IDIAP RESEARCH REPORT

Text Detection and Recognition in Images and Videos

Datong Chen, Jean-Marc Odobez and H. Bourlard
IDIAP, Switzerland
chen, odobez, bourlard@idiap.ch

IDIAP–RR 02-61

Dec. 2002

Institut Dalle Molle d'Intelligence Artificielle Perceptive • CP 592 • Martigny • Valais • Suisse

téléphone +41 − 27 − 721 77 11
télécopieur +41 − 27 − 721 77 12
adr.él. secretariat@idiap.ch
internet http://www.idiap.ch

Rapport de recherche de l'IDIAP 02-61

TEXT DETECTION AND RECOGNITION IN IMAGES AND VIDEOS

Datong Chen, Jean-Marc Odobez and H. Bourlard
IDIAP, Switzerland
chen, odobez, bourlard@idiap.ch

DEC. 2002

Abstract - Text embedded in images and videos represents a rich source of information for content-based indexing and retrieval applications. In this paper, we present a new method for localizing and recognizing text in complex images and videos. Text localization is performed in a two step approach that combines the speed of a focusing step with the strength of a machine learning based text verification step. The experiments conducted show that the support vector machine is more appropriate for the verification task than the more commonly used neural networks. To perform text recognition on the localized regions, we propose a new multi-hypotheses method. Assuming different models of the text image, several segmentation hypotheses are produced. They are processed by an optical character recognition (OCR) system, and the result is selected from the generated strings according to a confidence value computed using language modeling and OCR statistics. Experiments show that this approach leads to much better results than the conventional method that tries to improve the individual segmentation algorithm. The whole system has been tested on several hours of videos and showed good performance when integrated in a sports video annotation system and a video indexing system within the framework of two European projects.

Keywords: content-based indexing, text localization, text segmentation, text recognition, support vector machines, Markov random field, OCR, multiple hypotheses,.

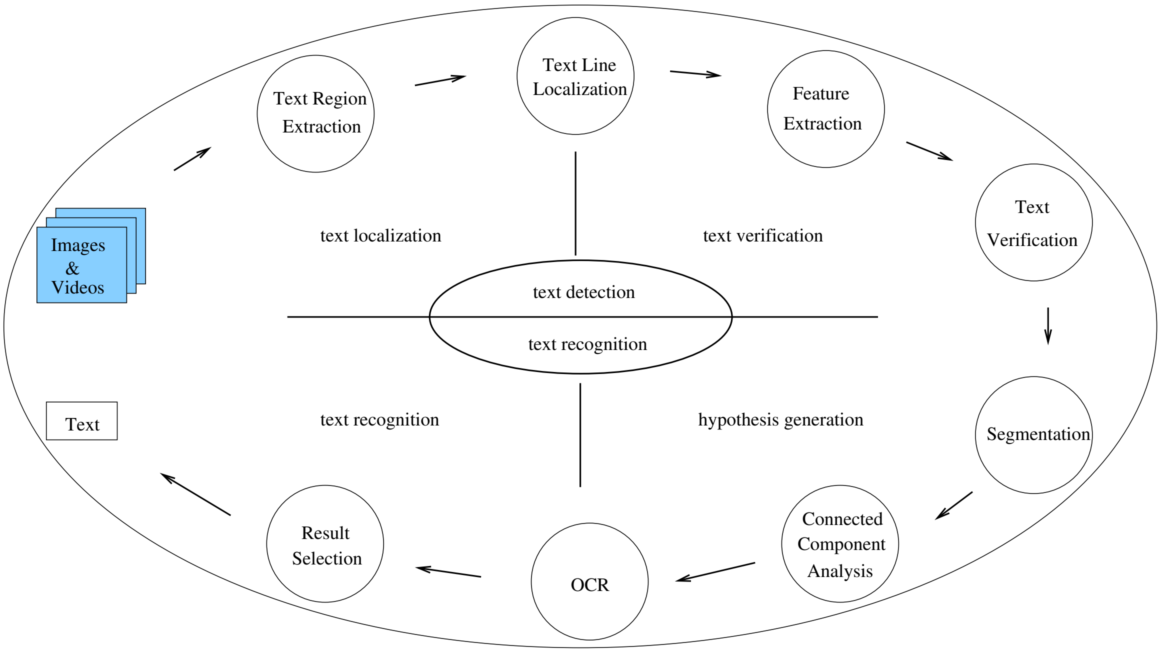

FIG. 1 – *Algorithm proposed for text detection and recognition.*

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 [16], or shape [17]. 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 [4] 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.

Text detection and recognition in images and videos, 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 text characters contained in images and videos can be any grayscale value (not always white), low resolution, variable size and embedded in complex backgrounds. Experiments show that applying conventional OCR technology directly leads to poor recognition rates. Therefore, efficient detection and segmentation of text characters from the background is necessary to fill the gap between image and video documents and the input of a standard OCR system. Previously proposed methods can be classified into bottom-up [14] [23] [29] and top-down methods [28] [13] [9] [29]. Bottom-up methods segment images into regions and then group "character" regions into words. The recognition performance therefore relies on the segmentation algorithm and the complexity of the images. Top-down algorithms first detect text regions in images and then segment each of them into text and background. They are able to process more complex images than bottom-up approaches but difficulties are still encountered at both the detection and segmentation/recognition stages.

The method we propose belongs to the top-down category, and consists of two main tasks as illustrated by Figure 1 : a text detection task and a text recognition task applied to the detected text regions. Following the cascade filtering idea, which consists of the sequential processing of data with more and more selective filters, the text detection task is decomposed into two subtasks. These are a text localization step, whose goal is to quickly extract potential text blocks in images with a very low missing rate and a reasonable false alarm rate, and a text verification step based on a powerful machine learning tool. Such an approach allows us to obtain high performances with a lower computational cost in comparixson to other methods.

To address the recognition task, we propose a multi-hypotheses approach. More precisely, the text image is segmented two or three times, assuming a different number of classes in the image each time. The different classes, all considered as text candidates, are processed by a commercial optical character recognition (OCR) engine and the final result is selected from the generated text string hypotheses using a confidence level evaluation based on language modeling. Additionally, we propose a segmentation method based on Markov Random Field to extract more accurate text characters. This methodology allowed us to handle background grayscale multimodality and unknown text grayscale values, problems that are often not taken into account in the existing literature. When applied to a database of several hours of sports video, it reduces by more than 50% the word recognition error rate with respect to a standard Otsu binarization step followed by the OCR.

The rest of the paper is organized as follows. Section 2 presents a more detailed review on text detection and segmentation/recognition. Section 3 describes the detection step, whereas Section 4 is devoted to the text recognition task. Section 5 describes our databases, which come from two European projects, together with the performance measures and the experimental results of our approach. Section 6 provides some discussion and concluding remarks.

2 Related work

In this section, we separately review the existing methods towards text detection and text recognition, as these two problems are often addressed separately in the literature.

2.1 Text detection

Text can be detected by exploiting the discriminate properties of text characters such as the vertical edge density, the texture or the edge orientation variance. One early approach for localizing text in covers of Journals or CDs [29] assumed that text characters were contained in regions of high horizontal variance satisfying certain spatial properties that 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 also produce many false alarms because many background regions may also have strong horizontal contrast. The method of Wu et al. [28] for text localization is based on texture segmentation. Texture features are computed at each pixel from the derivatives of the image at different scales. Using a K-means algorithm, pixels are classified into three classes in the feature space. The class with highest energy in this space indicates text while the two others indicate non-text and uncertainty. However, the segmentation quality is very sensitive to background noise and image content and the feature extraction is computationally expensive. More recently, Garcia et al. [9] proposed a new feature referred to as variance of edge orientation. This relies on the fact that text strings contain edges in many orientations. Variation of edge orientations was computed in local area from image gradient and combined with edge features for locating text blocks. The method, however, may exhibit some problems for characters with strong parallel edges characteristics such as “i” or “l”.

Besides the properties of individual characters, Sobettka 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 can be detected in order to assess the presence of a text string in an image block.

The above manually designed heuristic features usually perform fast detection but are not very robust when the background texture is very complex. As an alternative, a few systems considered machine learning tools to perform the text detection [13, 15]. These systems extracted wavelet [13] or derivative features [15] from fixed-size blocks of pixels and classified the feature vectors into text or non-text using artificial neural networks. However, since the neural network based classification was applied to all the possible positions of the whole image, the detection system was not efficient in terms of computation cost and produced unsatisfactory false alarm and rejection rates.

2.2 Text recognition review

Since commercial OCR engines achieve high recognition performance when processing black and white images at high resolution, almost all the methods in the literature that addressed the issue of text recognition in complex images and videos employed an OCR system. However, these OCR software can not be applied directly on regions previously extracted by a text localization procedure. Experience shows that OCR performance in this context is quite unstable, as already mentioned by others [14], and significantly depends on the segmentation quality, in the sense that errors made in the segmentation are directly forwarded to the OCR.

Some bottom-up based techniques addressed the segmentation problem for text recognition. For instance, Lienhart [14] and Bunke [23] clustered text pixels from images using standard image segmentation or color clustering algorithm. Although these methods can somehow avoid explicit text localization, they are very sensitive to character size, noise and background patterns. On the other hand, most top-down text segmentation methods are performed after text string localization. These methods assume that the grayscale distribution is bimodal and that characters a priori correspond to either the white part or the black part, but without providing a way of choosing the right one on-line. Great efforts are thus devoted to performing better binarization, combining global and local thresholding [1], M-estimation [11], or simple smoothing [28]. However, these methods are unable to filter out background regions with similar grayscale values to the characters. If the character grayscale value is known, text enhancement methods can help the binarization process [20]. However, without proper estimation of the character scale, the designed filters can not enhance character strokes with different thickness [5]. In videos, multi-frame enhancement [13] can also reduce the influence of background regions but only when text and background have different movements.

These methods mostly considered text segmentation as the main way to improve the text recognition results. In section 4, we will propose a multiple hypotheses framework to achieve the same goal.

3 Text detection

There are two problems in obtaining an efficient and robust text detection system using machine learning tools. 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 the 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 grayscale changes.

3.1 Text localization

The first part of the text localization procedure consists of detecting text blocks characterized by short horizontal and vertical edges connected to each other. The second part aims at extracting individual text lines from these blocks.

3.1.1 Candidate text region 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 site s ($s \in S$) in an image I the probability $P\,(T|s,I)$ that this site belongs to a text block and then grouping the pixels with high probabilities into regions. To this end, vertical and horizontal edge maps C_v and C_h are first computed from the directional second derivative zeros produced by a Canny filter [2]. Then, according to the type of edge, different dilation operators are used so that vertical edges extend in horizontal direction while horizontal edges extend in vertical direction :

$$D_v(s) = C_v(s) \oplus Rect_v \qquad \text{and} \qquad D_h(s) = C_h(s) \oplus Rect_h \tag{1}$$

The dilation operators $Rect_v$ and $Rect_h$ are defined to have the rectangle shapes 1x5 and 6x3. Figure 2 (b,c) displays the vertical and horizontal edges resulting of this process for the video frame showed in Figure 2 (a). The vertical and horizontal edge dilation results are shown in Figure 2 (d,e). Due to the connections between character strokes, vertical edges contained in text-like regions should be connected with some horizontal edges, and vice versa, we consider only the regions that are covered by both the vertical and horizontal edge dilation results as candidate text regions. Thus, the probability $P(T|s,I)$ can be estimated as :

$$P(T|s,I) = D_v(s)D_h(s) \tag{2}$$

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

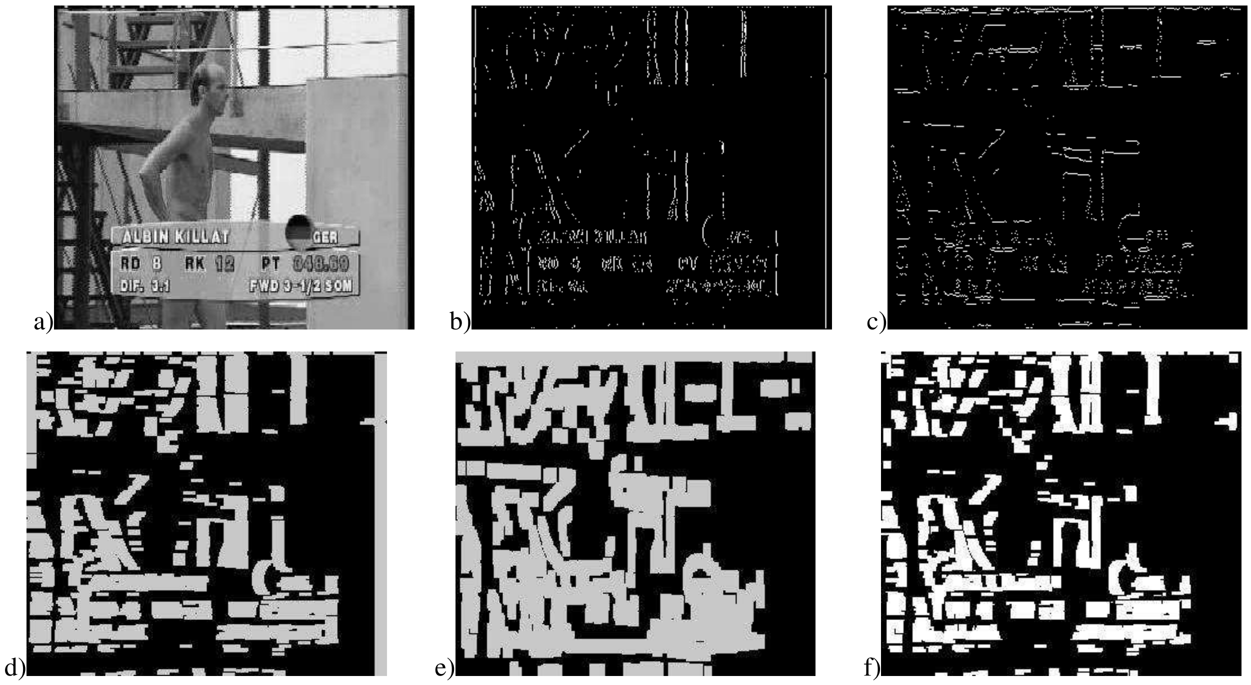

FIG. 2 – *Candidate text region extraction. (a) original image (b) vertical edges detected in image (a) (c) horizontal edges detected in image (a) (d) dilation result of vertical edges using 5x1 vertical operator (e) dilation result of horizontal edges using 3x6 horizontal operator (f) candidate text regions.*

The above text detection procedure is fast and invariant to text intensity changes. Also, ideally, the threshold of the edge detection step can be set in such a way so that no true text regions will be rejected. The false alarms resulting of this procedure are often slant stripes, corners, and groups of small patterns, for example human faces. Their number can be greatly reduced using the techniques introduced in the next sections.

3.1.2 Text line localization in candidate text region

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

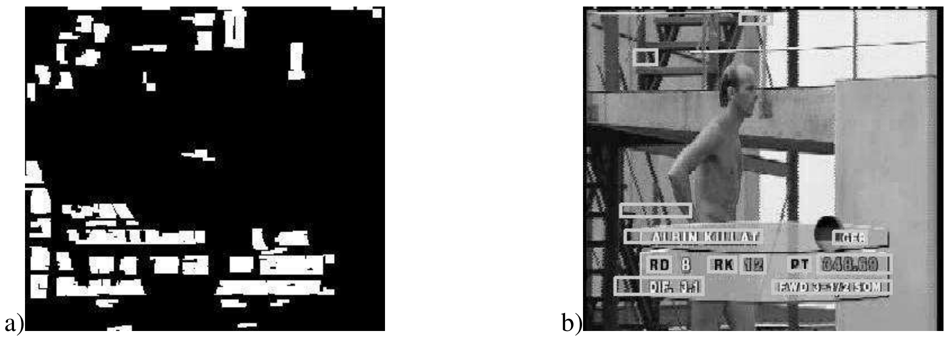

FIG. 3 – *Text line localization (a) candidate text region with located baselines (top and bottom boundaries) (b) the rectangle boundaries of candidate text lines.*

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 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 find the Y-coordinate y_0 which has the maximum absolute derivative of $h(y)$. If this maximum 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 [18]. 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.

Figure 3(a) illustrates the result of applying this text line localization step on Figure 2(f). Typical characteristics of text strings are then employed to select the resulting regions and the final candidate text line should satisfy the following constraints : it contains between 75 and 9000 pixels; the horizontal-vertical aspect ratio is more than 1.2; the height of the region is between 8 and 35. Figure 3(b) shows the rectangle boundaries of the candidate text lines. In general, the size of the text can vary greatly (more than 35 pixels high). Large characters can be detected by using the same algorithm on scaled image pyramid [28].

3.2 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, we used verifiers trained 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. multi-layer perceptrons (MLP), minimizes the error over the data set. On the contrary, structural risk minimization methods, e.g. support vector machines SVMs [27], aim at minimizing a bound on the generalization error of a model in a high dimensional space. The training examples that lie far 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 for text verification task.

3.2.1 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. A feature image I_f 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 16x16 sliding windows. Since the grayscale values of text and background are unknown, we tested four different kinds of features invariant to grayscale changes.

Grayscale spatial derivatives features :
To measure the contribution of contrast in the text verification process, the spatial derivatives of the image brightness function in both the X and Y directions are computed at each site s, resulting in feature vectors of 512 dimensions.

Distance map features :
Since the contrast of text character is background dependent, the brightness spatial derivatives 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. It is defined by [26] :

$$\forall s \in S,\ DM(s) = \min_{s_i \in E} d\,(s,s_i) \tag{3}$$

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

Constant gradient variance features :
To avoid the need for setting any threshold, we propose a new feature, called constant gradient variance (CGV), to normalize the contrast at a given point using the local contrast variance computed in a neighborhood of this point. More formally, let $g(s)$ denote the gradient magnitude at site s, and let $LM(s)$ (resp. $LV(s)$) denote the local mean (reps. the local variance) of the gradient defined by :

$$LM(s) = \frac{1}{|\mathcal{G}_s|} \sum_{s_i \in \mathcal{G}_s} g(s_i) \qquad \text{and} \qquad LV(s) = \frac{1}{|\mathcal{G}_s|} \sum_{s_i \in \mathcal{G}_s} (g(s_i) - LM(s))^2 \tag{4}$$

where $\mathcal{G}_s$ is a 9x9 neighborhood around s. Then, the CGV value at site s is defined as:

$$CGV(s) = (g(s) - LM(s)) \sqrt{\frac{GV}{LV(s)}} \tag{5}$$

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

$$\mathbb{E}[CGV(s)] = 0 \qquad \text{and} \qquad \mathbb{E}\left[(CGV(s))^2\right] = GV \tag{6}$$

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 also enhances the noise in regions with a uniform grayscale value. However such regions will be very rare since the localization step only provides candidate text images that contain many edges.

DCT coefficients :
The last feature vector we tested is composed of discrete cosine transform (DCT) coefficients computed over 16x16 blocks using a fast DCT algorithm presented by Feig [8]. This frequency domain features are commonly used in texture analysis.

3.2.2 Multi-Layer Perceptrons (MLP)

MLP is a widely used neural network, usually consisting of multiple layers of neurons : one input layer, hidden layers and one output layer. Each neuron in the hidden or output layers computes a weighted sum of its inputs (each output of the neurons in the previous layer) and then passes this sum through a non-linear transfer function to produce its output. In the binary classification case, the output layer usually consists of one neuron whose output encodes the class membership. In theory, MLPs can approximate any continuous function and the goal in practice consists of estimating the parameters of the best approximation from a set of training samples. This is usually done by optimizing a given criterion using a gradient descent algorithm.

3.2.3 Support vector machine (SVM)

SVM is a technique motivated by statistical learning theory which have shown their ability to generalize well in high-dimensional spaces [21, 6], such as those spanned by the texture patterns of characters. The key idea of SVM is to implicitly project the input space into a higher dimensional space (called feature space) where the two classes are more linearly separable. This projection, denoted ϕ, is implicit since the learning and decision process only involve an inner dot product in the feature space, which can be directly computed using a kernel K defined on the input space. An extensive discussion of SVMs can be found in [27]. In short, given m labeled training examples: $(x_1, y_1), \ldots, (x_m, y_m)$, where $y_i = \pm 1$ indicates the positive and negative classes, and assuming there exists a hyperplane defined by $w \cdot \phi(x) + b = 0$ in the feature space separating the two classes, it can be shown that w can be expressed as a linear combination of the training samples, i.e. $w = \sum_j \lambda_j y_j \phi(x_j)$ with $\lambda_j \geq 0$. The classification of an unknown example z is thus based on the sign of the SVM function:

$$G(z) = \sum_{j=1}^{m} \lambda_j y_j \phi(x_j).\phi(z) + b \ \doteq \ \sum_{j=1}^{m} \lambda_j y_j K(x_j, z) + b, \tag{7}$$

where $K(x_j, z) = \phi(x_j).\phi(z)$ is called the kernel function. The training of a SVM consists of estimating the λ_j (and b) to find the hyperplane that maximizes the margin, which is defined as the sum of the shortest distance from the hyperplane to the closest positive and negative examples.

3.2.4 Training

The database consists of samples extracted from the text and non-text examples resulting from the localization step. It was divided into a training set and a test set of equal size. Training and testing were performed using either an MLP or a SVM classifier.

The MLP network consists of one input layer, one hidden layer and one output layer with one neuron. We used the sigmoid as transfer function, and the network was trained using the backpropagation algorithm and the standard tricks regarding input normalization, initialization, learning rate decay. The number of hidden neurons, which is related to the capacity of the MLP, is chosen by performing a M-fold cross validation on the training set similar to the one presented hereafter for SVM.

The SVM classifier is trained using standard quadratic programing technique. As the kernel, we choose the Radial basis function (RBF) defined by :

$$K(x, x_j) = e^{\frac{-||x - x_j||^2}{2\sigma^2}} \tag{8}$$

where the kernel bandwidth σ is determined by M-fold cross validation. This M-fold cross-validation procedure can be outlined in the following way :

1. Partition the training data set into M parts of equal size called the "folds". Then, assign each fold a possible value of σ.
2. For i = 1 to M, train the SVM using the ith σ as parameter and all the folds except the ith as the training set. Evaluate the error rate of the resulting SVM on the ith fold.
3. Keep the value of σ corresponding to the lowest error rate as the optimal parameter and train the SVM using all the training data to obtain a good support vector set.

3.2.5 Text-line verification

In the text verification step, the feature vectors discussed in Subsection 3.2.1 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 **r**, we obtained a set of feature vectors $Z_r = (z_1^r, \ldots, z_l^r)$. The confidence of the whole candidate text line **r** is defined as:

$$\text{Conf}(r) = \sum_{i=1}^{l} \ G(z_i^r) \times \frac{1}{\sqrt{2\pi}\sigma_0} e^{-\frac{d_i^2}{2\sigma_0^2}} \tag{9}$$

where d_i is the distance from the geometric center of the ith sliding window to the geometric center of the text line **r**, σ_0 is a scale factor depending on the text line length, and $G(z_i^r)$ denotes the output of the MLP or the magnitude of the SVM (cf Eq. (7)), which indicates the confidence that the vector z_i^r belongs to a text line. Finally, the candidate text line **r** is classified as a real text region if Conf$(r) \geq 0$.

4 Text recognition

In this section, we first describe the overall text recognition scheme. We then describe more thoroughly the different elements of the algorithm.

FIG. 4 – *Examples of detected textlines.*

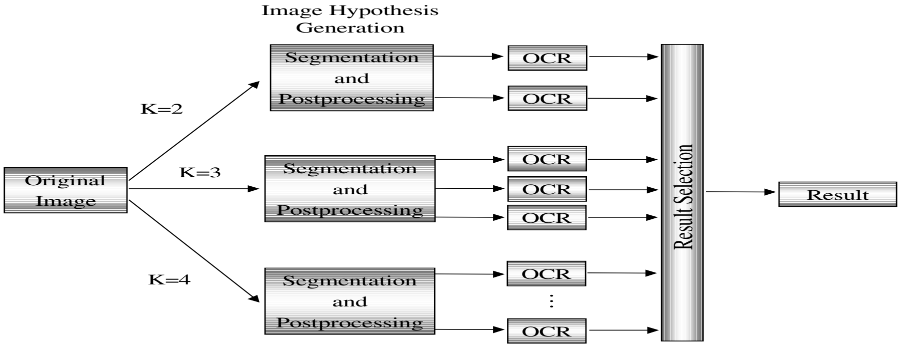

FIG. 5 – *Text recognition scheme*

4.1 Overall description

Most of the previous methods that addressed text recognition in complex images or video worked on improving the binarization method before applying an OCR module. However, an optimal binarization might be difficult to achieve when the background is complex and the grayscale distribution exhibits several modes. Moreover, the grayscale value of text may not be known in advance. These problems are illustrated by the image of Fig. 2a) and examples of detected text lines in Fig. 4.

Fig. 5 and 6 outline the multi-hypotheses approach we propose to handle these problems. A segmentation algorithm that classify the pixels into K classes is applied on the text image. Then, for each class label, a binary text image hypothesis is generated by assuming that this label corresponds to text and all other labels corresponds to background. This binary image is then passed through a connected component analysis and grayscale consistency constraint module and forwarded to the OCR system, producing a string hypothesis (see Fig. 6). Rather than trying to estimate the right number of classes K, using a minimum description length criterion for instance, we use a more conservative approach, by varying K from 2 to 3 (reps. 4), generating in this way five (resp. nine) string hypotheses from which the text result is selected.

4.2 Segmentation methods

Let o denote the observation field $o = \{o_s, s \in S\}$, where o_s corresponds to the gray-level value at site (pixel) s. We assume that the image intensity distribution is composed of K classes, also referred to as layers. Each class is expected to represent regions of the image having similar gray levels, one of them being text. The segmentation is thus stated as a statistical labeling problem, where the goal is to find the label field $e = \{e_s, 1 \leq e_s \leq K, s \in S\}$ that best accounts for the observations, according to a given criterion. To perform the segmentation, we tested 3 algorithms. In the first two cases, the probability that a gray value o_s arises at a given site s within a particular layer i is modeled by a gaussian, i.e. $\mathrm{p}_i(o_s) = \mathcal{N}(\mu_i, \sigma_i)$.

4.2.1 The basic EM algorithm

Here, individual processes are combined into a probabilistic mixture model according to :

$$\mathrm{p}(o_s) = \sum_{k=1}^{K} \mathrm{p}(o_s|e_s = k)\mathrm{p}(e_s = k) = \sum_{k=1}^{K} \pi_k \mathrm{p}_k(o_s) \tag{10}$$

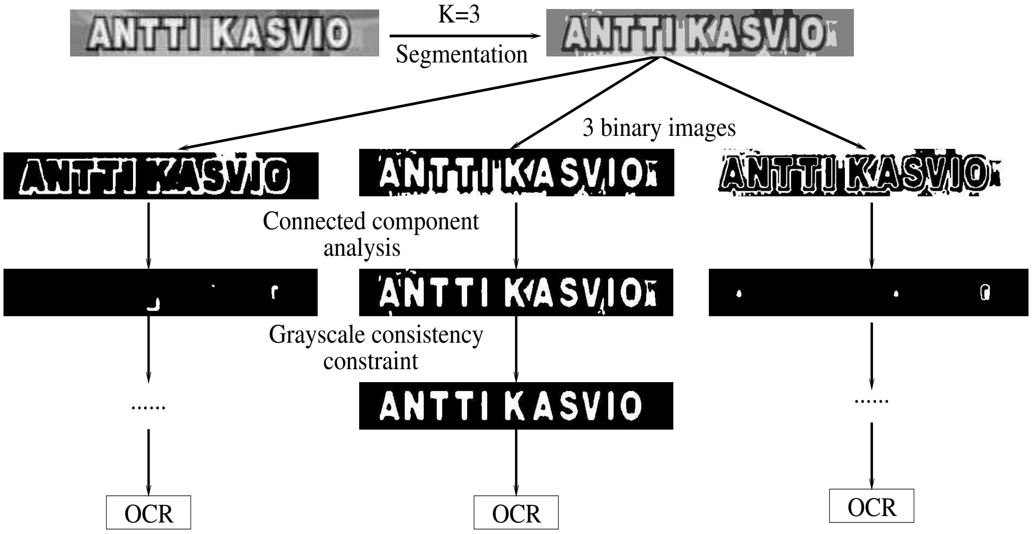

FIG. 6 – *Segmentation and postprocessing steps.*

Given an image, the goal is thus to find the set of parameters (φ,π)=$(\mu_i,\sigma_i,\pi_i)_{i=1\ldots K}$ maximizing the likelihood of the data set o defined as $\mathrm{L}^{\varphi}(o) = \ln \mathrm{p}(o) = \sum_{s\in S} \ln \mathrm{p}(o_s)$. Using the standard "expectation-maximization" (EM) algorithm [7], the expected log-likelihood of the complete data (i.e., observations and labels) can be iterativly maximized with respect to the unknown data (the labels). After maximization, the labels can be assigned to the most probable layer according to the following rule :

$$\forall s \qquad e_s = \operatorname*{argmax}_i \ \mathrm{p}_i(o_s) \tag{11}$$

4.2.2 The Gibbsian EM (GBEM) algorithm

While the EM algorithm is able to capture most of the gray level distribution properties, it does not model the spatial correlation between assignment of pixels to layers, resulting in noisy label images. To overcome this, we introduce some prior by modeling the label field as a Markov Random Field (MRF). Then, instead of using the simple rule (11) to estimate e, we perform a MAP optimization, i.e., we maximize the *a-posteriori* distribution of the labels given the observations. Due to the equivalence between MRF and Gibbs distribution [10], we have :

$$\mathrm{p}(e) = \frac{1}{Z(V)} e^{-U_1^V(e)}$$

where $Z(V)$ is a normalizing constant that can not be computed in practice due to the high dimension of the configuration space. Thus, the MAP optimization is equivalent to the minimization of an energy function $U(e,o)$ given by :

$$U(e,o) = U_1^V(e) + U_2^{\varphi}(e,o) \tag{12}$$

$$\text{with} \qquad U_2^{\varphi}(e,o) = \sum_{s\in S}(-\ln \mathrm{p}_{e_s}(o_s)) \tag{13}$$

expressing the adequacy between observations and labels, as in the EM algorithm. For the definition of U_1, we only considered second-order neighbors and set U_1 to :

$$U_1^V(e) = \sum_{s\in S} V_{11}(e_s) + \sum_{<s,t>\in \mathcal{C}_{hv}} V_{12}^{hv}(e_s,e_t) + \sum_{<s,t>\in \mathcal{C}_{diag}} V_{12}^{d}(e_s,e_t) \tag{14}$$

where $\mathcal{C}_{hv}$(resp. $\mathcal{C}_{diag}$) denotes the set of two elements cliques (i.e. two neighbor pixels) in the horizontal/vertical (resp. diagonal) direction. The V are the (local) interaction potentials which express the prior on the label field. One may wish to learn these parameters off-line, from examples. However, the use of the learned parameters in the optimization process

would require to know the correspondence between learned labels/layers and current ones.[1] Moreover, the scale of the characters plays an important role in the optimum value of these parameters.
The second algorithm we propose consists of estimating all the parameters $\Theta=(\varphi,V)$ using an EM procedure [3]. Recall that the expectation step involves the computation of :

$$\mathbb{E}\left[\ln \mathrm{p}_{eo}^{\Theta}|o,\Theta^n\right] = \sum_e \ln\left(\mathrm{p}_{o|e}^{\Theta}(e,o)\mathrm{p}_e(e)\right)\ \mathrm{p}_{e|o}^{\Theta^n}(e,o) \tag{15}$$

which is then maximized over Θ. Two problems arise here. Firstly, this expectation on $\mathrm{p}_{e|o}^{\Theta^n}$ can not be computed explicitly nor directly. Instead, this law will be sampled using Monte Carlo methods, and the expectation will be approximated along the obtained Markov chain. We used a Gibbs-sampler for the simulation.
Secondly, the joint log-likelihood probability p_{eo}^{Θ} is not completely known, because of the presence of the uncomputable normalizing constant $Z(V)$ in the expression of $\mathrm{p}(e)$. To avoid this difficulty, we use the pseudo-likelihhod function [3] as a new criterion, that is, in (15), we replace $\mathrm{p}(e)$ by its pseudo-likelihood $\mathrm{p_s}(e)$ defined from the local conditional probabilities :

$$\mathrm{p_s}^V(e) \doteq \prod_{s\in S} \mathrm{p}(e_s|e_{\mathcal{G}_s}) \tag{16}$$

where $e_{\mathcal{G}_s}$ represents the label in neighborhood of s. Using this new criterion, the maximization of the expectation (15) can be performed, providing new estimates of (μ_i,σ_i) and V. The complexity of the GBEM algorithm is approximately 4 times greater than the complexity of the EM algorithm.

4.2.3 The Kmeans algorithm

In this method, the goal is to find the K means of the disjoint subsets S_i (the layers) which minimizes the intra-class variance [18], that is :

$$IV = \sum_{i=1}^{K} \sum_{s\in S_i} \|o_s - \mu_i\|^2 \ = \sum_{s\in S} \|o_s - \mu_{e_s}\|^2$$

The minimization can be achieved using standard algorithms, which iteratively assign each data to the class of the nearest center and then recompute the means.

4.3 Postprocessing : connected component analysis (CCA) and grayscale consistency constraint (GCC)

To help the OCR system, a simple connected component analysis is used to eliminate non character regions in each hypothesis based on their geometric properties. We only keep connected component that satisfies the following constraints: size is bigger than 120 pixels; width/height ratio is between 4.5 and 0.1; the width of the connected component is less than 2.1 the height of the whole text region.

Since we only consider 2 to 4 layers, regions from the background with a gray value slightly different from that of characters may still belong to the text layer/class. We thus developed another module to enforce a more stringent gray consistency among the connected components. The procedure is the following and is applied to each layer (see Fig. 6). After the CCA step, we estimate with a robust estimator [19] the gray level mean m^* and standard deviation st^* of the set S_r of sites belonging to the remaining regions. More precisely, a least-median squares estimator is employed to identify the graylevel value that fit the majority of pixels graylevel values and eliminate the pixel with outlier graylevel values, and m^* and st^* are estimated on the remaining valid pixels using standard formula [19]. Finally, a connected component is eliminated from the layer if more than 50% of its pixels have a gray level value different than the majority of pixels, that is, verify :

$$\frac{|o_s - m^*|}{st^*} > k \tag{17}$$

An illustration of the result of this step is displayed in Fig. 4.3.

1. Remind that text may be black, white or gray.

a)
b)
c)
d)

FIG. 7 – *Applying grayscale consistency : a) original image b) text layer (3 layers, GBEM algorithm) c) same as b), after the connected component analysis d) same as c), after the gray level consistency step.*

4.4 OCR and result selection

The selection of the result from the set of strings generated by the segmentation processes (see Fig. 5) is based on a confidence value evaluation relying on language modeling and OCR statistics. From a qualitative point of view, when given text-like background or inaccurate segmentation, the OCR system produces mainly garbage characters like ., ,!, & etc and simple characters like i,l, and r. Let $T = (T_i)_{i=1..l_T}$ denote a string where l_T denotes the length of the string and each character T_i is an element of the character set $\mathcal{T} = (0,\ldots,9,a,\ldots,z,A,\ldots,Z,G_b)$, in which G_b corresponds to any other garbage character. Furthermore, let H_a (resp. H_n) denote the hypothesis that the string T or the characters T_i are generated from an accurate (resp. a noisy) segmentation. The confidence value is estimated using a variant of the log-likelihood ratio :

$$C_v(T) = \log\left(\frac{\mathrm{p}(H_a|T)}{\mathrm{p}(H_n|T)}\right) + l_T * b = \log\left(\mathrm{p}(T|H_a)\right) - \log\left(\mathrm{p}(T|H_n)\right) + l_T * b$$

when assuming an equal prior on the two hypotheses and b is a bias that is discussed below. We estimated the noise free language model $\mathrm{p}(.|H_a)$ by applying the CMU-Cambridge Statistical Language Modeling (SLM) toolkit on Gutenberg collections[2]. A bigram model was selected. Cutoff and backoff techniques [12] were employed to address the problems associated with sparse training data for special characters (e.g. numbers and garbage characters). The noisy language model $\mathrm{p}(.|H_n)$ was obtained by applying the same toolkit on a database of strings collected from the OCR system output when providing the OCR input with either badly segmented texts or text-like false alarms coming from the text detection process. Only a unigram model was used because the size of the background dataset was insufficient to obtain a good bigram model. The bias b is necessary to account for the string length. It was observed that without this bias, the likelihood ratio would quite often select strings with only a few quite reliable letters instead of the true string. By incorporating this bias in the confidence value, the selection module was encouraged to compare string results whose length was in accordance with the underlying text image width. Setting $b = 0.7$, the confidence value is defined as :

$$C_v(T) = \log \mathrm{p}(T_1|H_a) + \sum_{i=2}^{l_T} \log \mathrm{p}(T_i|T_{i-1},H_a) - \sum_{i=1}^{l_T} \log \mathrm{p}(T_i|H_n) + 0.7 * l_T.$$

5 Experiments and Results

In this section, we report results on text localization, text verification, and text recognition.

2. www.gutenberg.net

iX-based	RR	FPR	CPU costs
Derivative Texture	9.46%	3.48%	1.27
Vertical Edge	13.58%	16.17%	0.52
Proposed	5.49%	0.81%	0.54

TAB. 1 – *Performances and running costs of different text detection techniques. RR denotes the rejection rate and FPR denotes the false pixel alarm rate.*

5.1 Text localization results

The text localization step is evaluated on a half an hour video containing a Belgian news program [3] in french provided by Canal+ in the context of the CIMWOS [4] European project. The performance of the text localization step is measured in terms of rejection rate (RR), false pixel alarm rate (FPR) and CPU cost. The rejection rate is defined as :

$$RR = \frac{RP}{TP} \tag{18}$$

where RP denotes the total number of text pixels rejected 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 :

$$FPR = \frac{PF}{PI} \tag{19}$$

where PF denotes the number of false alarm pixels and PI denotes the total number of pixels in the images. The computation cost is measured in seconds on a Sun UltraSPARC-II with 333 MHz CPU without counting the I/O consumption.

In table 5.1, we compare the performance of the proposed algorithm with the derivative texture algorithm [28] and the vertical edge based algorithm [22] implemented by ourselves according to the referenced papers. It can be observed that the proposed feature yields the lowest rejection rate. The computation cost of the proposed method is lower than the derivative texture algorithm and similar to the vertical edge based method. This latter result can be explained by the fact 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 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 database. There were no rejected regions. The precision of this localization step on this database is $\frac{9369}{9369+7537} = 55.4\%$.

5.2 Text verification results

The text verification algorithms was designed and trained on a database consisting of still images and half an hour video recorded from TV. The videos contain excerpts from advertisements, sports, interviews, news and movies. The still images include covers of Journals, maps and flyers. The video frames have a size of 720x576 and are compressed in MPEG, while the still image have a size of 352x288 and are compressed in JPEG. Only the grayscale information is used in the experiments.

The feature extraction, training and testing procedures described in Subsection 3.2 were applied on this database. More precisely, 2,400 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 5.2 lists the error rate measured on the test set for each feature and for each classifier. First of all, we can see that these results are very good and better than those reported when running the classifier on the whole image whithout

3. From the Radio-Télévision Belge d'expression Française (RTBF).
4. Combined Image and Word Spotting

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

TAB. 2 – *Error rates of the SVM and MLP classifiers for the text verification task. 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.*

FIG. 8 – *Detected text regions in images or video frames.*

applying size normalization (13%-30% [15]). Additionally, the proposed scheme runs approximately five times faster. Secondly, whatever the considered feature, the SVM classifier gives better results than the MLP classifier, showing its ability to better generalize. Finally, we can see that the proposed constant gradient variance feature provides the best result. This can be explained by its better invariance to text/background contrast.

The SVM classifier together with the CGV feature was employed to verify the extracted text regions of the CIMWOS database, based on the confidence value given by Eq. 9. This verification scheme removed 7255 regions of the 7537 false alarms while only rejecting 23 true text lines, giving a 0.24% rejection rate and a 97% precision rate. Fig. 8 shows examples of detected text on some images in our databases.

5.3 Text recognition results

The multiple hypotheses recognition scheme was tested on a sports database of the Barcelona 1992 Olympic games provided by the BBC in the context of the ASSAVID[5] European project. From about five hours of video, we extracted around one hour of video containing text. The text localization and verification algorithms were applied. As the text regions located in consecutive video frames usually contain similar text and background, the video data results where sub-sampled in time, leading to a database of 1208 images containing 9579 characters and 2084 words. Text characters are embedded in complex background with JPEG compression noise, and the grayscale value of characters is not always the highest as the examples shown in Fig. 8.

To assess the performance of the different algorithms, we use character recognition rate (CRR) and character precision rate (CPR) that are computed on a ground truth basis as :

$$CRR = \frac{N_r}{N} \ and \ CPR = \frac{N_r}{N_e}$$

5. Automatic Segmentation and Semantic Annotation of Sports Videos.

K	Algorithm	Ext.	CRR	CPR	WRR
2	EM	7715	74.9%	93.1%	61.0%
	GBEM	9300	**92.8%**	**95.6%**	**83.5%**
	Kmeans	9260	**92.5%**	**95.7%**	**82.8%**
3	EM	9239	89.9%	93.2%	80.9%
	GBEM	9302	89.9%	92.6%	80.7%
	Kmeans	9394	91.3%	93.1%	82.2%
4	EM	9094	87.4%	92.1%	77.7%
	GBEM	9123	88.3%	92.8%	79.9%
	Kmeans	9156	88.0%	92.1%	79.7%

TAB. 3 – *Recognition results without grayscale consistency constraint (GCC): number of extracted characters (Ext.), character recognition rate (CRR), precision (CPR) and word recognition rate (WRR).*

K	Algorithm	Ext.	CRR	CPR	WRR
2	EM	7914	77.9%	94.2%	66.2%
	GBEM	9307	**94.0%**	**96.8%**	**87.1%**
	Kmeans	9291	93.8%	96.7%	86.8%
3	EM	9245	90.3%	93.6%	81.7%
	GBEM	9268	89.5%	92.5%	81.1%
	Kmeans	9395	91.2%	93.0%	83.4%
4	EM	9136	88.0%	92.3%	78.9%
	GBEM	9123	87.7%	92.1%	79.1%
	Kmeans	9195	88.9%	92.6%	80.4%

TAB. 4 – *Recognition results with GCC : number of extracted characters (Ext.), character recognition rate (CRR), character precision rate (CPR) and word recognition rate (WRR).*

N is the true total number of characters, N_r is the number of correctly recognized characters and N_e is the total number of extracted characters. These numbers are computed using an edit distance (counting the amount of character substitutions, insertions and deletions) between the ground truth and the recognized string. Additionally, we compute the word recognition rate (WRR) to get an idea of the coherency of character recognition within one solution. For each text image, we count the words from the ground truth of that image that appear in the string result. Thus, WRR is defined as the percentage of words from the ground truth that are recognized in the string results.

We first report results where the string result is selected from the hypotheses generated by applying the segmentation algorithm one time with a fixed K value, and when applying only the connected component analysis as a postprocessing step. This will serve as a baseline for the rest of the experiments. Table 3 lists the results obtained with the three described segmentation methods. It can be seen that the usual bi-modality (K=2) hypothesis yields the best character and word recognition rate with the GBEM and Kmeans algorithms. However, in the case of K=3, the Kmeans algorithm also gives quite similar results. Indeed, some text images are composed of the grayscale characters, contrast contours around characters, and background (see figure 4). In this case, the grayscale values are better modeled with 3 or more clusters. Under the bimodality assumption (K=2), the GBEM algorithm yields better results than the typical Otsu's binarization method (Kmeans with K=2) in terms of both CRR and WRR. This is probably due to the regularization power of the GBEM algorithm, which learns the spatial properties of the text and background layers. It helps in avoiding over segmentation, as can be seen from the example of Fig. 9. However, the improvement is not very important and is deceptive. It can be explained by the fact that the MRF approach mainly improves the character shape, a kind of noise the OCR has been trained on and to which it is probably not very sensitive.

The grayscale consistency constraint (GCC) technique described in Section 4 was added to the baseline system. The corresponding results are listed in table 4. When $K = 2$, they show an increase in absolute value of about 4% of both the CRR and WRR together with an increase of 1.2% of the CPR. This is due to the ability of the GCC to remove burst-like noise (i.e. significant background regions with a slightly different graylevel value than characters) which greatly impair the recognition of the OCR. For higher values of K, the increase is less important. Indeed, in these cases, the grayscale consistency constraint is inherently better respected.

	K	Algorithm	Ext.	CRR	CPR	WRR
without GCC	2,3	EM	9480	93.3%	94.3%	86.7%
		GBEM	9606	96.6%	96.3%	**93.1%**
		Kmeans	9565	96.6%	96.8%	92.8%
	2,3,4	EM	9417	93.2%	94.8%	86.8%
		GBEM	9604	96.6%	96.2%	**93.0%**
		Kmeans	9547	96.6%	97.0%	92.9%
with GCC	2,3	EM	9449	94.0%	95.3%	88.1%
		GBEM	9579	96.5%	96.5%	92.8%
		Kmeans	9587	**97.1%**	97.0%	93.7%
	2,3,4	EM	9411	93.9%	95.6%	88.1%
		GBEM	9557	96.6%	96.8%	93.0%
		Kmeans	9560	97.0%	**97.2%**	**93.8%**

TAB. 5 – *Recognition results from 5, 9 hypotheses, with or without GCC : number of extracted characters (Ext.), character recognition rate (CRR), character precision rate (CPR) and word recognition rate (WRR).*

EM.2 : “WOMEN3 RIFLE FINALISTS”

GBEM.2: “WOMEN3 A RIFLE FINALISTS”

Kmeans.2: “WOMEN’S AIR RIFLE FINALISTS”

EM.3: “’,˜Oi’.lErJ 5 AIFI RIFLE FIrJALIS r ”

GBEM.3: “ WOMEN’S AIR RIFLE FINALISTS”

Kmeans.3: “,.’.,Or.lErl S AIR FIIF E Flrlg IS”

FIG. 9 – *Segmentation output of the three studied algorithms and associated recognition results using 2 or 3 layers/classes.*

Table 5 lists the results obtained by generating 5 or 9 hypotheses (using K=2 to 4) in the multi-hypotheses framework. Without employing the GCC postprocessing, the method achieves a 96.6% CRR and a 93.1% WRR, which constitutes a reduction of more than 50% for both rates with respect to the best baseline result (GBEM with K=2). These results demonstrates firstly the complementary of the solutions provided when assuming different K values, and secondly the ability of our selection algorithm to choose the right solution. Interestingly, the results obtained with 9 hypotheses are not better than the results obtained using only 5 hypotheses. It probably means that the segmentation with K=4 doesn't generate additional interesting results with respect to the K=2 and K=3 cases.

When integrating the GCC algorithm in the multiple hypotheses framework, we can notice that the GCC postprocessing improves the result when using the Kmeans or EM segmentation algorithms and remain similar for the GBEM algorithm. This smaller improvement, when compared to the improvement obtained when adding the GCC to the baseline, can be explained by the fact that the multiple hypotheses framework has less need for burst-noise elimination, since it can select between alternative modelization of the graylevel distribution.

6 Discussion and conclusions

This paper presents a general scheme for extracting and recognizing embedded text of any grayscale value in images and videos. The method is split into two main parts : the detection of text lines, followed by the recognition of text in these lines.

Applying machine learning methods for text detection encounters difficulties due to character size and grayscale variations and heavy computation cost. To overcome these problem, we proposed a two step localization/verification scheme. The first step aims at quickly locating candidate text lines, enabling the normalization of characters into a unique size. In the verification step, a trained SVM or MLP is applied on background independent features to remove the false alarms. Experiments showed that the proposed scheme improved the detection result at a lower cost in comparison with the same machine learning tools applied without size normalization, and that SVM was more appropriate than MLP to address the text texture verification problem.

The text recognition method we propose embeds the traditional character segmentation step followed by an OCR algorithm within a multiple hypotheses framework. A new grayscale consistency constraint (GCC) algorithm was proposed to improve segmentation results. The experiments that were conducted on around 1 hour of sports video demonstrated the validity of our approach. More specifically, when compared to a baseline system consisting of the standard Otsu binarization algorithm, the GCC postprocessing step was able to reduce the character and word error rates of more than 20%, showing its ability to remove burst-like noise that greatly disturbs the OCR software. Morevover, added to the multiple hypotheses framework, the whole system yielded around 97% character recognition rate and more than 93% word recognition rate on our database, which constitutes a reduction of more than 50% w.r.t the baseline system. This clearly shows that (i) several text images may be better modeled with 3 or 4 classes rather than using the usual 2 class assumption (ii) multiple segmentation maps provide complementary solutions and (iii) the proposed selection algorithm based on language modeling and OCR statistics is often able to pick up the right solution.
We proposed to use a Maximum A Posteriori criterion with a MRF modeling to perform the segmentation. Used as a traditional binarization algorithm, it performed better than Otsu's method. However, embedded in the multi-hypotheses system with GCC, it yielded similar results to the Kmeans. Thus, the latter has been preferred in real application since it runs faster.

The performance of the proposed methodology are good enough to be used in video annotation and indexing system. In the context of the ASSAVID European project, it was integrated with other components (shot detector, speech recognizer, sports and event recognizers,...) in a user interface designed to produce and access sports video annotation. A simple complementary module combining the results from consecutive frames containing the same text was added. User experiments with librarians at the BBC showed that the text detection and recognition technology produced robust and useful results, i.e. did not produce many false alarms and the recognized text was accurate. The same proposed scheme is currently used in the CIMWOS project to index French news programs.

Acknowledgment

This work has been performed partially with in the frameworks of the "Automatic Segmentation and Semantic Annotation of Sports Videos (ASSAVID)" project and the "Combined Image and Word Spotting (CIMWOS)" project both granted by the European IST Programme.

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

Références

[1] H. Kamada amd K. Fujimoto. High-speed, high-accuracy binrization method for recognizing text in images of low spatial resolutions. In *Int. Conf. on Document Analysis and Recognition*, pages 139–142, Sept. 1999.

[2] J. F. Canny. A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(1):679–698, 1986.

[3] B. Chalmond. Image restoration using an estimated Markov model. *Signal Processing*, 15(2):115–129, Sept. 1988.

[4] D. Chen, H. Bourlard, and J-Ph. Thiran. Text identification in complex background using svm. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages 621–626, Dec. 2001.

[5] D. Chen, K. Shearer, and H. Bourlard. Text enhancement with asymmetric filter for video OCR. In *Proc. of the 11th Int. Conf. on Image Analysis and Processing*, pages 192–198, Sept. 2001.

[6] R. Collobert, S. Bengio, and Y. Bengio. A parallel mixture of svms for very large scale problems. *Neural Computation*, 14(5), 2002.

[7] A. Dempster, N. Laird, and D. Rubin. Maximum-likelihood from incomplete data via the em algorithm. *Royal Statistical Society*, B-39:1–38, 1977.

[8] E. Feig and S. Winograd. Fast algorithms for the discrete cosine transform. *IEEE Trans. Signal Processing*, 40(28):2174–2193, Sept. 1992.

[9] C. Garcia and X. Apostolidis. Text detection and segmentation in complex color images. In *Int. Conf. on Acoustics, Speech and Signal Processing*, pages 2326–2329, 2000.

[10] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 6(6):721–741, Nov. 1984.

[11] O. Hori. A video text extraction method for character recogntion. In *Int. Conf. on Document Analysis and Recognition*, pages 25–28, Sept. 1999.

[12] S.M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Trans. on Acoustics, Speech and Signal Processing*, 35:400–401, 1987.

[13] H. Li and D. Doermann. Text enhancement in digital video using multiple frame integration. In *ACM Multimedia*, pages 385–395, 1999.

[14] R. Lienhart. Automatic text recognition in digital videos. In *Proc. SPIE, Image and Video Processing IV*, pages 2666–2675, Jan. 1996.

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

[16] B.S. Manjunath and W.Y. Ma. Texture features for browsing and retrieval of image data. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(8):837–842, Aug. 1996.

[17] F. Mokhtarian, S. Abbasi, and J. Kittler. Robust and efficient shape indexing through carvature scale space. In *British Machine Vision Conference*, pages 9–12, 1996.

[18] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Trans. on Systems, Man and Cybernetics*, 1(9):62–66, 1979.

[19] P.J. Rousseeuw. Least median of squares regression. *American Statistical Association*, 79(388):871–880, Dec. 1984.

[20] T. Sato, T. Kanade, E. K. Hughes, and M. A. Smith. Video OCR for digital news archives. In *IEEE Workshop on Content Based Access of Image and Video Databases*, pages 52–60, Jan. 1998. Bombay.

[21] B. Schölkopf, K. Sung, C. Burges anf F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik. Comparing support vector machines with gaussian kernels to radial basis functions classififers. *IEEE Trans. Signal Processing*, 45(11):2758–2765, 1997.

[22] M. A. Smith and 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, and H. Kronenberg. Identification of text on colored book and journal covers. In *Int. Conf. on Document Analysis and Recognition*, pages 57–63, 1999.

[24] Rohini K. Srihari, Zhongfei Zhang, and Aibing Rao. Intelligent indexing and semantic retrieval of multimodal documents. *Information Retrieval*, 2(2/3):245–275, 2000.

[25] M. Swain and H. Ballard. Color indexing. *Int. Journal of Computer Vision*, 7:11–32, 1991.

[26] J. Toriwaki and S. Yokoi. Distance transformations and skeletons of digitized pictures with applications. *Pattern Recognition*, pages 187–264, 1981.

[27] V. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

[28] V. Wu, R. Manmatha, and E. M. Riseman. Finding text in images. In *Proc. ACM Int. Conf. Digital Libraries*, pages 23–26, 1997.

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