
Performance of the localization/verification scheme on the CIMWOS database

	RRR(%)	RPR(%)
Ideal result	100	100
Proposed method	99.76	97.0

RRR denotes the region recall rate and RPR denotes the region extraction precision rate.

4.2.2. Evaluation of the text verification

The SVM classifier together with the CGV feature was employed to verify the extracted text regions of the CIMWOS data base, based on the confidence value given by Eq. (18). This verification scheme removed 7255 regions from the 7537 false alarm regions while only reject 23 true text lines, which gives a 99.76% RRR and a 97% RPR as listed in Table 4.

Fig. 7 shows examples of detected text on some images of our general database. As can be seen, the algorithm is able to locate different text strings, with different colors and background, without reporting false alarms even in the presence of strong text texture like patterns.

Finally, let us mention that text recognition experiments have also been performed on the CIMWOS database, using the multiple hypotheses based segmentation approach proposed in [5]. A 97% character recognition rate and a 85% word recognition rate were obtained, showing that the precision of the detected text line regions is sufficient enough in order to achieve good recognition results.

Many results in terms of recall and precision are also reported in the literature. A 94.7% recall and

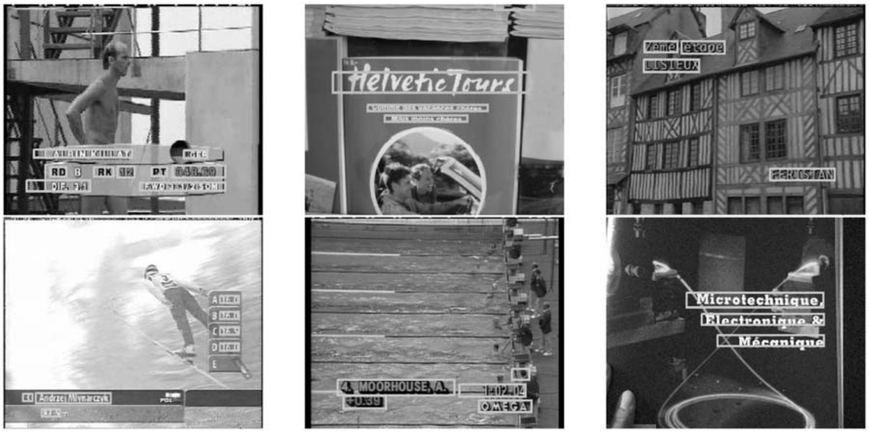

Fig. 7. Detected text regions in images or video frames.

100% precision are reported for video frames in [9]. A 97–100% recall and 100% precision are reported on five video sequences in [21]. A DCT-based method [31] reported a 99.2% recall and about 98.2% precision. Lienhart's MLP method [12] reported a 94.7% recall and 84% precision for video frames. Unfortunately, these results are not comparable because they are based on different databases.

5. Conclusion

Applying machine learning methods for text detection encounters difficulties of large range of character sizes, grayscale and contrast variations, and heavy computation cost. In this paper, a localization/verification scheme was proposed to overcome these problems. In the proposed scheme, the text detection problem was addressed by performing two subtasks, namely, text localization and text verification. In the first task, we extracted candidate text regions from images using a fast algorithm, so that the size of characters over a large range from 8 up to 45 pixels can be normalized into a unique scale. A vertical/horizontal edge based method was proposed in this task, and combined with a baseline detection technique, to perform fast localization with a very low rejection rate and a proper false alarm rate. In the verification task, background independent features are proposed for training MLP and SVM to remove the false alarms.

Our experiments have shown that the proposed scheme can improve the text detection greatly as compared with applying the same machine learning tools without performing size normalization. Furthermore, SVM obtains a better performance than MLP for addressing text texture verification problem in using any of the four features. Finally, within the four proposed features in the verification, the constant gradient variance feature provides the best performance in characterizing text textures of unknown grayscale values.

The presented localization/verification scheme leads to a 99.76% recall rate and a 97% precision rate for text detection in images or videos. Further recognition algorithms [5] able to recognize text of any grayscale values in complex background yielded a 97.1% character recognition rate and a 93.8% word recognition rate in the detected text area.

Acknowledgements

This work has been performed partially within the "Combined Image and Word Spotting (CIMWOS)" project granted by the European IST Programme, and the "Video Optical Character Recognition" (VOCR) project granted by the Swiss National Fund for Scientific Research.

The authors would like to thank Samy Bengio and Ronan Collobert for their help on this work.

References

[1] J.A. Anderson, A. Pelionisz, E. Rosenfeld, Neurocomputing, MIT Press, Cambridge, MA, 1990.

[2] J.F. Canny, A computational approach to edge detection, IEEE Trans. Pattern Anal. Mach. Intell. 8 (1) (1986) 679–698.

[3] D. Chen, H. Bourlard, J.-P. Thiran, Text identification in complex background using SVM, in: International Conference on Computer Vision and Pattern Recognition, 2001, pp. 621–626.

[4] D. Chen, J. Odobez, Sequential monte carlo video text segmentation, in: International Conference on Image Processing, 2003.

[5] D. Chen, J. Odobez, H. Bourlard, Text segmentation and recognition in complex background based on markov random field, in: International Conference on Pattern Recognition, Vol. 2, Quebec City, Canada, 2002.

[6] R. Collobert, S. Bengio, Y. Bengio, A parallel mixture of svms for very large scale problems, Neural Comput. 14 (5). URL citeseer.nj.nec.com/collobert02parallel.html

[7] E. Feig, S. Winograd, Fast algorithms for the discrete cosine transform, IEEE Trans. Signal Process. 40 (28) (1992) 2174–2193.

[8] C. Garcia, X. Apostolidis, Text detection and segmentation in complex color images, in: International Conference on Acoustics, Speech and Signal Processing, 2000, pp. 2326–2329.

[9] A.K. Jain, B. Yu, Automatic text localisation in images and video frames, Pattern Recognition 12 (31) (1998) 2055–2076.

[10] K. Jonsson, J. Kittler, Y. Li, J. Matas, Support vector machines for face authentication, in: British Machine Vision Conference, Nottingham, UK, 1999, pp. 543–553.

[11] H. Li, D. Doermann, Text enhancement in digital video using multiple frame integration, in: ACM Multimedia, Orlando, FL, 1999, pp. 385–395.

[12] R. Lienhart, A. Wernicke, Localizing and segmenting text in images and videos, IEEE Trans. Circuits Systems Video Technol. 12 (4) (2002) 256–268.

[13] B. Manjunath, W. Ma, Texture features for browsing and retrieval of image data, IEEE Trans. Pattern Anal. Mach. Intell. 18 (8) (1996) 837–842.

[14] F. Mokhtarian, S. Abbasi, J. Kittler, Robust and efficient shape indexing through curvature scale space, in: British Machine Vision Conference, 1996, pp. 9–12.

[15] N. Morgan, D. Ellis, E. Fosler, A. Janin, B. Kingsbury, Reducing errors by increasing the error rate: Mlp acoustic modeling for broadcast news transcription, in: DARPA Broadcast News Workshop, Herndon, VA, 1999.

[16] N. Otsu, A threshold selection method from gray-level histograms, IEEE Trans. Systems Man Cybernet. 1 (9) (1979) 62–66.

[17] T. Sato, T. Kanade, E.K. Hughes, M.A. Smith, Video OCR for digital news archives, in: IEEE Workshop on Content Based Access of Image and Video Databases, Bombay, 1998, pp. 52–60.

[18] T. Sato, T. Kanade, E.K. Hughes, M.A. Smith, S. Satoh, Video OCR: indexing digital news libraries by recognition of superimposed caption, in: ACM Multimedia System Special Issue on Video Libraries, 1998, pp. 52–60.

[19] B. Schiele, J.L. Crowley, Object recognition using multidimensional receptive field histograms, in: ECCV (1), Cambridge, UK, 1996, pp. 610–619.

[20] J. Serra, Image Analysis and Mathematical Morphology, Academic Press, London, 1982.

[21] J.-C. Shim, C. Dorai, R. Bolle, Automatic identification of text in digital video key frames, in: Proceedings of ICPR, Brisbane, Australia, 1998, pp. 618–620.

[22] M.A. Smith, T. Kanade, Video skimming for quick browsing based on audio and image characterization, Technical Report CMU-CS-95-186, Carnegie Mellon University, July 1995.

[23] K. Sobottka, H. Bunke, H. Kronenberg, Identification of text on colored book and journal covers, in: International Conference on Document Analysis and Recognition, 1999, pp. 57–63.

[24] R.K. Srihari, Z. Zhang, A. Rao, Intelligent indexing and semantic retrieval of multimodal documents, Inform. Retrieval 2 (2/3) (2000) 245–275.

[25] M. Swain, H. Ballard, Color indexing, Int. J. Comput. Vision 7 (1991) 11–32.

[26] J. Toriwaki, S. Yokoi, Distance transformations and skeletons of digitized pictures with applications, in: L.N. Kanal, A. Rosenfeld (Eds.), Progress in Pattern Recognition, North-Holland, Amsterdam, 1981, pp. 187–264.

[27] V. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.

[28] E. Wong, M. Chen, A new robust algorithm for video text extraction, Pattern Recognition 36 (6) (2003) 1397–1406.

[29] V. Wu, R. Manmatha, E.M. Riseman, Finding text in images, in: Proceedings of the ACM International Conference on Digital Libraries, 1997, pp. 23–26.

[30] Y. Zhong, K. Karu, A.K. Jain, Locating text in complex color images, Pattern Recognition 10 (28) (1995) 1523–1536.

[31] Y. Zhong, H. Zhang, A. Jain, Automatic caption localization in compressed video, IEEE Trans. Pattern Anal. Mach. Intell. 22 (4) (2000) 385–392.