**Accepted for publication in Pattern Recognition Letter, 2005**

# Video Text Recognition using Sequential Monte Carlo and Error Voting Methods

Jean-Marc Odobez and Datong Chen

IDIAP Research Institute,
Rue du Simplon 4, 1920 Martigny, Switzerland.
E-mail: {chen,odobez}@idiap.ch.

**Abstract**

This paper addresses the issue of segmentation and recognition of text embedded in video sequences from their associated text image sequence extracted by a text detection module. To this end, we propose a probabilistic algorithm based on Bayesian adaptive thresholding and Monte-Carlo sampling. The algorithm approximates the posterior distribution of segmentation thresholds of text pixels in an image by a set of weighted samples. The set of samples is initialized by applying a classical segmentation algorithm on the first video frame and further refined by random sampling under a temporal Bayesian framework. One important contribution of the paper is to show that, thanks to the proposed methodology, the likelihood of a segmentation parameter sample can be estimated not using a classification criterion or a visual quality criterion based on the produced segmentation map, but directly from the induced text recognition result, which is directly relevant to our task. Furthermore, as a second contribution of the paper, we propose to align text recognition results from high confidence samples gathered over time, to composite a final result using error voting technique (ROVER) at the character level. Experiments are conducted on a two hour video database. Character recognition rates higher than 93%, and word error rates higher than 90% are achieved, which are 4 and 3% more than state-of-the-art methods applied to the same database.

**Index Terms**

Video text recognition, text segmentation, sequential monte-carlo filter, language model, recognition output voting error reduction.

## I. Introduction

TExt recognition in video sequences, which aims at integrating Optical Character Recognition (OCR) and text-based searching technologies, is recognized as one of the key components in the development of content-based multimedia annotation and retrieval systems. Content-based multimedia database indexing and retrieval tasks require automatic extraction of descriptive features that are relevant to the

The authors would like to thank the Swiss National Fund for Scientific Research that supported this work through the Video OCR project. This work has been performed partially within the frameworks of the "Combined Image and Word Spotting (CIMWOS)" granted by the European IST Programme.

subject materials (images, video, etc.). The typical low level features that are extracted in images and videos include measures of color [1], texture [2], or shape [3]. Although these features can easily be extracted, the interpretation in terms of image content is hard to obtain. Extracting more descriptive features and higher level entities, for example text [4], [5] or human faces [6], has attracted more and more research interest recently. In this respect, text embedded in 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 segments can be considered a powerful feature (keyword) resource. They can be used directly in text-based search systems that have been successfully used in many applications. On the other hand, the robustness and computation cost of the feature matching and retrieval algorithms based on lower level features are problematic when applied to large databases, though promising results are currently achieved in this field [7].

The recognition of characters has become one of the most successful applications of technology in the field of pattern recognition and artificial intelligence. However, OCR systems are developed for recognizing characters printed on clean paper. Applying the current OCR systems directly on video text leads to poor character recognition rates, typically from 0% to 45% [8], [9]. This is because text characters contained in video can be of any grayscale values and embedded in multiple consecutive frames with complex backgrounds. For recognizing these video text characters, it is necessary to segment text from backgrounds even when the whole text string is already well located. Therefore, a large amount of work on text segmentation from complex backgrounds has been published in recent years. Generally, a segmentation of text images can be regarded as a process that searches for a couple of thresholds (lower and upper) covering the grayscale values of text pixels. Lienhart [10] and Sobottka [11] clustered text pixels from images using a standard image segmentation or color clustering algorithm. Although these methods can somehow avoid the text detection work, they are very sensitive to noise and character size. Most video text segmentation methods are performed after pre-locating the locations of the text strings in the images. These methods generally assume that the grayscale distribution is bimodal and devote efforts to perform better binarization such as combining global and local thresholding [12], M-estimation [13], simple smoothing [14] or Markov Random Field regularisation [15], [16]. Furthermore, multiple hypotheses segmentation method, which assumes that the grayscale distribution can be k-modal (k=2,3,4), has been proposed in [16] and shown to improve the recognition performance up to 94% character recognition rate, in some applications.

Video sequences have also been considered to improve the segmentation. Most methods construct an enhanced image from the temporal frames and then apply a standard segmentation algorithm on this image. Sato [9] and Lienhart [17] 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 in the video sequence will be pushed to black or white while the values of the text pixels are kept. For example, in the MAX value method, the text is assumed to be black. To extract these black text strings, the method expects to push all the background pixels to the white. However, this method can only be applied on black or white characters. This method will be referred to as the MAX-MIN method in the rest of this paper. Li

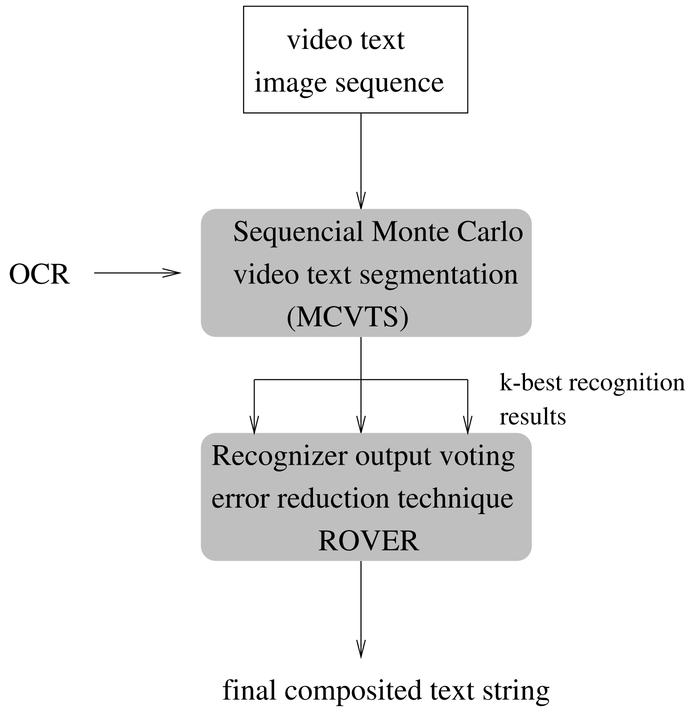


Fig. 1. Text segmentation and recognition in multiple video frames.

[18] proposed a multi-frame enhancement for unknown grayscale text which computes the average of pre-located text regions in multiple frames for further segmentation and recognition. It will be referred to as the AVE method. The average image has a smaller noise variance but may propagate blurred characters in frames. A common drawback of these temporal methods is that they require accurate text image alignment at the pixel level.

Previous methods for exploiting temporal video text information assume that the grayscale values of text characters are constant at the pixel level. A common goal of these methods is to construct text images with higher constrast so as to obtain a "better" segmentation of the text images. However, in these methods, no criterions for evaluating the quality of the text image segmentation are proposed. Besides, above a given level of quality, there is no obvious relationship between what would be called a good segmentation and the quality of the recognized text string. We can even show that, given the low resolution of text strings we need to deal with, an OCR system may output different results for two segmentations that look good and very similar to each other (see Fig. 2). Since the final task is to obtain good recognition results, the "best" way to measure a segmentation image quality is to evaluate the quality of its corresponding OCR output, if this is feasible. One important contribution of the paper is to propose a method based on this principle.

This paper addresses the issue of text recognition in video sequences. More specifically, we study how the use of temporal information about text strings can improve text recognition performances. In this view, we propose a method based on two main steps : a sequential Monte Carlo video text segmentation step, and a recognition results voting step. These two steps are applied sequentially, as shown in Fig. 1.

The first step, presented in Section II, is a probabilistic algorithm for segmenting and recognizing text embedded in video sequences based on adaptive thresholding using a Bayes filtering method. More precisely, text strings are first detected and tracked in consecutive video frames. Then, to segment these strings, the algorithm approximates the posterior distribution of segmentation thresholds of video text by a set of weighted samples. The set of samples is initialized by applying a classical segmentation algorithm on the first video frame and further refined by random sampling under a temporal Bayesian framework. This framework allows us to evaluate a text image segmentation operator on the basis of the text recognition result, which is directly relevant to our character recognition task, instead of on the basis of the visual segmentation result.

Moreover, in order to handle the multiple recognition results of the same text string provided by the first algorithm, we propose, in Section III, an algorithm to reduce the video character recognition error rates by using an output voting technique. The recognition outputs obtained from different frames are modeled as independent knowledge sources using a character transition network. The resulting network is exploited by an automatic voting process for selecting the output sequence. This method should help in reducing spurious recognition errors that occur due to the low resolution of characters and their unknown font, and which are more prone to affect the recognition of long text strings. In particular, it helps in combining results into a new string so that even if none of the OCR results are correct, the final output string may still be good. Some similar approaches have also been exploited in the context of handwritting recognition (e.g. [19]).

This method has been applied on a database of about two hours of broadcasted video (TV news programs and one documentary), and has been shown to improve the performance with respect to other methods (MAX-MIN, AVE). Section 4 reports the results, while Section 5 concludes the paper .

## II. Sequential Monte Carlo video text segmentation

Generally speaking, the segmentation of a text image can be regarded as a process that searches for parameters, a threshold couple in our case (lower and upper thresholds), that optimizes the discrimination between the grayscale values of text pixels and background pixels. When applied to consecutive text images of a given text string, the threshold couples computed in different frames may be different and therefore provide additional information in the recognition process. However, applying traditional segmentation on every frame has two drawbacks. The first one is that it is not efficient in terms of computation cost. For a video text string, the segmentation characteristics in different frames are varying but not completely unpredictable. Thus, the optimal threshold couple of the previous frame could be re-used instead of re-computing the optimal segmentation parameters again. The second drawback is that traditional segmentation algorithms usually rely on a predefined criterion which may not always lead to the optimal threshold couple that would conduct to good recognition [15]. In other words, the segmentation quality in our case should be validated using recognition results instead of any predefined criterion on grayscale values of the image. Figure 2 shows an example of two segmentation results of a given text image and their associated recognized strings. The OCR software we used is RTK from EXPERVISION, which has a character recognition rate higher than 99% on clean page characters. Although the two segmentations of the

Original text image


Segmentation (a)


$\Longrightarrow$ Fusion or i level audio and wdeo features to reoognlse

Segmentation (b)


$\Longrightarrow$ Fusion of lower level audio and video features to recogn se

Fig. 2. Different recognition results may be obtained from segmentation results that are visually quite similar. When the OCR does not recognize characters, it outputs nothing.

word "lower" seem to be visually very similar, they lead to different recognition results. Figure 3 illustrates another example of thresholding a text image. The optimal thresholds of the gray level distribution using Otsu's criterion are: T2 assuming a 2-class problem. T1 and T3 assuming a 3-class problem. None of these values allow us to extract a text binary map that leads to good recognition results. However, in this example, there exist threshold values (e.g. Tg) that lead to the recognition of the image text.

To address these two problems, we present in this section a particle filtering based approach for the segmentation of text characters of any grayscale values. The idea of particle filters was first developed in the statistical literature, and recently this methodology, also named sequential Monte Carlo filtering [20], [21] or CONDENSATION [22], has shown to be a successful approach in several applications of computer vision [22], [23], [24]. One important aspect of this method is to represent the posterior distribution of state variables given the image data by a set of weighted random samples, referred to as particles. For example, in our case, the state variables are text threshold couples. The method performs a traditional segmentation of the text image in the first frame and propagate the resulting threshold couples to other frames using particle filters. By introducing randomness in the exploration of the space of possible segmentation parameters in a Bayesian framework, the particle representation allows adaptation to changes of grayscale values both in the text and background by simultaneously maintaining multiple-hypotheses. In contrast to other filtering techniques that approximate posterior probabilities in parametric form, such as Kalman filters, this methodology allows us to evaluate the likelihood of the segmentation parameters directly from the corresponding recognized text string based on language modeling and OCR statistics. This is a key point of our approach, which allows us to compensate for OCR instabilities with respect to segmentation maps and which are due to the low resolution of characters (before resizing and interpolation), the short length of the strings and their unknown font.

In this section, we first introduce the Bayes filtering principle. Then, we define the specific components

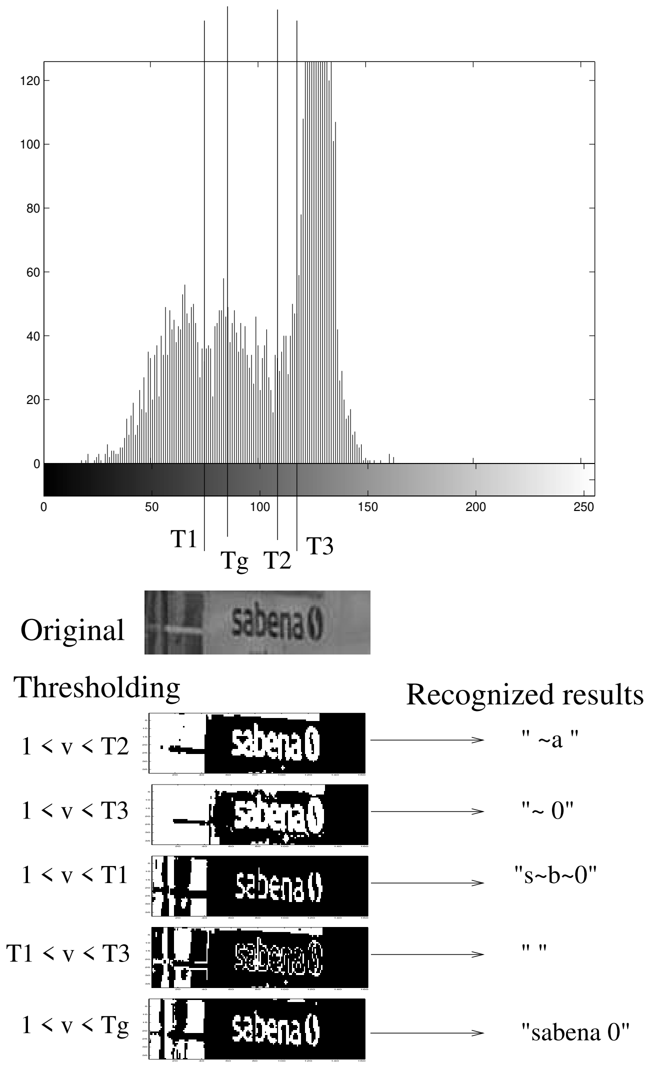


Fig. 3. Thresholding a text image according to its grayscale histogram (v is a gray level value). The optimal two mode threshold is T2. The optimal three mode thresholds are T1 and T3. However, the best recognition result is obtained using threshold Tg. The binary maps corresponding to the graylevels satisfying the equation on their left are displayed.

involved in the modeling of our filter. Finally, we describe our implementation of the particle filter algorithm.

### *A. Bayes filtering*

Bayes filters address the problem of estimating the posterior probability $p\left(x_t|o_{1:t}\right)$ of a dynamic state given a sequence of observations, where $x_t$ denotes the state $x$ at time $t$ and $o_{1:t}$ denotes the observation sequence from time $1$ to time $t$. For video text segmentation, the state $x_t = (l, u)_t$ is characterized by an upper (u) and a lower (l) segmentation threshold. The observations are the grayscale text images extracted and tracked in consecutive video frames. The goal of video text segmentation is to find the states that lead to an accurate segmentation or, better, to a correctly recognized string.

To derive a recursive update equation, we observe that posterior probability can be transformed by

Bayes rule to

$$p\left(x_t|o_{1:t}\right) = Z^{-1} p\left(o_t|x_t, o_{1:t-1}\right) p\left(x_t|o_{1:t-1}\right) \tag{1}$$

where $Z$ is the normalization constant

$$Z = p\left(o_t|o_{1:t-1}\right) \tag{2}$$

The prediction term $p\left(x_t|o_{1:t-1}\right)$ can be expanded by integrating over the state at time $t-1$:

$$p\left(x_t|o_{1:t-1}\right) = \int p\left(x_t|x_{t-1}, o_{1:t-1}\right) p\left(x_{t-1}|o_{1:t-1}\right) dx_{t-1} \tag{3}$$

Exploiting the two standard assumptions -independence of observations conditioned on the states ( i.e. $p\left(o_t|x_t, o_{1:t-1}\right) = p\left(o_t|x_t\right)$) and a first order Markov model for the state sequence (i.e. $p\left(x_t|x_{t-1}, o_{1:t-1}\right) = p\left(x_t|x_{t-1}\right)$), we obtain the following recursive equation for the posterior:

$$p\left(x_t|o_{1:t}\right) = Z^{-1} p\left(o_t|x_t\right) \int_{x_{t-1}} p\left(x_t|x_{t-1}\right) p\left(x_{t-1}|o_{1:t-1}\right) dx_{t-1} \tag{4}$$

The exploitation of equation (4) requires the definition of two conditional densities: the transition probability $p\left(x_t|x_{t-1}\right)$ and the data likelihood $p\left(o_t|x_t\right)$. Both models are typically time-invariant so that we can simplify the notation by denoting these models $p\left(x^{'}|x\right)$ and $p\left(o|x\right)$ respectively. They are presented in the next subsection, while the description of the particle filter implementation of the above equation is deferred to the end of the section.

### B. Probabilistic models for video text segmentation

*1) Transition probability:* In the context of video text segmentation, the transition probability $p\left(x^{'}|x\right)$ represents a probabilistic prior on text threshold variations. In practice, the goal of this term is also to allow the exploration of the state space. Broad priors allow for a fast exploration of the space, a necessity when the initialization is far from the best state value, while tigther priors allow for small refinements which are useful due to OCR instabilities. Different models could be considered. In this paper, we investigate the two following models.

**Adaptive uniform model** - In this case, the threshold variation is supposed to be due to background light shift, with the extent of threshold variations depending on the current state. Let $min = 0$ and $max = 255$ respectively denote the minimum and the maximum values of the grayscale in the image. Given $x = (l, u)$, a new sample has a uniform distribution in the ranges $[l_{min}, l_{max}]$ for $l^{'}$ and $[u_{min}, u_{max}]$ for $u^{'}$, with :

$$\begin{cases} l_{min} = l - \eta(l - min) \\ l_{max} = l + \eta(u - l) \end{cases} \quad \text{and} \quad \begin{cases} u_{min} = u - \eta(u - l) \\ u_{max} = u + \eta(max - u) \end{cases} \tag{5}$$

where $\eta$ is a constant. The distribution of $p\left(x^{'}|x = (150, 200)\right)$ in the adaptive uniform model is illustrated in figure 4a.

**Adaptive mixture model** - To allow for broader transition steps outside the uniform range, we can simply modify the above model by adding a Gaussian noise component on the state space out of the uniform range. The transition probability $p\left(x^{'}|x\right)$ is therefore defined as :

$$p\left(x^{'}|x\right) = \begin{cases} \gamma & \text{if } l^{'} \in [l_{min}, l_{max}] \,\&\, u^{'} \in [u_{min}, u_{max}] \\ \gamma e^{-\frac{(l^{'}-l_b)^2 + (u^{'}-u_b)^2}{2\sigma^2}} & \text{otherwise,} \end{cases} \tag{6}$$

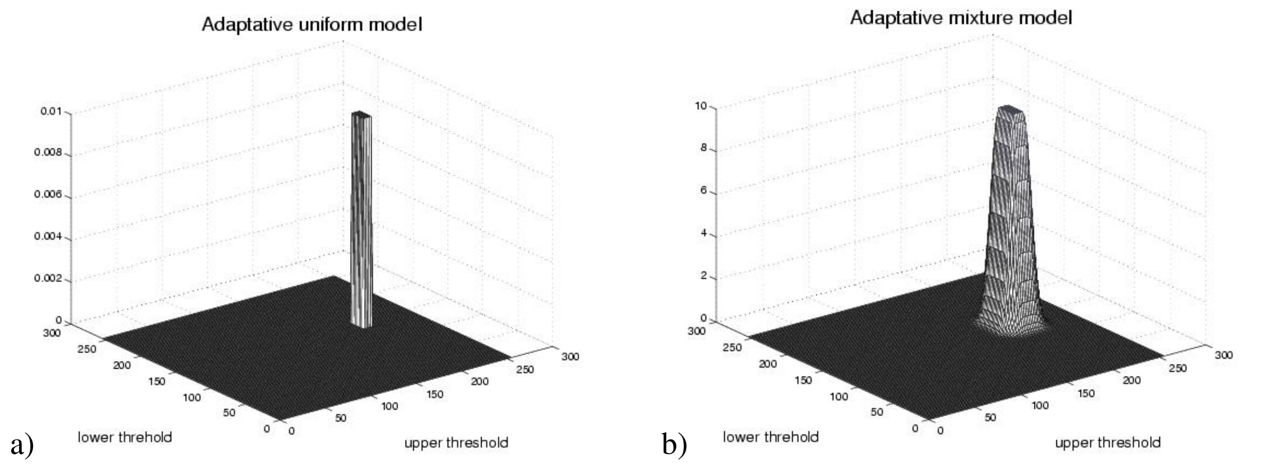


Fig. 4. a) Adaptive uniform model and b) Adaptive mixture model, of the transition probability $p\left(x^{'}|x=(150,200)\right)$.

where

$$l_b = \begin{cases} l_{min} & \text{if } l^{'} \leq l_{min} \\ l^{'} & \text{if } l^{'} > l_{min} \text{ and } l^{'} < l_{max} \\ l_{max} & \text{if } l^{'} \geq l_{max} \end{cases}, u_b = \begin{cases} u_{min} & \text{if } u^{'} \leq u_{min} \\ u^{'} & \text{if } u^{'} > u_{min} \text{ and } u^{'} < u_{max} \\ u_{max} & \text{if } u^{'} \geq u_{max} \end{cases} \tag{7}$$

and $\gamma$ is a normalization constant which does not affect the MCVTS algorithm. A typical transition probability distribution for the adaptive mixture model is illustrated in Fig. 4b.

*2) Data likelihood:* The data likelihood $p(o|x)$ provides an evaluation of the segmentation quality of the observed image $o$ given a pair of thresholds $x = (l, u)$. This evaluation could rely on the segmented image. However, computing accurate measures of segmentation quality in terms of character extraction is difficult without performing some character recognition analysis. Besides, visually well segmented image does not always lead to correct recognition. The OCR may produce errors due to the short length and the unknown font of the text string. Therefore, since ultimately we are interested in the recognized text string, the data likelihood will be evaluated based on the output $T$ of the OCR.

To extract the text string $T$, we follow the method described in [16]. In brief, we first binarize the image $o$ using $x$, and then remove noise regions using a connected component analysis step. We keep, as character components, the connected components that satisfy constraints on size, aspect ratio and fill-factor and apply OCR software on the resulting binary image to produce the text string $T$.

To model the data likelihood, we exploit some prior information on text strings and on the OCR performance based on language modeling (applied to character sequences) and OCR recognition statistics. From a qualitative point of view, the system works by identifying characters which are more reliably produced when the segmentation is ideal (i.e. the original text is recognized with no error) than when the segmentation is noisy. For instance, when given text-like backgrounds or inaccurate segmentations, the OCR system produces mainly garbage characters like ., ,!, & etc and simple characters like i,l, and r, whereas characters like A or G are rarely produced in these situations.

Let us define a text string $T$ as $T = (T_i)_{i=1..l_T}$ where $l_T$ denotes the length of the string and each character $T_i$ is an element of the character set $\mathcal{T}$ :

$$\mathcal{T} = (0, \ldots, 9, a, \ldots, z, A, \ldots, Z, G_b)$$

in which $G_b$ corresponds to any other garbage character. Finally, let us denote by $H_a$ (resp. $H_n$) the hypothesis that the string $T$ or the characters $T_i$ are generated from an accurate (resp. a noisy) segmentation. We then define the data likelihood as the probability of accurate segmentation $H_a$ given the string $T$:

$$p(o|x) \propto p(H_a|T) = \frac{p(T|H_a)p(H_a)}{p(T)} \tag{8}$$

Here $p(T)$ is given by:

$$p(T) = p(T|H_a)p(H_a) + p(T|H_n)p(H_n), \tag{9}$$

and the data likelihood is then proportional to:

$$p(o|x) \propto \frac{1}{1 + \frac{p(T|H_n)p(H_n)}{p(T|H_a)p(H_a)}}. \tag{10}$$

We estimated the noise free language model p$(.|H_a)$ by applying the toolkit on Gutenberg collections[1], which contains a large amount of book text. A bigram model was selected. Cutoff and backoff techniques [25] were employed to address the problems associated with sparse training data for special characters (e.g. numbers and garbage characters). The noise language model p$(.|H_n)$ was obtained by applying the same toolkit on a database of strings collected from the OCR (RTK from EXPERVISION) 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 prior ratio on the two hypotheses $\frac{p(H_n)}{p(H_a)}$ is modeled as:

$$\frac{p(H_n)}{p(H_a)} = b,$$

where $b$ is a bias that can be estimated from examples. In practice, we used a value of 0.7 . The data likelihood is then given by:

$$p(o|x) \propto \frac{1}{1 + \frac{\prod_{i=1}^{l_T} \mathrm{p}(T_i|H_n)}{\mathrm{p}(T_1|H_a)\prod_{i=2}^{l_T} \mathrm{p}(T_i|T_{i-1},H_a)} * b}. \tag{11}$$

Figure 5 illustrates the model on one example. In the third row, the targeted data likelihood is shown on the left. It is defined as $p(o|x) = black$ if not all the words in the groundtruth are recognized, and $p(o|x) = white^2$ otherwise . On the right of the third row, the figure also displays the proposed data likelihood for this image, at all possible states. It illustrates that our model is accurate, i.e. it provides high likelihoods when the words are correctly recognized. Even if the initial state (here provided by an Otsu algorithm [26] and shown with an arrow in the images) leads to an incorrectly recognized text

[1] www.gutenberg.net

[2] The output may contain additional characters, due to the image structures on the top left of the image.

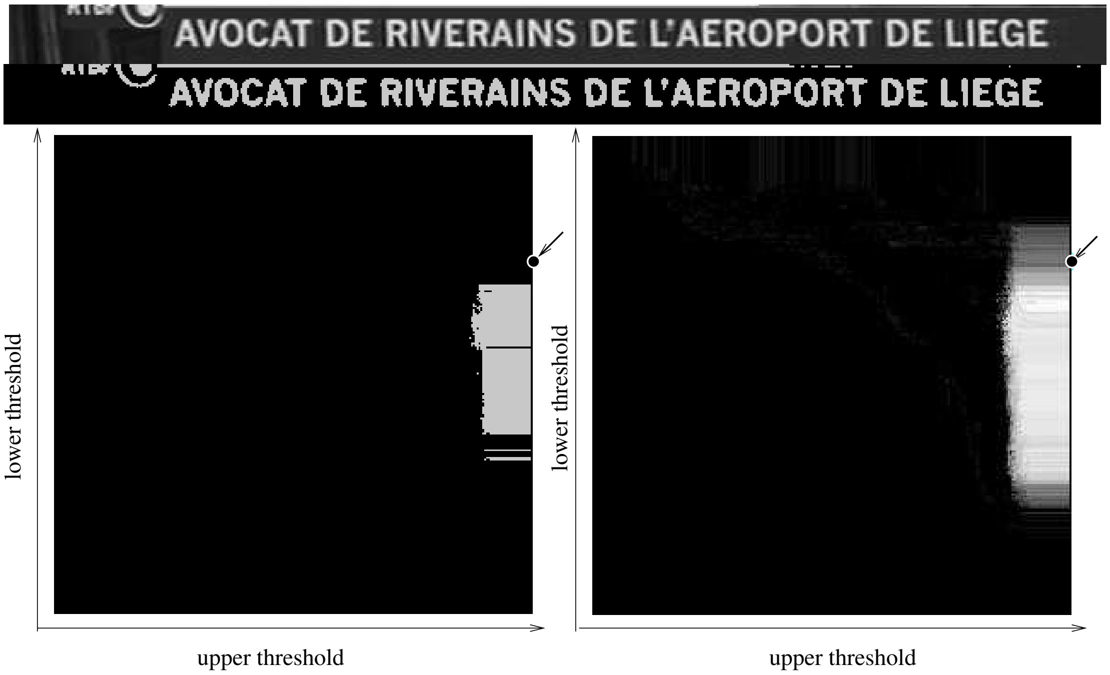


Fig. 5. Data likelihood approximation: the observed text image is displayed at the top. The second image displays the results of applying Otsu binarization, which corresponds to OCR output "V AVOCAT DE RIVERAINS DE L AEROPORT DE iIEGE". In the third row, the left image shows the states that lead to the recognition of *at least* all the words in the ground truth (but can contain other symbols due to noise on the top left of the image), the right image displays the proposed data likelihood at all the states.

string, the bayesian filtering methodology, thanks to the introduction of random perturbation and our data likelihood model, will still be able to find a state that provides the correct string. The bayesian filtering is implemented by a recursive particle filter that is described below.

### C. Particle approximation

The idea of the particle filter is to approximate the posterior $p(x_t|o_{1\ldots t})$ by a set of $N$ weighted samples $X_t = (x_t^j, w_t^j)_{j=1\ldots N}$ such that:

$$p(x_t|o_{1\ldots t}) \approx \sum_{j=1}^{N} w_t \delta\left(x_t^j - x_t\right)$$

where $\delta$ is the mass choice function ($\delta(0) = 1$, otherwise $\delta(x) = 0$). The initial set of samples represents the initial knowledge $p(x_1|o_1)$ and can be initialized using an Otsu algorithm applied on the first image. The recursive update is realized in three steps. First, sample $\tilde{x}_t^i$ from the approximated posterior $X_t$. Then, sample $x_{t+1}^i$ from the transition probability $p\left(x_{t+1}|\tilde{x}_t^i\right)$. Finally, assign $w_{t+1}^i = p\left(o_{t+1}|x_{t+1}^i\right)$ as the weight of the new sample $(x_{t+1}^i, w_{t+1}^i)$. In our case, since the number of samples per image is low, we add the new particles to the set $X_{t+1}$ of samples instead of replacing the old values with the new ones, as is normally done [20]. The following is the MCVTS algorithm presented in pseudo code.

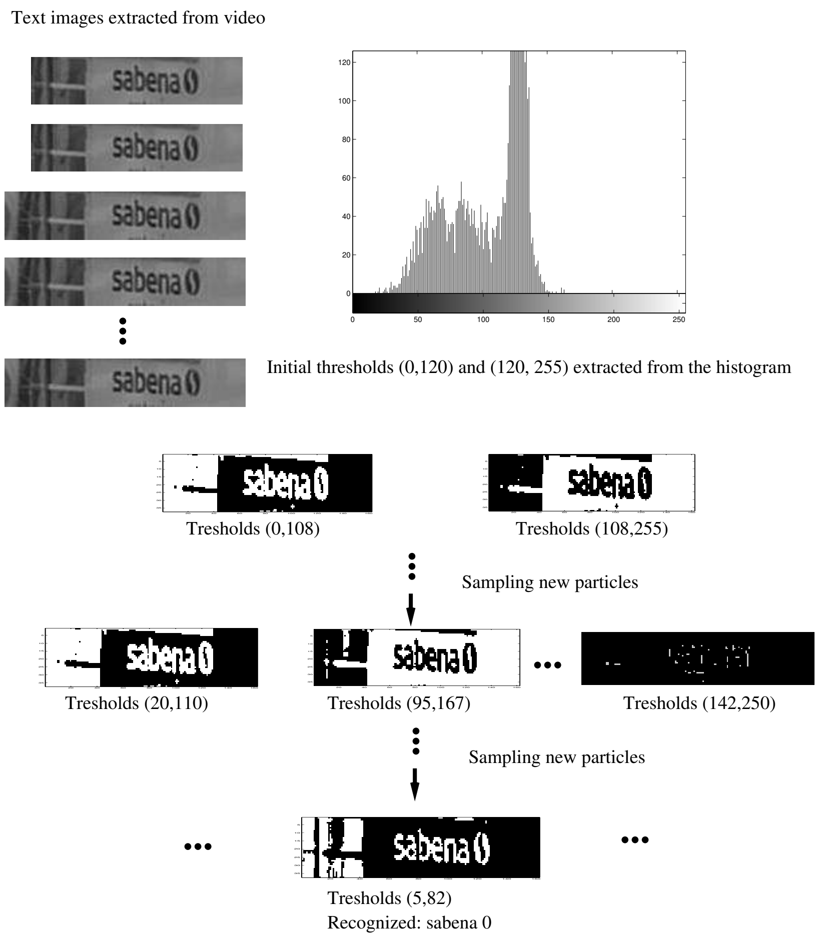


Fig. 6. Video text segmentation using particle filtering.

| |
|---|
| 1. initial $X_1$ using an Otsu algorithm; |
| 2. for each frame $t = 1, \ldots, n$ do step 3 and 4; |
| 3. for $i = 1$ to $m$ do<br>sample $\tilde{x}_t^i \sim X_t$;<br>sample $x_{t+1}^i \sim p\left(x_{t+1} \vert \tilde{x}_t^i\right)$;<br>set $w_{t+1}^i = p(o_{t+1} \vert x_{t+1}^i)$; |
| 4. $X_{t+1} = X_t$,<br>add the $m$ new samples $(x_{t+1}^i, w_{t+1}^i)$ to $X_{t+1}$. |
| 5. output the k-best text strings that corresponds to the segmentation with the highest data likelihood values. |

Figure 6 illustrates the procedure of the MCVTS algorithm. The initial threshold couple $x = (108, 255)$ and $x = (0, 108)$ are obtained by applying Otsu's thresholding algorithm on the first frame. This doesn't lead to a correct solution in this case. After several particle sampling steps, the states (threshold couples) cover a wide range of thresholds in the state space. At the end of the process, the threshold couple $x = (5, 82)$ yields the highest likelihood. The segmentation result using this threshold couple leads to a correct OCR output as shown in the figure, though the pictogram at the right of "sabena" is interpreted as a “0”.

Multiple string results are produced during the particle sampling procedure. In the good cases, the string with the highest likelihood is the correct one. However, in some cases, all the results contain one or several errors, not in the same place; or the correct result is recognized several times but with a lower likelihood than the optimal result, which occurs only one time and has an error that corresponds to a more reliable character than the corresponding one in the true answer. In order to exploit the redundancy and potential complementary nature of the results, we propose an algorithm to further compose the final recognition result from the characters of the more likely string results produced by the MCVTS algorithm. This algorithm is described in the next Section.

## III. Recognizer output voting error reduction technique (ROVER)

In this section, we present an algorithm for reducing video character recognition error rates using an output voting technique. The recognizer output voting error reduction technique (ROVER) [27] is used in automatic speech recognition systems (ASR) for compositing a new ASR output (sequence of words) from the outputs of multiple ASR systems. The system works by applying a sequential voting process to reconcile differences in ASR system outputs. The idea is to build a so called word transition network (WTN) via iterative applications of dynamic programming alignments so that voting can be applied at each position of the WTN. It has been shown that, in many cases, the composite output has a lower error rate than any of the individual systems due to the combination of complementary knowledge sources, such as acoustic and language models, that are embedded in each ASR system.

In the case of video text recognition, the OCR recognition results of a text string in different frames have differences that can be reconciled using the ROVER technique. Text strings are first detected, tracked in consecutive video frames and then segmented and recognized in each frame using the MCVTS algorithm. Since the basic unit of OCR recognition results are characters, we will build a character transition network instead of the word transition network in ASR systems. Figure 7 depicts the procedures of the algorithm. After having obtained the OCR recognition results of a text string in consecutive frames, we iteratively build a character transition network via dynamic programming alignments. Here, the recognition outputs obtained from different frames are assumed to be independent knowledge sources. Then the “best” output is composed by selecting, at each position of the character transition network, the character maximizing a criteria based on the occurence frequency of the character and its recognition confidence. The best output might be a new string that was not present in initial solutions.

### A. *Character transition network construction*

A character transition network (CTN) consists of an array of nodes and a set of arcs connecting two consecutive nodes. Each node represents a boundary between characters. Each arc indicates the presence of a character at a given position in the network. To make sure that there are no sequence of space characters in the OCR recognition results, we always keep only one space character in this case. Besides, we introduce a special label “NULL” to represent a character which is inserted by the dynamic programming alignment as a blank.

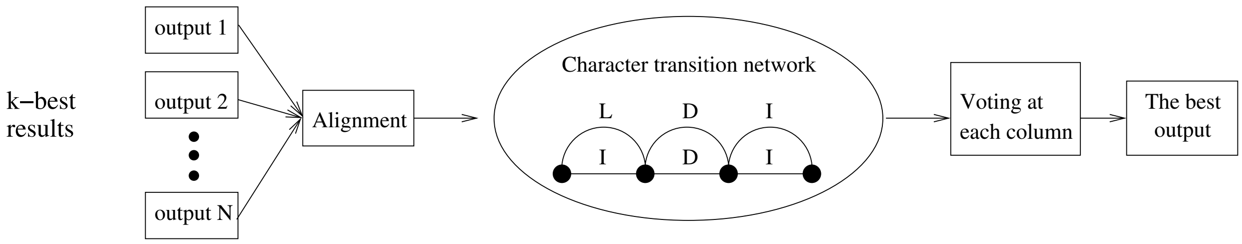


Fig. 7. Video Character Recognition Error Reduction Algorithm

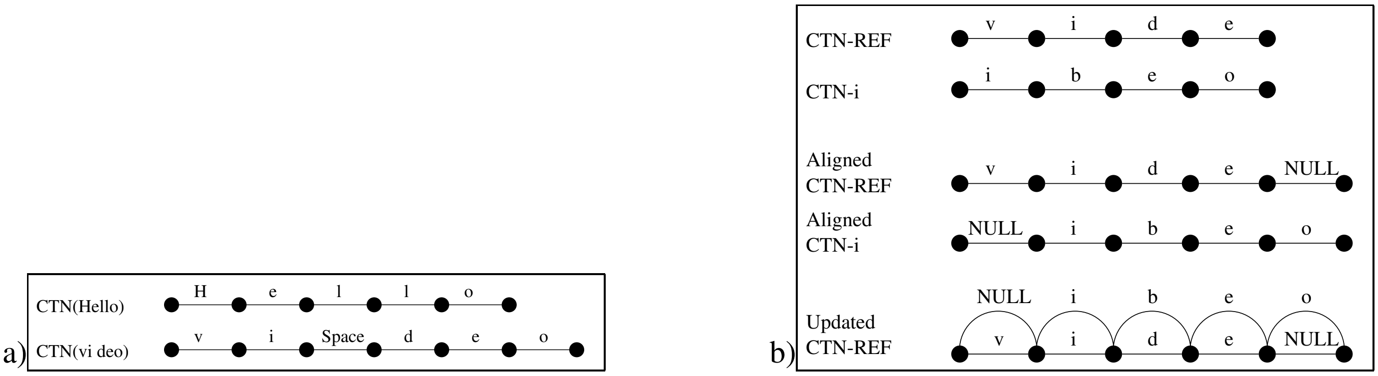


Fig. 8. a) Linear-topology character transition networks b) Alignment of two character transition networks

Every individual text string can be modeled by a linear topology CTN, as illustrated in Figure 8a. For multiple text outputs in consecutive frames, CTN with non-linear topology (more than one arc between two nodes) must be constructed by aligning the nodes of individual CTN. Let us examplify this alignment for two CTNs. Treating one CTN as a reference, denoted CTN-ref, the other CTN, denoted CTN-i, can be aligned using a 2-D dynamic programming alignment process[3]. This process yields two aligned CTNs with potentially inserted "NULL" arcs. Then, we can merge the two CTNs together by copying the corresponding arcs from CTN-i into CTN-REF. An example of such a process is depicted in Fig. 8b.

Iteratively merging the CTNs from all the outputs into a reference CTN-REF generates a composite CTN that models the temporal redundant outputs. However, this iterative composing process is affected by the order in which the CTNs are merged. Intuitively, the best strings should be introduced earlier in the CTN construction to serve as alignment references. We therefore introduce a confidence value for sorting the CTNs so that a "better" text string is merged as early as possible. As a confidence measure for the CTN modeling one text string $T$, we use the confidence $C(T)$ defined in equation 11 : $C(T) = p\left(o|x\right)$. We first sort the CTNs according to their confidences. Then, the CTN with the highest value is set as the initial CTN-REF. The remaining CTNs are merged into the reference CTN-REF one by one according to their confidences. This process produces a network model of all the outputs of a given string in consecutive

[3]We use SCLITE, a tool for scoring and evaluating the output of speech recognition systems. Sclite is part of the NIST SCTK Scoring Toolkit. http://www.icsi.berkeley.edu/Speech/docs/sctk-1.2/sclite.htm

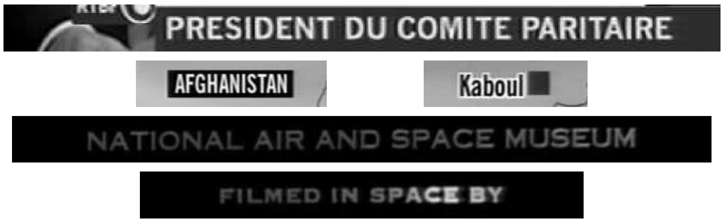


Fig. 9. Examples of located embedded text in video.

frames.

### B. Best character sequence searching in character transition network

The best character sequence is determined by a scoring process in the composite CTN. The traditional ROVER algorithm, developed for ASR systems, implements the scoring process as a voting procedure among different recognizers (voters). Given a composite CTN of a text string occurring in $m$ frames and that has $n+1$ nodes, we denote $c_{ij}$ as the $jth$ arc behind the $ith$ node. We further define the probability of occurrence $F(c_{ij})$ of $c_{ij}$ as:

$$F_i(c_{ij}) = \frac{\sum_{k=1}^{m} \delta(c_{ij}, c_{ik})}{m} \tag{12}$$

where $\delta$ is the mass choice function:
Besides the probability of occurrence, a confidence value of each character is also necessary. Let us follow the definition of the confidence value proposed in Subsection II-B.2 and define the confidence value of a character as:

$$Conf(c_{ij}) \propto p(H_a|c_{ij}) = \frac{1}{1 + \frac{\mathrm{p}(c_{ij}|H_n)}{\mathrm{p}(c_{ij}|H_a)} b} \tag{13}$$

which only uses the character unigram statistics of the noise-free langage model (as opposed to bigram in II-B.2). The confidence of the "NULL" transition arc is considered as a parameter $\beta$ of the scoring scheme that will be learned by cross-validation. The general scoring formula is given by :

$$S(c_{ij}) = \alpha F_i(c_{ij}) + (1-\alpha) Conf(c_{ij})$$

where $\alpha \in [0, 1]$ is a constant weight parameter. When $\alpha = 1$, the score only depends on the occurrence of the characters. When $\alpha = 0$, the score relies only on the individual confidence of each recognized character. The value of this parameter will also be learned using cross-validation. For each position $i$ in the CTN, the algorithm outputs the character whose score S is the highest among all candidates at this position.

## IV. Experiments and discussion

The MCVTS and the ROVER algorithm are tested on a dataset of two hours of broadcasted video : two 30 minute news programs gathered in the context of the CIMWOS[4] project, and a documentary of one

[4] "Combined Image and Word Spotting" project granted by the European IST Programme

TABLE I

PERFORMANCE COMPARISON BETWEEN THE 1-BEST MCVTS (M=3) METHOD, THE DIFFERENT MAX-MIN METHODS AND THE AVERAGE VALUE METHOD: EXTRACTED CHARACTERS (EXT.), CHARACTER RECOGNITION RATE (CRR) AND PRECISION (PREC.) THE BASELINE SYSTEM IS THE AVERAGE IMAGE METHOD RE-IMPLEMENTED ACCORDING TO [8].

| Methods | Ext. | CRR | Prec. | WRR |
|---|---|---|---|---|
| Minimum value | 2692 | 81.5% | 87.8% | 81.8% |
| Maximum value | 2472 | 73.3% | 85.9% | 78.0% |
| Best one in Max-Min | 2878 | 86.1% | 86.7% | 84.4% |
| Average value | 2929 | 88.9% | 87.9% | 86.8% |
| Multi-model | 2939 | 89.2% | 88.0% | 87.1% |
| 1-best adaptive uniform MCVTS | 2935 | 92.3% | 91.2% | 89.0% |
| 1-best adaptive mixture MCVTS | 2928 | 93.9% | 93.0% | 90.6% |

hour. The text strings were first detected and tracked using the system described in [5]. This resulted in a database of $247$ different text strings (2899 characters and 786 words), coming from $6944$ text images (about 28 images per text string in average). Figure 9 shows some image examples in the database. Among these, approximately $6\%$ of the text strings were scene texts.

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 :

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

$N$ 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. The number of correctly recognized characters is computed using an edit distance[5] between the recognized string and the ground truth[6]. More precisely, let $l_T$, $del$, $ins$ and $sub$ respectively denote the length of the recognized text string, the number of deletions, insertions, and substitutions obtained when computing the edit distance. The number $N_r$ of correctly recognized characters in this string is then defined as :

$$N_r = l_T - (del + sub)$$

Intuitively, if in order to match the ground truth, we need to delete a character or substitute a character, it means that this character is not in the ground truth. In addition to the above rates, we compute the word recognition rate (WRR) to get an idea of the coherency of character recognition within one solution. In an indexing application, this rate makes more sense than the CRR. For each text string, we count the words from the ground truth 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.

[5]The edit distance of two strings, s1 and s2, is defined as the minimum number of character operations needed to change s1 into s2, where an operation is one of the following : deletion, insertion, substitution.

[6]Only numeric, upper and lower case letters are kept.


Fig. 10. Examples of video image with compression noise. Despite the large size of text in this image, text distorsions are visible (see text).

### A. MCVTS recognition results

To evaluate the performance of the MCVTS algorithm without ROVER, we use the 1-best result, i.e. we output the result of the particle having the highest likelihood. The results are given in table I. They are compared with the performance of the different Max-Min methods and the average value method AVE implemented by us, following the descriptions in [17] and [8]. Since the Max (resp. Min) method is better adapted to recognize black (resp. white) characters, we need a way to automatically select which one applies best to the particular case at hand (this point is not addressed in [17]). From the two strings produced by the Max and the Min methods, we selected the one that had the largest likelihood according to Eq. 11. The table also provides the results of the multi-model scheme [16], which is applied on individual text images of the database, and whose principle is to select the best string result among those produced by a Kmeans segmentation algorithms applied with K equal to 2, 3 or 4 (and followed by OCR recognition).

The Max-Min methods do not provide better results than the multi-model based on static images. One important reason for this is the fact that, in the CIMWOS application, the broadcasted videos are compressed on the fly with an MPEG-2 encoder and stored for further processing. Text regions have strong edges, which are high frequency signals. They are affected by compression, especially at low scales, creating “bridges” between strokes or smoothing salient but small strokes (see Fig. 10). Such noise in one image of a text string image sequence, directly impacts the constructed Max or Min images, pushing text pixels into background and vice-versa.
Although more robust, the results of the average value method are similar to those of the multi-model method, which works on single images.

Both versions of the MCVTS algorithm performed better than the state-of-the-art methods. Applying a significance test on the difference of two proportions (the recognition rates), we obtained p-values of $10^{-14}$ and $10^{-35}$ (resp. $0.03$ and $6. \times 10^{-5}$) when comparing the CRR (resp. the WRR) of the two MCVTS algorithms with the best state-of-the-art method. This shows that the improvements are statistically significant. By checking the tested samples in the database, we found out that the MCVTS algorithm selected better segmentation parameters when the automatically detected text images were noisy, or when the grayscale values of characters spanned a wide range (e.g. last rows of Fig. 9). Thus, the method leads

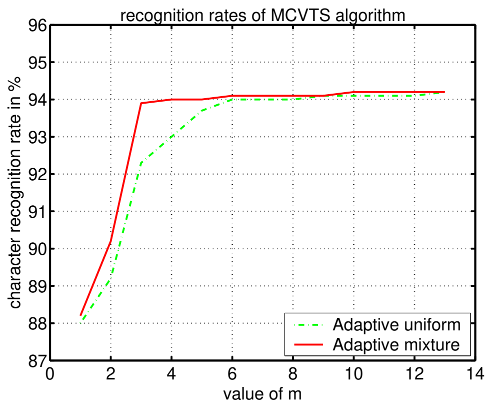


Fig. 11. Character recognition rates of MCVTS algorithms with varying "m".

to improved recognition rates on both types of text data present in our database, namely news text and credit text.

Regarding the remaining mistakes, we can identify two main sources of error. First, the presence of visual effects (see Fig. 13), which decreases the likelihood of the existence of a global threshold set leading to an entirely well recognized string. This problem is overcome by the ROVER algorithm. The second error type is due to the confusion often made by the OCR between similar looking characters, e.g. l, 1 (one) and i , especially when the dot is filtered out. Such confusion occurs mostly for small text sizes (but not only), due to the compression noise, and because of the short length of the character strings, which prevents the OCR system from building good statistics about the right font and the right size of the text to be recognized. On several occasions two alternative solutions recognized at different time instants involved only one character substitution. The langage model we introduced (see Eq. 11) is not specific enough to always select the appropriate solution, and many errors may occur in these situations. For instance, the news text 'Adjoint de la Directlon' was prefered over 'Adjoint de la Direction' by the current model. One solution to this problem is to use a dictionary, as done in [9]. However, common dictionaries lack common people's name, and some proper noun's, which are frequent in video text. In [9], this problem was circumvented by using the output of the closed captions coming along the news video.

From the two dynamic models proposed in the paper, the adaptive mixture model yields the best results in terms of character recognition rate and precision. Figure 11 illustrates the character recognition rates of the MCVTS algorithms with varying $m$. Both the dynamic models give similar results when $m$ is above $6$, which shows that all these dynamic models lead to the estimation of the same maximum mode in the likelihood. The dynamic model is an important factor only when the computation resource is limited ($m$ is small). In the context of the CIMWOS project, a value of $m = 3$ has been chosen. The average number of samples per text string is thus around $80$.

TABLE II

PERFORMANCE COMPARISON OF THE ROVER ALGORITHM, THE 1-BEST MCVTS ALGORITHM AND OTHER ALGORITHMS : EXTRACTED CHARACTER NUMBER (EXT.), CHARACTER RECOGNITION RATE (CRR), PRECISION (PREC.) AND WORD RECOGNITION RATE (WRR)

| methods | Ext. | CRR | Prec. | WRR |
|---|---|---|---|---|
| Multi-model | 2939 | 89.2% | 88.0% | 87.1% |
| Best one in Max-Min | 2878 | 86.1% | 86.7% | 84.4% |
| Average value | 2929 | 88.9% | 87.9% | 86.8% |
| 1-best MCVTS | 2928 | 93.9% | 93.0% | 90.6% |
| k-best MCVTS+ROVER | 2870 | 94.1% | 95.1% | 92.0% |

### *B. ROVER results*

To apply ROVER on a text image sequence, we first run the MCVTS algorithm on the sequence and then select the $k$ best recognition results from the recognition hypotheses using the confidence defined in Eq. 13. The value $k$ is experimentally set to be the number of frames of the text image sequence.
Additionally, the ROVER algorithm needs the setting of two parameters : the mixing parameter $\alpha$, and the confidence $\beta$ of the NULL transition. To avoid biasing the results, we applied a 2-fold cross-validation scheme. The database was split into two parts A and B. Then, the parameters were trained on set A to achieve the highest $CRR + Prec$ value. The method was then tested on set B. The method was repeated by exchanging the roles of A and B. The results in Table II are given by the test results on A (parameters trained on B) and B (parameters trained on A).

The results for ROVER show an improvement over the MCVTS method alone. More specifically, there is an improvement in the character precision rate, and an insignificant increase in the character recognition rate. This suggests that ROVER is helpful at removing the insertion of erroneous characters at the output of the OCR due either to noise or the recognition of one character as two characters, but is less able at correcting character substitutions. Nevertheless, ROVER offers a way to compose new words which translates into a better word recognition rate (note however that this improvement is not statistically significant). This is illustrated by the two examples of Fig. 12. In this case, none of the results from the individual frames contained the correct string. It is due to a special effect in the display of these texts (waving effect of high and low intensity), as shown by images in Fig. 13. However, in some cases, the numbers of errors for some particular letter in the individual results is still too important to be corrected. For instance, in the example of 14, the J at the beginning is missed most of the time because of its dark appearence.
Closer analysis of the results also shows that there is no significant improvement for the static close-captions of our two news programs. Indeed, in these cases, text is static, background changes are marginal, and even the compression noises do not seem to evolve. Therefore, as soon as good segmentation parameters have been found by the MCVTS algorithm, the outputs of the OCR remain almost identitical over time. This limits the usefulness of ROVER in this context, as the multiple OCR recognition results can not really be thought of as independent knowledge resources. The use of more than one OCR software

| | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | A | S | I | O | C | I | A | T | I | | P | R | O | D | U | C | r | R | s |
| | A | S | S | O | C | I | A | T | E | | P | R | O | D | U | C | E | i | s |
| | A | S | S | O | C | | A | 1 | [ | | P | R | O | D | U | C | E | R | S |
| | A | | | | | I | A | T | E | | P | R | O | D | U | C | E | t | B |
| | A | S | S | O | C | I | A | T | E | | P | R | O | D | U | C | [ | R | 5 |
| | A | | S | | | I | A | T | I | | f | R | O | D | U | C | | | |
| | A | S | S | O | C | I | A | I | I | | | | | D | U | C | E | R | S |
| | A | | | | | | o | | | | P | R | O | D | U | | | | |
| | | | | | o | | A | T | [ | | P | R | O | D | U | C | E | R | S |
| | | | | | o | | A | T | E | | P | : | ) | D | U | C | E | R | S |
| | | | | | | I | A | T | [ | | | | | D | U | C | E | R | S |
| | | | | | C | I | A | T | r | | P | R | O | D | U | C | E | R | S |
| | A | | g | O | C | I | A | T | E | | P | R | O | D | U | C | E | R | S |
| = | A | S | S | O | C | I | A | T | E | | P | R | O | D | U | C | E | R | S |

| | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | T | O | N | I | | f | Y | E | R | S | | G | R | A | E | M | E | | F | E | R | G | U | S | O | N |
| | T | | N | I | | M | Y | E | R | S | | | R | A | E | M | E | | F | E | R | G | U | S | O | N |
| | T | O | N | I | | | | | | S | | | | | | M | E | | F | E | R | G | U | S | O | N |
| | T | O | N | I | | | | | | | | | | | | M | E | | F | E | R | G | U | S | O | N |
| | T | O | N | I | | | | | | | | G | | | | | | | F | E | R | G | U | S | O | N |
| | | O | N | I | | M | Y | E | R | S | | G | R | A | | | | | | | | | U | S | O | N |
| | T | O | N | I | | M | Y | E | R | S | | G | | | | | | | | | | | U | S | O | N |
| | T | O | N | I | | M | Y | E | R | S | | | | | | M | E | | | | | | U | S | O | N |
| | T | O | N | I | | M | Y | E | R | S | | | | | | M | E | | | | | | | | | N |
| | C | ) | N | I | | M | Y | E | R | S | | G | R | A | E | M | E | | F | E | R | G | U | S | O | N |
| | | O | N | I | | M | Y | E | R | S | | G | R | A | E | M | E | | F | E | R | G | U | S | O | N |
| | | O | N | 1 | | M | Y | E | R | S | | G | R | A | E | M | E | | F | E | R | G | U | S | O | N |
| = | T | O | N | I | | M | Y | E | R | S | | G | R | A | E | M | E | | F | E | R | G | U | S | O | N |

Fig. 12. Voting of multiple recognition results of two video text strings in the database.

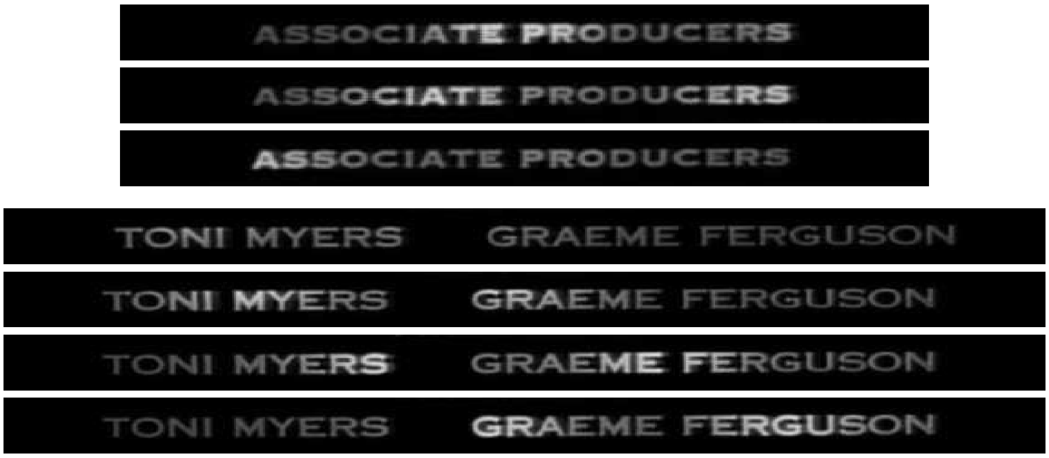


Fig. 13. Examples of text images associated with Fig. 12.

is a potential way to improve this issue.

## V. Conclusions

In this paper, we proposed and evaluated two algorithms, the MCVTS and the ROVER algorithms, in order to improve the recognition of video text by exploiting temporal information in multiple frames.

The MCVTS algorithm has three main advantages for segmenting video text. Firstly, the algorithm proposes a methodological way to search for segmentation parameters that lead to accurate string results. The algorithm uses a Bayesian framework and adapts itself to the data by sampling in proportion to the posterior likelihood. This enable us to propose an accurate probability model based directly on OCR results

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | U | D | Y | | C | A | R | R | O | L | L | | P | | | S | | S | | | F | E | R | G | U | S | O | N |
| | | U | D | Y | | C | A | R | R | O | L | L | | P | | | S | | | @ | | F | E | R | G | U | S | O | N |
| | | U | D | | | C | A | R | R | O | L | L | | P | H | Y | L | L | I | S | | F | E | R | G | U | S | O | P |
| | | U | D | Y | | C | A | R | R | O | L | L | | P | I | 4 | | L | f | S | | F | E | R | G | U | S | O | N |
| | | U | E | Y | | C | A | R | R | O | L | L | | P | H | | L | | I | S | | F | E | R | G | U | S | O | N |
| | | U | D | Y | | C | A | R | R | @ | L | | | P | H | Y | L | L | I | S | | F | E | R | G | U | S | O | I |
| | | t | J | D | Y | C | A | R | R | O | | | | P | N | Y | L | L | I | S | | F | E | R | G | U | S | O | I |
| | | U | D | Y | | C | A | R | R | O | | | | P | H | | L | L | I | S | | F | E | R | G | U | S | O | I |
| | | D | | | | C | A | R | R | O | | | | P | H | Y | L | L | I | S | | F | E | R | G | U | S | O | I |
| | | D | | C | , | A | , | R | R | | | | , | P | | | L | | I | S | | F | E | R | G | U | S | O | f |
| | | U | D | Y | | | | | | | | | | | | | | | | S | | F | E | R | G | U | S | O | r |
| | | U | D | Y | | | | | | | | | | | | | | | | S | | F | E | R | G | U | S | O | |
| | | U | D | Y | | | | | | | | | | | | | | | | S | | F | E | R | G | U | S | O | |
| | | U | D | Y | | | | | | | | | | | | | | | I | S | | F | E | R | G | U | S | O | |
| | | U | D | Y | | | | | | | | | | | i | | I | | I | S | | F | E | R | G | U | S | O | |
| | | U | D | Y | | | | | | | L | L | | P | t | Y | L | L | I | S | | F | E | R | G | U | S | O | |
| | | U | D | Y | | | | | | I | L | | | P | H | Y | L | L | I | S | | F | E | R | G | U | S | O | I |
| | | I | U | I | | | | | | | L | L | | P | N | Y | L | L | I | S | | F | E | R | G | U | S | O | I |
| | J | U | D | Y | | | | | | | | | | P | H | Y | L | L | I | S | | F | E | R | G | U | S | O | r |
| = | | U | D | Y | | C | A | R | R | O | L | L | | P | H | Y | L | L | I | S | | F | E | R | G | U | S | O | I |

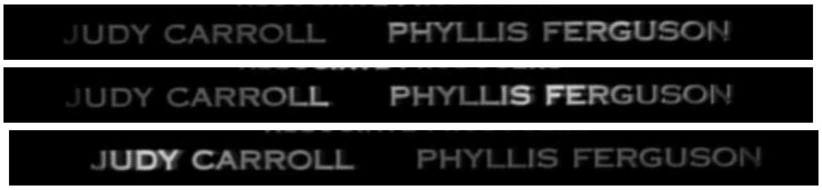


Fig. 14. Example of a text recognition sequence still leading to errors.

instead of estimating the posterior based on the quality of segmented images. Secondly, the algorithm does not require precise tracking and registration of text images among video frames at pixel or sub-pixel level. This is an interesting property as this means that the algorithm can be directly applied to sequences of moving scene text images which might not be easy to register due to some imaging distortions. Finally, the MCVTS algorithm is very easy to implement and also easy to extend to other state spaces, such as parameters of local thresholding techniques (e.g. Niblack binarization). The results of the conducted experiments have demonstrated the validity of this approach in comparison with state-of-the-art methods.

Also, we applied a voting technique (ROVER) to fuse, at the character level, the more likely string results produced by the MCVTS algorithm. The method is based on string alignment and the voting relies on character recognition confidence and frequency of occurence. The method has been shown to composite correct words or strings from OCR recognition results even if none of these results contained the correct answer. The improvement is insignificant, with respect to the MCVTS method alone, for static inserted text (the majority of strings in our database). However, the method has been found to be useful when a text string image sequence is affected by unstationary visual noise, like in the case of special video effects, lighting changes in scene text, or compression noise in moving text.

The proposed method, added to a text detection module [5], has been used in the context of the CIMWOS project to index French news programs. It was run every day for 3 months on the 30 minute daily news. It was integrated in an information retrieval application with other modules (speech transcripts, speaker identification, object localisation...). Journalists experimented with the retrieval system and reported, among other things, 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 very accurate. More recently, we ran our system on the video data of the TRECVID track of NIST to generate the text detection and recognition cues, which are available for all partners.

## References


[1] M. Swain and H. Ballard, "Color indexing," *Int. Journal of Computer Vision*, vol. 7, pp. 11–32, 1991.

[2] B.S. Manjunath and W.Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 837–842, Aug. 1996.

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

[4] P. Clark and M. Mirmehdi, "Recognising text in real scenes," *Int. Journal of Document Analysis and Recognition (IJDAR)*, pp. 243–257, 2002.

[5] Datong Chen, Jean-Marc Odobez, and Jean-Philippe Thiran, "A Localization/Verification Scheme for Finding Text in Images and Video Frames Based on Contrast Independent Features and Machine Learning Methods," *Signal Processing: Image Communication*, vol. 19, pp. 205–217, Mar. 2004.

[6] Rohini K. Srihari, Zhongfei Zhang, and Aibing Rao, "Intelligent indexing and semantic retrieval of multimodal documents," *Information Retrieval*, vol. 2, no. 2/3, pp. 245–275, 2000.

[7] Josef Sivic and Andrew Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Int. Conf. Computer Vision*, Nice, France, 2003, pp. 1470–1477.

[8] H. Li and D. Doermann, "Text enhancement in digital video using multiple frame integration," in *Proc. ACM Multimedia*, Orlando, Florida, USA, 1999, vol. 1, pp. 385–395.

[9] T. Sato, T. Kanade, E. K. Hughes, M. A. Smith, and S. Satoh, "Video OCR: indexing digital news libraries by recognition of superimposed caption," *Multimedia Systems*, vol. 7, no. 5, pp. 385–395, Sept. 1999.

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

[11] K. Sobottka, H. Bunke, and H. Kronenberg, "Identification of text on colored book and journal covers," in *Proc. Int. Conf. on Document Analysis and Recognition*, 1999, pp. 57–63.

[12] H. Kamada and K. Fujimoto, "High-speed, high-accuracy binarization method for recognizing text in images of low spatial resolutions," in *Proc. Int. Conf. on Document Analysis and Recognition*, Sept. 1999, pp. 139–142.

[13] O. Hori, "A video text extraction method for character recogntion," in *Proc. Int. Conf. on Document Analysis and Recognition*, Sept. 1999, pp. 25–28.

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

[15] Christian Wolf, Jean-Michel Jolion, and Francoise Chassaing, "Text localization, enhancement and binarization in multimedia documents," in *Proc. Int. Conf. on Pattern recognition*, Quebec City, Canada., Aug. 2002, pp. 1037–1040.

[16] D. Chen, J-M. Odobez, and H. Bourlard, "Text Detection and Recognition in Images and Videos," *Pattern Recognition*, vol. 37, no. 3, pp. 595–609, Mar. 2004.

[17] R. Lienhart and A. Wernicke, "Localizing and segmenting text in images and videos," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 12, no. 4, pp. 256–268, 2002.

[18] H. Li, D. Doermann, and O. Kia, "Automatic text detection and tracking in digital videos," *IEEE Trans. Image Processing*, vol. 9, no. 1, pp. 47–156, 2000.

[19] X. Ye, M. Cheriet, and C.Y. Suen, "StrCombo : Combination of String Recognizers," *Pattern Recognition Letters*, vol. 23, pp. 381–394, 2002.

[20] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for on-line non-linear/non-gaussian," *IEEE Trans. Signal Processing*, pp. 100–107, 2001.

[21] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*, Springer-Verlag, 2001.

[22] M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," in *4th European Conf. Computer Vision*, 1996, vol. 1, pp. 343–356.

[23] K. Nummiaro, E. Koller-Meier, and L. Van Gool, "Object tracking with an adaptive color-based particle filter," in *Proc. Symposium for Pattern Recognition of the DAGM*, Sep. 2000.

[24] P. Perez, A. Blake, and M. Gangnet, "Jetstream: Probabilistic contour extraction with particles," in *Proc. Int. Conf. on Computer Vision*, Vancouver, July 2001, pp. 424–531.

[25] 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*, vol. 35, pp. 400–401, 1987.

[26] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 1, no. 9, pp. 62–66, 1979.

[27] J.G. Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover)," in *Proc. IEEE Automatic Speech Recognition and Understanding Workshop*, 1997, pp. 347–352.