IDIAP RESEARCH REPORT

Text Detection and Recognition in Images and Video Sequences

Datong Chen
IDIAP

IDIAP–RR 03-44

Aug. 2003

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 03-44

TEXT DETECTION AND RECOGNITION IN IMAGES AND VIDEO SEQUENCES

Datong Chen
IDIAP

AUG. 2003

Summary

Text characters embedded in images and video sequences represents a rich source of information for content-based indexing and retrieval applications. However, these text characters are difficult to be detected and recognized due to their various sizes, grayscale values and complex backgrounds. This thesis investigates methods for building an efficient application system for detecting and recognizing text of any grayscale values embedded in images and video sequences. Both empirical image processing methods and statistical machine learning and modeling approaches are studied in two sub-problems : text detection and text recognition.

Applying machine learning methods for text detection encounters difficulties due to character size, grayscale variations and heavy computation cost. To overcome these problems, we propose a two-step localization/verification approach. The first step aims at quickly localizing candidate text lines, enabling the normalization of characters into a unique size. In the verification step, a trained support vector machine or multi-layer perceptrons is applied on background independent features to remove the false alarms.

Text recognition, even from the detected text lines, remains a challenging problem due to the variety of fonts, colors, the presence of complex backgrounds and the short length of the text strings. Two schemes are investigated addressing the text recognition problem : bi-modal enhancement scheme and multi-modal segmentation scheme. In the bi-modal scheme, we propose a set of filters to enhance the contrast of black and white characters and produce a better binarization before recognition. For more general cases, the text recognition is addressed by a text segmentation step followed by a traditional optical character recognition (OCR) algorithm within a multi-hypotheses framework. In the segmentation step, we model the distribution of grayscale values of pixels using a Gaussian mixture model or a Markov Random Field. The resulting multiple segmentation hypotheses are post-processed by a connected component analysis and a grayscale consistency constraint algorithm. Finally, they are processed by an OCR software. A selection algorithm based on language modeling and OCR statistics chooses the text result from all the produced text strings.

Additionally, methods for using temporal information of video text are investigated. A Monte Carlo video text segmentation method is proposed for adapting the segmentation parameters along temporal text frames. Furthermore, a ROVER (Recognizer Output Voting Error Reduction) algorithm is studied for improving the final recognition text string by voting the characters through temporal frames.

The whole system was successfully evaluated on large databases of camera-based images and

video sequences obtained in context of two European projects ASSAVID[1] and CIMWOS[2].

[1]European project ASSAVID : Automatic Segmentation and Semantic Annotation of Sports Videos, 5th Framework Programme, Information Society Technology, supported by OFES. Web site : http ://www.bpe-rnd.co.uk/assavid/

[2]European project CIMWOS : Combined IMages and WOrd Spotting , 5th Framework Programme, Information Society Technology, supported by OFES. Web site : http ://www.xanthi.ilsp.gr/cimwos/

Version abrégée

Les textes inclus dans des images et des séquences vidéos sont une source d'information très riche pour les applications d'indexation et de recherche automatique. Cependant, ces caractères sont difficiles à détecter et à reconnaître en raison de la variabilité de leurs tailles, de leurs niveaux de gris et de leurs arrière-fonds. Cette thèse étudie des méthodes génériques pour construire un système capable de détecter et de reconnaître de tels textes au sein d'images fixes et de vidéos. Des modélisations statistiques par apprentissage ainsi que des méthodes de traitement d'image plus empiriques sont proposées pour résoudre les deux sous-problèmes majeures posés par notre problème : d'un coté la détection et la localisation du texte dans les images, de l'autre la reconnaissance du texte détecté.

L'utilisation de méthodes par apprentissage en détection de texte se heurte aux difficultés causées par la variabilité de la taille et des valeurs de niveau de gris des caractères, ainsi qu'au coût de calcul de ces méthodes. Pour surmonter ces problèmes, nous proposons une approche en deux étapes : localisation de texte puis vérification. La première étape vise à localiser rapidement des régions horizontales de l'image qui contiennent potentiellement des lignes de texte. La hauteur de ces régions est ensuite normalisée, ce qui permet de réduire la variance de la taille des caractères en entrée de l'étape suivante. Lors de l'étape de vérification des régions extraites, une machine à vecteurs de supports (support vector machine) ou un perceptron multicouches est appliqué après entraînement sur des caractéristiques de l'image invariantes par repport au du niveau de gris et de l'arrière fond du texte, ceci afin d'éviter les fausses détections.

La reconnaissance de texte, mme appliquée aux lignes contenant potentiellement du texte, reste un problème difficile étant donné la diversité des polices et des couleurs, la présence d'arrière-fonds complexes et la faible longueur des chanes de caractères. Deux approches sont étudiées afin de résoudre le problème de la reconnaissance : une approche par augmentation de contraste reposant sur une hypothèse de bi-modalité des niveaux de gris, et une approche avec segmentation multi-modale. Pour l'approche bi-modale, nous proposons un ensemble de filtres permettant d'accentuer les caractères contrastés, ce qui conduit ensuite à une meilleure binarisation des caractères en noir et blanc avant l'application d'un algorithme commercial de reconnaissance optique de caractères (ROC). Dans l'approche multimodale, la reconnaissance est abordée par une étape de segmentation suivie par une étape de reconnaissance optique de caractères dans un cadre d'hypothèses multiples. Plus précisément, lors de l'étape de segmentation, nous modélisons la distribution des niveaux de gris par un mélange de gaussiennes ou un champ de Markov à K classes, K pouvant varier entre 2 et 4. Les segmentations qui en résultent sont post-traitées par un algorithme de décomposition en composantes connexes et d'un algorithme imposant une contrainte d'uniformité des niveaux de gris, puis traitées par un logiciel de ROC. Un algorithme

de sélection basé sur un modèle de langage et les statistiques ROC sélectionnent le texte parmi toutes les chanes de caractères proposées.

De plus, des méthodes permettant d'utiliser l'information temporelle des textes vidéo sont étudiées. On propose une méthode pour adapter des paramètres de segmentation au fil des trames vidéos de texte à l'aide d'une méthode de Monte-Carlo séquentielle. En outre, un algorithme ROVER (Recognizer Output Voting Error Reduction) est étudié afin d'améliorer la reconnaissance finale du texte en combinant, par un algorithme de vote appliqué à chaque caractère, les multiples chanes de caractères reconnues au cours du temps.

Le système complet a été évalué avec succès sur différentes bases de données d'images fixes ainsi que de séquences vidéos acquises dans le cadre des deux projets européens ASSAVID et CIMWOS.

Table des matières

Table des figures

Liste des tableaux

Acknowledgements

This thesis would not have been possible without my wife's help to me over these many years in Switzerland. Her and my family's love and support for me from afar has strengthened and encouraged me and has reminded me of how valuable they are to me.

My friends at IDIAP have shown much kindness and patience towards me. Their friendship have been a source of encouragement and refreshment for me. This is the wealthy that is written in my heart instead of being written in this thesis.

Thanks are given to my supervisor Hervé Bourlard for his support and encouragement on my work. Thanks are given to Dr. Jean-Marc Odobez. As one of my supervisor in IDIAP, he has had a substantial impact on my research and has offered excellent collaborations on my work. Thanks are also given to Dr. Jean-Philippe Thrian, my supervisor in EPFL, for his supervision and collaborations on this thesis. This work also benefited directly from discussions with Jürgen Lütin, Samy Bengio, Ronan Collobert and Andrew Morris. This work has also benefited from discussions with the research community at conferences, from reviews of papers written as part of this work. Florent Monay provided his French translation services.

Finally, This work was carried out in the framework of the Swiss National Center of Competence in Research (NCCR) on Interactive Multimodal Information Management (IM)2. The NCCR are managed by the Swiss National Science Foundation on behalf of the Federal Authorities. This work was also possible through financial support from the Swiss National Science Foundation under the grant SNSF 2100-057231.99/1 ; from European project ASSAVID[3] : Automatic Segmentation and Semantic Annotation of Sports Videos, 5th Framework Programme, Information Society Technology, supported by OFES ; from the European project CIMWOS[4] : Combined IMages and WOrd Spotting , 5th Framework Programme, Information Society Technology, supported by OFES.

[3]Web site : http ://www.bpe-rnd.co.uk/assavid/
[4]Web site : http ://www.xanthi.ilsp.gr/cimwos/

Chapitre 1

Introduction

Text detection and recognition in images and videos is a research area which attempts to develop a computer system with the ability to automatically read from images and videos the text content visually embedded in complex backgrounds. As an example of the general object recognition issues, this computer system, shown in figure 1.1, should answer two typical questions “Where & What” : “where is a text string ?” and “what does the text string say ?” in an image or a video. In other words, using such a system, text embedded in complex backgrounds can be automatically detected and each character or word can be recognized.

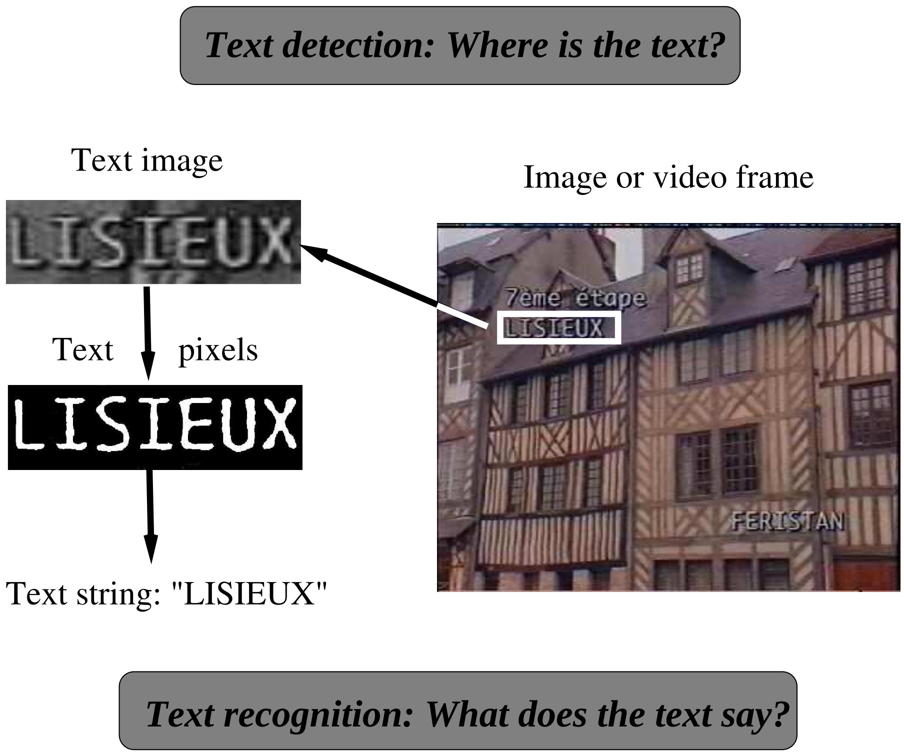


FIG. 1.1 – Text detection and recognition in images and videos

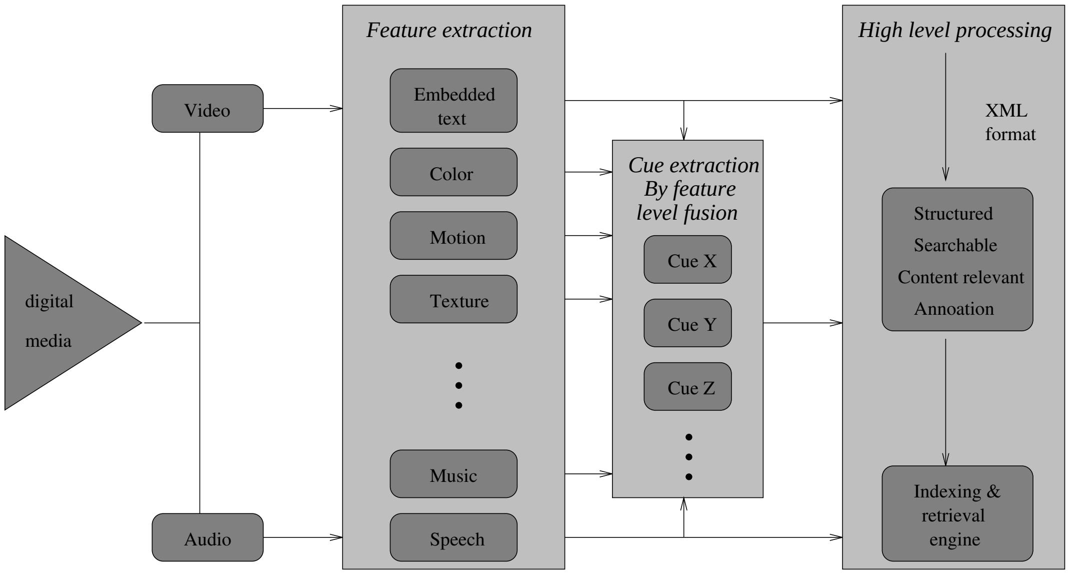


FIG. 1.2 – A simple model of video indexing and annotation system.

1.1 Goals

The investigation of text detection and recognition in complex background is motivated by cutting edge applications of digital multimedia. Today more and more audio and visual information is captured, stored, delivered and managed in digital forms. The wide usage of digital media files provokes many new challenges in mobile information acquisition and large multimedia database management. Among the most prominent are :

1. Automatic broadcast annotation : creates a structured, searchable view of archives of the broadcast content ;
2. Digital media asset management : archives digital media files for efficient media management ;
3. Video editing and cataloging : catalogs video databases on basis of content relevance ;
4. Library digitizing : digitizes cover of journals, magazines and various videos using advanced image and video OCR ;
5. Mobile visual sign translation : extracts and translates visual signs or foreign languages for tourist usage, for example, a handhold translator that recognizes and translates Asia signs into English or French.

The fundamental techniques and the scheme addressing these challenges, content-based multimedia retrieval and annotation, are illustrated in figure 1.2. Due to the fact that the CPU cost of image and video processing is very high, most of the applications rely on off-line content annotation. The annotation should be structured so that it can be easily processed by computers. It is also important that the annotation is content relevant and searchable, for example in textual format.

To obtain such an annotation automatically or interactively, content-based multimedia database indexing and retrieval tasks require extraction of descriptive features which are relevant to the subject materials (images, video, etc.). Features, such as color, texture, shape, motion, and layout, can be extracted at the low vision processing level. Many kinds of color features have been used as atomic units for image and video retrieval, for example the dominant color [53], regional or global histogram [53, 57, 100], average color vectors [99], and eigen image [79]. Texture features, such as edge-orientation histogram [53], Markov random field parameters [99, 79], wavelet or Fourier transform coefficients [57], and local binary pattern [100], are also widely used for indexing natural images. Shape features are employed mostly in map or object retrieval tasks. The typical shape features are, for instance, region overlap [67], Hough transform parameters [53], elementary descriptors [99, 53], bounding box [82], curvature scale space measures [63], elastic model parameters [79, 82], Fourier descriptors [57], and edge angles [31]. Motion features at the low level include motion magnitude and derivatives [124]. Layout features can be the spatial information of features in images [53, 16, 46, 100, 4, 73, 107, 57, 82, 103] or the shots of videos [124].

Low level features are easily extracted, but can not give a clear idea about what is present in the image. In order to obtain more descriptive features, low level features are usually fused into more meaningful entities and events, for example text [9], human face [4, 107, 79, 102], cars [98], indoors, outdoors sea, sky and beach etc. [1]. Both detection and recognition of high level entities have attracted more and more research interests recently. For instance, various methods for detecting and recognizing faces have been reported over the years [123, 78, 7, 91, 26, 42, 113, 52] and yield quite reasonable results. However, the results [95] of face recognition algorithms show that it works mostly in constrained environments. The detection and recognition of other objects, for example cars [98], are still great research challenges.

Text embedded in images and video sequences, especially captions, provide brief and important content information, such as the name of a player or speaker, the title, location and date of an event etc. On one hand, they carry clear meanings and can be powerful entities in media content annotation. On the other hand, text can also be combined with motion or other low level features to exploit more useful events. In this thesis, we investigate the extraction and recognition of text embedded in images and videos. There are two kinds of text in images or videos, scene text and superimposed text. Scene text are those text strings that written on some objects in the images or video frames. They usually have big sizes but can have occlusions and various alignments and movements. The superimposed text are the captions in the videos. They do not have occlusions but can be small sizes. The superimposed text are usually very relevant to the associated images or video frames, which are mainly focused on in this thesis. Some scene text are also useful, for example the sign of the road or the text on a cover of a journal. The images contains these scene text are usually carefully captured with constrained text alignment. The extraction and recognition of these constrained scene text is also investigated in this thesis. Before presenting the state of the art techniques in this domain, we describe and analyze two closely related issues : optical character recognition (OCR) and text-based retrieval.

1.1.1 Optical character recognition

Optical character recognition (OCR) addresses the problem of reading optically processed characters and has become one of the most successful applications of technology in the field of pattern recognition and artificial intelligence.

[1]See the project TRECVID : TREC Video Retrieval Evaluation funded by NIST, http ://www-nlpir.nist.gov/projects/trecvid/

History

The start of OCR was motivated by the requirement of high speed and electronic data processing. The first OCR equipment installed at the Reader's Digest in 1954 was employed to convert typewritten reports into computer readable punched cards. In the early 1960's, commercial OCR systems were developed for recognizing a small mount of letters and symbols specially designed for machine reading. Several years later, IBM exhibited its IBM 1287 system which was able to recognize regular machine printed characters. Toshiba also developed a postal code numbers recognizer that had hand-printed character processing capability. In the middle of the 1970's, some prototype OCR systems were able to process large printed and hand-written character sets in poor quality documents. OCR systems became available as software packages for home usage after 1986 since the prices of related hardware were getting cheaper.

Methods

A typical way of building an OCR system consists of first training a machine in order to obtain descriptions of what the characters look like, and then, to compare unknown characters to the previously learned descriptions to obtain the best match. In a typical commercial OCR system, the training process has been done in advance. The matching procedure usually consists of a pre-processing step, a recognition step and an optional post-processing step [65].

In the pre-processing step, the system digitizes a paper-made document into a grayscale image using an optical scanner and converts the grayscale image into a binary image. Furthermore, the regions containing text are located and each character is segmented from the word. Then, a smoothing and normalization processing is applied for eliminating noise and variation of size, slant and rotation before performing the recognition.

The recognition methods consist of feature extraction and classification. Feature extraction aims at capturing the essential characteristics of the characters. Most feature spaces are empirically designed on the basis of experiments and domain knowledge. The fundamental idea is to make the features invariant to distortions, fonts, and global deformations. Many different techniques have been presented and each of them has its own merits and drawbacks. Some good surveys summarize most of these features and methods in [112, 12, 65, 24, 69, 32, 58]. There are also non-feature extraction techniques, such as template-matching and correlation, which are sensitive to noise and fonts and were only used in the early age of OCR development. The objective of classification is to compute the probabilities or scores that an unknown character belongs to each character class and label it as the one class which has the highest probability or score. Common distance classifiers, such as k-nearest neighbor, quadratic Bayesian classifier and artificial neural networks are often used to fulfill this task.

In the post-processing step, the identified characters are grouped together to reconstruct the original words or numbers. Some techniques based on lexicon [13, 47], Markov chain and n-gram language models [45] are usually used here to detect and correct character recognition errors in words or even sentences.

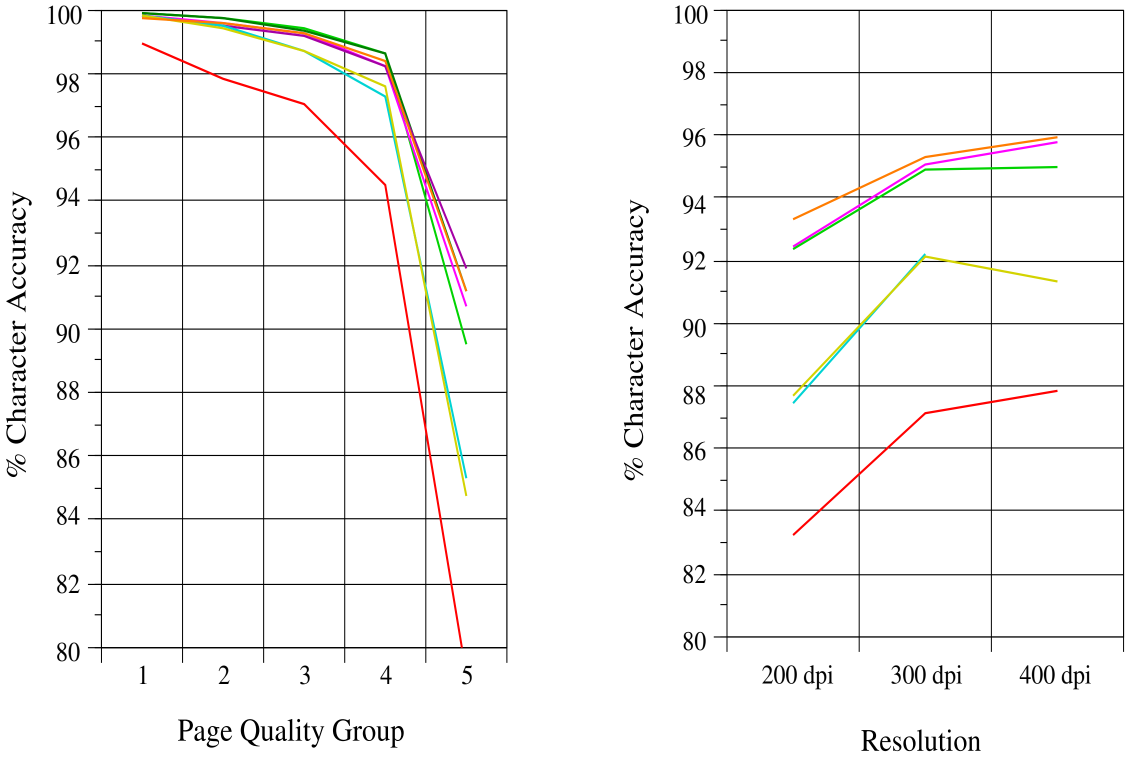


FIG. 1.3 – Typical commercial OCR software performance in function of page quality (left) and resolution (right) [86].

Capabilities and performance

More and more OCR systems now claim to be omnifont, which indicates their capability of recognizing most non-stylized fonts used by modern typesetters. Commercial OCR packages achieve a recognition rate ranging from 95% to 99.99% on common paper-printed documents. Intelligent character recognition (ICR) systems for recognizing constrained handwritten characters are also commercially available. However, these systems require characters to be written in a similar way of printed characters, referred to as standard hand-printed characters. The recognition of unconstrained handwritten characters, namely connected or cursive scripts, is still an area of research.

The performance of OCR systems closely rely on the quality of the targeting documents. Figure 1.3 illustrates the performance of some typical commercial OCR software in function of page quality and resolution. These statistical data come from the ISRI 1995 annual test of OCR accuracy [86]. The different curves in the figure represent the performance (in terms of word/character recognition rate) of different software. The page quality groups in the left figure, ranging from 1 to 5, represent the quality of the pages from high to low. Here the page quality only measures white papers, in which there are almost no background or just dots noise. We can see that the performance of OCR software drop abruptly because of low quality of page or poor resolution. This drawback of current OCR software raises an important question in the context of content-based indexing and retrieval. The question is : what kind of performance can lead to a good retrieval ?

1.1.2 Text-based retrieval

The basic level of multimedia retrieval on the basis of textual information is keyword searching. Numerous methods have been proposed in solving documents indexing and retrieval tasks based on only text content with noisy data, for example the output of an OCR system. These methods can improve the retrieval performance on top of simple word matching using fuzzy logic [56], confusion information for characters and a bi-gram model [72], finite state machine [28], similarity distance measure [108], and OCR error modeling [21].

In a survey of 1998, Doermann [22] illustrated that there is a direct correlation between OCR character accuracy and retrieval performance metrics. For 80-95% accuracy, enhanced retrieval techniques work well and above 95%, most retrieval techniques are completely unaffected by errors.

The speed of the text-based searching algorithms are very fast so that they have been successfully applied in many applications, such as the famous searching engines www.yahoo.com and www.google.com. One of the main reasons of this success is the efficiency of text-based techniques. On the contrary of text, the robustness and computation cost of the matching algorithms based on other high level features, for example the recognition of human face, car, beach etc., are not very efficient to be applied at a large scale (on a large database).

At the higher level, more semantic concepts, such as the topic of the documents, can be mined from the text contents for more advanced document retrieval. This research area is called text mining, which is not concerned and discussed in detail in this thesis.

1.1.3 Filling the gap between image and video documents and OCR system

An image and video text detection and recognition system aims at extending the capability of OCR and text-based retrieval technologies for accessing images and videos. However, text characters contained in images and videos can be :

- of any grayscale value (not always white or black)
- of low resolution ;
- of variable size ;
- embedded in complex backgrounds.

To have a knowledge on the OCR performance on images that contain text characters, experiments have been done based on a commercial OCR system RTK from Expervision, which is the OCR software used in the rest of the thesis and whose performance is comparable to the best two systems shown in Fig. 1.3. By simply feeding the four images shown in figure 1.4 as inputs to the OCR, we found that none of the characters was recognized. The OCR software refused to interpret these images or was not able to localize and segment characters in these images. This means that applying conventional OCR technology directly leads to very poor recognition rates. Therefore, efficient detection and recognition of text characters from background is necessary to fill the gap between image and video documents and the input of a standard OCR system.

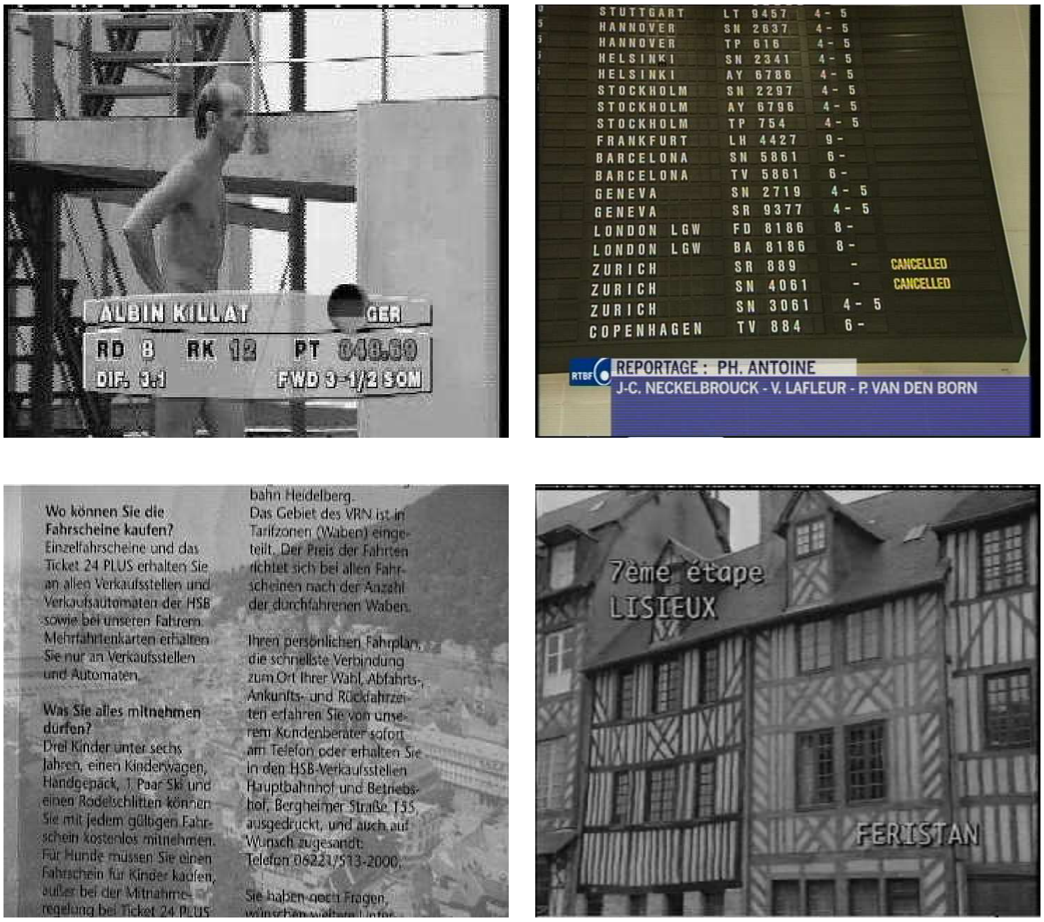


FIG. 1.4 – Examples of text strings contained in image and video frames

1.2 State-of-the-art of the text detection and recognition in images and videos

In this section, previous text detection and recognition systems described in the literature are discussed. To have a clear overview of these methods, we will only focus on describing the frameworks and the main procedures of the methods. Some technical details of these methods are also described in Chapter 2.

1.2.1 Text detection

Previous text detection methods in complex background can be classified into bottom-up [54] [106] [125], heuristic top-down methods [121] [29] [125] and machine learning based top-down methods [50, 55].

Bottom-up methods

Bottom-up methods do not really detect where the text is. The methods directly segment images into regions and then group "character" regions into words. Lienhart [54] segmented an image into regions using a split-merge [48] algorithm or a region grow [77] algorithm while Bunke [106] clustered text pixels from images using a color clustering algorithm. Geometric constraints, such as the size of the region, height and width ratio, are employed to select "character" regions. Candidate character regions are then grouped into words or even sentences according to their alignments. Although these methods can somehow avoid explicit text detection, they are very sensitive to character size, noise and background patterns.

Heuristic top-down methods

Heuristic top-down algorithms first detect text blocks in images using heuristic filters, then segment them into text and background regions. In other words, the first part of the algorithms aims at detecting text regions in images and the second part can be regarded as applying a bottom-up method on a local image. This enables the algorithms to process complex images but difficulties are encountered in both the detection and segmentation stages.

Characters can be detected by exploiting the characteristics on vertical edge, texture and edge orientations. One system for localizing text in covers of Journals or CDs [125] assumed that text were contained in regions with high horizontal variance, and satisfied certain spatial properties. The spatial properties could be exploited in a connected component analysis process. Smith et al. [104] localized text by first detecting vertical edges with a predefined template, then grouping vertical edges into text regions using a smoothing process. These two methods are fast but produce many false alarms because many background regions may also have strong horizontal contrast.

Wu et al. [121] described a text localization method based on texture segmentation. Texture features were 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 the feature space indicated text while the two others indicated non-text and uncertainty. The heavy duty of texture segmentation is the main drawback of this method and the segmentation quality is sensitive to background noises and image priors.

In a more recent work, Garcia et al. [29] proposed a feature, called variance of edge orientation, for text localization which exploited the fact that text string contains edges in different orientations. The variation of edge orientations was computed in a local area from image gradients, and combined with the edge features for locating text blocks. However, the method does not take care of characteristics coming from parallel character edges.

Besides the properties of individual character, Sobettka et al. [106] suggested that baseline detection could be used for text string localization. More precisely, text strings are characterized by specific top and bottom baselines, which thus should be detected in order to assess the presence of a text string in an image block.

Machine learning based top-down methods

Most of the heuristic top-down methods employ manually designed heuristic features and 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 [50, 55]. These systems extracted wavelet [50] or derivative features [55] 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.

1.2.2 Text recognition

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 to finally recognize characters. 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 in Section 1.1.1, and significantly depends on the segmentation quality, in the sense that errors made in the segmentation are directly forwarded to the OCR. To extend the recognition capability of the OCR for image and video text, the main research efforts focus on text segmentation and enhancement.

Text segmentation

Text segmentation methods are performed on the extracted text regions to remove the background surrounding text characters. These methods usually assume that the grayscale distribution is bimodal and that characters a priori correspond to either the white part or the black part. Great efforts are thus devoted to performing better binarization, combining global and local thresholding [43], M-estimation [36], or simple smoothing [121]. To eliminate the non-character regions in each binary image, a simple connected component analysis step is employed by setting constraints on size, height and width ratio and so on. However, these methods are unable to filter out background regions with similar grayscale values to the characters.

Text enhancement

If the character grayscale value is known, text enhancement methods can help the binarization process. In [93], a method for enhancing text in images exploits the characteristic that text characters consist of many stripe structures. Four directional filters (horizontal, vertical, diagonals) are used for blurring non-stripe structures in an input image while keeping the stripe structures in each direction. However, without proper estimation of the character scale, the designed filters can not enhance character strokes with different thickness [11].

In videos, multi-frame enhancement can also reduce the influence of background regions. The enhancement is performed on text images, which are blocks of image containing the same text string detected and tracked in consecutive video frames. In [55, 93], the maximum or minimum value at each pixel position is computed for enhancing characters that are known to have either the darkest or lightest grayscale values in video. In [50], an average image is computed from the detected and tracked text images for reducing the variations of the background, which can enhance the contrast of characters of any grayscale values. These methods assume that text and background have different movements and the grayscale of text characters is constant through consecutive frames.

1.3 Remaining problems

The previous techniques still have many unsolved problems in building an efficient image and video text detection and recognition system. These problems lead to unsatisfied performance of both text detection and recognition in real applications.

1.3.1 Problems in text detection

For text detection, heuristic top-down methods are empirical and most of the time rely on manually designed visual features. Since it is difficult to model complex backgrounds in images, these empirical methods usually detect too many false alarms. Typical ratio of text false alarm yielded by these methods is 80%-500% or more depending on the amount of text strings in the target images or videos. The machine learning methods are considered as improvements of the heuristic top-down methods by obtaining text detector through a set of training examples. However, there are two problems in building an efficient and robust text detection system using machine learning tools.

<u>Problem 1</u> : How to avoid performing a computational intensive classification on the whole image ? The computation costs of machine learning tools are associated with their capacities. For example, the computation cost of an artificial neural network is decided by the number of hidden units. Due to the huge mount of various backgrounds, training an artificial neural network for distinguishing general backgrounds and text will generate a complex model. Furthermore,

performing artificial neural network classification everywhere in the whole image is expensive when the images are large.

Problem 2 : What is the way to reduce the variances of character size and grayscale values in the feature space before training ? The wide range of variations of character sizes and grayscale values in images and videos is one of the obstacles in training a good model for text detection. Technologies of feature extraction, normalization and the investigation of different machine learning tools are expected to improve the performance of the text detection system.

1.3.2 Problems in text recognition

For text recognition, 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 values of text may not be known in advance or even inconstant in one text string (see figure 1.4). To well recognize text strings extracted from images or video frames, the following problems need to be investigated.

Problem 3 : How to enhance text characters in various sizes ? The text enhancement is one of the methods to remove backgrounds from text images by exploiting the characteristics of the substructures of characters. However, fixed-size filters are not able to enhance text characters of any sizes. Therefore, an enhancing technique for various sizes of characters is desired to have a better text binarization.

Problem 4 : How to segment and recognize text characters of unknown grayscale values ? Previous works in literature assume the text is either white or black in images or videos. This is not always true. As we showed in figure 1.4, characters can be any grayscale values and have even different grayscale values in one string or word. Moreover, the distribution of the grayscale values in a text image (extracted text region) is not always bi-modal. Some text images, especially from videos, have mainly three or more clusters of grayscale values. Applying text recognition system in real images or videos requires the capability of recognizing characters of any grayscale values.

Problem 5 : How to integrate temporal information contained in consecutive video frames to improve text recognition ? Former methods for using temporal information focus on removing background structures in frames, and can only be applied on black or white characters. The grayscale of characters is constrained to be constant through frames and an accurate alignment of text images is required. The problem of temporal information integration of text characters at both the image level and the recognized text string level need to be addressed.

1.4 Contributions of this dissertation

This thesis investigates methods for building an efficient application system for detecting and recognizing text of any grayscale values embedded in images and videos. Contributions of this dissertation are listed below and address the problems discussed in the last section.

First, as one of the major goals of this thesis, a localization/verification scheme of text detection is proposed that avoids performing intensive machine learning based classification on obvious non-text regions. This text detection scheme first employs a fast heuristic algorithm for locating text-like region in images with a very low rejection rate and then apply machine learning approaches to verify the true text images.

Second, inside the verification step of the proposed scheme, contrast independent features are proposed for training classifiers. These features together with a size normalization process reduce the variance of character sizes and grayscale values. Two kinds of machine learning approaches, namely multi-layer perceptrons and support vector machines, are compared based on various features.

Third, in the conventional bi-modal text recognition scheme, we proposed a set of Gabor filters for estimating the scales of substructures of text characters. The method enables the text enhancement at various size and improve the binarization of text images.

Fourth, a multi-hypotheses framework is presented for addressing text recognition, which can significantly improve the recognition performance. As the second major purposes of the thesis, multiple segmentation hypotheses are generated and processed by an OCR software. Under this framework, a Markov Random Field based segmentation method and a post-processing algorithm for removing non-character regions are proposed and evaluated. A selection algorithm based on language modeling and OCR statistics is also presented to select the text result from all the produced text strings.

Finally, a sequential Monte Carlo segmentation method is proposed for integrating temporal information of video text at the grayscale image level. Furthermore, we also investigate the method of integration of text in multiple frames at the level of OCR outputs using a voting technique.

The rest of the thesis is organized into 5 chapters. In Chapter 2, background knowledge of visual feature extraction, machine learning tools and statistical methods that will be used in the thesis are discussed. In Chapter 3, an overview of the text detection and recognition system is presented. In Chapter 4, the proposed text detection schemes are introduced and evaluated in details. In Chapter 5, the framework and methods related with text recognition are presented. The bi-modal text enhancement method and multi-modal methods are discussed and evaluated in detail. The Chapter 6 presents the text recognition methods in video by exploiting temporal information in multiple frames. The Chapter 7 consists of a concluding summary of the achievements and limitations of the text detection and recognition system and proposes some future research directions.

Chapitre 2

Background

In this chapter, we introduce image processing algorithms that are essential to image analysis in general and to image text detection in particular, as well as the mathematical background related to machine learning and stochastic modeling. Some algorithms that were used in previous work or are used in the next chapters of this thesis are presented in detail.

Firstly, visual feature detection techniques, such as edge detection and texture analysis, will be formally presented in this chapter. Both edge detection and texture analysis are fundamental topics in computer vision and image processing. We will give the general ideas about these topics and then highlight the techniques that are more closely related to text localization.

Secondly, as this thesis addresses the text detection problem based on machine learning principles, two approaches, namely multi-layer perceptrons and support vector machine that are employed for verifying text strings, are discussed in detail in the second section.

Finally, the mathematical background of two stochastic modeling methods, the gaussian mixture models and Markov random fields, are introduced in the third section. These two methods will be used for segmenting and recognizing text characters in Chapter 5.

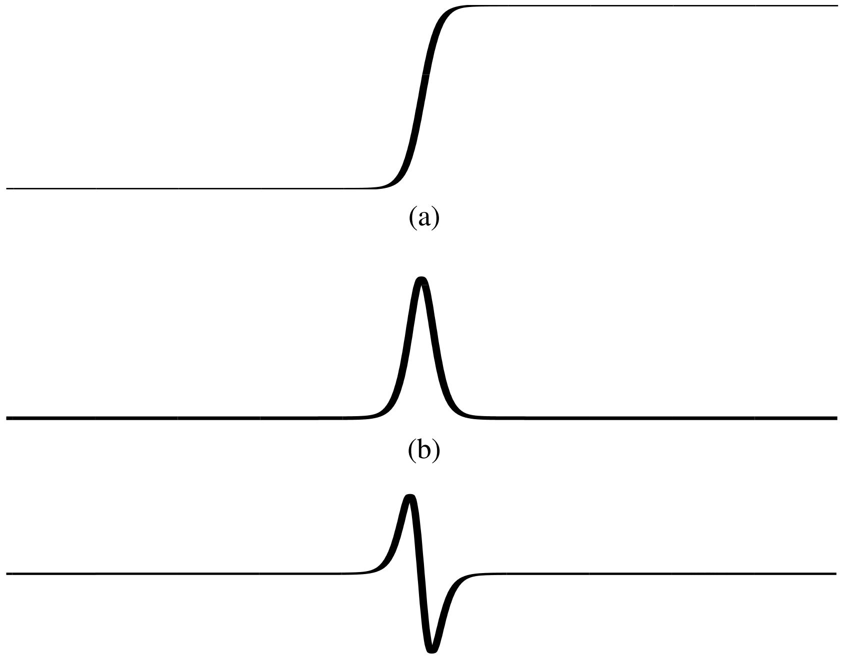
2.1 Visual feature extraction

2.1.1 Edge detection

Edges are the most important visual features that are used in text detection researches. An edge, as defined in [39], is a sharp discontinuity in a gray-level profile. Due to the presence of noise and the ambiguity of “sharp”, edge detection is a complex task. In 2D images, an edge is specified by its magnitude and its direction. Edge detection is often carried out by spatial derivation and thresholding. Figure 2.1 shows an edge and its first and second order derivatives in a 1D case.

First order derivative

The first order derivatives of a given image function I, referred to as gradient, are defined as the vector :



(a)

(b)

(c)

FIG. 2.1 – An edge (a), its first order derivative (b) and its second order derivative (c)

$$\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right) \tag{2.1}$$

Its magnitude is defined as :

$$|\nabla I| = \left[\left(\frac{\partial I}{\partial x} \right)^2 + \left(\frac{\partial I}{\partial y} \right)^2 \right]^{\frac{1}{2}} \tag{2.2}$$

and its direction θ is defined as :

$$\theta = \arctan \left(\frac{\frac{\partial I}{\partial x}}{\frac{\partial I}{\partial y}} \right) \tag{2.3}$$

For a digital image, the gradient can be approximated using a difference operation. Many of them have been proposed in the literature [87, 84, 105, 38, 18, 3, 83, 33, 89]. For example, the Sobel operators that will be used in our text localization algorithm are given by :

$$\frac{\partial I}{\partial x} = I \star \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{2.4}$$

$$\frac{\partial I}{\partial y} = I \star \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{2.5}$$

The Sobel operator with respect to the x direction consists of summing intensity values at the neighbours of each pixel (x, y) using the weights of with the operator :

$$\frac{\partial I(x,y)}{\partial x} = -I(x-1,y-1)+0+I(x+1,y-1) \tag{2.6}$$

$$-2I(x-1,y)+0+2I(x+1,y) \tag{2.7}$$

$$-I(x-1,y+1)+0+I(x+1,y+1) \tag{2.8}$$

$$\tag{2.9}$$

Edges are given by the local maxima of the gradient magnitude image (see figure 2.1b). Due to the presence of noise, the local maxima are often selected by using a thresholding procedure.

Second order derivative

The second order derivatives of a given image I are defined as :

$$\nabla^2 I = \left(\frac{\partial^2 I}{\partial^2 x}, \frac{\partial^2 I}{\partial x \partial y}, \frac{\partial^2 I}{\partial^2 y}\right) \tag{2.10}$$

Potential edges detected by using the second order derivative are zero-crossings, where the derivative values change signs as illustrated in the middle of the figure 2.1c. Second order derivatives are noisier then the first order derivative. However, one advantage of using zero-crossing for detecting edges is that it is easier to track the zero-crossings for obtaining close contours than to track the maximum gradient points with a threshold. The isotropic zero-crossing based edge operator is a direction-invariant Laplacian operator. To achieve stable edge detection in the presence of noise, Marr and Hildreth [59] suggest smoothing an image with Gaussian smoothers before applying the Laplacian operator. The resulting operator is referred to as the Laplacian of Gaussian (LOG) operator, which can also be implemented by difference of Gaussian (DOG).

A high-performance edge operator has been obtained by differentiation in a direction across an edge. Canny [6] suggests the use of both the first and the second order derivatives for edge detection. Zero-crossings of the second derivative are detected along a line in the gradient direction, which should also have high gradient magnitude values. In his algorithm, the second order directional derivatives are implemented as :

$$\frac{\partial^2 I}{\partial^2 x} = I \star \begin{bmatrix} 1 & -2 & 1 \\ 2 & -4 & 2 \\ 1 & -2 & 1 \end{bmatrix} \tag{2.11}$$

$$\frac{\partial^2 I}{\partial^2 y} = I \star \begin{bmatrix} 1 & 2 & 1 \\ -2 & -4 & -2 \\ 1 & 2 & 1 \end{bmatrix} \tag{2.12}$$

$$\frac{\partial^2 I}{\partial x \partial y} = I \star \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \tag{2.13}$$

Various implementations of the Canny algorithm differ in the details of the computation of the gradient and its direction.

Clearly visible text characters in images always contains strong edges. Therefore, the extraction of edges is regarded as an essential features in text detection task. We will come back to this in Chapter 4.

2.1.2 Texture feature extraction

Texture is also an important visual feature although it is difficult to find a clear definition for it. Usually, the term texture indicates some characteristics of a region in an image. Under the statistical framework, texture features can be characterized by histogram, grayscale co-occurrence [34, 127], mean and variance, moments [68], spectral density and autocorrelation [128], edgeness [37], mathematical morphological operators [117, 62], parameters of Markov random fields [111, 60] and mosaic models [96, 109]. There are three types of texture features employed in the context of text detection : gaussian space features, discrete cosine transform coefficients and wavelet features.

Gaussian space features

Wu [122] proposed a texture feature for detecting text in images. The feature is extracted by computing the second order derivatives of a given image smoothed by Gaussian functions. Formally, given an image I, nine feature images

$$(F_x^1, F_x^{\sqrt{2}}, F_x^2, F_y^1, F_y^{\sqrt{2}}, F_y^2, F_{xy}^1, F_{xy}^{\sqrt{2}}, F_{xy}^2)$$

are computed as :

$$\begin{aligned} F_x^i &= \tfrac{\partial^2 I \star G_i}{\partial^2 x} \\ F_y^i &= \tfrac{\partial^2 I \star G_i}{\partial^2 y} \\ F_{xy}^i &= \tfrac{\partial^2 I \star G_i}{\partial x \partial y} \end{aligned} \tag{2.14}$$

where the gaussian functions $G_i = \mathcal{N}(0, i)$. The feature vectors are extracted at each pixel site. Each feature vector characterizes the texture around a pixel in three Gaussian spaces specified by $i = 1, \sqrt{2}, 2$, i being the standard variation of the gaussian in terms of the number of pixels.

To extract more texture information of text images, his feature can also be extended into higher gaussian spaces. However, the computation cost of the feature extraction algorithm can become very high.

Discrete cosine transform coefficients

Another texture feature that is often used for text detection are based on discrete cosine transform (DCT) coefficients [126]. The DCT is a loss-less and reversible mathematical transformation that converts a spatial amplitude representation of data into a spatial frequency representation. One of the advantages of the DCT is its energy compacting property, that is the signal energy is concentrated on a few components while most other components are zero or are negligible. DCT is used in many applications such as filtering, trans-multiplexers, speech coding, image coding, pattern recognition and image enhancement. Formally, the 2D-DCT can be described as :

$$DCT(u,v) = \frac{2}{\sqrt{MN}}\alpha(u)\alpha(v)\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} I(x,y)\cos\left(\frac{(2x+1)u\pi}{2M}\right)\cos\left(\frac{(2y+1)v\pi}{2N}\right) \tag{2.15}$$

where M and N are the width and the height of the image and

$$\alpha(u) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if}\, u = 0 \\ 1 & \text{else} \end{cases}$$

There are three reasons why the DCT coefficients were used to characterize texture of text images. First, DCT coefficients have been proved to be very efficient in presenting visual information in the frequency domain . Second, a fast algorithm for calculating DCT is available. Finally, most of the images and videos are compressed using DCT. This may enable the fast text detection without completely decompressing the images and videos.

Wavelet feature

Li et. al. [50] presented a texture feature using Haar wavelet decomposition. Haar wavelets were chosen since they are adequate for local detection of line segments, which are supposed to be able to characterize text regions. Given an image I, the Haar wavelets decompose a given image I into four sub-images : lower frequency image (L), vertical high frequency image (V), horizontal high frequency image (H), and diagonal high frequency image (D). Figure 2.2 illustrates a Haar decomposition of an image.

$$L(x,y) = \frac{1}{4}\left(I(2x,2y) + I(2x,2y+1) + I(2x+1,2y) + I(2x+1,2y+1)\right) \tag{2.16}$$

$$V(x,y) = \frac{1}{4}\left(I(2x,2y) - I(2x,2y+1) + I(2x+1,2y) - I(2x+1,2y+1)\right) \tag{2.17}$$

$$H(x,y) = \frac{1}{4}\left(I(2x,2y) + I(2x,2y+1) - I(2x+1,2y) - I(2x+1,2y+1)\right) \tag{2.18}$$

$$D(x,y) = \frac{1}{4}\left(I(2x,2y) - I(2x,2y+1) - I(2x+1,2y) + I(2x+1,2y+1)\right) \tag{2.19}$$

FIG. 2.2 – An image and its Haar decomposition

The decomposition procedure can be applied iteratively until each sub-image contains only one pixel. In paper [50], this decomposition is applied not on the whole image but on 16×16 blocks and therefore the maximum decomposition level is 4.

Three features are then extracted in each $N \times N$ sub-image (sub-block), for example H : the mean $M(H)$, the second order central moment $M_2(H)$, and the third order central moment $M_3(H)$:

$$M(H) = \frac{1}{N^2} \sum_{x=1}^{N} \sum_{y=1}^{N} H(x,y) \tag{2.20}$$

$$M_2(H) = \frac{1}{N^2} \sum_{x=1}^{N} \sum_{y=1}^{N} (H(x,y) - M(H))^2 \tag{2.21}$$

$$M_3(H) = \frac{1}{N^2} \sum_{x=1}^{N} \sum_{y=1}^{N} (H(x,y) - M(H))^3 \tag{2.22}$$

The moments (M, M_2, M_3) are calculated in each of the (L, V, H, D) images. Some of these moments are then selected as features for classifying text and non-text blocks. This Haar wavelet decomposition has also a fast algorithm. Another advantage of the wavelet feature is that text characters in different scales can be detected at the different decomposition levels.

Conclusion

Both edge features and texture features are used to transform the original image space into feature spaces which contain more relevant information for fulfilling the text detection task. In text detection, text image pixels are classified into a text class and a non-text class. There are many methods that can be used for this classification task. They can be empirical, such as thresholding, or reasoning based on simple rules, or machine learnable. In the next section, we introduce two machine learning approaches that are investigated in our text detection system.

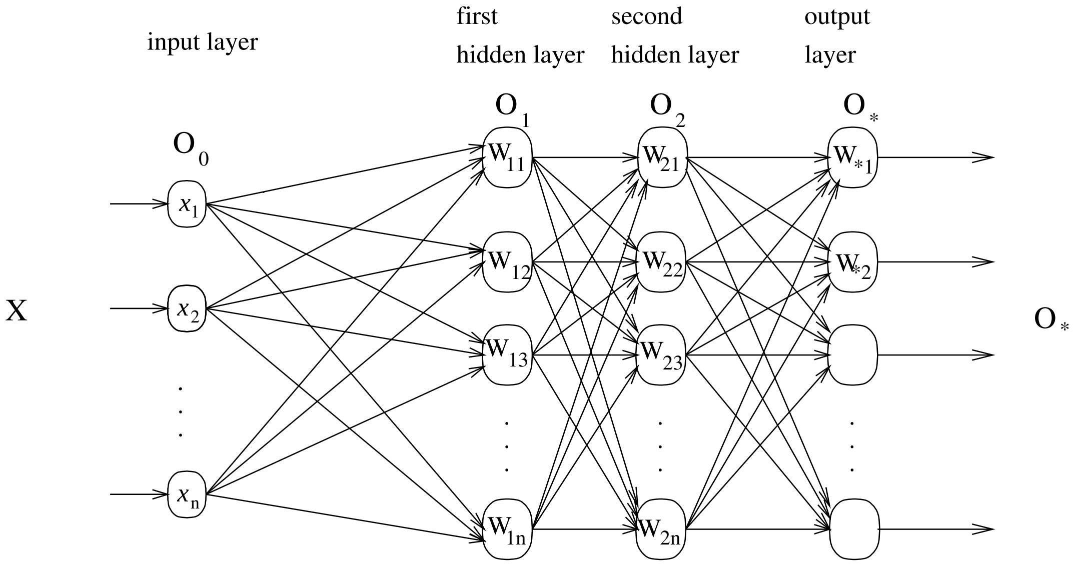


FIG. 2.3 – Multilayer perceptron network with two hidden layers

2.2 Machine learning approaches

The methodology of learning from experience has been more and more applied on problems where a proper mathematical modeling is impossible. For example, it is not known how to mathematically model the relationship between a block of pixels for distinguishing text and backgrounds. In most of generative statistical methods, such as the maximum likelihood ratio testing, large number of training examples are needed to obtain a good probabilities distribution estimation of target events. In comparison with these generative approaches, discriminative machine learning methods aim at find the boundaries between different target events and therefore potentially need less training data. Since there are only two targeting classes, text and non-text, the discriminative methods are usually more successful than the generative approaches. We therefor focus on discriminative machine learning methods in this section. Two supervised learning approaches, multilayer perceptrons and support vector machines, are now introduced. They will be employed in the text verification task described in Chapter 4.

2.2.1 Multilayer perceptrons

Perceptron is the first and simplest forms of feedforward network proposed by Rosenblatt in 1962 [88]. A perceptron is composed of an input layer and an output layer of neurons, in which an output node computes the weighted sum of input nodes. A multilayer perceptrons (MLP) model is a multilayer extension of the perceptron model, which consists of an input layer, an output layer, and one or more hidden layers of neurons, as shown in Figure 2.3.

The functions of the MLP consist of a classification operation and a training operation. In the classification operation, an input vector is presented in the input layer and each neuron computes a weighted sum of the outputs of the neurons in the previous layer, followed by an activity function until a set of outputs is finally obtained at the output layer. Since one neuron only has one output value at a time, we can simply write the output of a layer as a vector, for example $O_i = (o_{i1}, \ldots, o_{iN_i})$ indicates the output vector of the ith hidden layer with N_i neurons. Let us also denote the output of the input layer by O_0 and the output of the MLP by O_k if the MLP has $k-1$ hidden layers. A neuron in a hidden layer or the output layer has a weight vector corresponding to the output vector from the previous layer. Let us define the weight vector of the jth neuron in the ith hidden layer as W_{ij}, in the output layer as W_{kj} and in the input layer as an unit vector. Therefore, for each input vector $\mathbf{x} = (x_1,\ x_2,\ \ldots\ , x_{N_0})$, the output of the input layer is still the vector $O_0 = \mathbf{x}$. The output vector $O_i = (o_{i1}, \ldots, o_{iN_i})$of the ith hidden layer is computed as :

$$o_{ij}(\mathbf{x}) = f\left(W_{ij} \cdot O_{i-1}\right) \tag{2.23}$$

To avoid misunderstanding, we explicitly write the operator of a scalar product in this thesis. The output vector $O_k = (o_{k1}, \ldots, o_{kN_O})$ in the output layer is defined as :

$$o_{kj} = f(W_{kj} \cdot O_{k-1}) \tag{2.24}$$

where O_{k-1} is the output vector of the last hidden layer. The activity function f is selected as a sigmoid function, in this thesis :

$$f(x) = \frac{1}{1 + \exp(-x)} \tag{2.25}$$

The goal of the training operation is to obtain a set of optimal weights of an MLP. One of the popular way to implement this training process is the backpropagation (BP) algorithm. The earliest idea of the BP algorithm is described by Paul Werbos in 1974 [116]. Similar algorithms were also developed independently by LeCun [49] and Parker [76] around 1985 and became popular through the book Parallel Distributed Processing [92] written by Rumelhart, Hinton and Williams in 1986.

The backpropagation algorithm is a gradient descent procedure in the weight space of the MLP. Let us define one of the most common target function called cost function $F\left(W, T_{set}\right)$ on the basis of the weights of the MLP W, and a set of training data T_{set} which consists of N pairs of input vector $\mathbf{x}_i$ and desired output y_i, $T_{set} = \{(\mathbf{x}_i, y_i)\}_{i=1,\ldots,N}$:

$$F\left(W, T_{set}\right) = \frac{1}{2} \sum_{(\mathbf{x}_i, y_i) \in T_{set}} \|O_k(\mathbf{x}_i) - y_i\|^2 \tag{2.26}$$

The MLP training task consists of minimizing the cost function F with respect to the weight set W. Therefore, using a gradient descent procedure, W can be updated by computing the partial derivatives of the cost function :

$$W^{t+1} = W^t - \alpha \frac{\partial F\left(W^t, T_{set}\right)}{\partial W} \tag{2.27}$$

The detail expansion of the Eq. 2.27 can be given as the following. Let us first define the differential of the cost function with respect to the output of each neuron as :

$$\delta_{hj}^t = \frac{\partial F}{\partial o_{hj}^t} \tag{2.28}$$

The weight vector W_{hj} of the jth neuron in the h layer can be updated as :

$$W_{hj}^{t+1} = W_{hj}^{t} + \alpha\mu_{hj}^{t} \tag{2.29}$$

where α is the learning rate and the μ_{kj}^{t} can be expended as :

$$\begin{aligned}
\mu_{hj}^{t} &= \frac{\partial F}{\partial W_{hj}^{t}} && (2.30)\\
&= \frac{\partial F}{\partial o_{hj}^{t}}\frac{\partial o_{hj}^{t}}{\partial W_{hj}^{t}} && (2.31)\\
&= \delta_{hj}^{t}\frac{\partial o_{hj}^{t}}{\partial W_{hj}^{t}} && (2.32)\\
& && (2.33)
\end{aligned}$$

The last parameter can be computed as :

$$\frac{\partial o_{hj}^{t}}{\partial W_{hj}^{t}} = f'(W_{hj}^{t}O_{h-1}^{t})O_{h-1}^{t} \tag{2.34}$$

The first parameter δ_{hj}^{t} is extended as the following.

For a neuron in the output layer, we can compute the δ_{kj}^{t} as :

$$\begin{aligned}
\delta_{kj}^{t} &= \frac{\partial F}{\partial o_{kj}^{t}} && (2.35)\\
&= \sum_{(\mathbf{x}_i,y_i)\in T_{set}} \left(o_{kj}^{t}(\mathbf{x_i}) - y_i\right) && (2.36)\\
& && (2.37)
\end{aligned}$$

For a neuron that is not in the output layer, the δ_{hj}^{t} can be updated as :

$$\begin{aligned}
\delta_{hj}^{t} &= \frac{\partial F}{\partial o_{hj}^{t}} && (2.38)\\
&= \sum_{m=1}^{N_{h+1}} \frac{\partial F}{\partial o_{(h+1)m}^{t}}\frac{\partial o_{(h+1)m}^{t}}{\partial o_{hj}^{t}} && (2.39)\\
&= \sum_{m=1}^{N_{h+1}} \delta_{hj}^{t}\frac{\partial o_{(h+1)m}^{t}}{\partial o_{hj}^{t}} && (2.40)\\
& && (2.41)
\end{aligned}$$

where the last parameter can be extended as :

$$\frac{\partial o^t_{(h+1)m}}{\partial o^t_{hj}} = \frac{\partial f\left(W^t_{(h+1)m} O^t_h\right)}{\partial o^t_{hj}} \tag{2.42}$$

$$= \frac{\partial f\left(\sum_{l=1}^{N_h} w^t_{(h+1)ml} o^t_{hl}\right)}{\partial o^t_{hj}} \tag{2.43}$$

$$= f'(W^t_{(h+1)m} O^t_h) w^t_{(h+1)mj} \tag{2.44}$$

$$\tag{2.45}$$

Here, $f'(x) = \frac{\partial f(x)}{\partial x}$.

In practice, the training of the parameters W consists of two procedures : the forward procedure and the backward procedure. Firstly, the outputs $(O_{hj}(\mathbf{x}_i))$ of each neuron are computed in the forward procedure. Then, the δ_{hj} are computed from the output layer to the first hidden layer in the backward procedure. This training algorithm is therefore called the backpropogation algorithm.

One of the important theoretical results about MLP is that a configuration with only one hidden layer of neurons has been proven to be able to approximate any continuous function [15]. In practice, multiple perceptrons have been applied in many applications such as speech recognition [66], optical character recognition [49], face detection [114].

2.2.2 Support Vector Machine

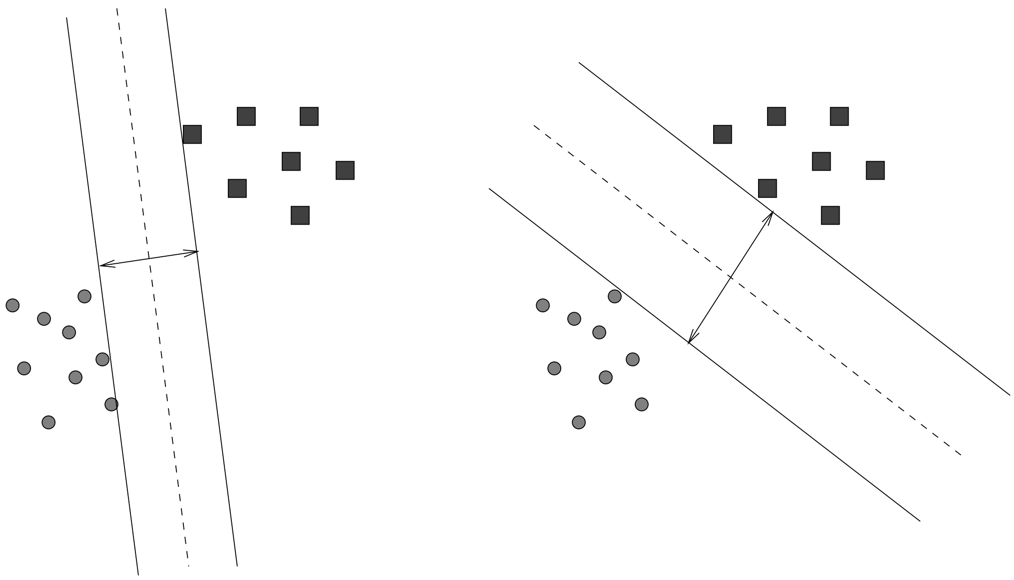
Support Vector Machine (SVM) is a technique motivated by statistical learning theory and has been successful applied to numerous classification tasks. The key idea is to transform the input data into high dimensional feature space and separate classes with a decision surface in this space. As opposite to the empirical risk minimization algorithm such as MLP, which minimizes the error over the data set, SVM is a structural risk minimization algorithm, which aims at minimizing a bound on the generalization error of a model in high dimensional space. Extensive discussions of SVMs can be found in [115, 5, 14].

In this thesis we will only use SVM for a binary classification task. Let us say that we have N labeled training examples : $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$, where $y_i = \pm 1$ indicating two different classes $i = 1, 2, \dots, N$.

In the linear separable case, there exists some hyperplanes $\mathbf{w} \cdot \mathbf{x} + b = 0$ (decision surfaces) that separate all the training examples :

$$\begin{aligned} \mathbf{x}_i \cdot \mathbf{w} + b &\geq +1 \text{ if } y_i = +1 \\ \mathbf{x}_i \cdot \mathbf{w} + b &\leq -1 \text{ if } y_i = -1 \end{aligned}$$

or equivalently :



(1) (2)

FIG. 2.4 – Solving classification problems with hyperplanes. (1) a hyperplane with small margin (2)a hyperplane with larger margin.

$$y_i\left(\mathbf{x}_i \cdot \mathbf{w} + b\right) \geq \pm 1 \;\forall i \tag{2.46}$$

where $\mathbf{w}$ is the normal to the hyperplane.

The decision function can therefore be given by

$$f(x) = sign(\mathbf{w} \cdot \mathbf{x} + b) \tag{2.47}$$

Notice that if the parameters $\mathbf{w}$ and b are scaled by the same quantity, the hyperplane given by 2.47 is unchanged. The following constraint is imposed to remove this redundancy and obtain a unique parameterization :

$$\min_{i=1,\ldots,m} |\mathbf{x}_i \cdot \mathbf{w} + b| = 1 \tag{2.48}$$

The margin of such a hyperplane $\mathbf{w} \cdot \mathbf{x} + b = 0$ is defined by the sum of the shortest distances from a hyperplane to the closest positive example and the closest negative example. Figure 2.4 illustrates separating hyperplanes with small and larger margins. According to the normalization 2.48, this margin is simply given by $\frac{2}{\|\mathbf{w}\|}$, where $\|\mathbf{w}\|$ is the Euclidean norm of $\mathbf{w}$. Therefore, the maximum margin can be obtained by minimizing $\|\mathbf{w}\|^2$ with respect to $\mathbf{w}$ and b, subject to the constraints Eq. 2.46.

Thus, this learning task can be reduced to the maximization of the Wolfe dual Lagrangian :

$$L(\mathbf{w}, b, \Lambda) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \lambda_i \left[y_i \left(\mathbf{w} \cdot \mathbf{x}_i + b\right) - 1\right] \tag{2.49}$$

where $\Lambda = (\lambda_1, \ldots, \lambda_m)$ is the vector of non-negative Lagrange multipliers corresponding to the constraints 2.46. Therefore, we have :

$$\frac{\partial L(\mathbf{w}, b, \Lambda)}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i = 0 \tag{2.50}$$

$$\frac{\partial L(\mathbf{w}, b, \Lambda)}{\partial b} = \sum_{i=1}^{N} \lambda_i y_i = 0 \tag{2.51}$$

The optimal $\mathbf{w}^*$ can be derived as :

$$\mathbf{w}^* = \sum_{i=1} m\lambda_i^* y_i \mathbf{x}_i \tag{2.52}$$

which shows that the optimal hyperplane can be written as a linear combination of the training vectors. Substituting 2.52 and 2.51 into the Lagrangian 2.49, we have :

$$W(\Lambda) = \sum_{i=1}^{m} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{m} \lambda_i \lambda_j y_i y_j \left(\mathbf{x}_i \cdot \mathbf{x}_j\right) \tag{2.53}$$

where the Lagrangian multipliers λ_i are subject to the constraints :

$$\begin{array}{rcl} \lambda_i & \geq & 0 \\ \sum_{i=1}^{m} \lambda_i y_i & = & 0 \end{array}$$

At this point, this problem can be solved using standard quadratic programming (QP). Notice that complementary slackness conditions of the form :

$$\lambda_i^* \left[y_i(\mathbf{w}^* \cdot \mathbf{x}_i + b^*) - 1\right] = 0 \quad i = 1, \ldots, m \tag{2.54}$$

shows that $\lambda_i^* > 0$ only when $y_i(\mathbf{w}^* \cdot \mathbf{x}_i + b^*) = 1$, where $\lambda^*, \mathbf{w}^*, b^*$ represent the optimal values. A training sample $\mathbf{x}_i$ is termed as support vector if the optimal λ_i^*. is positive. The b^* can be computed as :

$$b^* = y_i - \mathbf{w}^* \cdot \mathbf{x}_i \tag{2.55}$$

for any support vector $\mathbf{x}_i$. Substituting equation 2.52 into the decision function 2.47, we have :

$$f(\mathbf{x}) = sign\left(\sum_{i=1}^{m} y_i \lambda_i^* (\mathbf{x} \cdot \mathbf{x}_i) + b^*\right) \tag{2.56}$$

Soft margin : linear non-separable case

Considering that we still look for a linear separating surface, but that such a separating hyperplane does not exist, we introduce a soft margin hyperplane to solve the learning problem. The soft margin has a set of variables $\{\xi_i\}_{i=1}^m$ that are positive slack variables and measure penalties proportional to the amount of constraint violations. The learning task now involves the minimization of :

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\left(\sum_{i=1}^{N} \xi_i\right)^k$$

subject to the constraints :

$$\begin{array}{rcl} y_i(\mathbf{x}_i \cdot \mathbf{w} + b) & \geq & 1 - \xi_i \ \forall i \\ \xi_i & \geq & 0 \ \forall i \end{array}$$

where C and k are the parameters of the penalty to errors that have to be determined beforehand. Let us set $k = 1$ to simplify the discussion. This minimization can be solved using a similar Lagrangian technique discussed above. The training task is, in this case, to maximize

$$W(\lambda) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2}\sum_{i,j=1}^{N} \lambda_i \lambda_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \tag{2.57}$$

where the Lagrangian multipliers λ_i are subject to the constraints :

$$\begin{array}{rcl} \lambda_i & \geq & 0 \\ \lambda_i & \leq & C \\ \sum_{i=1}^{m} \lambda_i y_i & = & 0 \end{array}$$

Non-linear decision surfaces

This method can be easily generalized to the non-linear case. The more complex decision surfaces can be constructed by mapping the training examples $\mathbf{x}_i$ into an alternative higher dimensional space $\phi(\mathbf{x}_i)$, so called feature space, and by working with linear classification in that space. Noticing that training of SVM only depends on examples through inner products, this mapping can be implicitly given by choosing a kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$.

The learning task therefore is the maximization of the Lagrangian :

$$W(\lambda) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{N} \lambda_i \lambda_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{2.58}$$

subject to constraints

$$\begin{array}{rcl} \lambda_i & \geq & 0 \\ \lambda_i & \leq & C \\ \sum_{i=1}^{m} \lambda_i y_i & = & 0 \end{array}$$

$W(\lambda)$ can be solved using quadratic programming techniques. Once we have found the optimal λ_i^*, the classification of an unknown example $\mathbf{x}$ is based on the function :

$$f(\mathbf{x}) = sign\left(\sum_{i=1}^{N} \lambda_i^* y_i K(\mathbf{x}, \mathbf{x}_i) + b\right) \tag{2.59}$$

The kernel functions that commonly used are :

$$\begin{array}{rrcl} \text{Gaussian RBF :} & K(\mathbf{x}_i, \mathbf{x}_j) & = & \exp(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}) \\ \text{Polynomial of degree } d: & K(\mathbf{x}_i, \mathbf{x}_j) & = & (1 + \mathbf{x}_i \cdot \mathbf{x}_j)^d \\ \text{Multi-layer perceptron :} & K(\mathbf{x}_i, \mathbf{x}_j) & = & tanh(\mathbf{x}_i \cdot \mathbf{x}_j - \theta) \end{array} \tag{2.60}$$

In real applications, training a SVM using large data set ($N > 5000$) is a difficult problem due to the large number of parameters (about $m \times m$). Some solutions employing data or problem decomposition have been published in [74, 81, 41].

2.3 Statistical methods

To recognize text in images and videos, text pixels need to be segmented apart from background pixels even when the whole text string is well located. Two statistical approaches gaussian mixture models and Markov random field are employed to model the grayscale distributions of text pixels in text images. These two approaches are on the basis of the segmentation process and are introduced in detail in this section.

2.3.1 Gauss mixture models and the Expectation Maximization algorithm

Gaussian mixture models approximate a stochastic distribution using multiple gaussians. For example, we can model the distribution of the grayscale values of pixels in a text image using two or three gaussians. The parameters of these gaussians are usually optimized using the Expectation Maximization (EM) algorithm.

The EM algorithm is a general technique of maximum likelihood estimation with incomplete data. One of the earliest papers on EM algorithm can be tracked to [35]. The seminal reference that formalized the EM and provided a proof of convergence is the paper by Dempster, Laird, and Rubin [19]. A good book devoted entirely to EM and its applications can be found in [61].

Let O be the set of observable variables, H be the set of missing (or unobservable, hidden) variables. The event that H takes a value h is denoted as $H = h$. Let Ω_O and Ω_H be the set of values or sample space for these variables. A complete data (sample) $c = (o, h)$ indicates that o is the observable part and h is the missing data.

With only the incomplete data o, an EM procedure attempts to find a model that maximizes the data log-likelihood :

$$\Theta^* = \arg\max_{\Theta} \ln P(o|\Theta)$$

where Θ denotes all the parameters of the model. For example, let us model the distribution of the observed data o, which is a set of grayscale values of pixels in an image, as two gaussians. The missing data h labels each pixel as belonging to one of the two gaussians. The EM procedure enables us to find the parameters of these two gaussians that maximizes the log-likelihood of the grayscale values of pixels. This EM procedure iterates between the following two steps until convergence.

E-step : This step estimates the missing part H given the current Θ^n and then use it to augment the observed data o to form the complete data set c. The missing data h is estimated by computing the following conditional expectation of the complete data log likelihood :

$$Q(\Theta|\Theta^n) \doteq \mathbb{E}\left[\ln P(o,h|\Theta)|o,\Theta^n\right]. \tag{2.61}$$

In other words, the E-step first computes the conditional expectation of the missing data h given the observed data o and the current estimate Θ^n, then substitutes the expectations for the missing data.

The expectation is therefore :

$$\begin{aligned}\mathbb{E}\left[\ln P(o,h|\Theta)|o,\Theta^n\right] &= \textstyle\sum_{h\in\Omega_H} \ln P(o,h|\Theta^n) P(h|o,\Theta^n) \\ &= \textstyle\sum_{h\in\Omega_H} \left[\ln P(o|h,\Theta^n) + \ln P(h|\Theta^n)\right] P(h|o,\Theta^n)\end{aligned}$$

For continuous H, it is calculated as :

$$\mathbb{E}\left[\ln P(o,h|\Theta)|o,\Theta^n\right] = \int_{\Omega_H} \ln P(o,h|\Theta^n) P(h|o,\Theta^n) dh$$

M-step : Once the complete data c is formed, the complete data log likelihood can be maximized with respect to the model Θ :

$$\Theta^{n+1} = \arg\max_{\Theta} Q(\Theta|\Theta^n) \tag{2.62}$$

The fundamental property of this algorithm is that $(\ln P(o|\Theta^n))$ is monotonically increasing, with a possible infinite limit.[1] In our example, the resulting optimal model Θ^* can then be used to assign the a label of each pixel as one of the two Gaussians by computing the label expectations.

2.3.2 Markov Random Field

Markov Random Field (MRF) theory, a branch of probability theory, provides a convenient and consistent way of modeling spatial context [51]. The MRF technique started to be widely used in image processing problems after the equivalence between MRFs and Gibbs distributions was established by Hammersley and Clifford and further developed by Besag [2] in 1974. Ten years later, Geman and Geman [30] and others advocated a frame of MRF using Maximum A Posterior (MAP) that enabled to develop algorithms for a variety of image processing problems.

Let $E = \{E_1, \dots, E_{|S|}\}$ be a family of random variables defined on the set S, in which each random variable E_i takes a value e_i in a set Λ. The family E is called a random field. To simplify the discussion in this thesis, we only consider as a set S the 2-D grid with N sites (usually, the pixel positions, $|S| = N$) and Λ as a discrete set. The event that E_i takes a value e_i is denoted as $E_i = e_i$, and therefore the joint event is denoted as $(E_1 = e_1, \dots, E_N = e_N)$ and abbreviated as $E = e$, where $e = \{e_1, \dots, e_N\}$ is a *configuration* of E.

Neighborhood system

A neighborhood system $\mathcal{G}$ is composed of the union of sets $\mathcal{G}_i$ of S verifying the following properties :

$$\forall i \in S,\ i \notin \mathcal{G}_i \tag{2.63}$$

$$\forall i, j \in S,\ j \in \mathcal{G}_i \Leftrightarrow i \in \mathcal{G}_j \tag{2.64}$$

Moreover, we will denote by $\mathcal{C}$ the set of all cliques c, a clique c being a subset of S reduced to a singleton or whose sites are neighbors each other. Figure 2.5 shows neighborhood system of order 1 or 2 as well as the corresponding kinds of cliques.

[1] Redner and Walker [85] give a local convergence theorem which states that under suitable conditions, the series converges to the maximum likelihood estimate Θ^*, provided the initial estimate is sufficiently close to Θ^*.

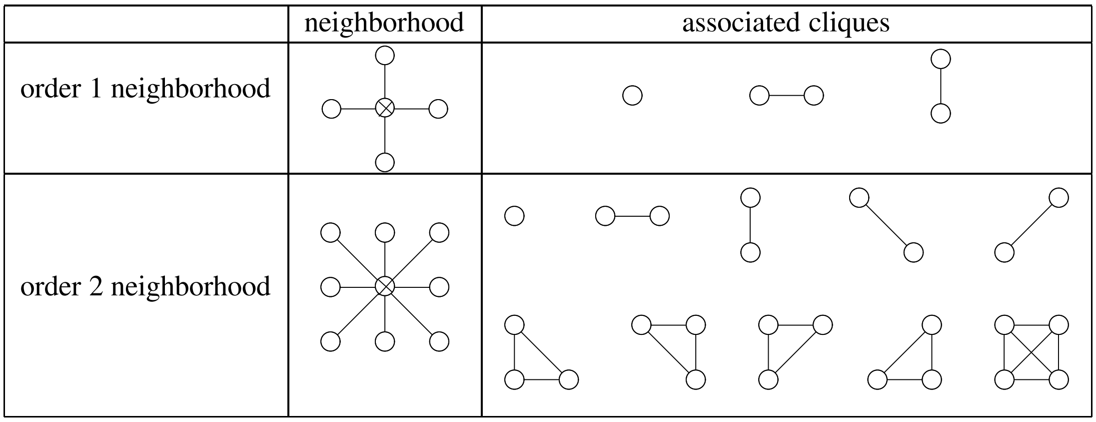


FIG. 2.5 – Example of neighborhood systems and associated cliques.

Markovian property

The space of possible configurations of E : $\Omega = \{e, e \in \Lambda^N\}$ is exponential to N, and can be very huge if N is the number of pixels in an image. To introduce a prior model on E defined by local properties, a Markovian *a priori* property is often imposed on this space. E is said to be a Markov random field on S with respect to a neighborhood system $\mathcal{G}$ if and only if the following two conditions are satisfied :

$$p(E = e) > 0, \ \forall e \in \Omega \tag{2.65}$$

$$\forall i \in S, \, p\left(E_i = e_i | (E_j = e_j)_{j \in S - \{i\}}\right) = p\left(E_i = e_i | (E_j = e_j)_{j \in \mathcal{G}_i}\right) \tag{2.66}$$

where $\mathcal{G}_i$ denotes the set of the sites neighboring i.

The first condition states that no configuration shall be forbidden *a priori* and the second one corresponds to the Markovian hypothesis, that is, at each site, the probability conditioned by the realization at all other sites is the same as the probability defined only with respect to the neighbors.

Markov-Gibbs equivalence

A very useful theorem established by Hammersley and Clifford [2], states that E is an MRF with respect to $\mathcal{G}$if and only if its configurations follow a Gibbs distribution, that is, it is of the form :

$$p(E = e) = \frac{1}{Z(\beta)} e^{-U_1^\beta(e)} \tag{2.67}$$

in which :
– β is a set of parameters ;

– $Z(\beta)$ is a normalization constant called "partition" function :

$$Z(\beta) = \sum_{e \in \Omega} e^{-U_1^\beta(e)} \tag{2.68}$$

Since the configuration space Ω is so huge, $Z(\beta)$ cannot be estimated in practice.

– $U_1(e)$, is an energy function, which can be decomposed as :

$$U_1^\beta(e) = \sum_{c \in \mathcal{C}} V_c^\beta(e) = \sum_{c \in \mathcal{C}} V_c^\beta(e_c) \qquad (\text{thus } \mathrm{P}_E(e) \propto \prod_c f_c^\beta) \tag{2.69}$$

where V_c, called interaction potential, depends only on the labels at the sites of c ($e_c = \{e_s, s \in c\}$). It is the definition of these potentials that provides the *a priori* properties of the label field.

Estimating an optimal configuration of an MRF

Among the different criterions of estimating the optimal configuration of an MRF, the Maximum A Posteriori (MAP) is one of the most popular criterions. Let O be a random field and E be an MRF both defined on the set S, where each variable in E takes a label e in Λ. Given a realization o of O, we are looking for the most *a posteriori* probable configuration $\hat{e}$ that gave rise to the observations, that is :

$$e^* = \arg\max_{e \in \Omega_E} \; p(E = e | O = o) \tag{2.70}$$

Using the Bayes rule, and since $p(O = o)$ is independent of e :

$$\hat{e} = \operatorname{argmax}_{e \in \Omega} \; p(O = o | E = e) \times p(E = e) \tag{2.71}$$

The first term of (2.71) is obtained by modeling the link between observations and labels, and provides the likelihood of the observed data, given the underlying (unknown) labels. Often, it is obtained through the modeling of the process (noise, blurring,. . .) transforming the labels (or primitives) into the observed data. It can also be some ad-hoc models. This probability is often parameterized, and we will call φ this set of parameters.

Let us assume the likelihood of an observation at a given site, given the labels, only depends on the label at the same site. Thus, this "point-to-point" likelihood has the general following form :

$$p(O = o | E = e) = \prod_{i \in S} p(O_i = o_i | E_i = e_i) = \exp(-U_2(e, o)), \tag{2.72}$$

where

$$-U_2(e, o) = \sum_{i \in S} \ln p(O_i = o_i | E_i = e_i). \tag{2.73}$$

Substituting equations (2.67) and (2.72) into (2.71), we can see that $\hat{e}$ defined by (2.71) corresponds to the minimum of an energy function $U^\Theta(e, o) = U_1^\beta(e) + U_2^\varphi(e, o)$:

$$\hat{e} = \operatorname{argmin}_{e \in \Omega} \; U^\Theta(o, e) = \operatorname{argmin}_{e \in \Omega} \; \left(U_1^\beta(e) + U_2^\varphi(e, o) \right) \tag{2.74}$$

The minimisation of the energy $\hat{e}$ defined in equation (2.74) is in the general case a difficult optimization problem. The size of Ω is too huge ($card(\Lambda)^{card(S)}$) to perform an exhaustive search. Moreover, the energy function $\hat{e}$ is generally non-convex, and does not lead to analytical solutions. Thus, we first consider an iterative deterministic solution.

Assuming that the parameters $\Theta = (\beta, \varphi)$ are known, a popular deterministic minimization approach is the ICM (Iterated Conditional Modes). It is derived from simulated annealing schemes, and consists in changing in the current configuration e, the label at a site s with one of the modes of the local conditional distribution ($p(e_s|o, e_t, t \in S, t \neq s)$ which, due to the Markovian property is equal to $p(e_s|o, t \in \mathcal{G}_s)$). This process is iterated at all sites until all sites are stable (or until only a small fraction of sites are unstable). A site is said to be stable if its current label optimize the local conditional probability.

Parameter estimation

Assuming that the parameters $\Theta = (\beta, \varphi)$ are unknown, we propose an algorithm for estimating all the parameters Θ=(φ, β) using an EM procedure. Recall that the expectation step involves the computation of :

$$\begin{aligned} Q(\Theta|\Theta^n) &\doteq \mathbb{E}\left[\ln \mathrm{p}(O = o, E = e|\Theta)|O = o, \Theta^n\right] \\ &= \textstyle\sum_{e\in\Omega} \ln\left(\mathrm{p}\left(O = o|E = e, \Theta^n\right) \mathrm{p}\left(E = e|\Theta^n\right)\right) \ \mathrm{p}(E = e|O = o, \Theta^n) \\ &= \textstyle\sum_{e\in\Omega} \left(U_2^{\varphi}(e, o) + \ln(Z(\beta)) + U_1^{\beta}(e)\right) \ \mathrm{p}(E = e|O = o, \Theta^n) \end{aligned} \tag{2.75}$$

which is then maximized over Θ.

Two problems arise here. Firstly, this expectation on $\mathrm{p}(E = e|O = o, \Theta^n)$ can not be computed explicitly nor directly because of the huge size of the label space Ω. 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 $\ln\left(\mathrm{p}\left(E = e, O = o|\Theta\right)\right)$ is not completely known, because of the presence of the incomputable normalizing constant $Z(\beta)$ in the expression of $\mathrm{p}(e)$:

$$-\ln\left(\mathrm{p}(E = e, O = o|\Theta)\right) = \ln(Z(\beta)) + U_1^{\beta}(e) + U_2^{\varphi}(e, o) \tag{2.76}$$

To avoid this difficulty, we can use the pseudo-Likelihood function [8] as a new criterion. The pseudo-likelihood of the MRF E, denoted by p_E is defined from the conditional probabilities 2.66 :

$$\mathrm{p}(E = e|\beta) \doteq \prod_{s\in S} \mathrm{p}(E = e_s|E_{\mathcal{G}_s} = e_{\mathcal{G}_s}, \beta) \tag{2.77}$$

Similarly, we get a joint pseudo-likelihood distribution :

$$\mathrm{p}(E = e, O = o|\Theta) = \mathrm{p}(O = o|E = e, \varphi) \times \prod_{s\in S} \mathrm{p}(E = e_s|E_{\mathcal{G}_s} = e_{\mathcal{G}_s}, \beta). \tag{2.78}$$

Since the conditional probabilities are known, this definition is now independent of the normalizing constant $Z(\beta)$.

We thus want to maximize the following criteria :

$$\mathbb{E}\left[\ \ln \mathrm{p}(O=o|E=e,\varphi)\ +\ \sum_{s\in S}\ln \mathrm{p}(E=e_s|E_{\mathcal{G}_s}=e_{\mathcal{G}_s},\beta)\ \ |\ \ O=o,\Theta^n\right] \tag{2.79}$$

with respect to $\Theta = (\varphi,\beta)$. However, from the current criterion, the estimation of the parameters $\beta = (V_{11}, V_{12})$ is indeed not directly necessary and we may only consider the conditional probabilities :

$$\mathrm{P}_{ij} = \mathrm{P}(e_s = i|e_{\mathcal{G}_s} \text{ is of type j}) \tag{2.80}$$

where a neighborhood of type j is characterized by the histogram of the labels on the neighborhood (H_j^{hv}, H_j^d). Using these parameters, the pseudo-likelihood function 2.77 can be rewritten :

$$\mathrm{p}(E=e) = \prod_{i,j}\mathrm{P}_{ij}^{n_{ij}(e)} \tag{2.81}$$

where :

$n_{ij}(e) =$ number of occurrences in e such that $e_s = i$ and $e_{\mathcal{G}_s}$ is of type j

Thus, and using a point-to-point likelihood (equation (2.72)), the maximization can be performed on $\Theta = (\varphi, (\mathrm{P}_{ij}))$ directly :

$$\mathbb{E}\left[\sum_{k=1}^{K}\left(\sum_s w_k(e_s)(-h^{\varphi_k}(o_s,k)\right)\ +\ \sum_{i,j} n_{ij}(e)\ln \mathrm{P}_{ij}\ \ |\ O=o,\Theta^n\right] \tag{2.82}$$

where :

$w_k(e_s) = 1$ if $e_s = k, 0$ otherwise.

After maximization, we get the following formula[2] :

$$\mathrm{P}_{ij}^{(n+1)} = \frac{\hat{n}_{ij}}{\sum_i \hat{n}_{ij}} \tag{2.83}$$

$$\varphi_k^{(n+1)} = \underset{\varphi_k}{\operatorname{argmin}}\ \sum_s\ \mathbb{E}[w_k(e_s)|o,\Theta^n]\ h^{\varphi_k}(o_s,k) \tag{2.84}$$

with

$\hat{n}_{ij} \doteq \mathbb{E}[n_{ij}|o,\Theta^n] = \sum_e n_{ij}(e)\,\mathrm{p}(E=e|O=o,\Theta^n)$

$\mathbb{E}[w_k(e_s)|o,\Theta^{(n)}] = \mathrm{P}(e_s=k|o,\Theta^{(n)}) = \sum_e w_k(e_s)\,\mathrm{p}(E=e|O=o,\Theta^n)$ is the marginal posterior distribution.

[2]We can point out that the pseudo-probability function $\mathrm{p}(E=e, O=o)$ is formally equivalent to a stochastic process where E would be generated by a Markov chain whose transition probabilities would be the conditional probabilities. Therefore, the solution to the EM problem can be related to the Re-estimation formula appearing in the Markov chain domain.

As pointed out before, these expectations will be approximated using a Monte Carlo method based on the Gibbs sampler, associated with the posterior distribution p$(E|O,\Theta^n)$.

If we come back to the estimation of the parameters β=$(V_{11}, V_{12}^{hv}, V_{12}^{d})$. From the definition of P_{ij}, we have :

$$\mathrm{P}_{ij} \propto \exp - \left(V_{11}(i) + \sum_{k=1}^{K} (H_j^{hv}(k) V_{12}^{hv}(i,k) + H_j^{d}(k) V_{12}^{d}(i,k)) \right) \qquad \forall i \in 1 \dots K \quad (2.85)$$

Thus, for every type j of neighborhood, we get the $(K-1)$ following equations :

$$\begin{aligned} -\ln\left(\frac{\mathrm{P}_{ij}}{\mathrm{P}_{1j}}\right) &= V_{11}(i) - V_{11}(1) & (2.86)\\ &+ \sum_{k=1}^{K} H_j^{hv}(k) \left(V_{12}^{hv}(i,k) - V_{12}^{hv}(1,k) \right) & (2.87)\\ &+ \sum_{k=1}^{K} H_j^{d}(k) \left(V_{12}^{d}(i,k) - V_{12}^{d}(1,k) \right) & (2.88)\\ & \forall i \in 2 \dots K & (2.89) \end{aligned}$$

which are linear with respect to the parameters to estimate. Since we have more equations than unknown, a standard least-squares solution can be obtained, by replacing the P_{ij} by their current estimate P_{ij}^{n}. Two problem arises with this method. Firstly, many conditional probabilities might be 0,[3] leading to a problem when evaluating the left-hand side of the above equation. To avoid this, we put a minimum probabilities on each P_{ij}. Secondly, some of the local neighborhood might not be observed, leading to useless equations. To deal with this fact, we weight each equation with the number of occurrence of the corresponding neighborhood type in the course of the Monte Carlo simulation.

In the preceding method, the "transition" probabilities P_{ij} are estimated in a first step using the (Gibbs) EM algorithm ; then a prior model of type 2.77, parameterized by the potentials V_{11} and V_{12}, is fitted to the obtained Pij values. From equation (2.81), we may also have estimated the potentials directly, by maximizing the expectation in the EM algorithm with respect to these parameters, using a gradient descent algorithm for instance :

$$\frac{\partial \mathbf{E}[...]}{\partial V_r} = \sum_{i,j} \hat{n}_{ij} \frac{\partial \ln \mathrm{P}_{ij}^{V}}{\partial V_r} \qquad (2.90)$$

for all V_r potential parameters, and where V denotes the set of all potentials.

2.4 Conclusion

In this chapter, we have introduced the visual feature extraction methods and machine learning approaches that will be used for text detection. These techniques are used in our system or used

[3]Note however that, thanks to the Gibbs sampling, this problem is not as dramatic as that of estimating the parameters from a single realization of the labels.

as comparisons for evaluating our system. The statistical methods, the GMM-EM and the MRF, introduced in this chapter are mainly used for text recognition. We also used another important statistical method for modeling video text recognition, namely particle filtering. To simplify the discussion, this method is introduced using our real problem as an example in Chapter 6.

Chapitre 3

Text detection and recognition system

This chapter gives a global view of the proposed system for detecting and recognizing text embedded in images and videos. This short chapter is organized in two sections. In the first section, an overview of our text detection and recognition system is briefly introduced, which is helpful to understand the related techniques that will be proposed in the rest of chapters. The second section mainly presents and discusses the characteristics of the databases used in this thesis.

3.1 Text detection and recognition scheme

As mentioned in the introduction, the previous developments of text detection and recognition system can be classified into three typical categories :

1. Bottom-up methods : they segment images into regions and group "character" regions into words. The methods, to some degree, can avoid performing text detection. Due to the difficulty of developing an efficient segmentation algorithm for text in complex background, the methods are not robust for detecting text in many camera based images and videos.
2. Heuristic top-down methods : they first detect text regions in images using heuristic filters and then perform bottom-up techniques inside the text regions. These methods are able to process more complex images than bottom-up approaches. However, the manually designed filters are empirical and therefore produce many false text regions, which are referred to as false alarms.
3. Training-based top-down methods : they detect text regions based on filters that are trained using machine learning tools.

The method we propose belongs to the top-down category, and consists of two main tasks as illustrated by Figure 3.1 : 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 to obtain high performance with a lower computational cost in comparison 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

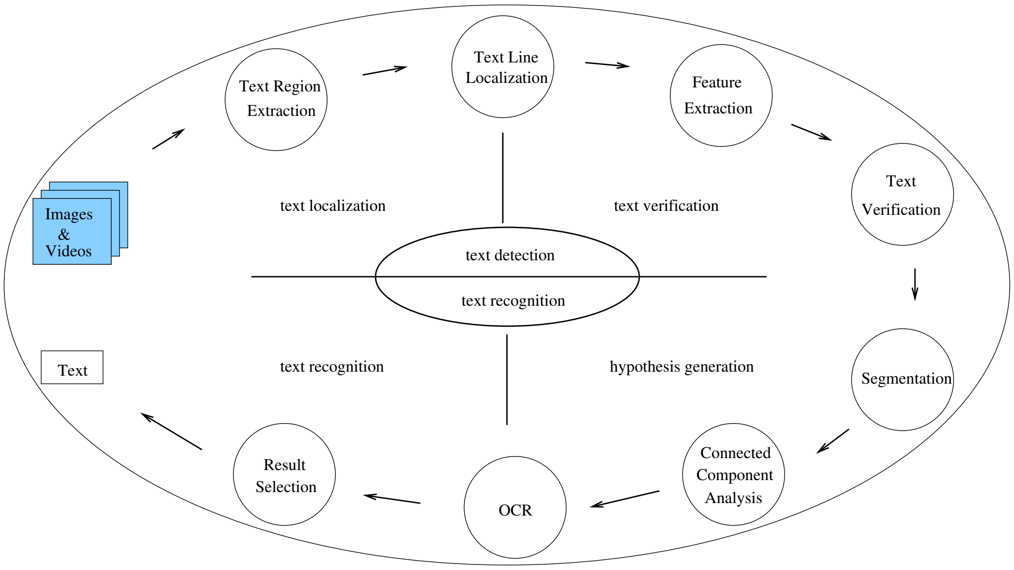


FIG. 3.1 – Algorithm proposed for text detection and recognition.

	images / video sequenses	characters	words
DB_TrainImages	50/30 mins	9579 (in test set)	2084 (in test set)
DB_SceneImages	50	2863	506
DB_NewsVideo	30 mins	2472	424
DB_Temporal	40 mins	2899	548

TAB. 3.1 – Databases used for training and evaluating algorithms in this thesis.

each time. The different regions, 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 accurately text characters. This methodology allowed to handle background grayscale multi-modality and unknown text grayscale values, problems that are very often not taken into account in the existing literature. When applied to real video text, it reduces by more than 50% the word recognition error rate with respect to a standard Otsu thresholding followed by the OCR [10, 71].

3.2 Databases for experiments

The databases for training and testing algorithms in this thesis are built for both images (including single video frames) and videos. The summary information of these databases is list in the Table 3.1.

3.2.1 Training image database (DB_TrainImages)

The first image database (DB_TrainImages) consists of 50 images of journal covers, flyers, maps, and frames extracted from videos of sports, news, advertisements and so on with a total length of one hour. The text characters contained in these images have different colors, grayscale values, fonts, sizes. Some characters are transparent and decorated with contours. Examples are shown in figure 3.2. This database will be used for selecting features and training text detection algorithms and testing text segmentation and recognition performance. All the images (frames) are first compressed using JPEG or MPEG-1, 2 and decompressed before performing any detection and recognition. Some frames which are grabbed during the fade-in or fade-out video effects are also involved in the database. We manually labeled all the text characters of this database.

3.2.2 Scene text image database (DB_SceneImages)

The scene text image database (DB_SceneImages) consists of 50 images of city streets, buildings, journal covers, and so on. The images are taken by a digital camera in 1280×960 resolution in various lighting conditions. Focus problems and slight transforms also appear in some images of this database. Figure 3.3 shows some of the images in DB_SceneImages.

3.2.3 News video database (DB_NewsVideo)

News video database (DB_NewsVideo) is a 30-minutes-long video compressed in MPEG-2 format in 720×576 resolution, at 25 frames per-second. This video contains a Belgian news program[1] in French provided by Canal+ in the context of the CIMWOS[2] European project. This database is used to have a benchmark of the text detection performance. One of the important criterions for validating text detection performance is precision, which is closely related on how frequent the text strings appear in images and videos. Therefore, we choose a real news video as a benchmark for this precision. The recognition performance will be measured mostly base on DB_TrainImages but not DB_NewsVideo because the style of character (fonts, sizes, colors ...) in DB_NewsVideo is not varying a lot. Figure 3.4 shows some frames in DB_NewsVideo.

3.2.4 Temporal character recognition database (DB_Temporal)

The temporal character recognition database (DB_Temporal) is a database for testing algorithms that recognize characters that appear in multiple video frames. The temporal information carried in multiple frames is explored in this thesis for improving text segmentation and recognition. This database is built on the basis of DB_NewsVideo, with the addition of the title credits of a documentary. The database consists of 230 sequences of text images that are tracked and extracted from DB_NewsVideo and 17 sequences of text images from the documentary credits. The sequences extracted from the documentary have special visual effects which are shown in figure 3.5. Ground-truth of the location and content of text strings in both DB_NewsVideo and DB_Temporal are manually obtained for further experimental usage.

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

[2]Combined Image and Word Spotting : Information Society Technologies IST :IST-1999-12203 Programme

FIG. 3.2 – Examples of images and video frames in DB_TrainImages

FIG. 3.3 – Examples of images in DB_SceneImages

FIG. 3.4 – Examples of video frames in DB_NewsVideo.

FIG. 3.5 – Examples of video frames in DB_Temporal.

Chapitre 4

Text detection

Text detection is defined as the task that localizes text strings in complex background without recognizing individual characters. In a computer-based application, there are many advantages of detecting text before performing segmentation and recognition. First, text embedded in images or videos usually doesn't cover the majority of pixels. It is obviously not an economic way to perform text region segmentation and character recognition on non-text regions. Second, a precise localization of text may offer the information about the scale of the text, which is very helpful for segmenting text regions from background. Moreover, the background inside a localized text region is usually less complex than the whole image if text characters are clearly visible. Therefore, a good text location may lead to a better text segmentation and recognition. Third, the characteristic of text string can be exploited in finding text. Since text strings have typical shapes and are aligned line by line, it implies that the localization of text strings is easier and more robust than the localization of individual characters.

As introduced in the Section 1.2.1, the state-of-the-art of image and video text detection techniques are classified into bottom-up, heuristic top-down methods and machine learning based top-down methods. In these three kinds of methods, the machine learning based top-down methods have the largest potential to yield the best performance. However, as discussed in the Sections 1.2.1 and 1.3, the current machine learning based top-down scheme has the drawback of high computational cost and difficulty for modeling various grayscale values and text sizes of strings embedded in complex backgrounds.

In this chapter, a machine learning based localization/verification scheme for text detection is proposed. It consists of two steps as illustrated in figure 4.1. The first step locates candidate text blocks in images with a fast algorithm. This localization process avoids applying the machine learning classifiers on the whole images as well as to further reduce the variation of text size by extracting individual text strings (lines). To obtain a fast algorithm, candidate text blocks are located by exploring heuristic characteristics. We use a threshold in this algorithm to adjust the weakness of the heuristic feature based classifiers in distinguishing text and backgrounds. A low threshold is useful to avoid rejecting text blocks (low rejection rate). The resulting false alarms will be removed in the following verification step. However, a too low threshold will also yields many false regions, making the text line extraction task more difficult. Fortunately, a good threshold can be found in practice. In the verification step, a size normalization is first performed on the candidate text lines. Then, a machine learning approach, either a multi-layer perceptrons (MLP) or a support vector machines (SVM), is employed to separate text regions from background regions in the candidates. Due to the large variance of the grayscale values of text characters, training of MLPs and SVMs and the verification of text lines are all performed in feature spaces. Four kinds of such features are proposed and compared in this chapter, which are robust to variable characters grayscale values.

4.1 Text region localization

The first goal of locating candidate text regions is to quickly filter out obvious background from images and video frames. Localization methods are therefore expected to be fast and ideally do not classify text regions as backgrounds. The second goal is to extract individual text lines (text strings in one line) so that the size of the text lines can be normalized. Therefore, the text region localization task is fulfilled by two sub-tasks : candidate text block extraction and individual text line extraction.

4.1.1 Candidate text block extraction

A text block refers to a block of image that contains one or more lines of text. Let S denote the set of sites (pixels) in an input image I. For any site s ($s \in S$) in the image I, we denote $T(s) = True$ as the event that pixel $I(s)$ is contained in a text block. The task of extracting text blocks, without recognizing individual characters, can be addressed by estimating at each site $s(s \in S)$ in the image I the probability $p\left(T(s) = True|I\right)$ and then grouping the pixels with high probabilities into regions.

In order to have a fast algorithm, we exploit the fact that text blocks contain short edges in vertical and horizontal orientations, and that these edges are connected to each other due to the connections between character strokes.

First, vertical and horizontal edges are detected individually by using Canny filters [6]. Let us use C_v and C_h to denote the vertical and horizontal edge images, where

$$C_v(s) = \begin{cases} 1 & \text{if s is a vertical edge point} \\ 0 & \text{otherwise} \end{cases}$$

and

$$C_h(s) = \begin{cases} 1 & \text{if s is a horizontal edge point} \\ 0 & \text{otherwise} \end{cases}$$

for any $s \in S$. Figure 4.2 (b,c) displays the vertical and horizontal edges resulting of this process for the video frame showed in Figure 4.2 (a).

Then, we connect the vertical edges in horizontal direction while connect the horizontal edges in vertical direction according using a mathematical morphology operator [101], namely dilation. To this end, two different structuring elements $Rect_v$ and $Rect_h$ are used according to the type of edge (vertical or horizontal). The dilated images D_v and D_h are defined as :

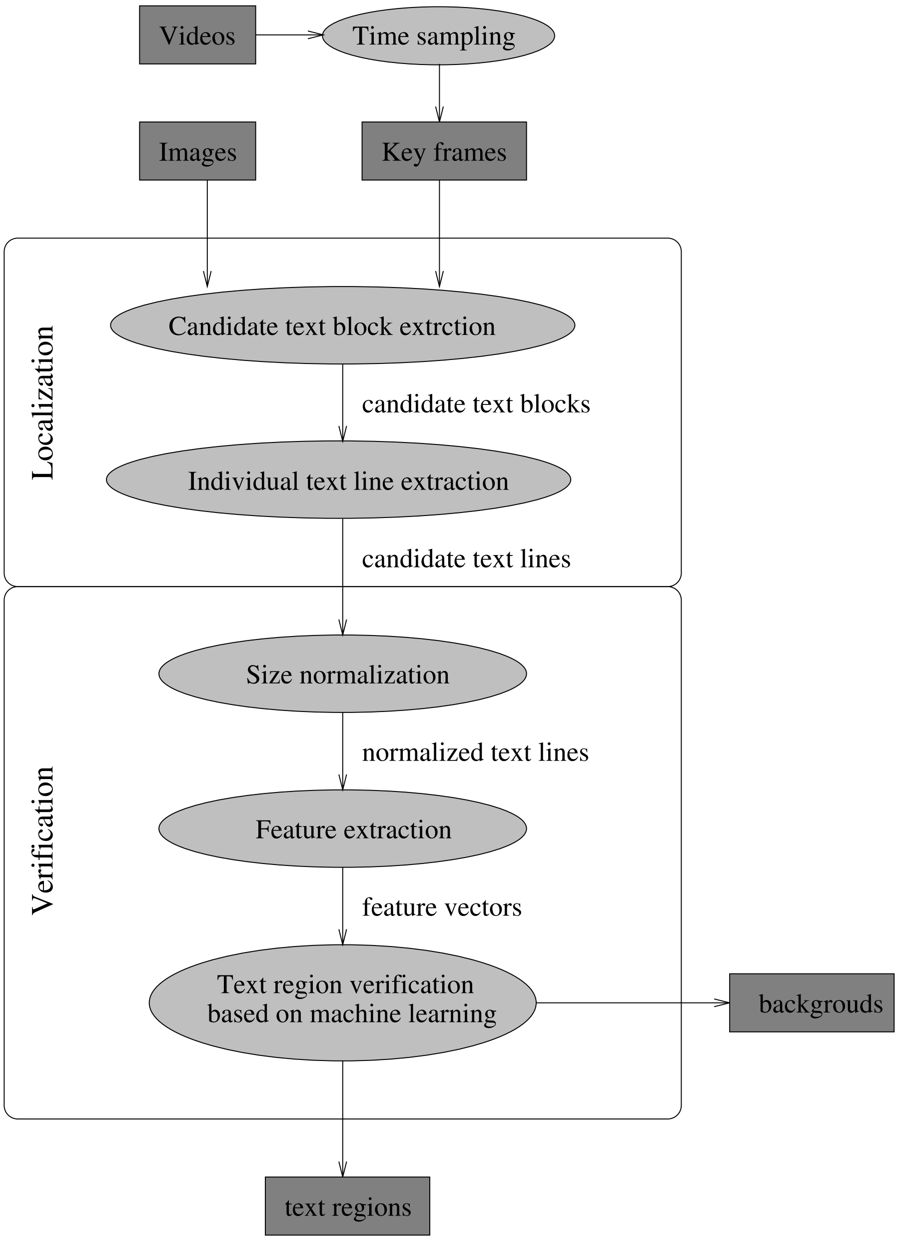


FIG. 4.1 – The localization/verification scheme for detecting text regions in images and video frames based on machine learning

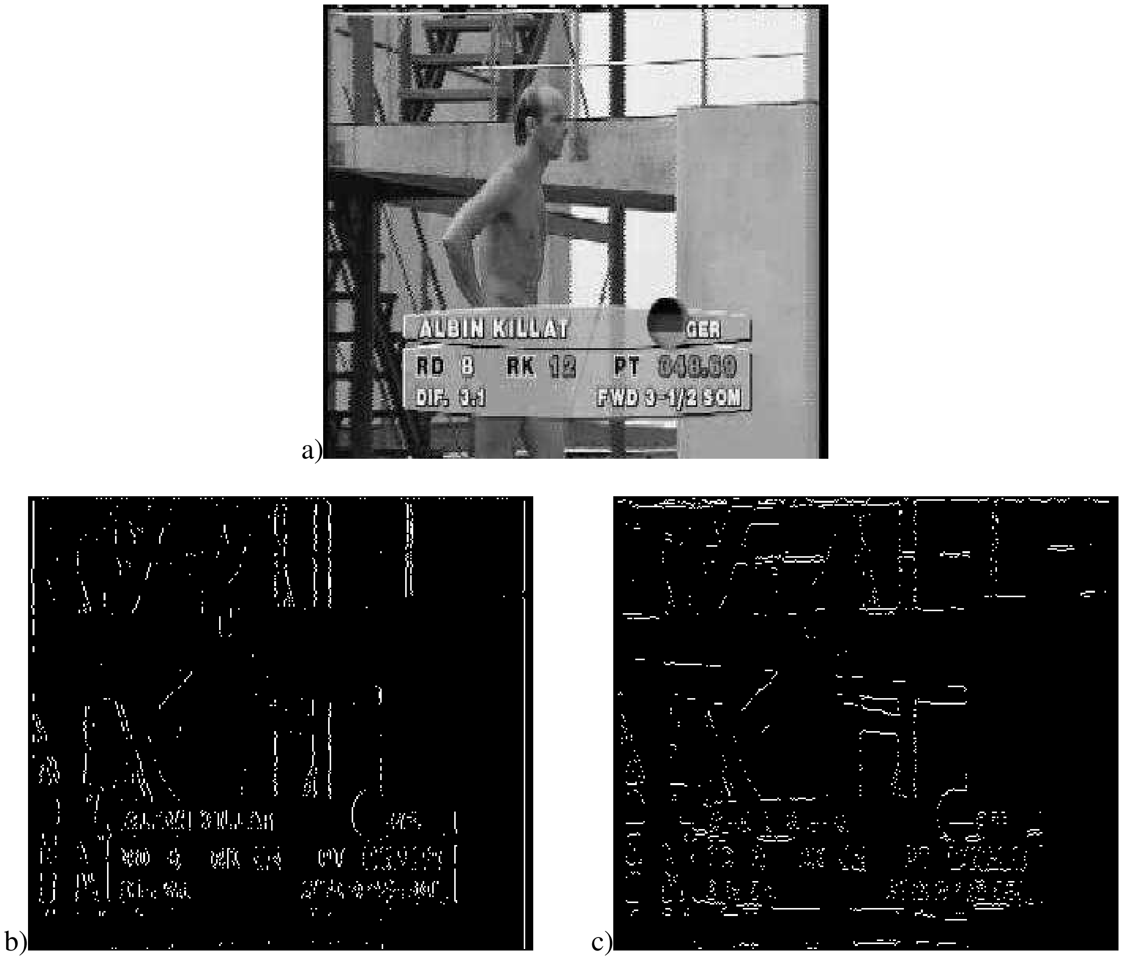
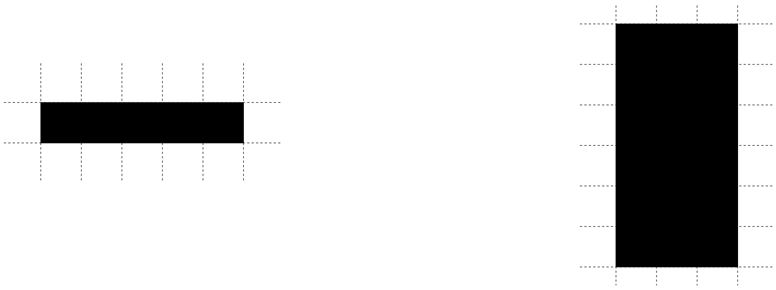


FIG. 4.2 – Edge detection in vertical and horizontal direction : (a) original image ; (b) vertical edges detected in image (a) ; (c) horizontal edges detected in image (a).



a) b)

FIG. 4.3 – (a) 5x1 structuring element for vertical edge dilation ; (b) 3x6 structuring element for horizontal edge dilation.

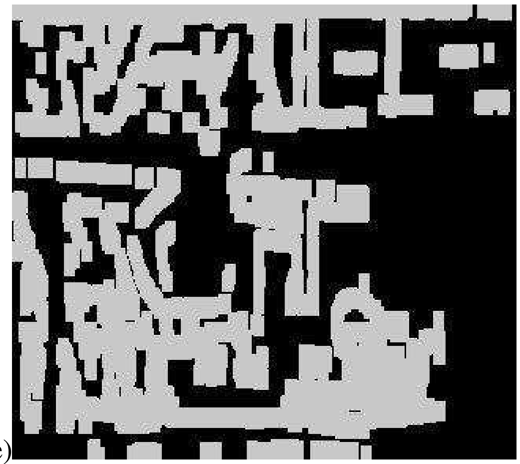
d)  e) 

FIG. 4.4 – Dilation of edge images. (d) dilation result of vertical edges in Figure 4.2 (b) using 5x1 vertical operator ; (e) dilation result of horizontal edges in Figure 4.2 (c) using 3x6 horizontal operator.

$$D_v = C_v \oplus Rect_v \tag{4.1}$$

and

$$D_h = C_h \oplus Rect_h \tag{4.2}$$

The structuring elements used for the dilation $Rect_v$ and $Rect_h$ are defined to have the rectangle shapes illustrated in Figure 4.3. The exact shape of $Rect_v$ and $Rect_h$ are related to the target text sizes. We found out through experiments that 5×1 $Rect_v$ and 3×6 $Rect_h$ are robust to detect text around 6 pixels to 40 pixels high and 8×1 $Rect_v$ and 4×8 $Rect_h$ are robust to the text from 10 pixels to 55 pixels. Since most of the text in video frames are 8 to 40 pixels high at 352×288 resolution, we selected 5×1 $Rect_v$ and 3×6 $Rect_h$ for detecting text in the images that are normalized into 352×288 resolution. A structuring element specifies a dilation pattern (or shape) of a neighborhood of a given pixel. The dilation in our case performs an expansion of the edge points. This operation can be described as the following : if the center pixel of the pattern is an edge point (value 1), then the neighboring pixels specified by a structuring element are also set to 1 in the dilated image. The vertical and horizontal edge dilation of the Fig. 4.2 (b) and (c) are shown in Figure 4.4 (d,e).

We then consider only the regions that are covered by both the vertical and horizontal edge dilation results as candidate text blocks. Thus, we estimate the probability $p(T(s) = True \,|\, I)$ as :

$$p\left(T\left(s\right) = True \,|\, I\right) = D_v(s) D_h(s) \tag{4.3}$$

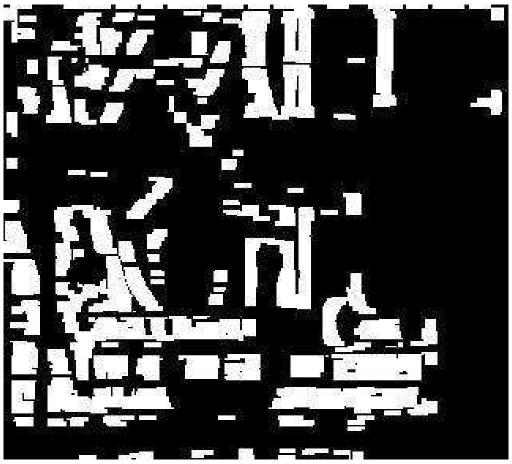


FIG. 4.5 – Candidate text region extraction. The white regions are extracted candidate text regions combining the clues from the two edge dilation images shown in the Figure 4.4.

In this case, the probability is a binary value. Figure 4.5 illustrates the result of this step. It can be observed that most background regions that contain only long edges are removed and also the regions are better segmented in comparison with both vertical and horizontal dilated images.

There are two advantages of using this text block extraction algorithm. The first is that the algorithm involves only edge detection and morphological operation on binary images, which enable fast text block extraction using a fast implementation of dilation algorithm. The second advantage is that the algorithm only relies on edges, which is quite robust to text intensity changes. Also, ideally, the threshold of the edge detection step can be set to a value low enough such that true text regions will be rejected as less as possible. The low value threshold sometimes leads to large mount of false detected regions (false alarms). 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 following section.

4.1.2 Individual text line extraction

In order to normalize text sizes, we need to extract individual text lines from candidate text blocks. 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 eliminates false alarms, such as slant stripes, which do not contain any well defined baselines. Secondly, it refines the location of text strings in candidate regions that contain text connected with some background objects.

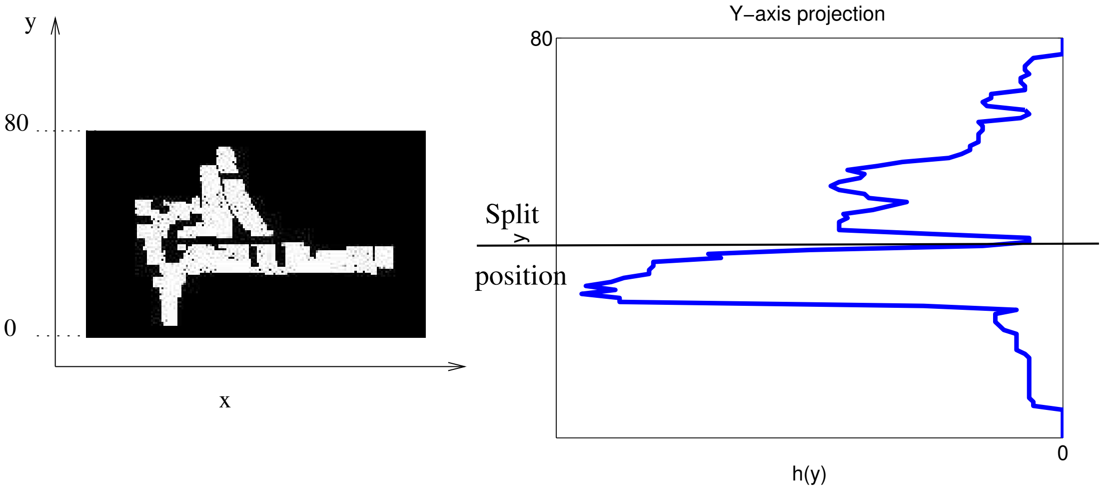


FIG. 4.6 – An example of a text region and its Y-axis projection. The split position is located using maximum derivative

Baseline detection starts by computing the Y-axis projection $h(y)$, where $h(y)$ denotes the number of text pixels in line y. Figure 4.6 illustrates a text block and its Y-axis projection.

The criteria of a text line with "accurate baselines" is measured mainly on the fill-factor F which is defined as the density of text pixels (white pixels shown in the Figure 4.6) 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. Various thresholds (from 70% to 75%) of the fill-factor are given in the previous literature. By manually labeling some real text strings in images and video frames, we found that $TF = 70\%$ is a reasonable value to avoid the rejection of text strings. The splitting positions (line numbers) are located using the three following algorithms, applied sequentially in three different cases.

The three splitting algorithms are :

1 : Varying length text lines splitting algorithm. This algorithm is used to split text lines from connected background regions or split two text lines of very different lengths
2 : Equal length text lines splitting algorithm aims at splitting two text lines of similar lengths.
3 : Baseline refinement algorithm is employed when a text line cannot be split any more. Its aim is to extract the top and bottom baselines more accurately.

We now describe these three algorithms in more detail.

Algorithm 1 : varying length text lines splitting

This algorithm aims at splitting a text region containing text strings of different lengths or text strings connected with background regions. In fact, most of these connected background regions are usually shorter than text strings. Let us define the Y-coordinate y_d which has the maximum absolute derivative of $h(y)$:

$$y_d = \arg\max_y \left| \frac{\delta h(y)}{\delta y} \right| \tag{4.4}$$

Let us denote $d(y_d)$ as this maximum absolute derivative value and define y_l the longest line in the region :

$$y_l = \arg\max_y h(y) \tag{4.5}$$

The region is split at line y_d if the following two conditions are satisfied :

$$d(y_d) > t_g \quad \text{and} \quad \frac{h(y_d)}{h(y_l)} < 0.5 \tag{4.6}$$

where t_g is a given threshold to impose a sufficiently large variation of the $h(y)$. We found that $t_g = 25$ is a good value. Figure 4.6 shows split position located using this varying length text lines splitting algorithm based on maximum derivative.

Recursively performing this algorithm on the given text region and its resulting sub-regions until no new splitting line is found, we are able to split most of text regions into individual text lines.

In practice, the derivative of $h(y)$ with respect to y is approximated by

$$\frac{\delta h(y)}{\delta y} = h(y) - h(y-1)$$

This approximation has two drawbacks. First, it can miss the maximum derivative point in a larger scale because it computes the derivative values using only the two positions next to each other. The way of somehow overcoming this drawback is to estimate the smoothness of the $h(y)$ by extending h_{y-1} to h_{y-t}. Second, the value $d(y_d)$ may correspond to the derivative at either y_d or y_{d-1}. We must choose the one that has the smaller $h(y)$ as y_d.

Algorithm 2 : equal length text lines splitting

The above algorithm may fail when a region consists of two text lines of similar lengths and its $h(y)$ function is very smooth. To split text lines in such a region, the $h(y)$ function is considered as a one dimension histogram and the text line splitting task is to search for a proper threshold y_c that split this histogram into two parts : the part that is above the y_c line and the part that is below.

This threshold can be found using the class variance method proposed by Otsu [75]. Let us define the threshold y_c as :

$$y_c = \arg\max_{y^*} \frac{\mathbf{E}\left(\left(h(y) - \mathbf{E}\left(h(y)\right)\right)^2_{y<y^*}\right) + \mathbf{E}\left(\left(h(y) - \mathbf{E}\left(h(y)\right)\right)^2_{y\geq y^*}\right)}{\mathbf{E}\left(\left(h(y) - \mathbf{E}\left(h(y)\right)\right)^2_{\forall y}\right)} \tag{4.7}$$

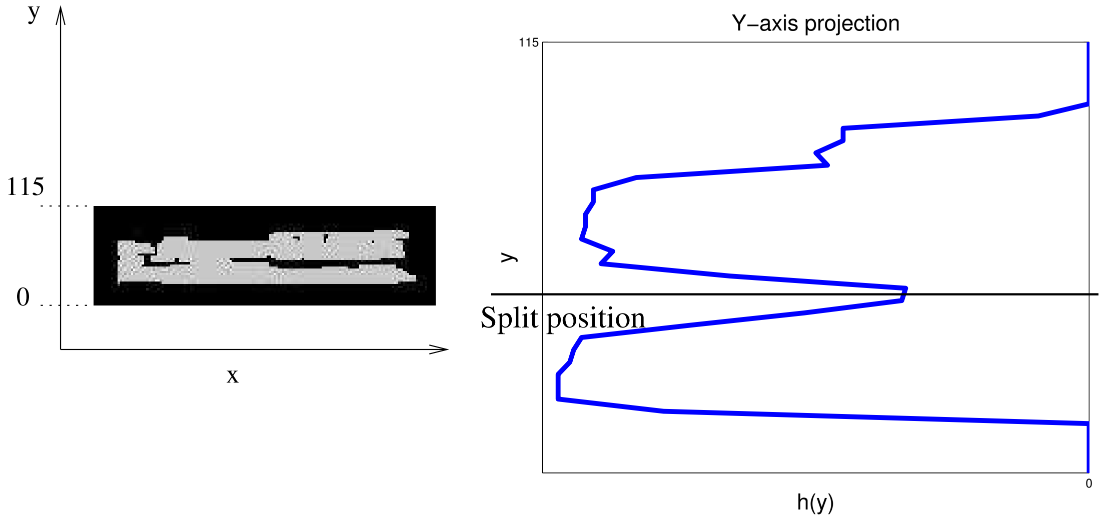


FIG. 4.7 – Split two text lines using Otsu thresholding. The text lines are not split by the Algorithm 1 because of its large scale edge.

The resulting threshold y_c maximizes the inter-class variance while minimizing the intra-class variance of the two parts (above and below) of $h(y)$. Then, if $h(y_c)$ is less than 50% of the longest line in this region, we split the region at line y_c. Figure 4.7 illustrates this equal length text lines splitting algorithm.

Algorithm 3 : baseline refinement

If a region cannot be split by the above two algorithms, we assume that it may contain only one text line. In this case, we apply a baseline refinement algorithm to yield more precise location of the top and bottom boundaries (baselines) of the region.

More clearly, this algorithm searches for the greatest sub-region (in height) whose fill-factor is above the given threshold TF. Given a region r and its $h(y)$ function, the center line of the sub-region r_{sub} is initialized as the line of the gravity center of the whole region r :

$$y_{Center} = \frac{\sum_y y \cdot h(y)}{\sum_y h(y)} \tag{4.8}$$

The height of the sub-region is initialized to be two pixels less than the minimum target text height. For example, if the smallest text we want to detected is 8 pixels, we can initialize the sub-region r_{sub} with a height of 6 pixels. Therefore, the initial top line T and the bottom line B of the sub-region are :

$$T = y_{Center} - 3 \quad \text{and} \quad B = y_{Center} + 3$$

The algorithm can then be described as the following :

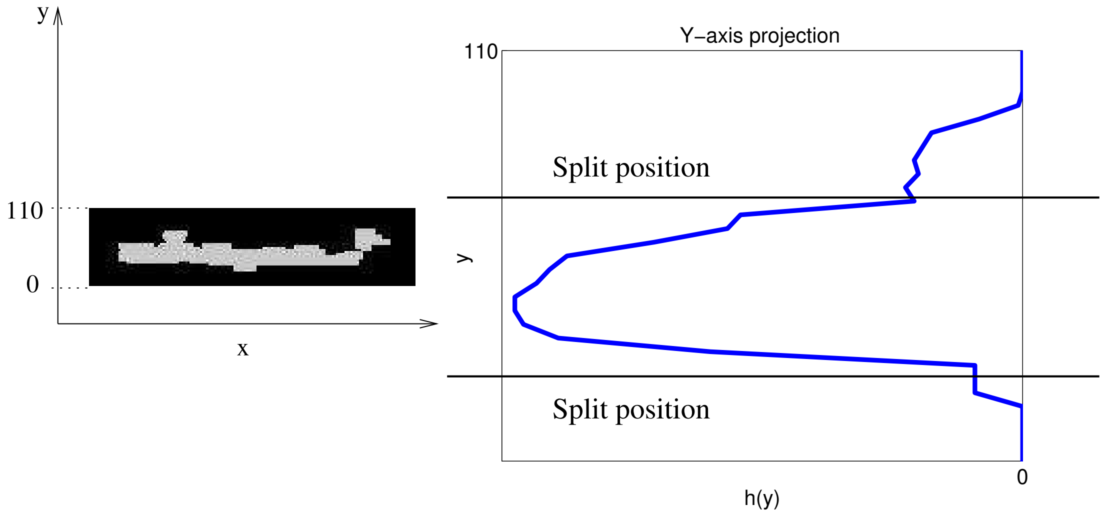


FIG. 4.8 – Refinement of top and bottom baselines of a text string.

1. Initialize T and B as above ;
2. If the fill-factor of the current region delimited by T and B is below the threshold TF, the region is eliminated as a non-text region.
3. $T = T - 1$; $B = B + 1$;
4. If the fill-factor of the updated region is above the threshold TF, go to 3.
5. $T = T + 1$; $B = B - 1$; set T and B as the new top and bottom baselines of the region r.

Figure 4.8 shows an example of using this baseline refinement algorithm to find more accurate baselines of a text string.

FIG. 4.9 – Baseline detection for refining and selecting text lines.

FIG. 4.10 – Text line localization : the rectangle boundaries of candidate text lines.

Text line selection using heuristic constraints

Figure 4.9 illustrates the result of applying text line extraction step on the image of Figure 4.5. Typical geometric characteristics of text strings are then employed to select the resulting regions. The final candidate text line should satisfy the following constraints :

1) The number of pixels contained in this text line is greater than a specified value. This value is decided by experiments according to the structuring elements used in the dilation operations. For instance, the value is 75 pixels if the structuring elements of the dilation are 5×1 and 3×6.

2) The horizontal-vertical aspect ratio of the text line is greater than a threshold T_{hv}. This threshold is selected according to the minimum number of characters in each text string. Experimentally, we figure out the relationship between the minimum number of characters n in a text string and this threshold T_{hv} :

$$T_{hv} = 0.6\,n$$

3) The height of the text line is greater than 8 pixels. The text lines that are than 8 pixels high have too poor resolution to lead a good recognition.

Figure 4.10 shows the rectangle boundaries of the candidate text line images. In general, by adopting the parameters given above, the detectable sizes of the text characters can cover a large range (from 8 to more than 40 pixels high). An extension of the proposed localization algorithm for larger characters consists of applying the algorithm on scaled image pyramid, as described in [121].

4.2 Text region verification

As in many other works, the text localization procedure described in the previous section is rather empirical and may therefore produce false alarms (i.e. non text regions). To remove these false alarms, a verification process relying on well established machine learning classifiers is employed. The role of the classifier is to separate the outputs provided by the text localization step into the text and non-text classes. The localization step produces very unbalanced class priors, i.e. it produces much more non-text samples than text samples. Thus, most learning algorithms based on empirical risk minimization, e.g. multi-layer perceptrons (MLP), will have a tendency to classify correctly only the non-text samples to minimize the error over the data set, which might not be desirable. On the contrary, structural risk minimization based methods, such as support vector machine (SVM), aim at minimizing a bound on the generalization error rather than minimizing the error over the data set, thus enabling a good classification performance on the text class.

Both MLP and SVM have been studied for the verification task and are presented next. Details about MLP and SVM can be found in the second chapter. The remaining of this section describes the character size normalization step, the input features we considered, the training methodology, and, finally, the text line verification procedure.

4.2.1 Character size normalization

The height of a candidate text line extracted from the text localization procedure does precisely indicates the size of its inside characters. It is helpful to normalize the whole text line in order to reduce the variance of the character size. Keeping the same aspect ratio, each candidate text line is normalized using bilinear interpolation into an image having a 16 pixels height.

4.2.2 Feature extraction

Characters in candidate text lines can be any grayscale values. To reduce this variation, for each normalized candidate text line I, a feature image I_f is then computed from I. Fixed size input feature vectors z_i to the MLP or SVM are directly extracted from I_f on 16x16 windows W sliding over the image grid S, which is normalized into 16 pixels high. Since the grayscale values of text and background are supposed to be unknown, we proposed and evaluated four different kinds of features invariant to the absolute grayscale values. These features are :

1. Grayscale spatial derivatives features.
2. Distance map features.
3. Constant gradient variance features.
4. DCT coefficients.

F1 : grayscale spatial derivatives features

The first features we used are the first order spatial derivatives of the image brightness function in both the X and Y directions, $\frac{\partial I}{\partial x}(s)$ and $\frac{\partial I}{\partial y}(s)$, computed at each site s. The feature vector corresponding to a sliding window is the concatenation of the 16×16 pairs of derivatives

$$z_i = \left(\frac{\partial I}{\partial x}(s_1), \frac{\partial I}{\partial y}(s_1), \ldots, \frac{\partial I}{\partial x}(s_{256}), \frac{\partial I}{\partial y}(s_{256}) \right)$$

The derivative features, as shown in Figure 4.11, are measures of the contrast between characters and backgrounds. It is invariant to text characters of any grayscale values that have constant contrast to the background.

F2 : distance map features

Since the grayscale values of both characters and background may vary, the contrast of text characters is background dependent. It implies that the brightness spatial derivatives may have big variations in the context of text images. Thus, we considered as a second feature image the distance map DM, which only relies on the position of strong edges in the image. DM is defined by [110] :

$$\forall s \in S,\ DM(s) = \min_{s_i \in E} d\left(s, s_i\right) \tag{4.9}$$

where $E \subseteq S$ is a set of edge points, and d is a distance function, in our case an approximation of the Euclidean distance. Some examples of distance map features are shown in Figure 4.11.

The distance map feature only relies on the positions of the edges in the edge set E. This makes the distance map feature more independent to the contrast between characters and background than the grayscale spatial derivative feature. However, the computation of the edge set E still depends on the text/background contrast and a threshold employed in edge detection.

F3 : constant gradient variance features

To avoid the need for setting any threshold, we propose a feature called constant gradient variance (CGV). The variance normalization technique is usually used to enhance grayscale images or to preprocess input data in speech [64, 97]. Here, we apply this technique on image of the gradient magnitude 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 :

$$LM(s) = \frac{1}{|\mathcal{G}_s|} \sum_{s_i \in \mathcal{G}_s} g\left(s_i\right) \tag{4.10}$$

and

$$LV(s) = \frac{1}{|\mathcal{G}_s|} \sum_{s_i \in \mathcal{G}_s} \left(g(s_i) - LM(s)\right)^2 \tag{4.11}$$

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

$$CGV(s) = (g(s) - LM(s))\sqrt{\frac{GV}{LV(s)}} \tag{4.12}$$

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 $LM(s)$ as the mean and $LV(s)$ as the variance, it is easy to show that :

$$\mathbb{E}\left[CGV(s)\right] = 0 \tag{4.13}$$

and

$$\mathbb{E}\left[(\,CGV(s)\,)^2\right] = GV \tag{4.14}$$

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

F4 : 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 [25]. This frequency domain feature is widely used in JPEG and MPEG compression schemes, as well as in texture and object recognition. They have also been used for text detection [126]. The mathematical expression of the DCT can be found in the second chapter of this thesis. Out of all 256 coefficients only the last 255 are used in the feature vector. The first coefficient, which is proportional to the mean, does not contain relevant information.

Figure 4.11 illustrates three of these features. The grayscale values shown in the images are scaled into the range of 0-255 for display reasons. DCT feature images are not shown in this figure because visually they are not very meaningful.

4.2.3 Multi-layer perceptrons model for text verification

MLP, as introduced in the second chapter, is a widely used neural network. The text verification is a binary classification problem for an MLP, in which the output layer usually consists of one neuron whose output encodes the class membership. The MLP model is illustrated in Figure 4.12.

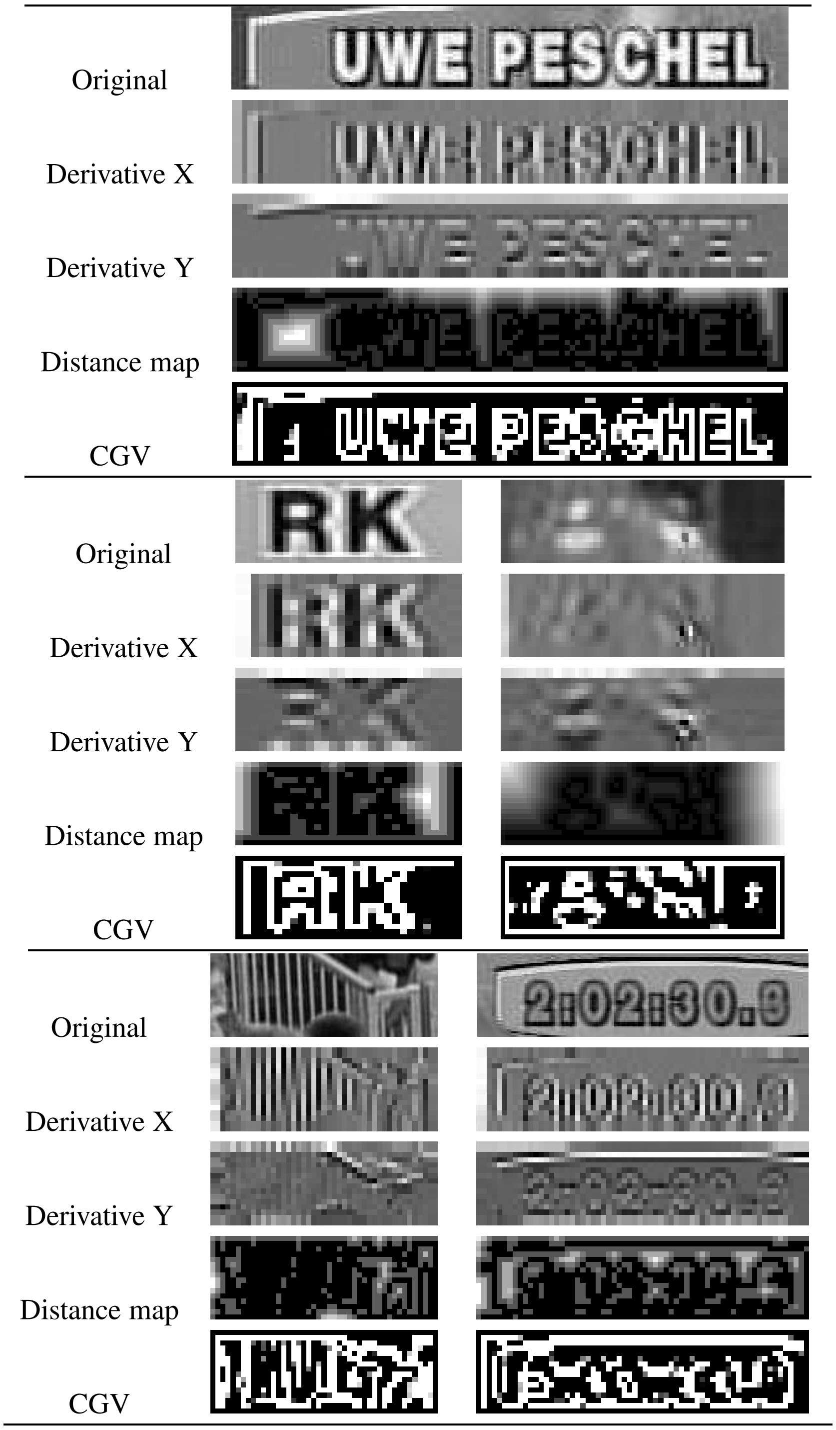


FIG. 4.11 – Examples of three features. The grayscale values shown in the images are scaled into the range of 0-255 for display reasons. DCT feature images are not shown in this figure since they are not very meaningful from a visual point of view.

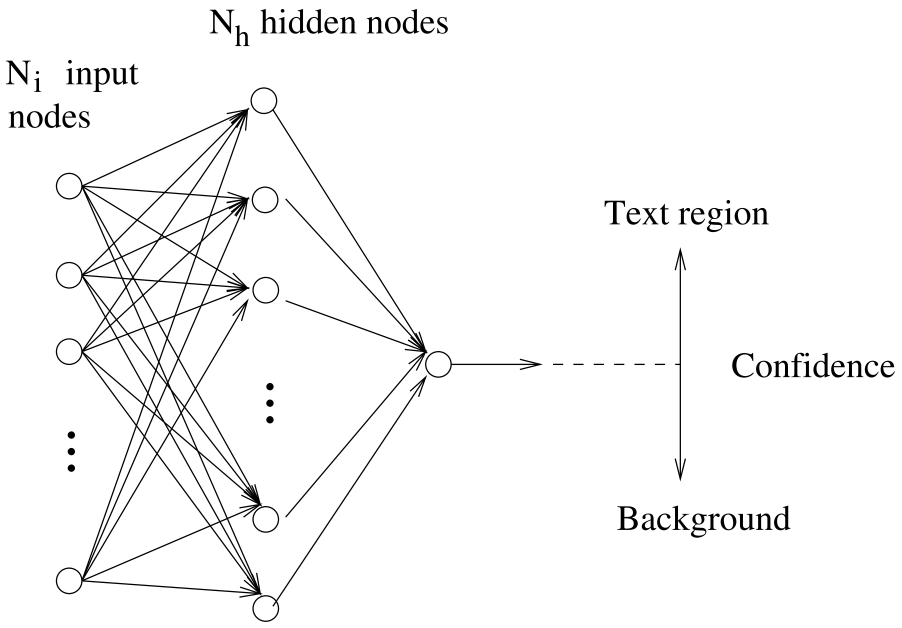


FIG. 4.12 – MLP model for text verification.

In our case, the MLP has only one hidden layer. The number of hidden neurons N_h is decided by a techniques, referred to as cross-validation, which will be discussed in 4.2.3. The number of input nodes N_i depends on the input feature space. For example, the derivative feature space has 512 input nodes and the distance map feature space has 256 input nodes.

Therefore, following the definition of MLP in chapter 2, the forward process of the above MLP for an input vector X can be simply written as :

$$\begin{aligned} O_*(X) &= W_* \cdot O_1 \\ &= W_* \cdot (W_{11} \cdot X, \ldots, W_{1N_h} \cdot X) \end{aligned} \tag{4.15}$$

where O_* (resp. W_*) indicates output value (resp. the weight vector) of the MLP final output neuron.

Multi-layer perceptrons training

The text verification training set consists of text and non-text examples. These samples are coming from text regions extracted from the DB_TrainImage database using the text localization algorithm presented in the previous chapter.

The MLP training algorithm, the backpropagation algorithm, is a gradient descent procedure in the weight space of the MLP :

$$W^{t+1} = W^{t} - \alpha \frac{\delta F\left(W, T_{set}\right)}{\delta W} \tag{4.16}$$

for a training set T_{set} and where we used the following cost function :

$$F\left(W, T_{set}\right) = \frac{1}{2} \sum_{(X_i, Y_i) \in T_{set}} \| O_*(X_i) - Y_i \|^2 \tag{4.17}$$

The detail expansion of equation 4.16 can be found in the Chapter 2.

Weights initialization, input normalization and learning rate

The weights of the MLP are initialized randomly and updated using eq. 4.16 iteratively. Before training the input vectors are normalized to have equal mean and unit variance. The learning rate α is first initialized as a big value 0.01 and is decayed every iteration with the rate of 0.01.

Number of hidden neurons

The number of hidden neurons k, which is related to the capacity of the MLP, is chosen by performing a M-fold cross validation on the training set, which is described as following :

1. Partition the training data set into M parts of equal size called the ”folds”. Then, assign each fold i a possible value of k_i.
2. For i = 1 to M, build an MLP with k_i hidden units and train the MLP using examples in all the folds except the ith as the training set. Evaluate the error rate of the resulting MLP on the ith fold considered as the test set.
3. Keep the value of $k = k_i$ corresponding to the lowest error rate as the optimal number of hidden units and built the final MLP with k hidden units. The final MLP is then trained on all the training data to obtain a good model (weight set).

The optimal numbers of hidden neurons of the four features range from 80 to 250 based on 5-fold cross validation processes.

MLP based verification on feature vector

As mentioned in the feature extraction section, feature vectors are extracted from sub-windows of a candidate text region. The confidence G of an input feature vector Z belonging to a text region is measured by the output of the trained MLP, that is :

$$G(Z) = O_*(Z)$$

4.2.4 Model of support vector machine for text verification

The SVM technique has been introduced in the Chapter 2. We performed tests on the three typical kernels, Gaussian RBF, polynomial and multi-layer perceptrons kernels, using small data sets and found the three typical kernels achieved quite similar results. In our text verification task, we thus choose one, the Radial basis function (RBF) as the kernel of the SVMs, recalled as :

$$K(X_i, X_j) = \exp\left(\frac{-\parallel X_i - X_j \parallel^2}{2\sigma^2}\right) \tag{4.18}$$

where X_i and X_j are feature vectors.

The kernel bandwidth σ is determined also by an 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 considered as the test set.
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.

For the four types of features, we obtained the optimal kernel bandwidths ranging from 1.25 to 15. The extracted numbers of support vectors are ranging from 62 to 500.

SVM based verification on feature vector

As in the MLP, the output of the SVM is used as a confidence value at the feature vector level. The confidence of an input feature vector Z belonging to a text region is measured by the output of the trained SVM as :

$$G(Z) = \left(\sum_{i=1}^{m} \lambda_i^* Y_i K\left(Z, X_i\right) + b\right) \tag{4.19}$$

4.2.5 Text-line verification

In the text verification step, the feature vectors discussed in Subsection 4.2.2 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 l feature vectors $Z^r = (Z_1^r, \ldots, Z_l^r)$.

The verification of a candidate **r** is performed by combining the confidence of the individual feature vectors Z_i^r. This confidence is expressed by the magnitude of the SVM function $G(Z_i^r)$ given by decision function 2.47 where the sum runs only on the support vector set (i.e. the samples for which λ_i is strictly greater than 0) resulting from the training step. The confidence of the whole candidate text line **r** is then defined as :

$$\text{Conf}(\mathbf{r}) = \sum_{i=1}^{l} G(Z_i^r) \times \frac{1}{\sqrt{2\pi}\sigma_0} e^{-\frac{d_i^2}{2\sigma_0^2}} \tag{4.20}$$

where d_i is the distance from the geometric center of the ith sliding window to the geometric center of the text line **r**, and σ_0 is a scale factor depending on the text line length L_t in terms of pixels, which is defined as :

$$\sigma_0 = \sqrt{\frac{L_t}{2.77}}$$

The weights of the confidences of the sliding windows are decayed from the center of a text line. This is mainly because the detected text line often has additional background at the left and right ends. We obtained better results by giving less weights to the confidences that are close to the left and right ends. Finally, the candidate text line **r** is classified as a real text region if

$$\text{Conf}(\mathbf{r}) \geq 0$$

4.3 Experiments and comparison

DB_TrainImagesExperiments are performed to evaluate the text localization and verification algorithms presented in the previous sections. In this section, we first discuss the criterions for evaluating the performance of the algorithms. We then report separately experimental results on text localization and text verification.

4.3.1 Evaluation criterions

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

$$PRR = \frac{RP}{TP} \tag{4.21}$$

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

$$PFR = \frac{PF}{PI} \tag{4.22}$$

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

The second set of evaluation criterions is defined at the region level. They make more sense for text recognition and content-based retrieval tasks. The region recall rate (RRR) is defined as :

$$RRR = \frac{RR}{RT} \tag{4.23}$$

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

The region precision rate (RPR) is defined as :

$$RPR = \frac{RR}{RE} \tag{4.24}$$

where RE denotes the total number of text regions extracted by the text localization algorithm or the whole text localization/verification approach.

4.3.2 Results of text localization

Comparison of text block extraction algorithms

At the lower (pixel) level, we compare the performance of the proposed text block localization algorithm with two other algorithms, namely the derivative texture algorithm [121] and the vertical edge based algorithm [104] in table 4.1, which can provide pixel-level localization and are often adopted in recent text detection systems [119, 93]. These two algorithms are implemented by ourselves according to the referenced papers. Let us first recall these two algorithms whose mathematical details have been discussed in the Chapter 2.

The derivative texture algorithm :

1. Produce three smoothed images from the given image using Gaussian function with zero mean and three variances : 1, $\sqrt{2}$ and 2.
2. At each corresponding pixel position, extract 3 second-order derivatives from every smoothed image and build a 9-dimentional vector feature.
3. Classify the feature vectors of all the pixel positions into three classes (text, non-text and uncertain) using k-means.

The vertical edge algorithm :

1. Compute first-order derivative in the X direction using the vertical Sobel operator.
2. Smooth the derivative image using a Gaussian function.
3. Threshold the smoothed image. In the resulting image, "1" indicates the text pixels and "0" indicates the background.

For a good evaluation process, the algorithms should be applied on characters of various grayscale values, various scales and various backgrounds. Therefore, the evaluation data set is defined in the following way.

First, we selected 40 video frames from the DB_TrainImages database as backgrounds. These frames do not contain any text. Then, we embed text strings at different places of these backgrounds. The length of the text strings are modulated from 4 characters to 40 characters, aligned

iX-based	PRR	PFR	CPU costs
Derivative Texture	90.54%	3.48%	$225(\times), 325(+)$
Vertical Edge	86.42%	16.17%	$34(\times), 35(+)$
Our algorithm	94.51%	1.73%	$36(\times), 44(+)$

TAB. 4.1 – Performance and running costs of different text detection techniques. RPR denotes the recall pixel rate and FPR denotes the false pixel alarm rate. The computational cost is measured by numbers of addition and multiply operation per pixel in average.

in the same line and grouped into less than five words. The grayscale values of the characters can be 10, 80, 160 and 200. The character sizes are also controlled, ranging from 10 pixels to 45 pixels high. After the embedding of a text string into a background image, gaussian noise with a 10 db signal to noise ratio is added to the image. The image is then compressed using JPEG codec (with a 75% quality rate) to include compression noise. We selected and applied these noise introductions and compression procedure so that the resulting images will more likely be representative of real camera-based images or video frames.

It can be observed that the proposed feature, which is based on short and connected vertical and horizontal edges, yields the highest recall rate. The computation cost of the proposed method is lower than the derivative texture algorithm and similar to the vertical edge based method.

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

Evaluation of text localization

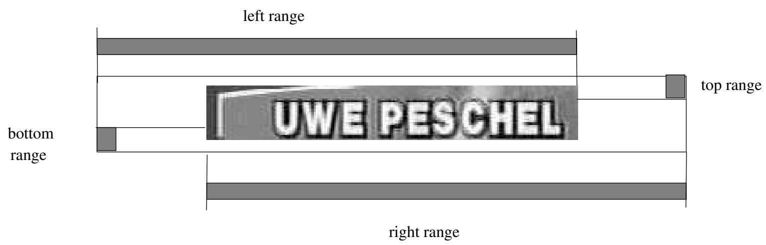


FIG. 4.13 – Valid baseline ranges : the shading parts indicate the valid baseline range.

The proposed text localization method composed of the fast text block extraction and baseline extraction is evaluated at region level on the databases the DB_S and DB_NewsVideo databases.

A text line is considered to be correctly located if it matches a ground-truth text region. A text region R in ground-truth is labeled as a rectangle with the top (R_t), the bottom (R_b), the left (R_l) and the right (R_r) boundaries. Let us define the top, bottom,left and right boundaries of a detected text line L as L_t, L_b, L_l and L_r. A match between the text line L and a ground-truth region R fulfills all the following conditions :

$$R_t - \frac{R_b - R_t}{5} < L_t < R_t + \frac{R_b - R_t}{5}$$

and

$$R_b - \frac{R_b - R_t}{5} < L_b < R_b + \frac{R_b - R_t}{5}$$

and

$$R_L - \frac{R_r - R_l}{4} < L_l < R_r$$

and

$$R_l < L_r < R_r + \frac{R_r - R_l}{4}$$

The area constrained in the above conditions is illustrated in Figure 4.13 using shadows.

The proposed method extracted 250 out of the 260 text lines and 78 false alarms in the DB_SceneImages database. It extracted all the 9369 text lines and 7537 false alarms in the DB_NewsVideo database. The obtained the RRR and RP are shown in Table 4.2.

database	RRR	RPR
DB_SceneImages	95.8%	23.8%
DB_NewsVideo	100%	55.4%

TAB. 4.2 – Performance of the proposed text localization method evaluated on the DB_SceneImages and DB_NewsVideo databases. RRR denotes the region recall rate and RPR denotes the region precision rate.

The method is tuned to optimize only the RRR because a higher precision is expected from the verification step. It is necessary to emphasis that the RPR criterion depends on the frequency of text strings appearing in the video. The results obtained on the DB_NewsVideo database can only give an idea about the region extraction precision in news videos.

4.3.3 Results of text verification

The text verification algorithms were trained and tested on the candidate text regions extracted by the text localization algorithm. We first train the classifiers (MLPs and SVMs) on the database DB_TrainImages using derivative features, distance map features, constant gradient variance features and DCT coefficients. The optimal combination of a feature and a classifier that yields the lowest error rate is selected from the experimental results obtained on the DB_TrainImages database. Then, this optimal model is applied on the DB_NewsVideo database to evaluate the verification algorithm at the region level.

The feature extraction, training and testing procedures described in Subsection 4.2 were applied on the DB_TrainImages 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. We equally divided the 76,470 vectors into a training set and a test set. It was ensured that the training sets of different features contained vectors extracted from the same windows (i.e. same image and location). The same was ensured for the the test set. Experiments are characterized by couples (classifier, feature) with two kinds of classifiers (MLPs and SVMs) and four kinds of features.

	Features			
Classifiers	DIS	DERI	CGV	DCT
MLP	5.28%	4.88%	4.40%	4.72%
SVM	2.56%	3.99%	1.07%	2.0%

TAB. 4.3 – Error rates of the SVM and MLP classifiers for the text verification task. DIS denotes the distance map features. DERI denotes the grayscale spatial derivative features. CGV denotes the constant gradient variance features. DCT denotes the DCT coefficients.

Table 4.3 lists the error rates measured on the test set for each feature/classifier combination. The error rate denotes the miss classification of both text and non-text feature vectors. First of all, we can see that these results are very good (less than 6% errors), taking into account the fact that they are based on a single feature vector. Secondly, whatever the considered feature, the SVM classifiers give better results than the MLP classifiers. This might be explained by the nature of the SVM classifiers, which optimizes a bound on the generalization error rather than the empirical risk. The experiments show that the RBF kernels provide an hyperplane as shown in Figure 4.14. This implies that the RBF kernel SVMs have a chance to make better generalization on unseen background regions.

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.

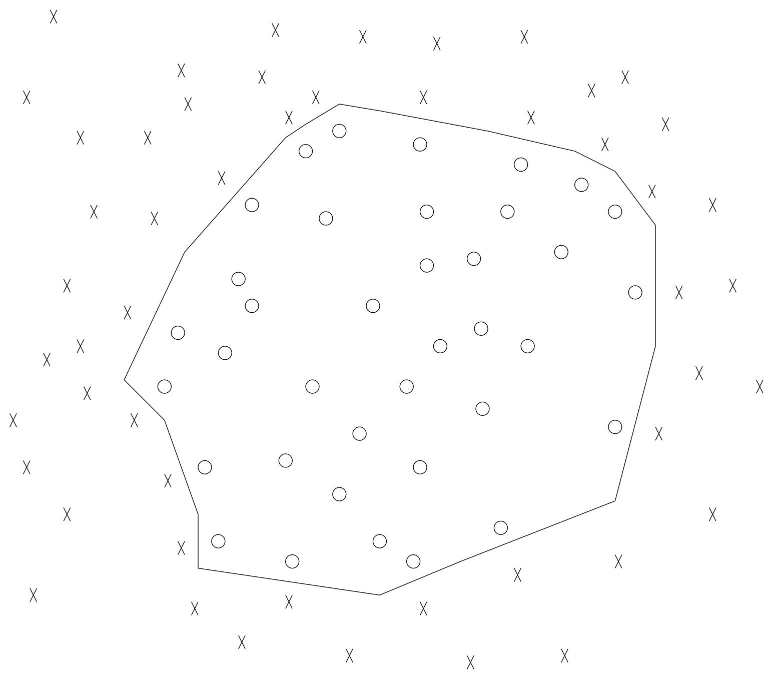


FIG. 4.14 – The decision surface of a RBF kernel based support vector machine.

4.3.4 Results of combining localization and verification

The performance of the whole localization/verification scheme was evaluated on the DB_NewsVideo database. The SVM classifier together with the CGV feature was employed to verify the candidate text regions based on the confidence value given by Eq. 4.20. This verification scheme removed 51 regions out of the 78 false alarms in the DB_SceneImages database and rejected 2 true text regions. In the DB_NewsVideo database, it removed 7255 regions of the 7537 false alarms while only rejecting 23 true text lines, leading to a 99.76% region recall rate (RRR) and a 97% region precision rate (RPR) as listed in Table 4.4. Figure 4.15 shows examples of detected text on some images in our databases.

database	RRR	RPR
DB_SceneImages	95.1%	94.7%
DB_NewsVideo	99.76%	97.0%

TAB. 4.4 – Performance of the localization/verification scheme on DB_NewsVideo database. RRR denotes the region recall rate and RPR denotes the region precision rate.

4.4 Conclusion of the text detection

In this chapter, we presented a scheme for detecting text in images and video frames using machine learning approaches. With the discussions and experiments in this chapter, we can have the following conclusions.

Some hard text detection problems can be addressed by two major steps : candidate text line localization and verification. Since there are obvious characteristics of text strings that can be exploited for localizing text, a low rejection rate text localization can be achieved in most cases. Then, the remaining problem is to distinguish between true text lines and false located regions after normalization, which is a little easier than detecting various sizes and grayscale values text in whole images. Experiments have shown that although the localization step may sometimes miss some text, mainly the scene text, the whole detection scheme still achieved more than 95% region recall rates.

The experiments have shown that machine learning classifiers can be efficiently developed and used for text detection in our localization/verification scheme. The size normalization and the grayscale value invariant feature extraction are the key steps to obtain good classifiers.

Although we collect large number of images and video frames in for training the classifiers, the text verification is still database dependent. To obtain a good model for new data in a real application, especially for news videos from different TV station, it may require to re-train the model.

FIG. 4.15 – Some examples of detected text regions in images or video frames.

Chapitre 5

Text recognition

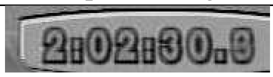
As an extension of conventional OCR system, text recognition in images and video sequences aims at recognizing the characters or words in the detected text regions. The detected text regions are usually local image regions of rectangle shapes as shown in Figure 5.1. Under each text region we have displayed the text recognited by an OCR software (RTK[1]). Obviously, these recognition results are not acceptable for content-based multimedia applications.

The main reason leading to these bad results is the poor segmentation quality of the text images. Traditional OCR systems recognize characters using the shape information of characters. In the case of characters printed against clean papers, the shape information of these characters is easy to obtain using binarization algorithms. More formally, we can say that a recognition system requires the characters to be somehow segmented from the background at first. Most commercial OCR systems are applying a binarization algorithm to the input image. When applied on the text images extracted from images and video frames, just like the images shown in the Figure 5.1, this binarization algorithm usually can not segment the characters well. Therefore this leads to poor recognition results.

Figure 5.2 shows the two schemes, bi-modal text enhancement approach and multi-modal segmentation approach that we propose and evaluate in this chapter to address the recognition issue. On the left of the figure, we illustrated the conventional bi-modal method, which directly binarize an input text image and apply the OCR software for the recognition. The term bi-modal refers to that the algorithm models the grayscale values of a text image with two segmentation layers. In the middle, the figure shows our bi-modal text enhancement approach. It aims at enhancing the contrast between text characters and background objects so that they are easier to be segmented by a binarization algorithm. Alternatively, multi-modal approaches are proposed to improve text image segmentation qualities by modeling the grayscale values of a text image with multiple segmentation layers. OCR is applied on every segmentation layer from specified modals that is referred to as segmentation hypothesis. The final recognition result is then selected from the OCR output results of multiple segmentation hypotheses. Both bi-modal and multi-modal scheme may produce false segmented regions. These false segmented regions are mixed with real character regions and often lead to poor recognition results of the OCR software. Therefore, post-segmentation processing is necessary to remove the segmented regions that are obviously not characters.

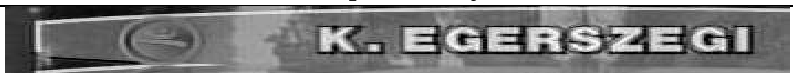
[1]OCR engine produced by EXPERVISION

output nothing



output nothing

output nothing



“m ”

output nothing

“./” PRESIDENT DU COMiTE PARiTAIRE”

output nothing

“KaboutI”

FIG. 5.1 – Text regions extracted by using the text detection approach described in the last chapter. The recognition results displayed under the text regions are obtaining by using a commercial OCR software (RTK Expervision).

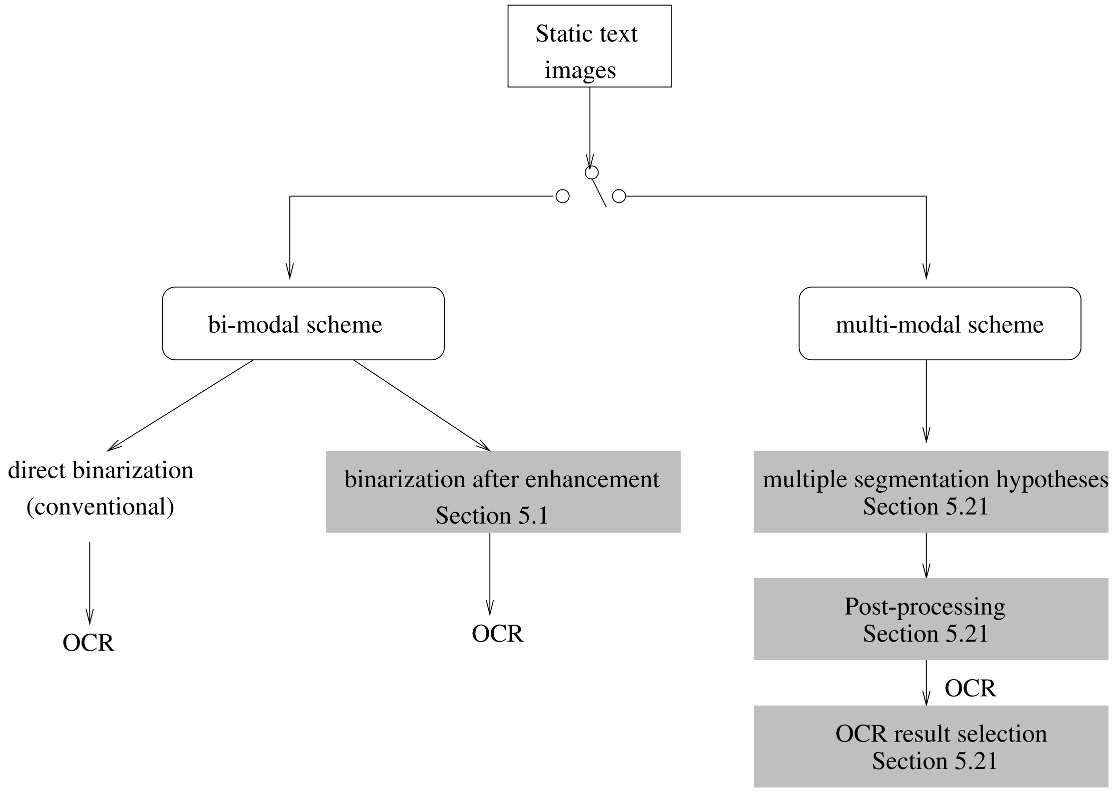


FIG. 5.2 – Structure of schemes and related methods in this chapter.

5.1 Bi-modal text enhancement

To obtain better binarization results, text enhancement approaches can be applied to selectively enlarge the contrast between characters and backgrounds.

In [120], Wu simply smoothes the detected text region using a Gaussian low pass filter. Smoothing eliminates noise from both text characters and backgrounds. However, it also blurs the characters with the background. Therefore, this method is not very effective on text in complex background, such as video text. Sato [93] [94] enhances the text on the basis of the line elements of characters. To this end, he used filters with four orientations : vertical, horizontal, left diagonal and right diagonal. However, because real scales are unknown, it is difficult to design a good enhancing filter that can be applied successfully on the line elements of characters with different stroke widths.

Noticing that stripes are common sub-structures of text characters, we propose a scale adaptive stripe filtering method for enhancing text characters using the orientation and scales of local sub-structures (stripes). We first locate the sub-structures of the text using an edge detection algorithm and estimate the orientation and scale of each sub-structure using a family of filters, which are derived from Gabor filters. We then select three scale ranges and enhance the contrast of these sub-structures as character strokes in each scale range individually to improve the performance of binarization.

5.1.1 Stripe localization

Stripes, the common sub-structures of text characters, usually form strong edges against their background. To find candidate character stripes, a Canny edge operator is first employed to detect the strong edges. In practice, close points associated with the same edge have similar scales. The sharp orientation changes exist mostly at corners, which are small areas and can be omitted in the enhancement. Therefore, to reduce the computation, the image is segmented into $n \times n$ blocks, where n is set equal to half the size of the smallest thickness of the substructures of the text (here $n = 2$). We keep only one edge point in each block, which has the highest gradient magnitude to perform the orientation and scale estimation. Commonly, the number of resulting candidate points is less than 10% of the sum of all pixels in one image. This reduces the number of pixels to be processed in the next step, yielding a more efficient algorithm.

5.1.2 Asymmetric Filter

The orientations and scales of the pixels on the edge of a stripe can be estimated by applying band selective filters, for example the Gabor filters. The family of two-dimensional Gabor filters $G_{\lambda,\theta,\varphi}(x, y)$, which was proposed by Daugman [17], are often used to obtain the spatial frequency of the local pattern in an image :

$$\begin{array}{rcl} G_{\lambda,\theta,\varphi}(x,y) & = & \cos\left(2\pi\frac{x'}{\lambda}+\varphi\right)e^{-\frac{\left(x'^2+\gamma^2y'^2\right)}{2\sigma^2}} \\ x' & = & x\cos\theta + y\sin\theta \\ y' & = & -x\sin\theta + y\cos\theta \end{array} \tag{5.1}$$

where the arguments x and y represent the pixel coordinates, φ is a phase parameter, σ is the variance of Gaussian for smoothing, parameter θ specifies the orientation of the filter, the constant γ determines the spatial aspect ratio, and λ is called the spatial bandwidth. Applying these filters on a given stripe, the highest filter response is given by the filter with the optimal orientation and the optimal spatial frequency of a local image region. However, the highest responses are always located on the center axle of the stripe. The conventional Gabor filter responses on the edge pixels since are close to zero.

In order to perform scale and orientation estimation directly on edge points, we introduce two groups of Gabor-based asymmetric filters : edge-form filters and stripe-form filters. The edge-form filters $E_{\lambda,\theta}(x, y)$ are the Gabor filters with $\varphi = \pi/2$:

$$
\begin{aligned}
E_{\lambda,\theta}\left(x,y\right) &= \cos\left(2\pi\frac{x^{'}}{\lambda}+\frac{\pi}{2}\right)e^{-\frac{\left(x^{'2}+\gamma^{2}y^{'2}\right)}{2\sigma^{2}}} \\
&= \sin\left(2\pi\frac{x^{'}}{\lambda}\right)e^{-\frac{\left(x^{'2}+\gamma^{2}y^{'2}\right)}{2\sigma^{2}}} \\
x^{'} &= x\cos\theta+y\sin\theta \\
y^{'} &= -x\sin\theta+y\cos\theta
\end{aligned}
$$

The stripe-form filters $S_{\lambda,\theta}\left(x,y\right)$ are defined as a Gabor filter with a translation $(t_x,t_y) = \left(-\frac{\lambda}{4},0\right)$:

$$
\begin{aligned}
S_{\lambda,\theta}\left(x,y\right) &= \cos\left(2\pi\frac{x'}{\lambda}\right)e^{-\frac{\left(x^{'2}+\gamma^{2}y^{'2}\right)}{2\sigma^{2}}} \\
x^{'} &= x\cos\theta+y\sin\theta+t_x \\
y^{'} &= -x\sin\theta+y\cos\theta+t_y
\end{aligned}
$$

This additional translation makes the original Gabor filters to be asymmetric. The reason is that when applying these asymmetric filters to a stripe with optimal orientation and scale, the highest filter responses are located on edge points instead of on the middle axle. Therefore, we can save the computation of estimating the scale of the stripes by only apply my stripe-form filters on the edge points.

Figure 5.3 shows the pattern of the edge-form filters (Fig. 1a) and stripe-form filters (Fig. 1bc) for 8 orientations with $\gamma = 0.92$.

Applying the stripe-form filters on a given edge point x_0, y_0 of a character stripe in the input image I, we can obtain the highest filter response at the optimal θ^* and λ^* :

$$(\theta^*,\lambda^*)_{(x_0,y_0)} = \arg\max_{\theta,\lambda} I \star S_{\lambda,\theta}(x_0,y_0) \tag{5.2}$$

The parameter θ^* corresponds to the orientation of the stripe and λ^* indicates the thickness of the stripe.

5.1.3 Orientation estimation

To have a fast searching of these two optimal parameters, we found that the orientation parameter can be first estimated from the output of the edge-form filters :

$$\theta^*_{(x_0,y_0)} = \arg\max_{\theta} I \star E_{\lambda,\theta}(x_0,y_0) \tag{5.3}$$

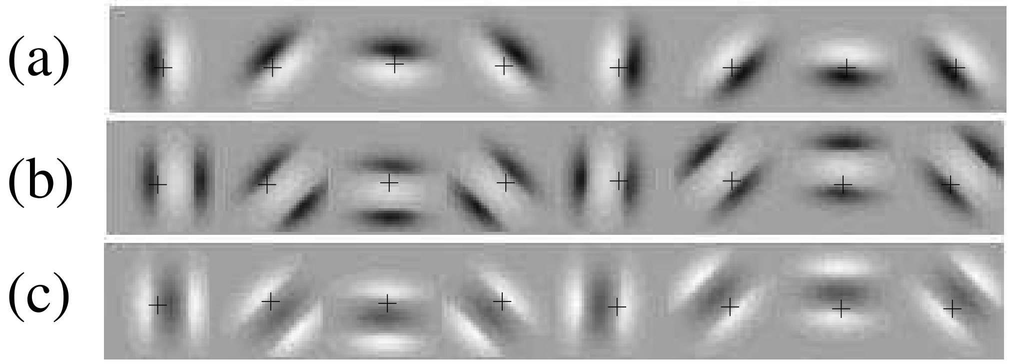


FIG. 5.3 – Asymmetric filters in 8 orientations with $\gamma = 0.92$: (a) edge-form filters , (b) stripe-form filters

In other words, the optimal orientation of an edge point of a potential stripe is computed by maximizing the edge-form filtering at the point with respect to the orientation θ. The bandwidth or thickness λ is set to be the smallest value for the sake of computation complexity (i.e. $\lambda = 2$). The resulting optimal orientation is denoted as θ^*.

5.1.4 Thickness estimation

Theoretically, the optimal thickness of an edge point of a potential stripe can be obtained by maximizing the stripe-form filtering 5.2 at the point with respect to the thickness factor λ through the optimal orientation θ^* :

$$\lambda^*_{(x_0,y_0)} = \arg\max_{\lambda} I \star E_{\lambda,\theta^*}(x_0, y_0) \tag{5.4}$$

In practice, this maximization can be done by a fast procedure using both edge and stripe-form filters instead of performing a full search. The key of this fast procedure is to find a good initial thickness λ, which is close to the optimal thickness λ^*. The idea comes from the filters responses on the ideal case. We apply edge-form and stripe-form filters with different λ and known optimal orientation ($\theta = 0$) to a vertical bar image. Figure 5.4 illustrates the responses of these filters. The results show that if the pixel (x, y) is on the edge of a stripe structure, the responses of the stripe-form filters are smaller than the responses of the edge-form filters when the scale of the filters are rather smaller than the scale of the stripe ($S_{\lambda,\theta}(x, y) < E_{\lambda,\theta}(x, y)$), but greater when the thickness of the filters are rather larger than the thickness of the stripe ($E_{\lambda,\theta}(x, y) < S_{\lambda,\theta}(x, y)$.

The thickness at which the stripe-form filter response intersects the edge-form filter response is called the intersection point. This intersection point $\overline{\lambda}$ is close to the optimal thickness λ^*. We

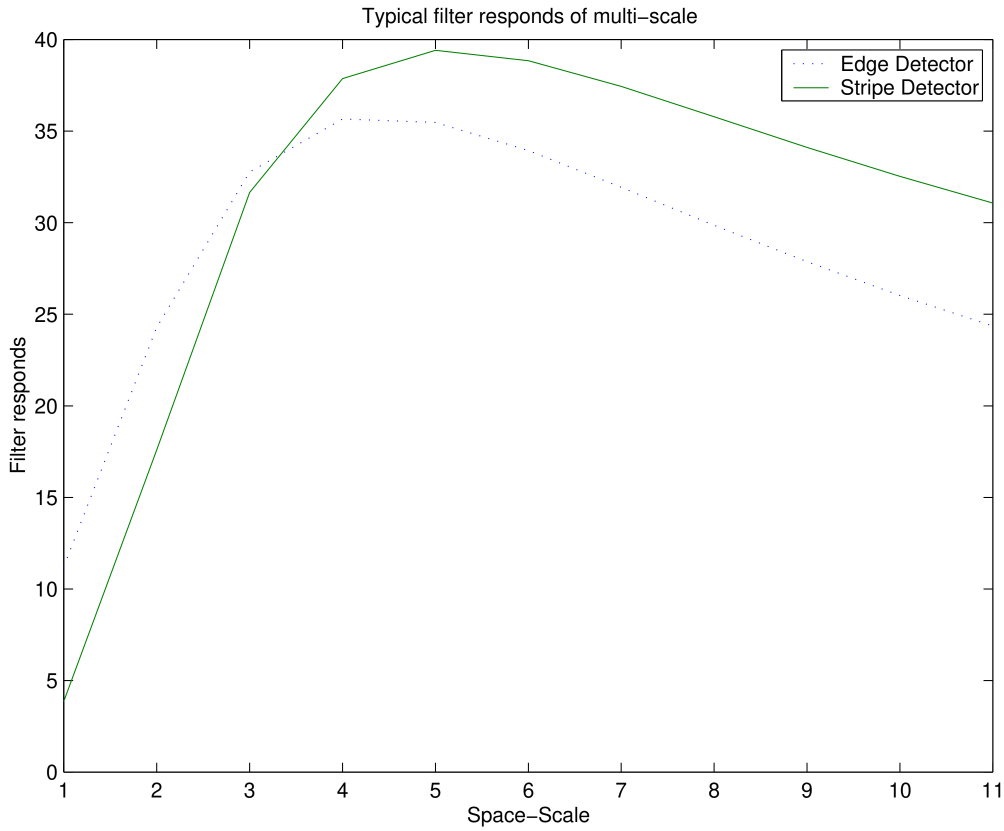


FIG. 5.4 – Responses of asymmetric filters on the edge of a vertical bar image.

thus initialize thickness $\lambda = \overline{\lambda}$. The intersection point $\overline{\lambda}$ can be located using a binary search method, which is described below :

1) At the given point x, y, initialize the searching range of thickness $[\lambda_1, \lambda_2]$; For example the searching range can be $[6, 30]$ in a text image normalized to be 100 pixels high.
2) Compute the middle thickness in the searching range $\overline{\lambda} = \frac{\lambda_1+\lambda_2}{2}$;
3) Compute the edge-form and stripe-form filtering responses $E_{\overline{\lambda},\theta^*}(x, y)$ and $S_{\overline{\lambda},\theta^*}(x, y)$;
4) If $(S_{\overline{\lambda},\theta^*}(x, y) < E_{\overline{\lambda},\theta^*}(x, y))$ then $\lambda_1 = \overline{\lambda}$, and goto step 2.
5) If $(S_{\overline{\lambda},\theta^*}(x, y) > E_{\overline{\lambda},\theta^*}(x, y))$ then $\lambda_2 = \overline{\lambda}$, and goto step 2.
6) If $(\mid \lambda_1 - \lambda_2 \mid < \varepsilon)$ return $\overline{\lambda}$.

Then, the Eq. 5.4 can be maximized using a full search in the rang of $[\overline{\lambda}-\tau, \overline{\lambda}+\tau]$. Experimentally, we found that $\tau = \frac{\overline{\lambda}}{3}$ is good for most of the cases.

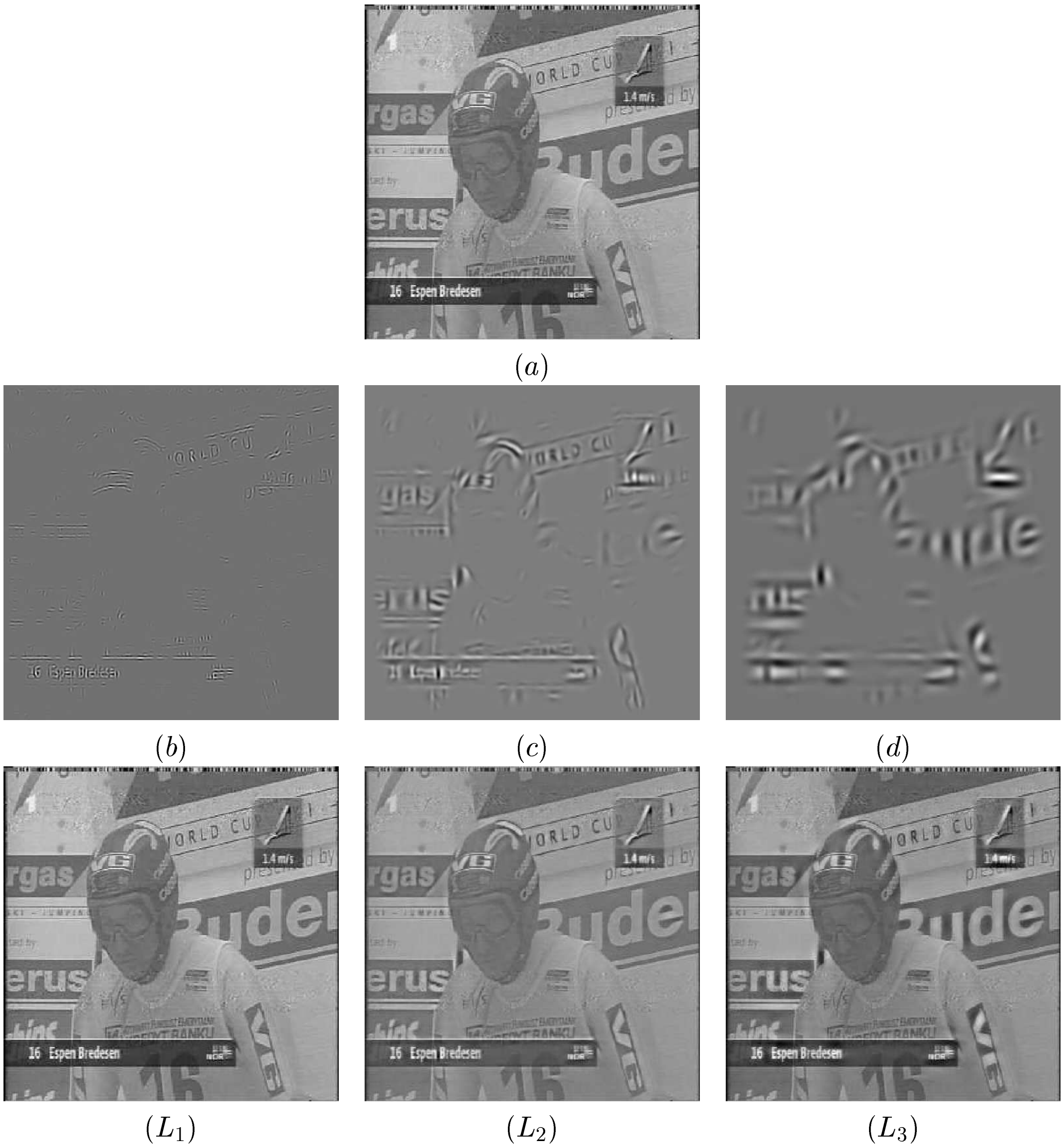


FIG. 5.5 – (a) is original frame image ; (b, c, d) are reconstructed image patterns in 3 scale ranges ; (L_1, L_2, L_3) are enhanced images in 3 scale ranges.

5.1.5 Enhancement

The estimated orientations and thickness of the stripes can be used to enhance the input video frame before the text is detected so that the enhanced image can benefit the text segmentation. The enhanced image I_e of a given text image I is defined as :

$$I_e(x,y) = I(x,y) + \alpha\, I_r(x,y)$$

where α is a weight parameter of the enhancement. We found that the weight $\alpha = 0.8$ is a good value. The enhancing image I_r is defined as :

$$I_r = \sum_{all\, edge\, points\, (x_e,y_e)} S_{\theta^*_{(x_e,y_e)}\lambda^*_{(x_e,y_e)}(x-x_e,y-y_e)}$$

To illustrate the result of this enhancement procedure on an original image, we applied the proposed enhancement algorithm on the image of Fig. 5.5 (a). Since we do not normalize the image, the enhancements are performed in three different thickness ranges L_1, L_2, and L_3, by set them as initial searching range in the proposed algorithm. The first thickness range $L_1 = [\lambda_1, \lambda_2] = [3, 9]$ enhances the small stripes. The second thickness range $L_2 = [\lambda_1, \lambda_2] = [7, 30]$ enhances the middle size stripes and the last range $L_3 = [\lambda_1, \lambda_2] = [26, 50]$ targets large characters. The enhancing images I_r in these three ranges are shown in figure 5.5 (b,c,d). The corresponding enhanced images are displayed below them.

5.1.6 Performing character recognition on enhanced images

We now apply the bi-modal enhancement method to recognize characters in text images. The detected text images are first normalized to the height of 100 pixels with bilinear interpolation. We then enhance the images at the thickness range of $[6, 30]$. The enhanced images are binarized using Otsu's thresholding method [75]. Before applying the OCR, we employed a connected component analysis algorithm to remove non-character regions from the binarized images. This algorithm is discussed in the Section 5.2.2. Figure 5.6 shows the result of direct binarization of a text image, enhanced image and the binarization of this enhanced image. The character recognition results obtained using a commercial OCR package (RTK) are displayed under the two binary images. The character recognition performance of this bi-modal enhancement method is reported in the Section 5.3.2 on the basis of large databases.

5.2 Multiple hypotheses segmentation scheme

The enhancement technique aims at improving the binarization result of text images 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

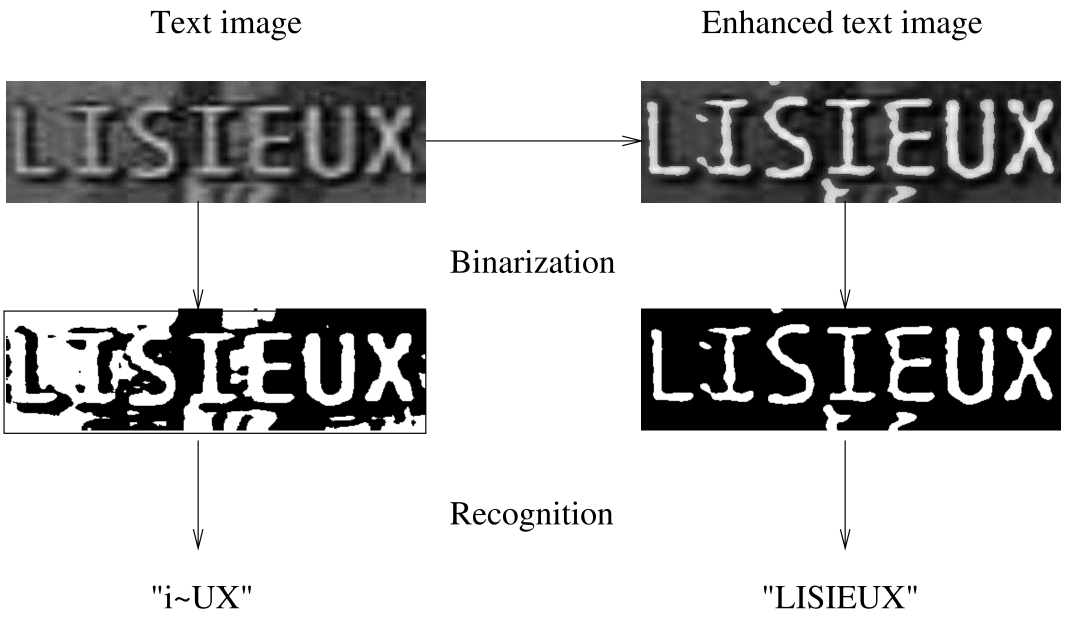


FIG. 5.6 – Illustration of the text enhancement algorithm. The recognition results of original text image and enhanced text image are displayed under the two binarized image.

grayscale value of text may not be known in advance. These problems are illustrated in Figure 5.8 by examples of detected text images .

To handle the problems encountered with the bi-modal assumption, multiple segmentation results of a given text image can provide complementary information for the character recognition system and therefore have potential possibilities to improve the recognition results. Figure 5.9 describes the multi-hypotheses approach we propose for segmenting and recognizing characters in images. In this approach, 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 Figure 5.10). Therefore, if we have K different labels, we get K hypotheses. The choice of K is an important and difficult issue. One general way to address this problem consists in checking whether the increase in model complexity really provides a better fit of the data, using the minimum description length criterion for instance. However, this information theoretic approach may not be appropriate for qualifying good text segmentation. Therefore, we used 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.

Three different segmentation algorithms have been tested. They are described in the next subsection. The post-processing steps and the result selection algorithm are presented afterwards.

(a)
(b)


(a)
(b)


FIG. 5.7 – Illustration of the results of applying the text enhancement algorithm on multi-model text images. (a) original images ; (b) enhanced images.

5.2.1 Segmentation algorithms

The gray-level segmentation of an image can be formally defined as a labeling problem. Let S denote the set of sites (pixels), and o be an assignment of an observation field O, $o = \{o_s, s \in S\}$, where o_s corresponds to the gray-level value at site s. We model the image intensities in terms of the combination of K simple random processes, also referred to as layers. Each simple process is expected to represent regions of the image having similar gray levels, one of them being text. Thus, the segmentation consists in the mapping of pixels to processes. It is stated as a statistical labeling problem, where the goal is to find the assignment e of a label field E, $e = \{e_s, 1 \leq e_s \leq K, s \in S\}$ that best accounts for the observations, according to a given criterion.

To perform the segmentation, we proposed 3 algorithms :

1. Standard EM algorithm, a maximum likelihood classifier with a standard EM procedure for the class parameter estimation.
2. Gibbsian EM algorithm, a Maximum a Posteriori classifier with a prior (Markov) model on the local field, and a Gibbsian EM algorithm estimation.
3. k-means algorithm.

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. $\mathrm{p}_i(o_s) = \mathcal{N}(\mu_i, \sigma_i)$. In the third case, the gray values of pixels in a given image are classified to minimize the intra-class variance while maximizing the inter-class variance.

The standard EM algorithm

In this algorithm, the K individual processes are combined into a probabilistic mixture model according to :

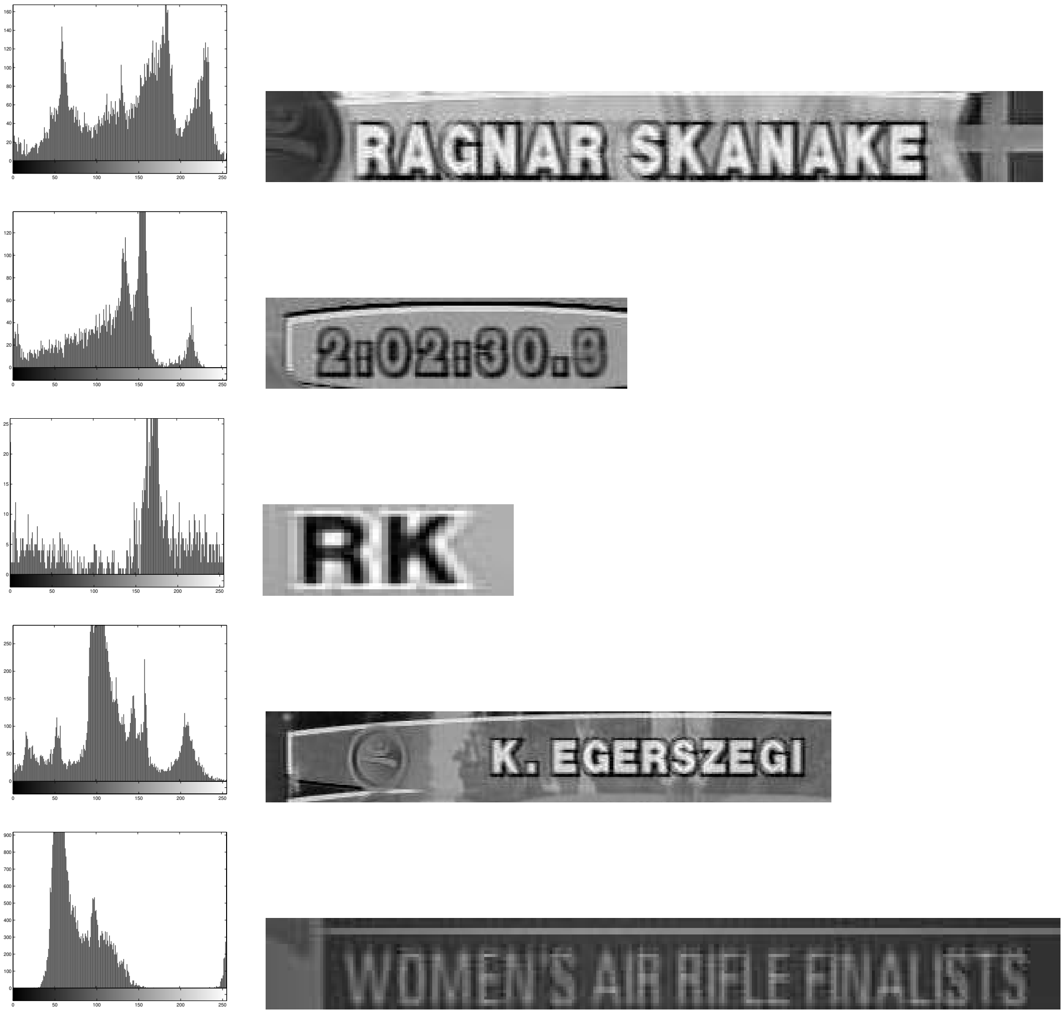


FIG. 5.8 – Examples of detected text images and their histograms. They illustrate the complex backgrounds and multi-modal grayscale value distributions that can be encountered in text images.

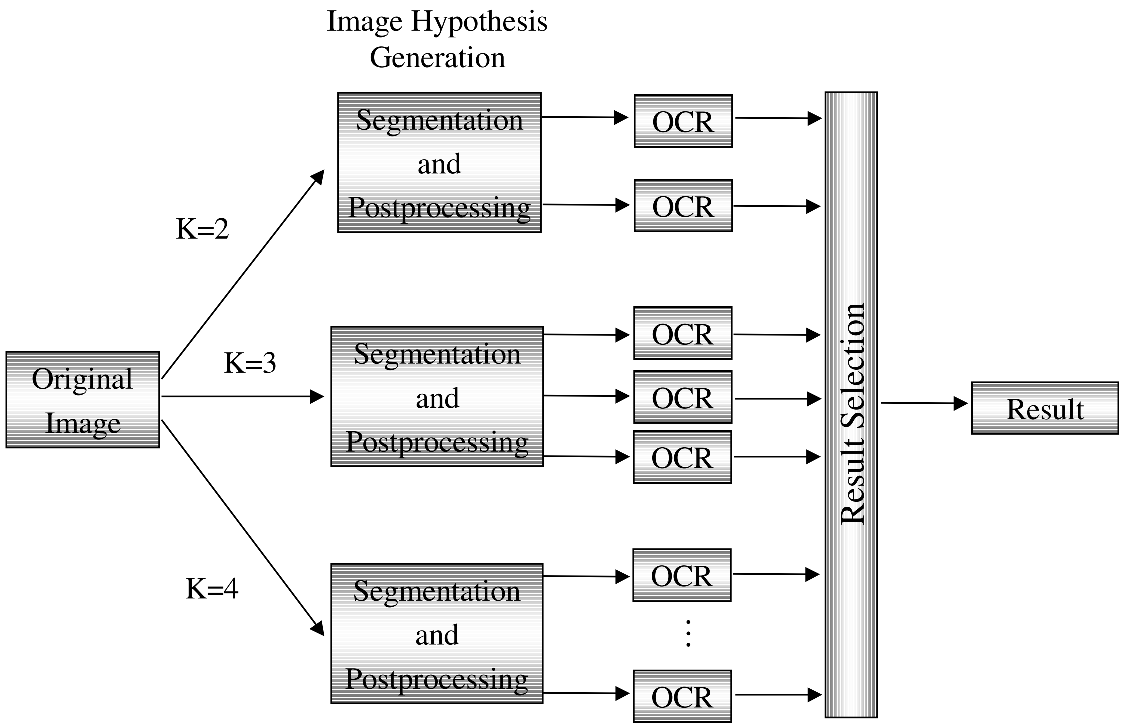


FIG. 5.9 – The multiple hypotheses text segmentation and recognition scheme.

$$\mathrm{p}(o_s) = \sum_{k=1}^{K} \mathrm{p}(o_s|e_s = k)\mathrm{p}(e_s = k) = \sum_{k=1}^{K} \pi_k \mathrm{p}_k(o_s) \tag{5.5}$$

Given an image, the goal is thus to find the set of parameters (φ, π)=$(\mu_i, \sigma_i, \pi_i)_{i=1...K}$ maximizing the likelihood of the data set o defined as :

$$\ln \mathrm{p}(o|\varphi) = \sum_{s \in S} \ln \mathrm{p}(o_s|\varphi) \tag{5.6}$$

Using the standard "expectation-maximization" (EM) algorithm (in the Chapter 2 [20]), the expected log-likelihood of the complete data (i.e., observations and labels) can be iteratively maximized with respect to the unknown data (the labels). After maximization, the labels can be assigned according to the resulting optimal parameters φ^* to the most probable layer according to the Maximum likelihood rule :

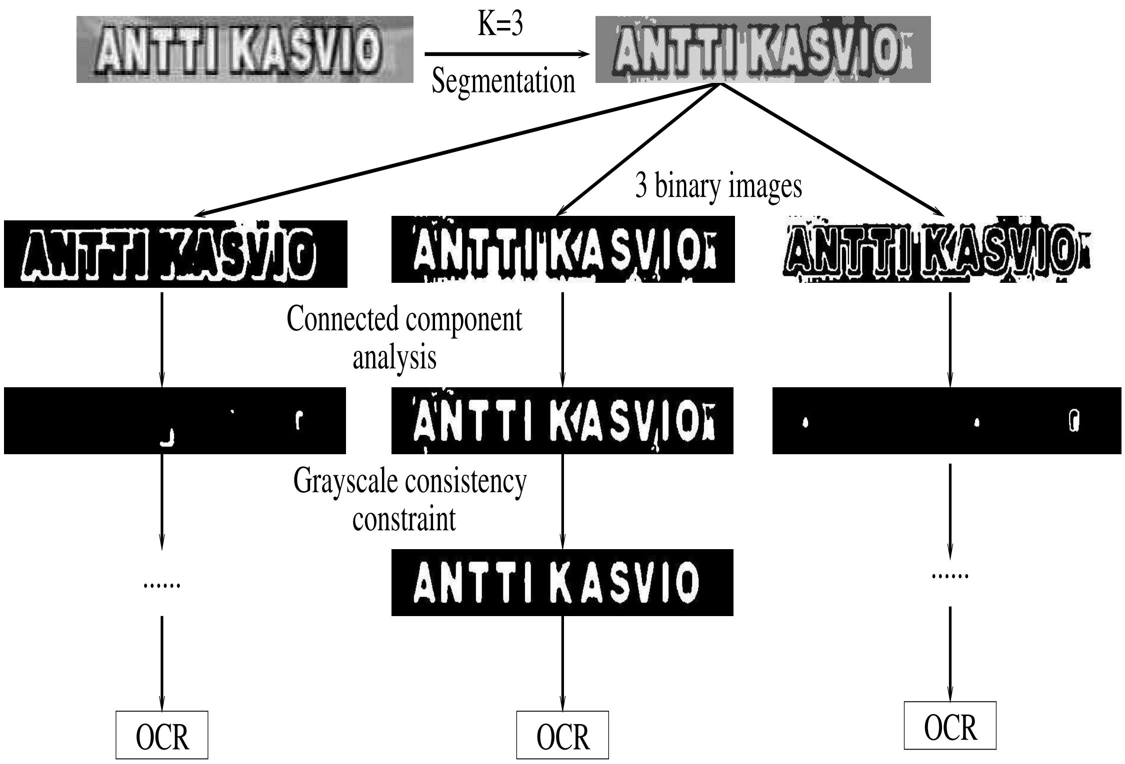


FIG. 5.10 – Segmentation and post-processing steps in the multiple hypotheses segmentation scheme.

$$\forall s \qquad e_s = \operatorname*{argmax}_i \ \mathrm{p}_i(o_s|\varphi*) \tag{5.7}$$

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 E as a Markov Random Field (MRF). Therefore, the goal of segmenting an image (an observation) o is to search for the most a posteriori probable configuration e^* that give rise to o. Following the discussion of the section 2.3.2, it is equivalent to minimizing an energy function :

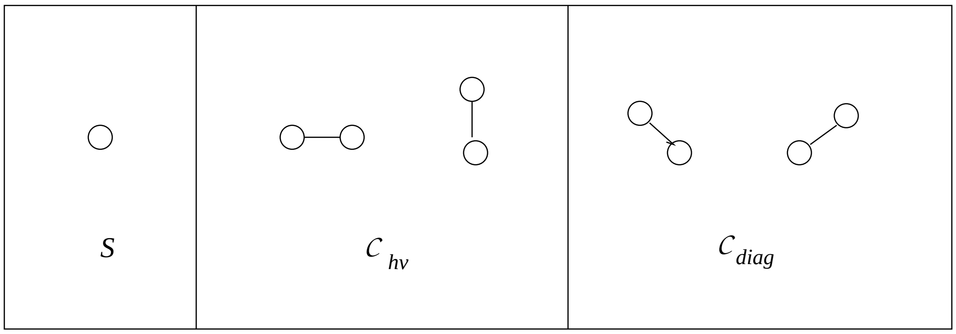


FIG. 5.11 – Neighbors defined in GBEM algorithm.

$$e^* = \arg\max_{e \in \Omega_E} \; p(E = e|O = o) \doteq \arg\min_{e \in \Omega_E} \; \left(U_1^{\beta}(e) + U_2^{\varphi}(e, o) \right) \tag{5.8}$$

with

$$U_2^{\varphi}(e, o) = \sum_{s \in S} (- \ln p(o_s|e_s)) \tag{5.9}$$

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 :

$$U_1^{\beta}(e) = \sum_{s \in S} V_{11}(e_s) + \sum_{<s,t> \in \mathcal{C}_{hv}} V_{12}^{hv}(e_s, e_t) + \sum_{<s,t> \in \mathcal{C}_{diag}} V_{12}^{d}(e_s, e_t) \tag{5.10}$$

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, as shown in Figure 5.11. The V are the (local) interaction potentials which express the prior on the label field. For instance, if we have :

$$V_{12}^{hv}(k, l) = V_{12}^{hv}(l, k) > V_{12}^{hv}(l, l) \qquad \forall l, \forall k \neq l \tag{5.11}$$

the prior will favor label fields where horizontal and vertical neighbor sites have similar labels. For another example, if we assume all the layers to be spatially homogeneous, we can select the potentials according to :

$$\forall i, V_{11}(i) = 0 \qquad \forall k, j \;\; V_{12}^{hv}(k, j) = \beta_{hv}(1 - \delta_{k=j}), \;\; V_{12}^{d}(k, j) = \beta_d(1 - \delta_{k=j}) \tag{5.12}$$

where δ denotes the kronecker function. β_d and β_{hv} therefore represent the costs to have different labels at neighboring pixels in the corresponding direction.

The method of minimization of the energy $U(e, o) = \left(U_1^{\beta}(e) + U_2^{\varphi}(e, o) \right)$ can be found in Chapter 2. We first train the parameters (φ, β) and then obtain the segmentation using MAP. The complexity of this GBEM algorithm is approximately 4 times greater than the complexity of the basic EM algorithm.

The K-means 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 [75], that is :

$$IV = \sum_{i=1}^{K} \sum_{s \in S_i} \|o_s - \mu_i\|^2 = \sum_{s \in S} \|o_s - \mu_{e_s}\|^2 \tag{5.13}$$

where μ_i is the mean value of pixels in each layer.

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

5.2.2 Post-processing

The goal of the post-processing step is to remove false (non-character) regions in an given segmented text image (a binary image) before applying OCR on it. The existence of the false regions, such as noise points and bar-style background regions, lead the OCR software to errors of locating or segmenting text characters. Therefore, the OCR software produces bad recognition results. In this section, we propose two approaches to remove false regions. One approach is based one the analysis of connected component regions. In this approach, simple measures of the shape of a region are exploited to distinguish characters from false regions. The other approach is based on a grayscale consistent constraint. It rejects connected regions whose pixel grayscale values are different from that of the majority of pixels in the original image. These two approaches are discussed in the following two sections.

Connected component analysis

A simple connected component analysis is used to eliminate non-character regions in each hypothesis to help the OCR system. We exploit some simple measures of the shape of a given region (connected component) and set constraints according to the statistical results of real characters in images and video frames. All the input text images are normalized to the same height (see Fig. 5.8), which is 100 pixels. We only keep connected component that satisfies the following constraints :

1) The size of the connected component is bigger than 120 pixels, which is the minimum size of a “dot” in the character “i” or “j” ;
2) The width/height ratio of the connected component is between 2.5 and 0.1, which is set according to the character “i” and “W” in different fonts ;
3) The width of the connected component is less than 2.1 the height of the whole text region, which is also set according to the character “W” in different fonts.

Grayscale consistent constraint

Since we only consider 2 to 4 layers in the segmentation, regions from the background with a gray value slightly different from that of characters may still belong to the text layer/class.

Unfortunately, the rather crude connected component analysis described previously is unable to always eliminate such regions. 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 Figure 5.10).

After the connected component analysis step, which is especially useful in eliminating rather large and elongated non-text regions, we estimate with a robust estimator the gray level mean m^* and standard deviation st^* of the set S_r of sites belonging to the remaining regions. More precisely, we first compute :

$$m_1 = \arg\min_m Med_{s\in S_r}|o_s - m| \tag{5.14}$$

and

$$st_1 = 1.48 \times (1 + \frac{5}{|S_r| - 1}) \times r_1 \tag{5.15}$$

where

$$r_1 = Med_{s\in S_r}|o_s - m_1| \tag{5.16}$$

Med denotes the median operator [90]. Valid pixels are then identified by checking that their gray value is within a given range of m_1, that is :

$$\frac{|o_s - m_1|}{st_1} < k \tag{5.17}$$

with k a given ratio (we use a value of 2). Then the mean m^* and standard deviation st^* are computed from the valid pixels only, using the usual formula. 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 :

$$\frac{|o_s - m^*|}{st^*} > k \tag{5.18}$$

An illustration of the result of this step is displayed in Fig. 5.12.

5.2.3 OCR and result selection

The selection of the result from the set of strings generated by the segmentation processes (see Figure 5.9) is based on a confidence value evaluation relying on language modeling. The confidence value should account for both the measurement quality and the prior on the strings' characters.

FIG. 5.12 – Applying grayscale consistency constraint : 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.

From a qualitative point of view, we can notice that, 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 us denote by a string $T = (T_i)_{i=1..l_T}$ where l_T denotes the length of the string and each character T_i is an element of the character set $\mathcal{T}$:

$$\mathcal{T} = (0, \ldots, 9, a, \ldots, z, A, \ldots, Z, G_b) \tag{5.19}$$

in which G_b corresponds to any other garbage character. Then, 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. The confidence value is estimated using a variant of the log-likelihood ratio :

$$C_v(T) \quad = \quad \log\left(\frac{\mathrm{p}(H_a|T)}{\mathrm{p}(H_n|T)}\right) + l_T\, b \tag{5.20}$$

where b is a bias that is discussed below. Using the Bayers rule, the probability of the accurate segmentation given the string T is equal to :

$$\mathrm{p}(H_a|T) = \frac{\mathrm{p}(T|H_a)\mathrm{p}(H_a)}{\mathrm{p}(T)} \tag{5.21}$$

The probability of the noise segmentation given the string T is equal to :

$$\mathrm{p}(H_n|T) = \frac{\mathrm{p}(T|H_n)\mathrm{p}(H_n)}{\mathrm{p}(T)} \tag{5.22}$$

Here, we assume an equal prior on the two hypotheses :

$$\mathrm{p}(H_a) = \mathrm{p}(H_n) \tag{5.23}$$

Therefore, the confidence value is equal to :

$$C_v(T) \;=\; \log\left(\,\mathrm{p}(T|H_a)\,\right) - \log\left(\,\mathrm{p}(T|H_n)\,\right) \,+\, l_T\, b \tag{5.24}$$

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], which contains a huge mount of text of books. A bi-gram model was selected. Cutoff and backoff techniques [44] 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 uni-gram model was used because the size of the background data set was insufficient to obtain a good bi-gram 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 finally defined as :

$$\begin{aligned} C_v(T) \;=\; & \log \mathrm{p}(T_1|H_a) + \textstyle\sum_{i=2}^{l_T} \log \mathrm{p}(T_i|T_{i-1}, H_a) \\ & - \textstyle\sum_{i=1}^{l_T} \log \mathrm{p}(T_i|H_n) + 0.7\, l_T \end{aligned} \tag{5.25}$$

5.3 Experiments and results on stationary images

5.3.1 Criterions for evaluating the recognition performance

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

$$CRR = \frac{N_r}{N} \quad \text{and} \quad CPR = \frac{N_r}{N_e} \tag{5.26}$$

where 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

[2] www.gutenberg.net

characters is computed using an edit distance[3] between the recognized string and the ground truth[4]. 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. 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, the word recognition rate is defined as the percentage of words from the ground truth that are recognized in the string results (it doesn't need any dictionary).

5.3.2 Performance of the bi-modal enhancement algorithm

We tested the text enhancement algorithm on the databases DB_TrainImages, DB_SceneImages and DB_Temporal. From DB_TrainImages we extract 1208 images containing 9579 characters and 2084 words, which are randomly selected. Most of the text characters in this set have artificial contours. These characters have various grayscale values. For the DB_Temporal database, which contains 247 sequences of text images, we only keep the image that is in the middle of each sequence as the key images for testing.

Test set	images	characters	CRR direct binaization	CRR of enhenced
DB_TrainImages	1208	9579	92.5%	92.6%
DB_SceneImages	50	2863	47.7%	70.9%
DB_Temporal	247	2899	87.8%	88.0%

TAB. 5.1 – Recognition results of three test sets using bi-modal algorithms. CRR : character recognition rate.

In all the experiments, we performed the post-processing with only the connected component analysis step before applying the OCR software. Otherwise, the OCR will yields very poor recognition results because of false regions in the binary images. The results show that the proposed enhancement algorithm can improve the DB_SceneImages database in terms of the character recognition rate. The proposed two groups of asymmetric Gabor filters can efficiently extract the orientations and thickness of the stripes of characters. Using the resulting orientations and thickness the contrast of stripes can be selectively enhanced and therefore lead to better recognition results.

However, the results also show that this bi-modal text enhancement algorithm is less helpful for the images of the DB_TrainImages and DB_Temporal databases. There are two main reasons for this. Firstly, in these two test sets, characters mostly have contour or graphic backgrounds (see Fig. 3.2 and Fig. 3.5. The bi-modal enhancement is not working well on these contour characters

[3]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.

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

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

because there are also many stripes structures in the contours themselves. The algorithm makes wrong decisions and enhances the contour instead of the character stripes. Secondly, the grayscale values of these characters are not only white and black but any grayscale values. The enhancement algorithm can enhance the contrast of the text images. However, the images remain multimodal. In the bi-modal framework, these images still can not be correctly segmented. To obtain a better solution for recognizing characters in any grayscale values, we will introduce a multi-modal framework in the next section.

5.3.3 Performance of the multiple hypotheses scheme

Since the DB_TrainImages database contains characters in various grayscale values, we first perform multiple hypotheses recognition scheme for evaluating the three segmentation algorithms proposed in the Section 5.2.1 and the post-processing methods CCA and GCC on the test set 2. Figure 4.15 shows some text images in the DB_TrainImages database. We can see that the grayscale value of characters is not always the highest (or lowest).

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 post-processing step. This will serve as a baseline for the rest of the experiments. Table 5.2 lists the results obtained with the three described segmentation methods. It can be seen that the usual bi-modality (K=2) hypothesis yield the best character and word recognition rate with the GBEM and Kmeans algorithms. This may explain why most of previous efforts are made in improving binarization 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 5.8). 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 character recognition rate and word recognition rate. This is probably due to the better adaptability of the GBEM algorithm, which learns the spatial properties of the text and background layers. This helps in avoiding over segmentation, as can be seen from the example of Figure 5.13.

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

K	Algorithm	Ext.	CRR	CPR	WRR
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%

TAB. 5.4 – Recognition results from 5, 9 hypotheses without GCC : number of extracted characters (Ext.), character recognition rate (CRR), character precision rate (CPR) and word recognition rate (WRR).

The grayscale consistency constraint (GCC) technique described in Section 5.2.2 was added to the baseline system. The corresponding results are listed in table 5.3. 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 grayscale 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.

In the proposed multiple hypotheses framework, more hypotheses can be generated by running each segmentation algorithm several times with different K values and selecting the "best" one according to the algorithm described in Subsection 5.2.3. Table 5.4 lists the results obtained by generating 5 or 9 hypotheses (using K=2 to 4), without employing the GCC post-processing. The method achieves a 96.6% CRR and a 93.1% WRR, which constitutes a reduction of more than 50% of both the character recognition and word recognition error 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.

Table 5.5 lists the results obtained by integrating the GCC algorithm in the multiple hypotheses

K	Algorithm	Ext.	CRR	CPR	WRR
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. 5.5 – Recognition results from 5, 9 hypotheses with GCC : number of extracted characters (Ext.), character recognition rate (CRR), precision (CPR) and word recognition rate (WRR).

framework. Comparing them with the figures in Table 5.4, we can notice that the GCC post-processing 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 model of the grayscale distribution. Hopefully, these different alternatives are not all corrupted by noise. This is especially true for the GBEM algorithm which already performs noise removal.

5.3.4 Summary of recognition performance

In table 5.6, we compare the recognition performance for both bi-modal and multi-modal schemes on the DB_TrainImages, DB_SceneImages and DB_Temporal database test sets. In the multi-modal case, we list only the results obtained using the GBEM method with 9 hypotheses.

Scheme	Test set	images	characters	baseline CRR	CRR of proposed methods
	DB_TrainImages	1208	9579	92.5%	92.6%
bi-modal	DB_SceneImages	50	2863	47.7%	70.9%
enhancement	DB_Temporal	247	2899	87.8%	88.0%
	DB_TrainImages	1208	9579	92.5%	96.5%
multi-modal	DB_SceneImages	50	2863	47.7%	90.0%
	DB_Temporal	247	2899	87.8%	89.2%

TAB. 5.6 – Recognition results of three test sets. CRR : character recognition rate.

From these results we can see that the multi-modal scheme obtains better performance in all the cases.

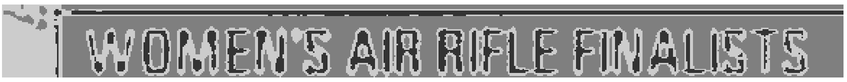
EM.2 : “WOMEN3 RIFLE FINALISTS”

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

Chapitre 6

Text recognition with multi-frame integration

In a video sequence, a text string usually appears in multiple consecutive frames. This temporal information can be exploited to help both text detection and recognition.

In text detection, potential detected text regions should be tracked consistently over a given number of frames to be validated. Text region tracking assumes that a text region has been firstly detected in a frame and then it aims at searching for the same text regions in consecutive frames. Given a text line B in the frame I_t, the matching value of B in another frame I_{t+k} is often computed using a correlation measure such as the percentage points in B that have a corresponding point in the I_{t+k} frame, assuming a translation motion $(\Delta x, \Delta y)$

$$Cor(I_t, I_{t+k}, \Delta x, \Delta y) = \frac{\sum_{(x,y)\in B} Threshold(|I_t(x,y) - I_{t+k}(x+\Delta x, y+\Delta y)|)}{|B|}$$

where the corresponding function $Threshold(x)$ is defined as :

$$Threshold(x) = \begin{cases} 1 & \text{if} \quad x < T_c \\ 0 & \text{otherwise} \end{cases}$$

The threshold T_c is a constant. Additionally, motion constraints on $(\Delta x\,,\,\Delta y)$ can also be exploited to remove false detected text regions. This is especially true for caption since they are either static or moving horizontally or vertically at a constant speed. Experimentally, we found out that text last more than 12 frames in news video, and last more than 3-8 frames in sports video. Therefore, in general, a detection algorithm assumes that a text region should at least be contained in 3 frames. As the methods using temporal information for detecting text are well discussed in Lienhart's work, in this thesis we will focus on exploiting temporal information for text recognition.

6.1 Text recognition in videos

In order to use the temporal information contained in consecutive frames of the same text string for improving the recognition results, Sato [94] and Lienhart [55] computed the maximum or

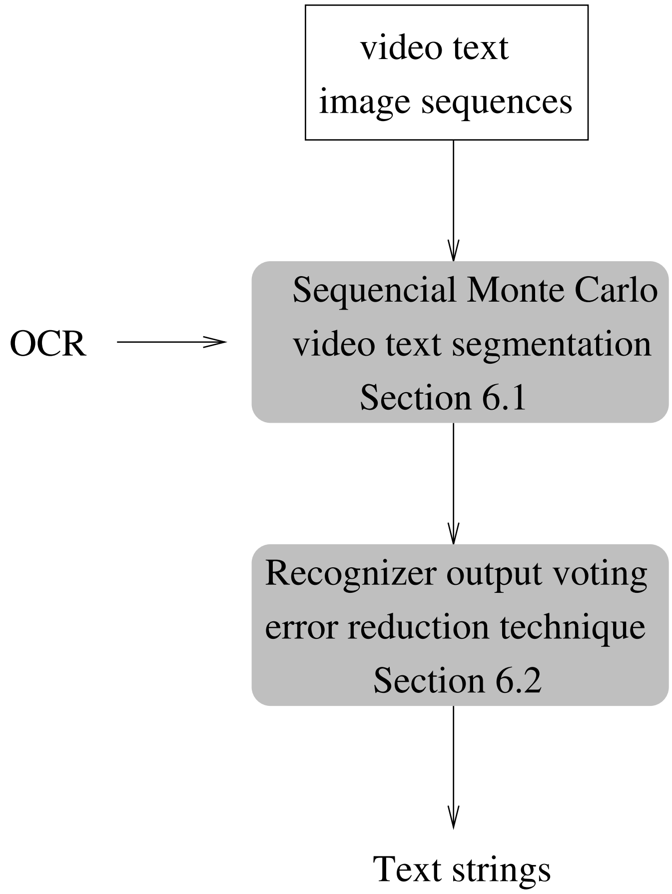


FIG. 6.1 – Text segmentation and recognition in multiple video frames.

minimum value at each pixel position over frames, which is referred to as MAX-MIN methods in the result section (6.2.4). 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. Li [50] 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, which is 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. Some methods, such as the MAX-MIN methods, even have to constrain the grayscale values of text to be the brightest or the darkest in the text images. A common goal of these methods is to obtain "better" segmentations of the text images.

However, none of these methods proposed criterion for evaluating the quality of the text image segmentation. The reason is that it is very hard to define a good measure of the quality of the segmentation. Since the final task is to obtain good recognition results, the "best" way to measure a segmentation is to evaluate its OCR output. We can even show that, giving the low resolution of text strings we can dealing with, an OCR system may output different results for segmentations that looks very similar to each other (see Fig. 6.2). In order to exploit temporal information for the recognition of text in video sequence, we proposed two algorithms in this chapter : the sequential Monte Carlo video text segmentation algorithm and the recognition results voting algorithm, which can be applied sequentially, as shown in the figure 6.1.

The first algorithm, presented in Section 6.2, is a probabilistic algorithm for segmenting and recognizing text embedded in video sequences based on adaptive thresholding using a Bayes filtering method. 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 instead of selecting the best recognition result from the temporal hypotheses using the criterion defined in Section 5.2.3, we propose in Section 6.3 an algorithm to reduce the video character recognition error rates by using output voting technique. Text characters are first detected, tracked in consecutive video frames and then segmented and recognized in each frame. These 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 may reduce recognition errors due to the low resolution of characters (before resizing and interpolation), the short length of the string and their unknown font. 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. This method is very helpful in improving the recognition of long text strings, for example more than 4 words.

6.2 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 reused 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 [118]. In other words, the segmentation quality in our case should be validated using recognition results instead of any predefined criterion on grayscale values of

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. 6.2 – Different recognition results may be obtained from segmentation results, which are visually quite similar.

the image. Figure 6.2 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 about 99% recognition rate on clean page characters. Although the segmentations of the word "lower" seems to be visually similar, they lead to different recognition results. The figure 6.3 illustrates an example of thresholding an text image. The optimal bi-modal threshold is (4), and the optimal multi-modal (K=3) thresholds are (2) and (3). However, the best recognition result is given by the threshold (1).

To address these two problems, we present in this section a particle filtering based Monte Carlo method 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 namely sequential Monte Carlo filtering [23, 1] or CONDENSATION [40] has shown to be a successful approach in several applications of computer vision [40, 70, 80].

One of the key point 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 to adapt to changes of grayscale values both in the text and background by simultaneously maintaining multiple-hypotheses. The advantage of the particle filtering in the presence of ambiguities and instabilities compensate OCR errors encountered when applying current OCR systems on video text due to the low resolution of characters (before resizing and interpolation), the short length of the string and their unknown font. In contrast to other filtering techniques that approximating posterior probabilities in parametric form, such as Kalman filters, this methodology allows to evaluate the likelihood of the segmentation parameters directly from the corresponding recognized text string based on language modeling and OCR statistics.

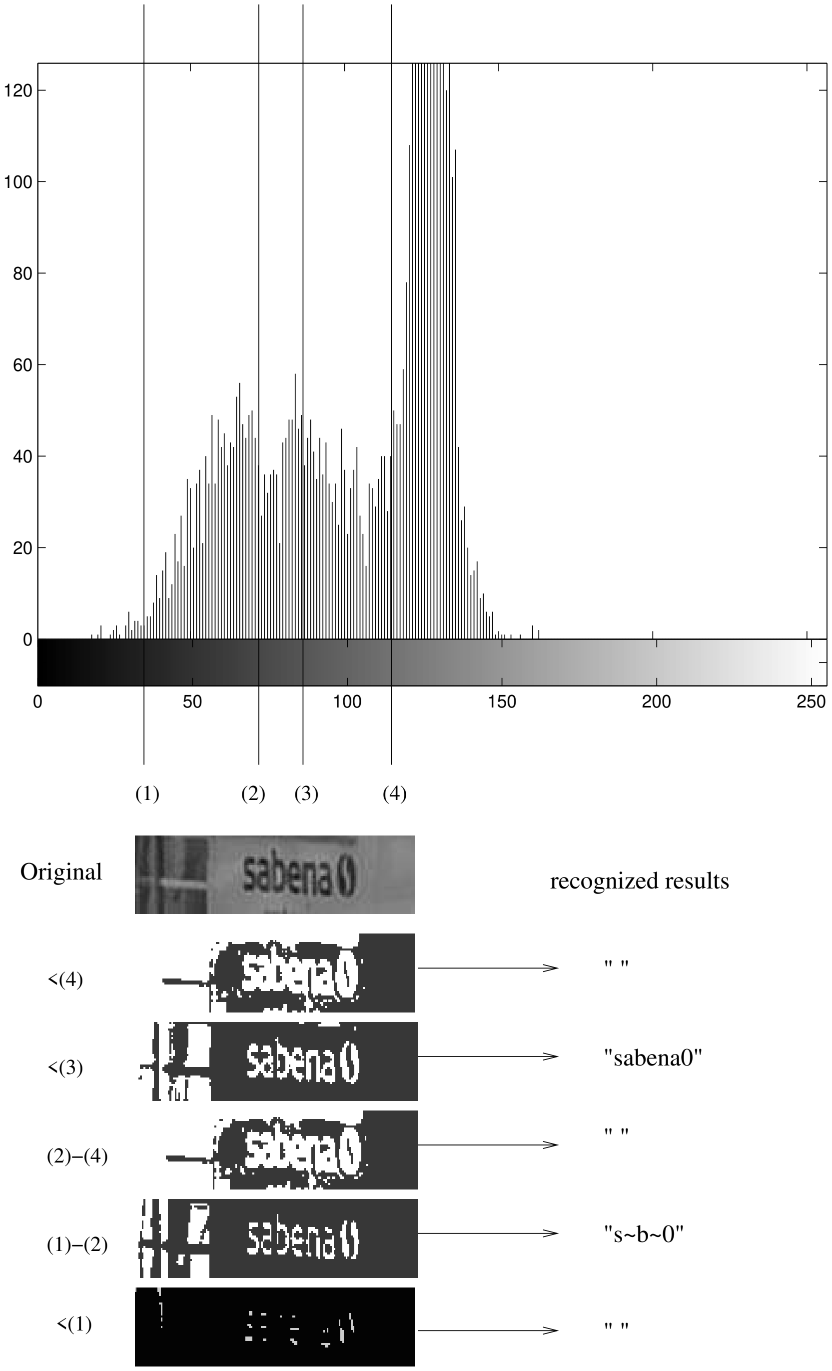


FIG. 6.3 – Thresholding an text image according to its grayscale histogram. The optimal bi-modal threshold is (4). The optimal multi-modal (K=3) thresholds are (2) and (4). The optimal multi-model (K=4) thresholds are (1), (2) and (4). However, the best recognition result is given by the threshold (3). The pixel values of some images are reversed for displaying.

6.2.1 Bayes filtering

Bayes filters address the problem of estimating the posterior probability $p(x_t|O_{1:t})$ of a dynamic state given a sequence of observations, where x_t denotes the state x at time t and $O_{1:t}$ denote the observation sequence from time 1 to time t. For video text segmentation, the state $x_t = (l, u)_t$ is characterized by a 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

$$p(x_t|O_{1:t}) = \alpha p(O_t|x_t, O_{1:t-1})\, p(x_t|O_{1:t-1}) \tag{6.1}$$

where α is the normalization constant

$$\alpha = p(O_t|O_{1:t-1})^{-1} \tag{6.2}$$

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

$$p(x_t|O_{1:t-1}) = \int p(x_t|x_{t-1}, O_{1:t-1})\, p(x_{t-1}|O_{1:t-1})\, dx_{t-1} \tag{6.3}$$

Assuming independence of observations conditioned on the states (i.e. $p(O_t|x_t, O_{1:t-1}) = p(O_t|x_t)$) and a first order Markov model for the sequence of states (i.e. $p(x_t|x_{t-1}, O_{1:t-1}) = p(x_t|x_{t-1})$). we obtain the following recursive equation for the posterior :

$$p(x_t|O_{1:t}) = \alpha p(O_t|x_t) \int_{x_{t-1}} p(x_t|x_{t-1})\, p(x_{t-1}|O_{1:t-1})\, dx_{t-1} \tag{6.4}$$

The exploitation of the of the equation (6.4) requires the definition of two conditional densities : the transition probability $p(x_t|x_{t-1})$ and the data likelihood $p(O_t|x_t)$. Both models are typically time-invariant so that we can simplify the notation by denoting these models $p\left(x^{'}|x\right)$ and $p(O|x)$ 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.

6.2.2 Probabilistic models for video text segmentation

Transition probability

In the context of video text segmentation, the transition probability $p\left(x^{'}|x\right)$ is a probabilistic prior on text threshold variations. The state space is a 2-D space constructed by the upper (u) and lower (l) thresholds of text grayscales $x = (l, u)$. In this paper, we investigate four methods to model the transition probability.

Gaussian model - In this model, the change of the text thresholds is assumed to be due to additive noise, which is modeled as a Gaussian process with a constant variance σ. The transition probability is thus defined as :

$$p\left(x^{'}|x\right) = \frac{1}{2\pi\sigma^2} e^{-\frac{(l^{'}-l)^2+(u^{'}-u)^2}{2\sigma^2}} \tag{6.5}$$

Uniform model - The second method considers the transition model as a result of illumination or lighting change in the video sequence. The grayscale values of all or part of text characters increase or decrease by a constant value due to the background motion behind transparent text or special visual effects. The transition probability is therefore defined as a uniform process :

$$p\left(x^{'}|x\right) = \begin{cases} \frac{1}{(l_{max}-l_{min})(u_{max}-u_{min})} & \text{if } l^{'} \in [l_{min}, l_{max}] \,\&\, u^{'} \in [u_{min}, u_{max}] \\ 0 & \text{otherwise}, \end{cases} \tag{6.6}$$

where the shifting range is modeled by a constant parameter α :

$$l_{min} = l - \alpha \quad \text{and} \quad l_{max} = l + \alpha$$

and

$$u_{min} = u - \alpha \quad \text{and} \quad u_{max} = u + \alpha$$

Adaptive uniform model - It is related to the uniform model. In this case, the amount of shifting values of the thresholds depends on current state. Let $min = 0$ and $max = 255$ respectively denote the minimum and the maximum values of the grayscale in the image. Given $x = (l, u)$, the shifting range l_{min} in equation (6.6) is adjusted by the distance between l and the min :

$$l_{min} = l - \alpha^{'}(l - min) \tag{6.7}$$

where $\alpha^{'} = 0.1$ is a constant. Similarly, we can defined :

$$l_{max} = l + \alpha^{'}(u - l) \tag{6.8}$$

and the shifting ranges of $u^{'}$ are defined as :

$$u_{min} = u - \alpha^{'}(u - l) \quad \text{and} \quad u_{max} = u + \alpha^{'}(max - u) \tag{6.9}$$

The distribution of $p\left(x^{'}|x = (150, 200)\right)$ in the adaptive uniform model is illustrated in figure 6.4.

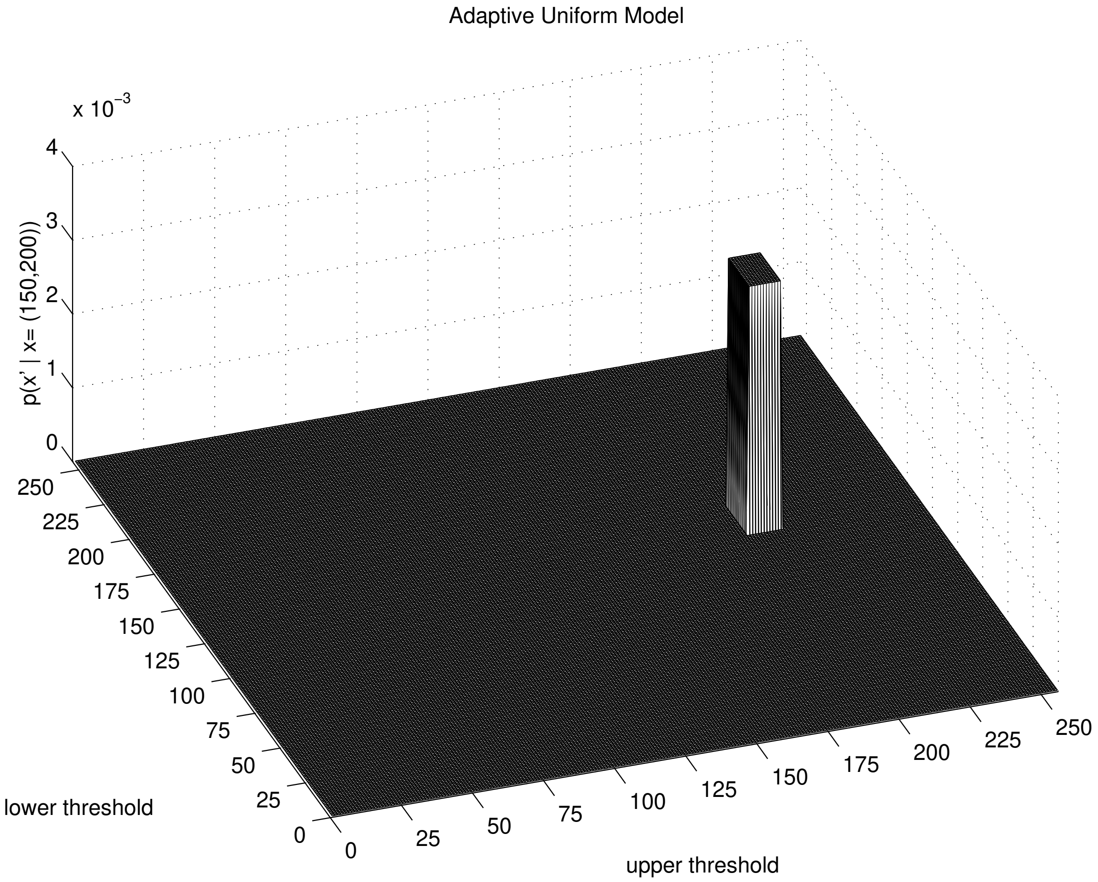


FIG. 6.4 – Adaptive uniform model of transition probability $p\left(x^{'}|x=(150,200)\right)$.

Adaptive mixture model - To model the transition probability using both noise and light shifting, we can modify the above adaptive uniform model by applying a Gaussian noise model on the state space out of the shifting range. Following the same definitions as in equation (6.7), (6.8) and (6.9), the transition probability $p\left(x^{'}|x\right)$ is therefore defined as :

$$p\left(x^{'}|x\right)=\begin{cases}\frac{1}{\gamma} & \text{if}\, l^{'}\in[l_{min},l_{max}]\,\&\,u^{'}\in[u_{min},u_{max}]\\ \frac{1}{\gamma}e^{-\frac{(l^{'}-l_{min}^{max})^2+(u^{'}-u_{min}^{max})^2}{2\sigma^2}} & \text{otherwise,}\end{cases} \tag{6.10}$$

where

$$l_{min}^{max}=\begin{matrix} l_{min} & \text{if}\, l<l_{min}\\ l_{max} & \text{if}\, l>l_{max}\end{matrix} \tag{6.11}$$

and

$$u_{min}^{max}=\begin{matrix} u_{min} & \text{if}\, u<u_{min}\\ u_{max} & \text{if}\, u>u_{max}\end{matrix} \tag{6.12}$$

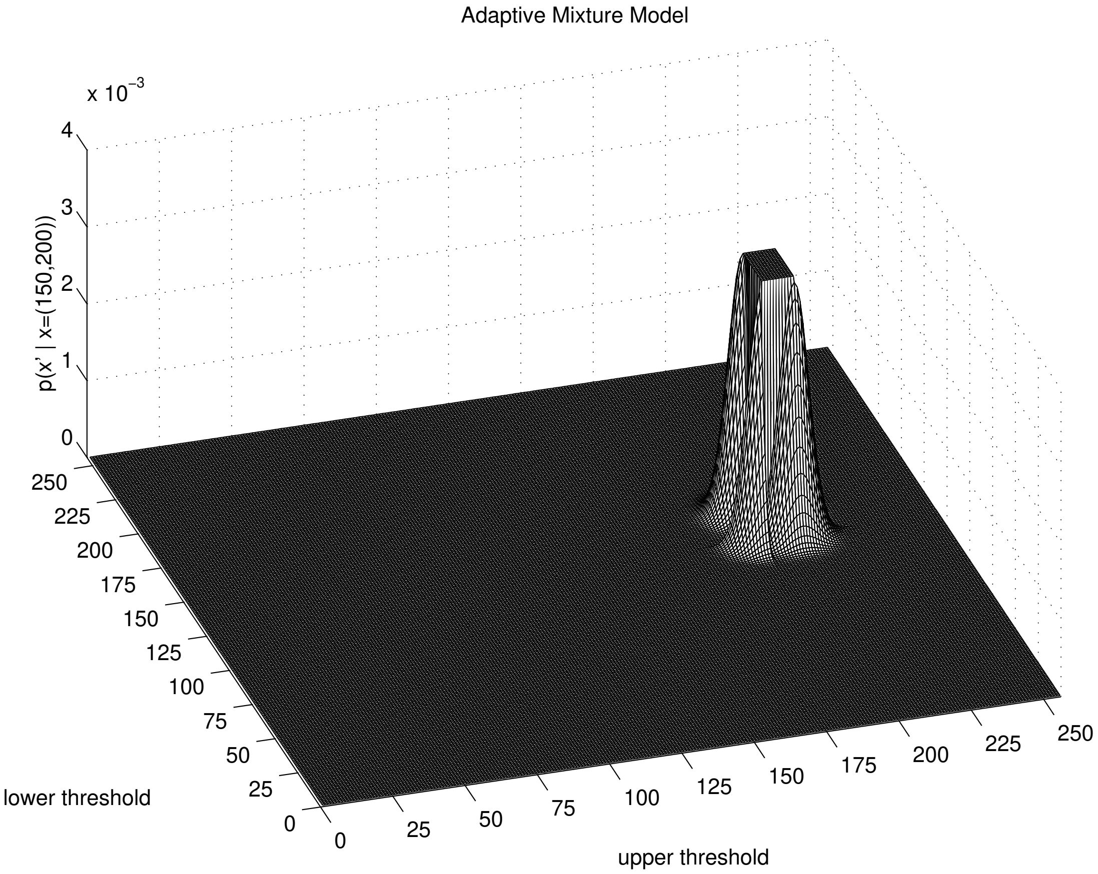


FIG. 6.5 – Adaptive mixture model of transition probability $p\left(x^{'}|x=(150,200)\right)$.

γ is a normalization constant which does not affect the MCVTS algorithm. The typical distribution of $p\left(x^{'}|x=(150,200)\right)$ in the adaptive mixture model is illustrated in figure 6.5.

Data likelihood

The data likelihood $p\left(O|x\right)$ 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 term of character extraction is difficult without performing some character recognition analysis. Besides, visually well segmented image does not always lead to correct recognition. Therefore, since ultimately we are interested in the recognized text string, the data likelihood will be evaluated on the output T of the OCR. The extraction of the text string T, we use the technique discussed in Chapter 5.

To evaluate the data likelihood using string T, we follow the definitions given in section 5.2.3. However, instead of computing the likelihood ratio 5.24, we need to approximate the data likelihood of the string T under the hypotheses that it comes from a correct segmentation. The data likelihood is thus defined as the probability of accurate segmentation H_a given the string T :

$$p\left(O|x\right) \propto p(H_a|T) = \frac{p(T|H_a)p(H_a)}{p(T)} \tag{6.13}$$

Here $p(T)$ is given by :

$$p(T) = p(T|H_a)p(H_a) + p(T|H_n)p(H_n)$$

and the data likelihood is then proportional to :

$$p\left(O|x\right) \propto \frac{1}{1 + \frac{p(T|H_n)p(H_n)}{p(T|H_a)p(H_a)}} \tag{6.14}$$

We use the same noise free language model p$(.|H_a)$ and noise language model p$(.|H_n)$ as introduced in subsection 5.2.3. The data likelihood is thus given by :

$$p\left(O|x\right) \propto \frac{1}{1 + \frac{\prod_{i=1}^{l_T} \mathrm{p}(T_i|H_n)}{\mathrm{p}(T_1|H_a)\prod_{i=2}^{l_T} \mathrm{p}(T_i|T_{i-1},H_a)} b} \tag{6.15}$$

Figure 6.6 illustrates this likelihood definition. It shows the ground-truth data likelihood, which we have defined here as $p(O|x) = 0$ if not all the words in the ground-truth are recognized, and $p(O|x) = 1$ otherwise. The figure also shows the proposed data likelihood of the image at all the possible states, illustrating that our probabilistic model is accurate. Even if the initial state (here provided by an Otsu algorithm [75] and shown with an arrow in the images) leads to an incorrectly recognized text 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.

6.2.3 Particle approximation

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

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

where δ 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

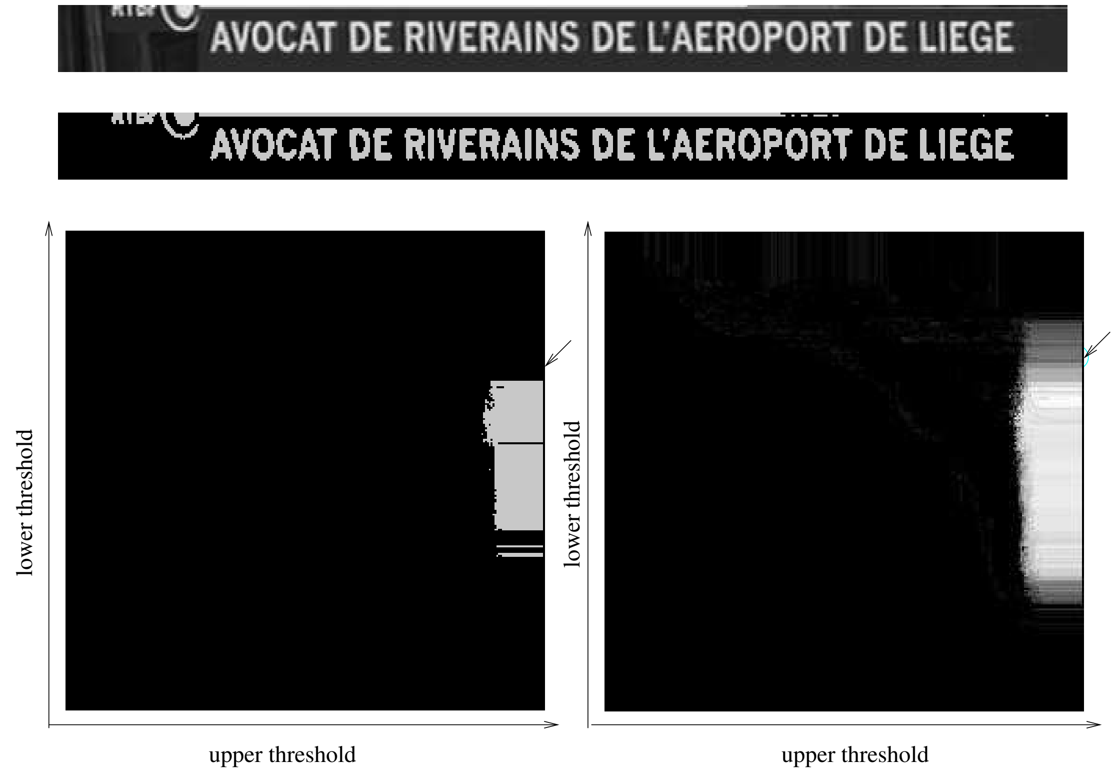


FIG. 6.6 – Data likelihood approximation : the observed text image is displayed at the top. The second image displays the results of applying Otsu binarization, which corresponds to OCR output "V AVOCAT DE RIVERAINS DE L AEROPORT DE iIEGE". In the last row, the left image shows the states that lead to the recognition of all the words in the ground truth, the right image displays the proposed data likelihood at all the states.

instead of replacing the old values with the new ones. The following is the MCVTS algorithm presented in pseudo code.

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 sample $\tilde{x}_t^i \sim X_t$; sample $x_{t+1}^i \sim p\left(x_{t+1} \vert \tilde{x}_t^i\right)$; set $w_{t+1}^i = p(O_{t+1} \vert x_{t+1}^i)$;
4. $X_{t+1} = X_t$, add the m new samples (x_{t+1}^i, w_{t+1}^i) to X_{t+1}.
5. output the text string that corresponds to the segmentation with the highest data likelihood.

Figure 6.7 illustrates the procedure of the MCVTS algorithm. The initial threshold couple $x = (120, 255)$ and $x = (0, 120)$ are obtained by using Otsu thresholding algorithm, which doesn't

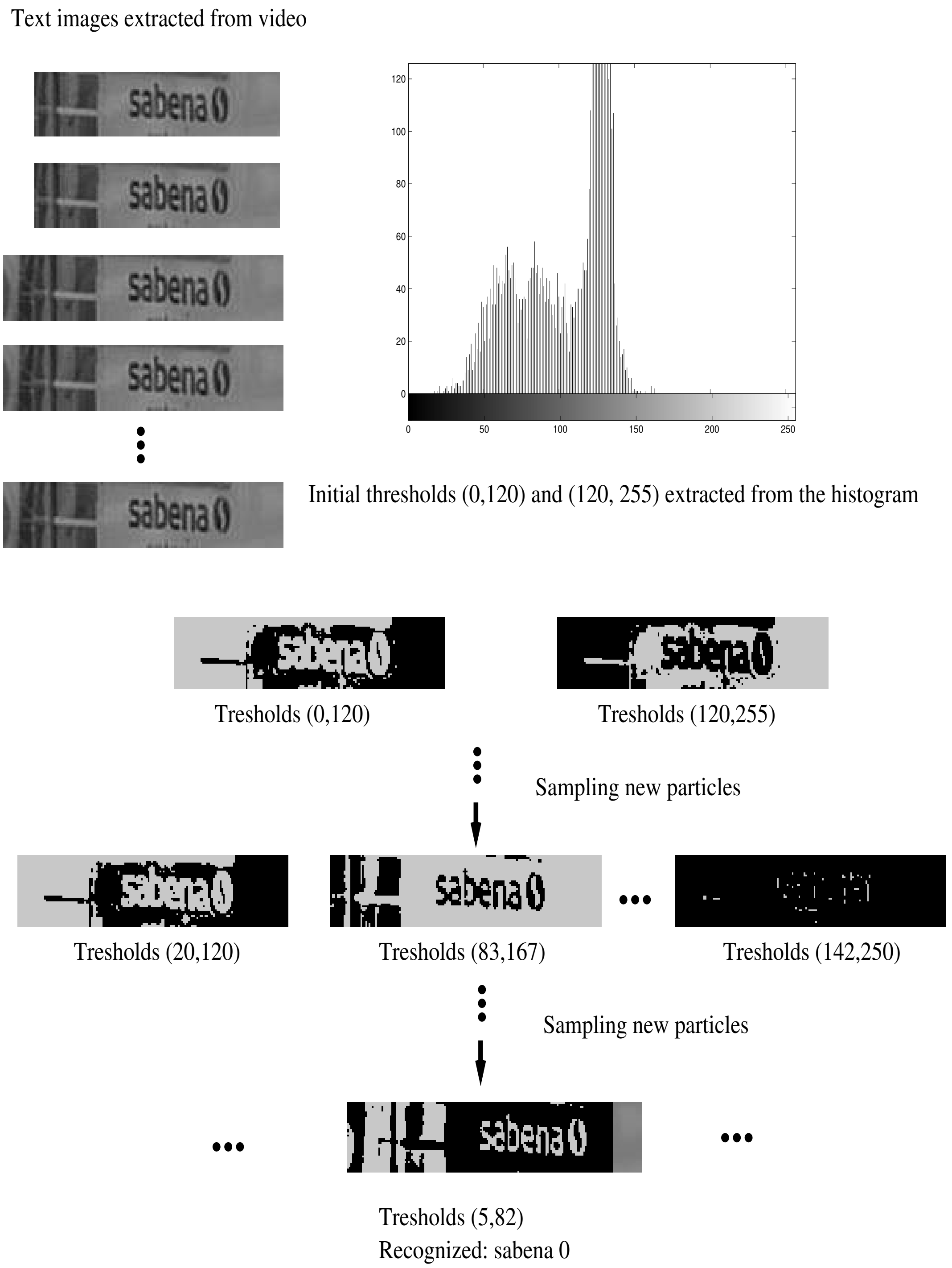


FIG. 6.7 – Video text segmentation using particle filtering.

Methods	Ext.	CRR	Prec.	WRR
Multi-model	2939	89.2%	88.0%	87.1%
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%
Gaussian MCVTS	2920	89.7%	88.3%	87.2%
Uniform MCVTS	2912	90.5%	90.1%	87.3%
Adaptive uniform MCVTS	2935	92.3%	91.2%	89.0%
Adaptive mixture MCVTS	2928	93.9%	93.0%	90.6%

TAB. 6.1 – Performance comparison between the MCVTS (m=3), the 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 [50].

FIG. 6.8 – Examples of located embedded text in video.

lead to a correct solution in this case. After particle sampling in several frames, the states (threshold couples) cover a wide range of thresholds in the state space. At the end, the threshold couple $x = (5, 82)$ gives the highest likelihood. The segmentation result using this optimal 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".

6.2.4 Experiments

The MCVTS algorithm was tested on the DB_Temporal database which consists of 247 text strings (2899 characters and 548 words) in 6944 text images (about 28 images per text string in average). Figure 6.8 shows some image examples.

In order to compare the performance of the MCVTS algorithm with previous work, we implemented the Max-Min methods [55] and average value method [50]. Table 1 lists the results of the Max-Min value methods, average value method and the MCVTS algorithm with $m = 3$. The best one in Max-Min consists of selecting the best result from the minimum and maximum outputs using the confidence defined in Eq. 6.1.2. The table also recalls the multi-modal recognition results on the DB_Temporal database.

The Max-Min methods do not provide better results than the multi-model based on static images. The main reason is that the text regions in video frames contain strong edges, which are high frequency signals. The current compression of the video frame, for example the MPEG codec, has the tendency to omit high frequency signals in the video. Therefore, a lot of noise is introduced around the text region. When applying the Max-Min methods on video frames, text pixels are sometime pushed into background because of the noise.

We can also see that the results of the average value method produces similar results as the multi-model method without introducing any randomness. All the four MCVTS algorithms gained some improvements in comparison. By checking the tested samples in the database, we found that the MCVTS algorithms performed better segmentation when the automatically detected text images were noisy, contained perturbation, or when the grayscale values of characters spanned a wide range, as shown in Figure 6.8. The results in table 1 also illustrates that the MCVTS algorithms not only significantly improves the character recognition but also the precision.

From all the four dynamic models proposed in the paper, the adaptive mixture model yields the best results in terms of character recognition rate and precision. Figure 6.9 illustrates the character recognition rates of MCVTS algorithms with varying m. All the four dynamic models give similar results when m is above 10, 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).

The CPU cost of the MCVTS algorithm depends on the size of the state space, the number of samples, the thresholding operation and the OCR computation. All these factors are associated to the number of new samples m in each iteration. The experiments show that using more than $m = 3$ particles per image with the adaptive mixture model does not change a lot the performance of the algorithm (adaptive mixture model). The average number of samples per text string is thus around 80.

6.2.5 Discussion of MCVTS

In this section, we proposed a Monte Carlo method for segmenting and recognizing embedded text of any grayscale value in image and video based on particle filter. The MCVTS algorithm has four main advantages for segmenting video text. Firstly, the algorithm proposes a methodological way to search for segmentation parameters that lead to accurate results. Secondly, the algorithm adapts itself to the data by sampling in proportion to the posterior likelihood. This enable us to propose an accurate probability model based on OCR results instead of estimating the posterior of segmentation based on segmented images. Thirdly, the algorithm does not require precise tracking of text images among video frames at pixel level. Finally, the MCVTS algorithm is very easy to implement and also easy to be extended to other state spaces, such as parameters of local thresholding techniques (e.g. Niblack binarization).

An additional improvement of the MCVTS algorithm can be made by combining multiple recognition results of the same text string of the character level instead of outputting the best result.

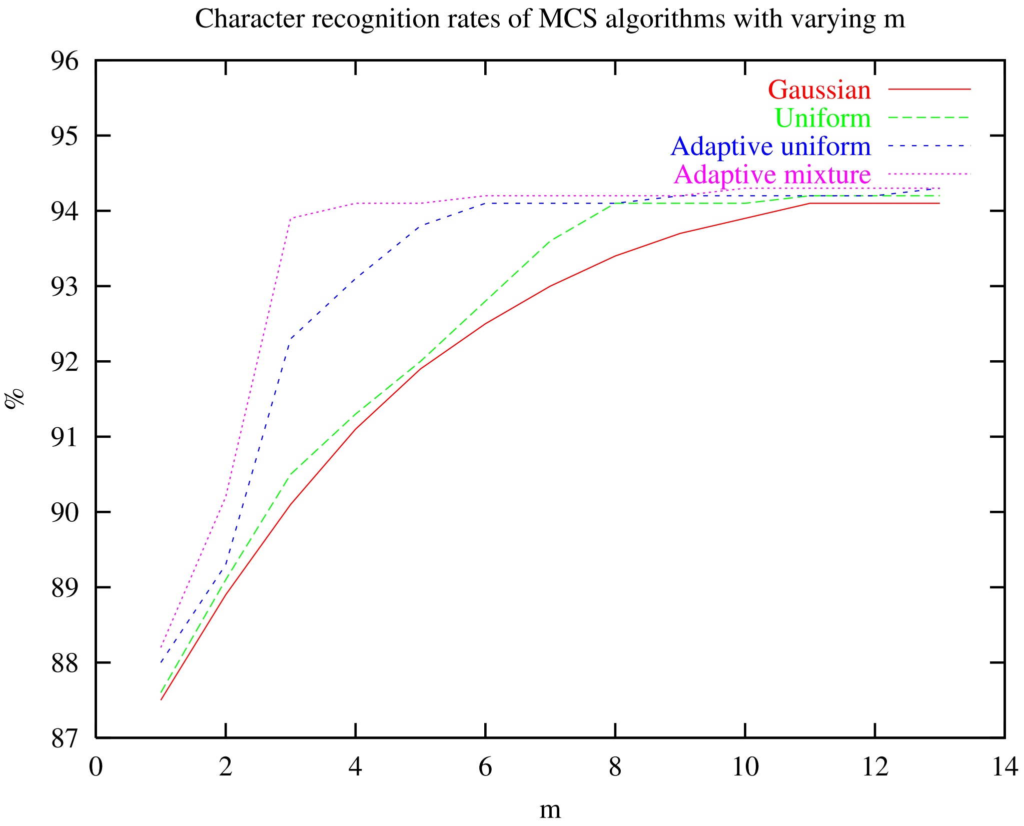


FIG. 6.9 – Character recognition rates of MCVTS algorithms with varying m.

This issue is addressed in the next section.

6.3 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 [27] is widely used in automatic speech recognition system (ASR) for compositing ASR output from the outputs of multiple ASR systems. The system developed a sequential voting process to reconcile differences in ASR system outputs (sequences of words). The key 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 additional knowledge sources such as acoustic and language models.

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. Since the basic element of OCR recognition results are characters, we will build a character transition network instead of the word transition network in ASR systems. Figure 6.10 shows 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 modeled as independent knowledge sources. Then the "best" output is composited using a criteria defined on the frequency of the character and a language model at each position of the character transition network. This "best output might be a new string that is not presented in initial solutions.

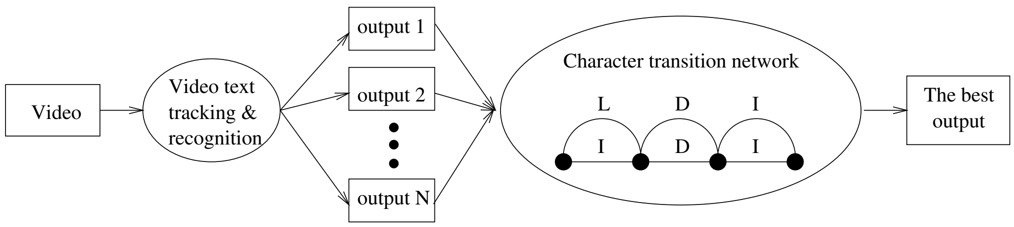


FIG. 6.10 – Video Character Recognition Error Reduction Algorithm

6.3.1 Character transition network

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. We introduce a special label "NULL" to represent a character which is inserted by the dynamic programming alignment as a blank. In practice, we treat both the space character in OCR recognition results and the blank generated by the dynamic programming alignment as the same character. 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. Figure 6.11 illustrates some linear-topology CTNs and corresponding text strings.

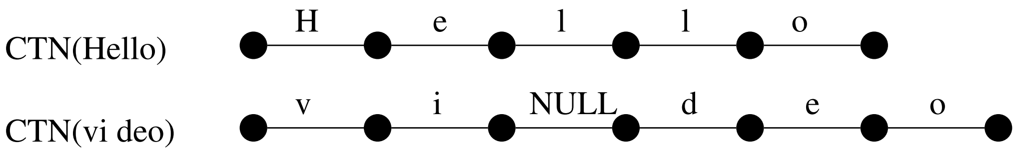


FIG. 6.11 – Linear-topology character transition networks

6.3.2 Character transition network construction

Every individual text string can be modeled by a linear topology CTN. Multiple outputs of consecutive frames can be modeled by a CTN with non-linear topology (more than one arc between two nodes). To obtain such a CTN, we first have to align the nodes of the individual CTN corresponding to each output.

Let us focus on the alignment of two CTNs. Regarding one CTN as a reference, the other CTN can be aligned using a 2-D dynamic programming alignment process. In our case, we use SCLITE [1] a dynamic programming alignment engine developed by NIST. The dynamic programming alignment process yields two aligned CTNs with potentially inserted "NULL" arcs as illustrated in figure 6.12. Then, we can merge the two CTNs together by copying the corresponding arcs from CTN-i into CTN-REF. The updated CTN-REF is shown in figure 6.12. It is a model of the two strings "vide" and "ibeo".

Iteratively merging CTNs of all the outputs into a reference CTN-REF generates a composite CTN that models the temporal redundant outputs.

However, this iterative composing process does not guarantee an optimal CTN because the composite CTN 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 the 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 of a CTN modeling one text string T, we use the confidence $C(T)$ of the text string T defined in equation 6.14.

$$C(T) = p\left(O|x\right) \tag{6.16}$$

[1]The program sclite is a tool for scoring and evaluating the output of speech recognition systems. Sclite is part of the NIST SCTK Scoring Tookit. http ://www.icsi.berkeley.edu/Speech/docs/sctk-1.2/sclite.htm

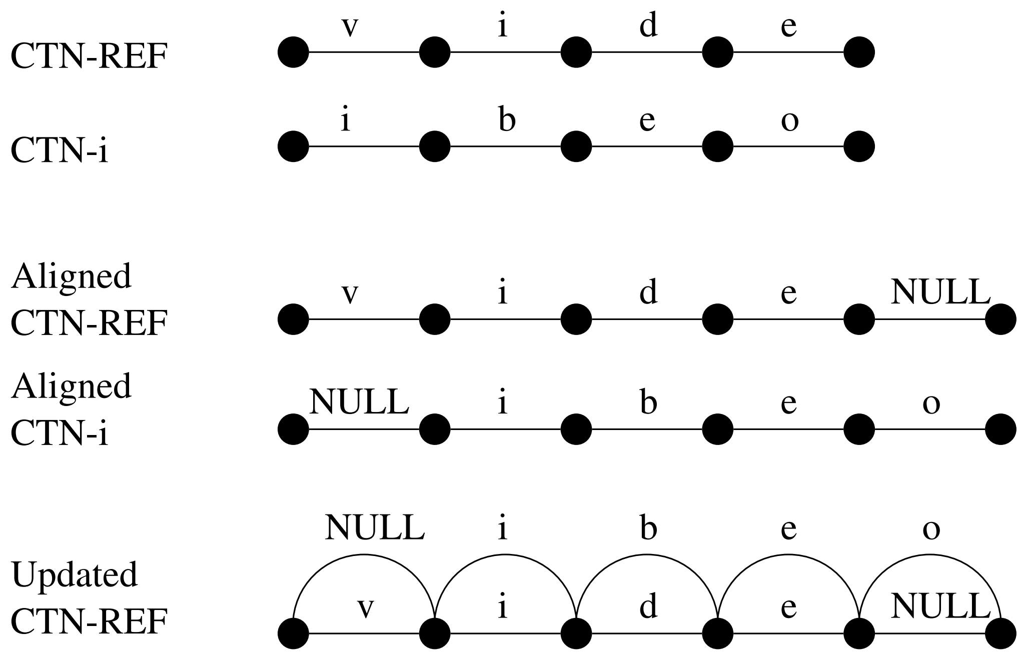


FIG. 6.12 – Alignment of two character transition networks

We first sort the CTN's 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 from high to low. We thus obtain a model of all the outputs of a given string in consecutive frames.

6.3.3 Best character sequence searching in character transition network

The best character sequence is searched by a scoring process in the composite CTN. Traditional ROVER algorithm developed in ASR system 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 frequency of occurrence $F(c_{ij})$ of c_{ij} as :

$$F_i(c_{ij}) = \frac{\sum_{k=1}^{m} \delta(c_{ij}, c_{ik})}{m} \tag{6.17}$$

where δ is a mass choice function defined as :

$$\delta(a,b) = \begin{cases} 1 & \text{if} \quad a = b \\ 0 & \text{otherwise} \end{cases} \tag{6.18}$$

Besides the frequency of occurrence, a confidence value of each character is also necessary. Let us follow the definition of the confidence value proposed in section 6.1.2 and define the confidence value of a character as :

$$Conf(c_{ij}) \propto p(c_{ij}|H_a) = \frac{1}{1 + \frac{\mathrm{p}(c_{ij}|H_n)}{\mathrm{p}(c_{ij}|H_a)} b} \tag{6.19}$$

The confidence of the "NULL" transition arc is considered as a parameter β of the scoring scheme. In practice, we can train it. The general scoring formula is :

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

where $\alpha \in [0, 1]$ is a constant weight parameter. In the extreme cases, if $\alpha = 1$, the score only depends on the occurrence frequency of the characters ; if $\alpha = 1$, the score only depends on the confidence of the characters.

6.3.4 Experiments and discussion

The MCVTS and the ROVER algorithm are tested together on the DB_Temporal database. In the MCVTS algorithm, multiple recognition hypotheses are generated in a text image sequence and the final recognition result is selected from all the hypotheses through all the frames. To apply ROVER on the text image sequence, we select the best one recognition result from the recognition hypotheses resulted from each frame using the confidence defined in Eq. 6.19, We therefore obtain a recognition result sequence corresponding to the original text image sequence frame by frame.

Table 6.2 lists two recognition results sequences : "ASSOCIATE PRODUCERS" and "TONI MYERS GRAEME FERGUSON". None of the results from the individual frames contains the correct string. The reason leading to these errors is coming from the special effect in the video. Figure 6.13 shows some text images of these two strings. The table also gives the voting results by using the ROVER algorithm at the end of each sequence.

Table 6.3 compares the performance of the combined MCVTS and ROVER algorithm, the raw adaptive mixture MCVTS algorithm, the best one Max-Min value method, the average value method and the multi-model method. Different parameters (α, β) lead to different recognition results. The optimal parameters for the DB_Temporal database are $\alpha = 0.1$ and $\beta = 0.5$. With the optimal parameters, the ROVER can improve only a little the character recognition rate, but can significantly improve the precision and word recognition rate. The results show that this voting method is helpful to remove false recognition characters, which leads to higher precision. It also enables a better chance to composite words by voting out the correct characters in the CTN and therefore leads to a better recognition of words. However, they also illustrate that the multiple OCR recognition results from video are not really independent knowledge resources. The outputs of only one OCR software are not very complementary. Using more than one OCR software in the ROVER algorithm is a potential way to improve the character recognition in video, which may be a good future research direction.

f1	A	S	l	O	C	I	A	T	I		P	R	O	D	U	C	r	R	s
f2	A	S	S	O	C	I	A	T	E		P	R	O	D	U	C	E	i	s
f3	A	S	S	O	C		A	1	[		P	R	O	D	U	C	E	R	S
f4	A					l	A	T	E		P	R	O	D	U	C	E	t	B
f5	A	S	S	O	C	I	A	T	E		P	R	O	D	U	C	[	R	5
f6	A		S			I	A	T	I		f	R	O	D	U	C			
f7	A	S	S	O	C	I	A	I	I					D	U	C	E	R	S
f8	A						o				P	R	O	D	U				
f9					o		A	T	[		P	R	O	D	U	C	E	R	S
f10					o		A	T	E		P	:	)	D	U	C	E	R	S
f11						I	A	T	[					D	U	C	E	R	S
f12					C	I	A	T	r		P	R	O	D	U	C	E	R	S
f13	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

f1	T	O	N	I		f	Y	E	R	S		G	R	A	E	M	E		F	E	R	G	U	S	O	N
f2	T		N	I		M	Y	E	R	S			R	A	E	M	E		F	E	R	G	U	S	O	N
f3	T	O	N	I						S						M	E		F	E	R	G	U	S	O	N
f4	T	O	N	I												M	E		F	E	R	G	U	S	O	N
f5	T	O	N	I								G							F	E	R	G	U	S	O	N
f6		O	N	I		M	Y	E	R	S		G	R	A									U	S	O	N
f7	T	O	N	I		M	Y	E	R	S		G											U	S	O	N
f8	T	O	N	I		M	Y	E	R	S						M	E						U	S	O	N
f9	T	O	N	I		M	Y	E	R	S						M	E									N
f10	C	)	N	I		M	Y	E	R	S		G	R	A	E	M	E		F	E	R	G	U	S	O	N
f11		O	N	I		M	Y	E	R	S		G	R	A	E	M	E		F	E	R	G	U	S	O	N
f12		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

TAB. 6.2 – Voting of multiple recognition results of two video text strings in the DB_Temporal database.

FIG. 6.13 – Examples of text images associated with Table 6.2.

6.4 Conclusion of video text recognition

In this chapter, 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. A common advantage of these two methods is that the temporal information is exploited at the level of text strings, so that additional knowledge resources, for example the language models, can be used to reduce the errors.

On the basis of the experiments and results in this chapter, we conclude that visual effects of the segmentations of text images are not the best way to measure the segmentations with respect to the final OCR recognition results. Under the Bayesian framework, the MCVTS method is helpful to search for an optimal segmentation based on language models of both real text and OCR recognition errors. Also, we found out that the sequential voting techniques on recognition outputs (the ROVER algorithm) is able to composite a correct word or string from OCR recognition results even if none of these results contains a correct answer. However, both the MCVTS and ROVER algorithms have higher computation cost than the average value method because segmentation and recognition have to be performed in each frames. In building a real video text recognition application, a trade of between speed and accuracy can be made by choosing the average value method (with multi-model) and the MCVTS-ROVER method.

methods	Ext.	CRR	Prec.	WRR
Multi-model	2939	89.2%	88.0%	82.1%
One-Max-Min	2878	86.1%	86.7%	84.4%
Average value	2929	88.9%	87.9%	86.8%
MCVTS	2928	93.9%	93.0%	90.6%
ROVER(0,0)	2983	90.1%	87.5%	87.1%
ROVER(0.1,0.2)	2983	90.9%	88.3%	87.6%
ROVER(0.5,0.2)	2948	90.7%	89.2%	87.5%
ROVER(0.1,0.5)	2870	**94.1%**	95.1%	**92.0%**
ROVER(0.1,0.6)	2812	93.9%	96.8%	91.9%
ROVER(0.1,0.8)	2765	93.3%	**98.0%**	91.6%
ROVER(0.9,0.1)	2542	84.5%	96.4%	82.7%

TAB. 6.3 – Performance comparison of the ROVER(α, β) algorithm, the raw MCVTS algorithm and the baseline (average image) algorithm : extracted character number (Ext.), character recognition rate (CRR), precision (Prec.) and word recognition rate (WRR)

Chapitre 7

Conclusions

This thesis has investigated methods in building an efficient system for detecting and recognizing text of any grayscale values embedded in images and video sequences. The main purpose for building such a system was to fulfill the needs of growing content-based multimedia indexing, retrieval and management. We constructed the whole system using sub-tasks : the localization task and the verification task in text detection and the segmentation task, the character recognition task and the temporal processing task in text recognition.

7.1 System performance

As an overview of the results of the previous chapters, we can conclude that 95% to 99% text strings contained in images and videos can be detected with a precision (the ratio between the number of the detected true text strings and the number of all the extracted regions) of 94% to 97%. Considering the errors made in the detection step, image and video text recognition (including both superimposed text and scene text) achieves a 93% to 96% character recognition rate. 90% to 97% extracted characters correspond to the original characters, referred to as precision, in image or video frames are true characters. We also obtained a 88% to 92% word recognition rate without using a lexicon.

The optimal configuration of the system in terms of word/character recognition rate is :

1. edge-based text localization ;
2. constant gradient variance feature based text verification using support vector machine ;
3. for static images, the multiple hypotheses recognition scheme using the Markov random field based segmentation method (GBEM) ;
4. for video text, the MCVTS algorithm and the ROVER algorithm for text recognition.

We can use the K-means algorithm instead of the Markov random field (GBEM) approach to make the segmentation step faster. Also, as we already said in Chapter 6, the MCVTS and ROVER algorithms are optional, depending on a compromise between speed and accuracy in real application.

A real application system has been implemented and applied in two European projects : ASSAVID (Automatic Segmentation and Semantic Annotation of Sports Videos) and CIMWOS

(Combined Image and Word Spotting) both granted by the European IST Programme. The performance of the system has fulfilled the requirements of the cooperating industrial partners, the BBC and Canal+.

7.2 Achievements in solving text detection and recognition problems

The contributions of the thesis are the following.

First, machine learning approaches, MLP and SVM, can be used to achieve efficient text detection by using the localization/verification scheme proposed in Chapter 4. The difficulty of intensive computation of these machine learning approaches can be avoided by pre-performing a heuristic localization with a low rejection rate (low precision) followed by size and contrast normalization.

Second, contrast independent features have been proposed and shown to achieve good text verification performance using MLP and SVM classifiers. Comparing combinations of features and classifiers, SVM using the constant gradient variance feature has shown to be the most effective one.

Third, in a bi-modal grayscale text recognition scheme, the recognition performance can be improved by enhancing the contrast of sub-structure of characters. Asymmetric Gabor filters have been shown to be able to enhance characters of various size. However, the enhancement only works for black or white characters without inserted surrounding contrast layout.

Fourth, a multi-hypotheses framework based on a set of statistical modeling methods, such as the Markov random field based segmentation method (GBEM) and the grayscale consistent constraint post-processing algorithm, can significantly improve the recognition performance with respect to bi-modal schemes. Under this framework, we have proposed an efficient string selection algorithm based on language modeling and OCR statistics to extract the best string from the recognition results of multiple segmentation hypotheses.

Finally, we proposed two more specific methods for text recognition in video. The first one estimates the optimal segmentation parameters based on text recognition results. The second combines text recognition outputs in consecutive frames using a sequential voting technique. The two methods combined has been shown to improve standard techniques in video text recognition.

7.3 Limitations of the current work and possible future research efforts

In the current system, the text strings are required to be aligned horizontally. Although the system also works for text strings that have a little slant, which are the cases of video text and some of scene text, it can not be applied on the applications, in which it is not possible to capture all the scene text strings horizontally. One way to extend the system for detecting slanted text is to rotate the input image and apply the current detection algorithm at the same time. However, this procedure require much more computation cost.

There is no lexicon used in the current system due to the difficulty of having a good general lexicon. Text strings in images and videos can be names of people and locations that are not very famous. Therefore, to have a good general lexicon is almost not possible. In a specific application, the usage of a lexicon, if available, will allow to enable the use of retrieval techniques to account for potential errors in text recognition, especially the word recognition, and therefore will lead to better indexing and annotation.

Current OCR techniques, for example the OCR software used in this thesis, do not perform very well when the extracted characters have a too low resolution. For characters that are less than 8 pixels high in the original image or video frame, simply re-scaling the text image can not really help the OCR software to do a better recognition. The development of new OCR techniques to recognize low resolution characters is still necessary.

Future investigations on other two aspects need to be pursued for developing solid image and video text detection and recognition applications and related multimedia retrieval and annotation applications.

One aspect is semantic mining for text-based multimedia retrieval and annotation. The exploitation of context information of text strings, such as the positions or movements of the text strings in an image or a video, is an important semantic resource to annotate the image or the video. Future investigations can focus on mining the relationship between these context information together with the content of the corresponding text and categories of images and video shots.

The other aspect is computation reduction for mobile image text recognition applications, such as traffic sign recognition. Most mobile devices, such as personal digital assistants (PDAs) and mobile phones, have less computation power and less memory resources than a desktop computer. In order to build an image or video text extraction application on these devices, the algorithms proposed in this thesis need to be optimized or even modified to reduce the computation cost.

Bibliographie

[1] 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*, pages 100–107, 2001.

[2] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Royal Statistical Society*, 36(B) :192–236, 1974.

[3] M. J. Brooks. Rationalizing edge tetectors. *Computer graphics and image processing*, 8 :277–285, 1978.

[4] Jean Marie Buijs and Michael Lews. Visual learning of simple semantics in imagescape. In *Proc. 3rd Int. Conference VISUAL*, pages 131–138, 1999.

[5] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2) :1–47, 1998.

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

[7] Fabien Cardinaux and Sébastien Marcel. Face verification using MLP and SVM. In *XI Journees NeuroSciences et Sciences pour l'Ingenieur (NSI 2002)*, La Londe Les Maures, France, Sep. 2002.

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

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

[10] D. Chen, J.M. Odobez, and H. Bourlard. Text segmentation and recognition in complex background based on markov random field. In *Proc. IAPR Int. Conf. Pattern Recognition*, volume 2, Quebec City, Canada, 2002.

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

[12] F.H. Cheng, W.H. Hsu, and M.C. Kuo. Recognition of handprinted chinese characters via stroke relaxation. *Pattern Recognition*, pages 579–593, 1993.

[13] K.W. Church and P. Hanks. Word association norms mutual information and lexicography. *Computational Linguistics*, 16(1) :23–29, 1990.

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

[15] G. Cybenko and J. Moody. Approximation by superposition of a sigmoidal function. *Mathematics of Control, Signals and Systems*, pages 303–314, 1988.

[16] M. Das, E. Riseman, and B. Draper. Focus : Searching for multi-colored objects in a diverse image database. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 756–761, 1997.

[17] G. Daugman. Uncertainty relations for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *Journal of the Optical Society of America*, 2 :1160–1169, 1985.

[18] L. S. Davis. A survey of edge detection techniques. *Computer graphics and image processing*, 4 :248–270, 1976.

[19] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39(1) :1–38, 1958.

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

[21] D. Doermann and S. Yao. Generating synthetic data for text analysis systems. In *Proc. Symposium on Document Analysis and Information Retrieval*, pages 449–467, 1995.

[22] David Doermann. The indexing and retrieval of document images : A survey. *Computer Vision and Image Understanding : CVIU*, 70(3) :287–298, 1998.

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

[24] D. G. Elliman and I. T. Lancaster. A review of segmentation and contextual analysis for text recognition. *Pattern Recognition*, pages 337–346, 1990.

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

[26] Raphaël Féraud, Olivier Bernier, and Jean-Emmanuel Viall. A fast and accurate face detector based on neural networks. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(1), 2001.

[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*, pages 347–352, 1997.

[28] H. Fujisawa and K. Marukawa. Full text search and document recognition of japanese text. In *Proc. Symposium on Document Analysis and Information Retrieval*, pages 55–80, 1995.

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

[30] 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.

[31] T. Gevers and A. W. M. Smeulders. Pictoseek : Combining color and shape invariantfeatures for image retrieval. *IEEE Trans. Image Processing*, 9(1) :102–119, 2000.

[32] V. K. Govindan and A. P. Shivaprasad. Character recognition- a review. *Pattern Recognition*, pages 671–683, 1990.

[33] R. M. Haralick. Edge and region analysis for digital image data. *Computer graphics and image processing*, 12 :60–73, 1980.

[34] R. M. K. Haralick. Textural feature for image classification. *IEEE Trans. on Systems, Man, and Cybernetics*, 3 :610–621, 1973.

[35] H. Hartley. Maximum likelihood estimation from incomplete data. *Bio-metrics*, 14 :174–194, 1958.

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

[37] S-Y. Hsu. A texuture-tone analysis for automated landuse mapping with panchromatic images. In *Proc. American Society for Photogrammetry*, pages 203–215, 1977.

[38] M. Hueckel. A local operator which recognizes edges and lines. *Journal of the ACM*, 20 :634–647, 1973.

[39] Z. Hussain. *Digital image processing : practical applications of parallel processing techniques*. ELLIS HORWOOD, Redwood Press Ltd, 1991.

[40] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *4th European Conf. Computer Vision*, volume 1, pages 343–356, 1996.

[41] T. Joachims. Making large-scale support vector machine learning practical. In A. Smola B. Scholkopf, C. Burges, editor, *Advances in Kernel Methods : Support Vector Machines*. MIT Press, Cambridge, MA, 1998.

[42] K. Jonsson, J. Matas, J. Kittler, and Y.P. Li. Learning support vectors for face verification and recognition. In *Proc.* 4*th Int. Conf. on Automatic Face and Gesture Recognition*, pages 208–213, 2000.

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

[44] 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.

[45] M.D. Kernighan, K.W. Church, and W.A. Gale. A spelling correction program based on a noisy channel model. In *Proc. Int. Conf. Computational Linguistics*, pages 205–210, 1990.

[46] J. Kreyss, M. oper, P. Alshuth, T. Hermes, and O. Herzog. Video retrieval by still image analysis with imageminer. In *Proc. SPIE Symposium on Electronic Imaging : Science and Technologie*, pages 8–14, 1997.

[47] K. Kukich. Techniques for automatically correcting words in text. *Computing Surveys*, 24(4) :377–440, 1992.

[48] R. Laprade and M. Doherty. Split-and-merge segmentation using an F test criterion. *SPIE image understanding and man-machine interface*, pages 74–79, 1987.

[49] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, and L.D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, pages 541–551, 1989.

[50] H. Li and D. Doermann. Text enhancement in digital video using multiple frame integration. In *Proc. ACM Multimedia*, volume 1, pages 385–395, Orlando, Florida, USA, 1999.

[51] S.Z. Li. *Markov random field modeling in computer vision*. Springer-Verlag, 1995.

[52] Y. Li, J. Kittler, and J. Matas. On matching scores of LDA-based face verification. In T. Pridmore and D. Elliman, editors, *Proc. British Machine Vision Conference BMVC2000*. British Machine Vision Association, 2000.

[53] Z. Li, O. Zaiane, and Z. Tauber. Illumination invariance and object model in content-based image and video retrieval. *Vis. Commun. and Image Rep.*, pages 219–244, 10 1999.

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

[55] 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.

[56] Daniel Lopresti and Jiangying Zhou. Retrieval strategies for noisy text. In *Proc. Fifth Annual Symposium on Document Analysis and Information Retrieval*, pages 255–269, 1996.

[57] Wei-Ying Ma and B. S. Manjunath. Netra : A toolbox for navigating large image databases. *Multimedia Systems*, 7(3) :184–198, 1999.

[58] J. Mantas. An overview of character recognition methodologies. *Pattern Recognition*, pages 425–430, 1986.

[59] D. Marr and E. Hildreth. Theory of edge detection. *Proceeding of the royal society of London*, B-207 :186–217, 1980.

[60] B. H. Mccormick and S. N. Jayaramamurthy. Time series model for texture sythesis. *Int. Journal of Computer and Information Sciences*, 3 :329–343, 1974.

[61] G. McLachlan and T. Krishnan. *The EM algorithm and extensions*. John Wiley & Sons, 1997.

[62] F. Meyer. Iteractive image transformations for an automatic screening of cervical smears. *Journal of Histochemistry and Cytochemistry*, 17 :128–135, 1979.

[63] F. Mokhtarian, S. Abbasi, and J. Kittler. Ecient and robust retrieval by shape content through curvature scale space. In *Proc. Int. Workshop on Image DataBases and MultiMedia Search*, pages 35–42, 1996.

[64] N. Morgan, D. Ellis, E. Fosler, A. Janin, and B. Kingsbury. Reducing errors by increasing the error rate : MLP acoustic modeling for broadcast news transcription. In *Proc. DARPA Broadcast News Workshop*, Herndon, Virginia USA, Feb. 1999.

[65] S. Mori, C. Y. Suen, and K. Yamamoto. Historical review of ocr research and developement. *Proceedings of the IEEE*, pages 1029–1058, 1992.

[66] Andrew Morris, Ljubomir Josifovski, Hervé Bourlard, Martin Cooke, and Phil Green. A neural network for classification with incomplete data : application to robust asr. In *Proc. Int. Conf. Speech and Language Processing*, Beijing, China, October 17-20 2000.

[67] S. Mukherjea, K. Hirata, and Y. Hara. Amore : a world wide web image retrieval engine. *World Wide Web*, 2(3) :115–132, 1999.

[68] M. Nagao, H. Tanabe, and K. Ito. Agricultural land use classification of aerial photographs by histogram similarity method. In *Proc. 3rd IEEE Joint Conf. on Pattern Recogntion*, pages 669–672, 1976.

[69] G. Nagy. At the frontiers of ocr. *Proceedings of the IEEE*, pages 1093–1100, 1992.

[70] 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.

[71] J.M. Odobez and D. Chen. Robust video text segmentation and recognition with multiple hypotheses. In *Proc. IEEE Int. Conf. on Image Processing*, volume 2, pages 433–436, Sep. 2002.

[72] M. Ohta, A. Takasu, and J. Adachi. Retrieval methods for english text with misrecognized ocr characters. In *Proc. Int. Conf. on Document Analysis and Recognition*, pages 950–956, 1997.

[73] Michael Ortega, Yong Rui, Kaushik Chakrabarti, Sharad Mehrotra, and Thomas S. Huang. Supporting similarity queries in MARS. In *Proc. ACM Multimedia*, pages 403–413, 1997.

[74] Edgar Osuna, Robert Freund, and Federico Girosi. Support vector machines : Training and applications. Technical Report AIM-1602, MIT, 1997.

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

[76] D.B. Parker. Learning logic. Technical Report 47, Center for Computational Research in Economics and Management Sciences, MIT, Cambridge, 1985.

[77] T. Pavlidis and Y.T. Liow. Integrating region growing and edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, pages 225–233, 1990.

[78] A. Pentland, B. Moghaddam, and T. Starner. View-based and modular eigenspaces for face recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Seattle, WA, Jun. 1994.

[79] A. Pentland, R. Picard, and S. Sclaroff. Photobook : Content-based manipulation of image databases. *Int. Journal of Computer Vision*, 18(3) :233–254, 1996.

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

[81] J. Platt. Fast training of support vector machines using sequential minimal optimization. In A. Smola B. Scholkopf, C. Burges, editor, *Advances in Kernel Methods : Support Vector Machines*. MIT Press, Cambridge, MA, 1998.

[82] P.Pala and S. Santini. Image retrieval by shape and texture. *Pattern Recognition*, 32(3) :517–527, 1999.

[83] W. K. Pratt. *Digital image processing*. John Wiley & Sons Inc., New York, 1978.

[84] J. M. S. Prewitt. Object enhancement and extraction. In B. S. Lipkin and A. Rosenfeld, editors, *Picture processing and psychopictorics*. Academic Press, New York, 1970.

[85] R. Redner and H. Walker. Mixture densities, maximum likelihood and the em algorithm. *SIAM*, 26 :195–239, 1984.

[86] S. V. Rice, F. R. Jenkins, and T. A. Nartker. The forth annual test of OCR accuracy. Technical report, ISRI, 1995.

[87] L .G. Roberts. Machine perception of three-dimensional solids. In J. P. Tippett, editor, *Optical and electro-optical information processing*. MIT Press, Cambridge, MA, 1965.

[88] F. Rosenblatt. *Principles of neurodynamics*. New York : Spartan, 1962.

[89] A. Rosenfeld and A. C. Kak. *Digital picture processing*. Academic Press, New York, 1982.

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

[91] Henry A. Rowley, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(1), 1998.

[92] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. *Learning internal representations by error propagation. In Parallel distributed processing Vol. 1*. MIT Press, Cambridge, 1986.

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

[94] T. Sato, T. Kanade, E. K. Hughes, M. A. Smith, and S. Satoh. Video OCR : indexing digital news libraries by recognition of superimposed caption. In *Proc. ACM Multimedia System Special Issue on Video Libraries*, pages 52–60, Feb. 1998.

[95] Shin'ichi Satoh, Yuichi Nakamura, and Takeo Kanade. Name-It : Naming and detecting faces in news videos. *IEEE MultiMedia*, 6(1) :22–35, 1999.

[96] B. Schacter, A. Rosenfeld, and L.S. Davis. Randommosaic models for textures. *IEEE Trans. on Systems, Man, and Cybernetics*, 8 :694–702, 1978.

[97] Bernt Schiele and James L. Crowley. Object recognition using multidimensional receptive field histograms. In *Proc. European Conf. Computer Vision*, pages 610–619, Cambridge, UK, 1996.

[98] H. Schneiderman and T. Kanade. Probabilistic modeling of local appearance and spatial relationships for object recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 45–51, 1998.

[99] Eugenio Di Sciascio, G. Mingolla, and Marina Mongiello. Content-based image retrieval over the web using query by sketch and relevance feedback. In *Proc. Visual Information and Information Systems*, pages 123–130, 1999.

[100] Stan Sclaroff, Leonid Taycher, and Marco LaCascia. Imagerover : A content-based image browser for the world wide web. In *Proc. Workshop on Content-based Access of Image and Video Libraries*, 24 1997.

[101] Jean Serra. *Image Analysis and Mathematical Morphology*. Academic Press, London, 1982.

[102] K. Shearer, H. Bunke, and S. Venkatesh. Video indexing and similarity retrieval by largest common subgraph detection using decision trees. *Pattern Recognition*, 34(5), 2000.

[103] J. Smith and S. Chang. Querying by color regions using the visualseek content-based visual query system. In *Proc. Int. Joint Conf. Artificial Intelligence*, 1996.

[104] 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.

[105] I. E. Sobel. *Camera models and machine perception*. Ph.D thesis, Stanford university, 1970.

[106] 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*, pages 57–63, 1999.

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

[108] A. TAKASU, S. SATOH, and E. ATSURA. A document understanding method for database construction of an electronic library. In *Proc. Int. Conf. Pattern Recognition*, pages 463–466, 1994.

[109] C. W. Therrien. An estimation-theoretic approach to Terrain image segmentation. *Computer Graphics and Image Processing*, 22 :313–326, 1983.

[110] J. Toriwaki and S. Yokoi. Distance transformations and skeletons of digitized pictures with applications. In L. N. Kanal and A. Rosenfeld, editors, *Progress in Pattern Recognition*, pages 187–264. North–Holland, Amsterdam, 1981.

[111] J. T. Tou and Y. S. Chang. An approach to texuture pattern analysis and recognition. In *Proc. IEEE Conf. on Decesion and Control*, pages 1–7, 1976.

[112] O. Trier, A. Jain, and T. Taxt. Feature extraction methods for character recognition - a survey. *Pattern Recognition*, 29 :641–662, 1993.

[113] M. Turk and A. Pentland. Eigenface for recognition. *Journal of Cognitive Neuro-science*, 3(1) :70–86, 1991.

[114] D. Valentin, H. Abdi, A. O'Toole, and G.W. Cotrell. Connectionist models of face processing : A survey. *Pattern Recognition*, pages 1209–1230, 1994.

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

[116] P. werbos. *Beyond regression : new tools for prediction and analysis in the behavioral sciences*. PhD Thesis, Harvard University, 1974.

[117] M. Werman and S. Peleg. Multiresolution texture signatures using Min-Max operators. In *Proc. 7th Int. Conf. on Pattern Recogntion*, pages 97–99, 1984.

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

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

[120] V. Wu and R. Manmatha. Document image clean-up and binarization. In *Proc. SPIE Symposium on Electronic Imaging*, pages 263–273, 1998.

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

[122] V. Wu, R. Manmatha, and E. M. Riseman. Textfinder : An automatic system to detect and recognize text in images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(20) :1224–1229, 1999.

[123] Ming-Hsuan Yang and Narendra Ahuja. Detecting human faces in color images. In *Proc. Int. Conf. on Image Processing*, volume 1, pages 127–130, 1998.

[124] Di Zhong, HongJiang Zhang, and Shih-Fu Chang. Clustering methods for video browsing and annotation. In *Proc. Storage and Retrieval for Image and Video Databases*, pages 239–246, 1996.

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

[126] Y. Zhong, H. Zhang, and A.K. Jain. Automatic caption localization in compressed video. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(4) :385–392, 2000.

[127] A. I. Zobrist and W. B. Thompson. Building a distance function for Gestalt grouping. *IEEE Trans. on Computers*, 4 :718–728, 1985.

[128] S. W. Zucker. On the foundations of texture : A transformational approach. Technical Report 311, University of Maryland, 1974.

Datong Chen

Address :	Rue du Simplon 4	Phone :	+41 27 721 7756
	CH-1920 Martigny	email :	chen@idiap.ch
	Switzerland		

Education

2000–	Docteur ès Sciences (anticipated Oct. 2003) The Swiss Federal Institute of Technology Lausanne (EPFL), Switzerland Dissertation : *Text Detection and Recognition in Images and Video Sequences*.
1995–1997	Master of Computer Science and Engineering Harbin Institute of Technology, Harbin, P. R. China Thesis : *Camera-Based Head Gesture Analysis*.
1991–1995	Bachelor of Computer Science and Engineering Harbin Institute of Technology, Harbin, P. R. China Thesis : *3-D Human Face Reconstruction Based on Structure Lighting*.

Professional Experience

2000–	Dalle Molle Institute for Perceptual Artificial Intelligence (IDIAP), Switzerland Computer Vision Group Research Assistant
1998–1999	Karlsruhe University, Germany Telecooperation Office, Telecommunication Institute Research Assistant
1997–1998	Motorola-NCIC Joint R&D Lab, P. R. China MPEG-4 Group Project Manager and Software Engineer
1996–1997	Motorola-NCIC Joint R&D Lab, P. R. China MPEG-4 Group Research Assistant

Computer Experience

Operating Systems : UNIX/Linux, Windows, Palm OS

Languages : C++, C, Java, Matlab, Assembly, Pascal, Basic

Science Duties

Reviewer for journals : Signal Processing
Machine Vision and Applications

Languages

Mandrin (native), English (fluent), French (moderate), German (basic)

Publications in Journals

In submitting

1. D. Chen and J-M. Odobez, "Video Text Recognition Using Sequential Monte Carlo and Error Voting Methods".

Submitted

2. D. Chen, J-M. Odobez and J-P. Thiran, "Particle Filtering Based Video Text Segmentation", submitted to International Journal on Pattern Recognition and Artificial Intelligence (IJPRAI).

To be Published

3. D. Chen, J-M. Odobez, and H. Bourlard, "Text Detection and Recognition in Images and Videos", to be published in Pattern Recognition (PR).
4. D. Chen, J-M. Odobez and J-P. Thiran, "A Localization/Verification Scheme for Finding Text in Images and Videos Based on Contrast Independent Features and Machine Learning Methods", to be published in Signal processing : Image Communication (SPIC).

Publications in Conferences and Workshops

5. D. Chen and J-M. Odobez, "Sequential Monte Carlo Video Text Segmentation", in Proc. IEEE International Conference on Image Processing (ICIP), Barcelona, 2003.
6. D. Chen, J-M. Odobez, and H. Bourlard, "Text Segmentation and Recognition in Complex Background Based on Markov Random Field", in Proc. International Conference on Pattern Recognition (ICPR), Quebec City, 2002.
7. J-M. Odobez and D. Chen, "Video Text Recognition Based on Markov Random Field and grayscale consistency constraint", in Proc. IEEE International Conference on Image Processing (ICIP), Rochester, 2002.
8. D. Chen, H. Bourlard, and J-P. Thiran, "Text identification in complex background using SVM", in Proc. International Conference on Computer Vision and Pattern Recognition (CVPR), Hawaii, 2001.

9. D. Chen, K. Shearer, and H. Bourlard, "Text enhancement with asymmetric filter for video OCR", in Proc. International Conference on Image Analysis and Processing (ICIAP), Palermo, 2001.
10. D. Chen, K. Shearer, and H. Bourlard, "Video OCR for sport video annotation and retrieval", in Proc. IEEE International Conference on Mechatronics and Machine Vision in Practice (ICM2VP), Hong Kong, 2001.
11. D. Chen and J. Luettin, "Multiple hypothesis Video OCR", in Proc. IAPR workshop on Document Analysis System (DAS), Rio de Janeiro, 2000.
12. D. Chen, H. Gellerson, "Cognition Based Collaboration Log for CSCW", in Proc. International Conference on Cognitive Science (ICCS), Tokyo, 1999.
13. D. Chen, A. Schmidt, H. Gellerson, "An Architecture for Multi-Sensor Fusion in Mobile Environments", in Proc. International Conference on Information Fusion (ICIF), Sunnyvale, California USA, 1999.
14. D. Chen, H. Gellerson, "Recognition and Reasoning in an Awareness Support system for Generation of Storyboard-like Views of Recent Activity", in Proc. ACM International Conference on Supporting Group Work (ACM SIGGROUP), Phinex, 1999.
15. F. Wu, W. Gao, Y. Xiang, and D. Chen, "On-line sprite encoding with large global motion estimation", in Proc IEEE International Conference on Data Compression (ICDC), Snowbird, 1998.
16. D. Chen, W. Gao, X. Chen, "A New Approach to Recover 3D Shape Based on Structure-Lighting", in Proc IEEE International Conference on Signal Processing (ICSP), Beijing, 1996.
17. D. Chen, W. Gao, X. Chen, "Build Perception Model To Multimodal Interface", in Proc International Conference on Multimodal Interface (ICMI), Beijing, 1996.

Selection of Technical Reports

18. D. Chen and J-M. Odobez, "A new method of contrast normalization for verification of extracted video text having complex backgrounds", RR-02-16, IDIAP, Nov. 08 2002.
19. D. Chen and J-M. Odobez, "Comparison of support vector machine and neural network for text texture verification", RR-02-19, IDIAP, Apr. 12 2002.
20. D. Chen and J. Luettin, "A survey of text detection and recognition in images and videos", RR-00-38, IDIAP, Aug. 2000.

Publications in Chinese

(7 papers unlisted here)