Submitted to *ACM/Springer Multimedia Systems Magazine*

REIHE INFORMATIK
9/98

Automatic Text Segmentation and Text Recognition for Video Indexing

Rainer Lienhart and Wolfgang Effelsberg
Universität Mannheim
Praktische Informatik IV
L15,16
D-68131 Mannheim

revised Mai 1998

Automatic Text Segmentation and Text Recognition for Video Indexing

Rainer Lienhart and Wolfgang Effelsberg
University of Mannheim
Praktische Informatik IV
68131 Mannheim, Germany
{lienhart, effelsberg}@pi4.informatik.uni-mannheim.de
phone: +49-621-292-3381 fax: +49-621-292-5745

ABSTRACT
Efficient indexing and retrieval of digital video is an important function of video databases. One powerful index for retrieval is the text appearing in them. It enables content-based browsing. We present our methods for automatic segmentation of text in digital videos. The output is directly passed to a standard OCR software package in order to translate the segmented text into ASCII. The algorithms we propose make use of typical characteristics of text in videos in order to enable and enhance segmentation performance. Especially the inter-frame dependencies of the characters provide new possibilities for their refinement. Then, a straightforward indexing and retrieval scheme is introduced. It is used in the experiments to demonstrate that the proposed text segmentation algorithms together with existing text recognition algorithms are suitable for indexing and retrieval of relevant video sequences in and from a video database. Our experimental results are very encouraging and suggest that these algorithms can be used in video retrieval applications as well as to recognize higher semantics in videos.

KEYWORDS: video processing, character segmentation, character recognition, OCR, video indexing, video content analysis

1 INTRODUCTION

There is no doubt that video is an increasingly important modern information medium. Setting free its complete potential and usefulness requires efficient content-based indexing and access. One powerful high-level index for retrieval is the text contained in videos. This index can be built by detecting, extracting and recognizing such text. It enables the user to submit sophisticated queries such as a listing of all movies featuring John Wayne or produced by Steven Spielberg. Or it can be used to jump to news stories about a specific topic, since captions in newscasts often provide a condensation of the underlying news story. For example, one can search for the term “Financial News” to get the financial news of the day. The index can also be used to record the broadcast time and date of commercials, helping the agents checking to see that a client’s commercial has been broadcasted at the arranged time on the arranged television channel. Many other useful high-level applications are imaginable once text can be recognized automatically and reliably in digital video.

In this paper we present our methods for automatic text segmentation in digital videos. The output is directly passed to a standard OCR software package in order to translate the segmented text into ASCII. We also demonstrate that these two processing steps enable semantic indexing and retrieval. To ensure better segmentation performance our algorithms analyze typical characteristics of text in video. Interframe dependencies of text incidences promise further refinement.

Text features are presented in Section 2, followed by a description of our segmentation algorithms in Section 3 which are based on the features stated in Section 2. Information about the text recognition step is given in Section 4. Then, in Section 5 we introduce a straightforward indexing and retrieval scheme, which is used in our experiments to demonstrate the suitability of our algorithms for indexing and retrieval of video sequences. The experimental results of each step - segmentation, recognition and retrieval - are discussed in Section 6. They are investigated independently for three different film genres: feature films, commercials and newscasts. Section 7 reviews related work, and Section 8 concludes the paper.

2 TEXT FEATURES

Text may appear anywhere in the video and in different contexts. It is sometimes a carrier of important information, at other times its content is of minor importance and its appearance is only accidental. Its significance is related to the nature of its appearance. We discriminate between two kinds of text: *scene text* and *artificial text.* Scene text appears as a part of and was recorded with the scene, whereas artificial text was produced separately from the video shooting and is laid over the scene in a post-processing stage, e.g. by video title machines.

Scene text (e.g. street names or shop names in the scene) mostly appears accidentally and is seldom intended (An exception, for instance, are the commercials in the new James Bond movie). However, when it appears unplanned, it is often of minor importance and generally not suitable for indexing and retrieval. Moreover, due to its incidental and the thus resulting unlimited variety of its appearance, it is hard to detect, extract and recognize. It seems to be impossible to identify common features, since the characters can appear under any slant, tilt, in any lighting and upon straight or wavy surfaces (e.g. on a T-shirt). Scene text may also be partially occluded.

In contrast, the appearance of artificial text is carefully directed. It is often an important carrier of information and herewith suitable for indexing and retrieval. For instance, embedded captions in TV programs represent a highly condensed form of key information on the content of the video [23]. There, as in commercials, the product and company name are often part of the text shown. Here, the product name is often scene text but used like artificial text. Therefore, in this paper we concentrate on extraction of artificial text. Fortunately, its appearance is subjected to many more constraints than that of scene text since it is made to be read easily by viewers.

The mainstream of artificial text appearances is characterized by the following features:

- Characters are in the foreground. They are never partially occluded.
- Characters are monochrome.
- Characters are rigid. They do not change their shape, size or orientation from frame to frame.
- Characters have size restrictions. A letter is not as large as the whole screen, nor are letters smaller than a certain number of pixels as they would otherwise be illegible to viewers.
- Character are mostly upright.
- Characters are either stationary or linearly moving. Moving characters also have a dominant translation direction: horizontally from right to left or vertically from bottom to top.
- Characters contrast with their background since artificial text is designed to be read easily.
- The same characters appear in multiple consecutive frames.
- Characters appear in clusters at a limited distance aligned to a horizontal line, since that is the natural method of writing down words and word groups.

Our text segmentation algorithms are based on these features. However, they also take into account that some of these features are relaxed in practice due to artifacts caused by the narrow bandwidth of the TV signal or other technical imperfections.

3 TEXT SEGMENTATION

Prior to presenting our feature-based text segmentation approach we want to outline clearly what the text segmentation step should do. This not only affects the algorithms employed during the text segmentation but also those which can be used during the text recognition step. Unlike in our previous work [10][11], where individual characters still may consist of several regions of different colors after the text segmentation step, and most related work, the objective of the text segmentation step here is to produce a binary image that depicts the text appearing in the video (see Figure 9). Hence, standard OCR software packages can be used to recognize the segmented text.

Note that all processing steps are performed on color images in the RGB color space, not on grayscale images.

3.1 Color Segmentation

Most of the character features listed in Section 2 cannot be applied to raw images; rather, objects must be already available. In addition, some of the features require that actual characters be described by exactly one object in order to discriminate between character and non-character objects.

Therefore, in an initial step, each frame is to be segmented into suitable objects. The character features monochromaticity and contrast with the local environment qualify as a grouping and separation criterion for pixels, respectively. Together with a segmentation procedure which is capable of extracting monochrome regions that contrast highly to their environment under significant noise, suitable objects can be constructed. Such a segmentation procedure preserves the characters of artificial text occurrences. Its effect on multicolored objects and/or objects lacking contrast to their local environment is insignificant here. Subsequent segmentation steps are likely to identify the regions of such objects as non-character regions and thus eliminate them.

As a starting point we over-segment each frame by a simple yet fast region-growing algorithm [27]. The threshold value for the color distance is selected by the criterion to preclude that occurring characters merge with their surroundings. Hence, the objective of the region-growing is to strictly avoid any under-segmentation of characters (under normal conditions).

By and by, then, regions are merged to remove the over-segmentation of characters while at the same time avoiding their under-segmentation. The merger process is based on the idea that the use of standard color segmentation algorithms such as region-growing [27] or split-and-merge [6] is improper in highly noisy images such as video frames, since these algorithms are unable to distinguish isotropic image structures from image structures with local orientation. Given a monochrome object in the frame under high additive noise, these segmentation algorithms would always split up the object randomly into different regions. It is the objective of the merger process to detect and merge such random split-ups of objects.

We identify random split-ups via a frame's edge and orientation map. If the border between two regions does not coincident with a roughly perpendicular edge or local orientation in the close neighborhood, the separation of the regions is regarded as incidentally due to noise, and they are merged. The advantage of edges over orientation is that they show good localization but may not be closed and/or too short to give a reliable estimate of the angle of the edge. In contrast, local orientation shows poor localization but determines precisely the angle of contrast change. Together they allow to detect most random split-ups of objects.

Edges are localized by means of the Canny edge detector extended to color images, i.e. the standard Canny edge detector is applied to each image band. Then, the results are integrated by vector addition. Edge detection is completed by non-maximum suppression and contrast enhancement. Dominant local orientation is determined by the inertia tensor method as presented in [7].

The color segmentation is completed by merging regions of similar colors. This segmentation algorithms yields an excellent segmentation of a video with respect to the artificial characters. Usually most of them will now consist of one region.

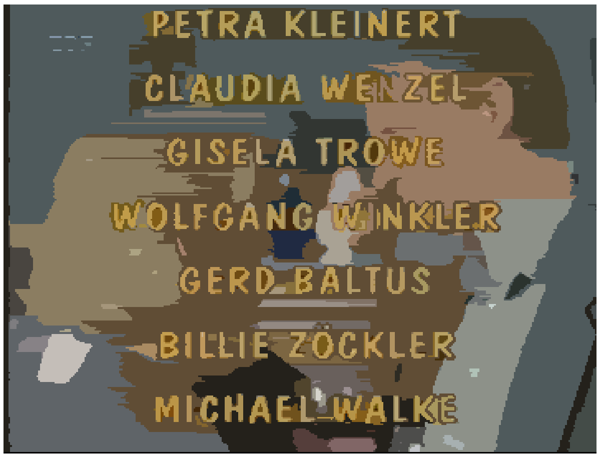


Figure 1: Result of the color segmentation

Note, that the color segmentation employed can be classified as an inhomogeneous and anisotropic segmentation algorithm which preserves fine structures such as characters. More details about this segmentation step can be found in [9].

3.2 Contrast Segmentation

For our task a video frame can also be segmented properly by means of the high contrast of the character contours to their surroundings and by the fact that the strength of the stroke of a character is considerably less than the maximum character size.

For each video frame a binary contrast image is derived in which set pixels mark locations of sufficiently high absolute local contrast. The absolute local color contrast at position $I(x,y)$ is measured by

$$Contrast_{abs,color}(x,y) = \sum_{k=-r}^{r}\sum_{l=-r}^{r} G_{k,l} \cdot \|I_{x,y} - I_{x-k,y-l}\| ,$$

where $\| \|$ denote the color metric employed (here City-block distance), $G_{k,l}$ a Gaussian filter mask, and r the size of the local neighborhood.

Next, each set pixel is dilated by half the maximum expected strength of the stroke of a character. As a result, all character pixels as well as some non-character pixels which also show high local color contrast are registered in the binary contrast image (see Figure 2). Likewise for color segmentation, the contrast threshold is selected in such a way that, under normal conditions, all character-pixels are captured by the binary contrast image.

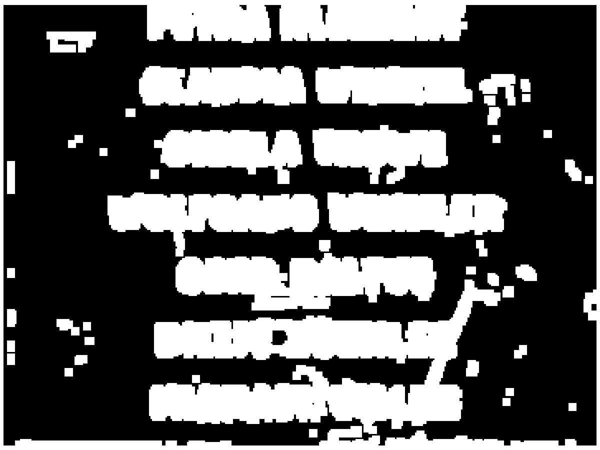


Figure 2: Contrast Segmentation

Finally, all regions which overlap by less than 80% with the set pixels in the binary contrast image are discarded.

3.3 Geometry Analysis

Characters are subjected to certain geometric restrictions. Their height, width, width-to-height ratio and compactness do not take on any value, but usually fall into specific ranges of values. If a region's geometric features do not fall into these ranges of values the region does not meet the requirements of a character region and is thus discarded.

The precise values of these restrictions depend on the range of the character sizes selected for segmentation. In our work, the geometric restrictions have been determined empirically based on the bold and bold italic versions of the four TrueType fonts Arial, Courier, Courier New and Times New Roman at the sizes of 12pt, 24pt, and 36 pt (4*3*4 = 24 fonts in total). The measured ranges of width, height, width-to-height ratio and compactness are listed in Table 1.

Geometric Restriction	Min	Max
width	1	31
height	4	29
width-to-height-ratio	0.56	7.00

Geometric Restriction	Min	Max
compactness	0.21	1.00

Table 1:Empirical measured ranges of values using 24 bold TrueType fonts

Since we have assumed that each character consists of exactly one region after the color segmentation step, the empirical values can be used directly to rule out non-character regions. All regions which do not comply with the measured geometric restrictions are discarded. Since the 24 fonts analyzed are only a small sample of all possible fonts, the measured ranges were extended slightly. The following ranges have been used to describe potential character regions:

- $height \in [4, 90]$
- $width \in [1, 120]$
- $ratio \in [0.4, 1]$
- $compactness \in [0.15, 1]$

The segmentation result after applying the geometric restrictions to the sample video is shown in Figure 3.

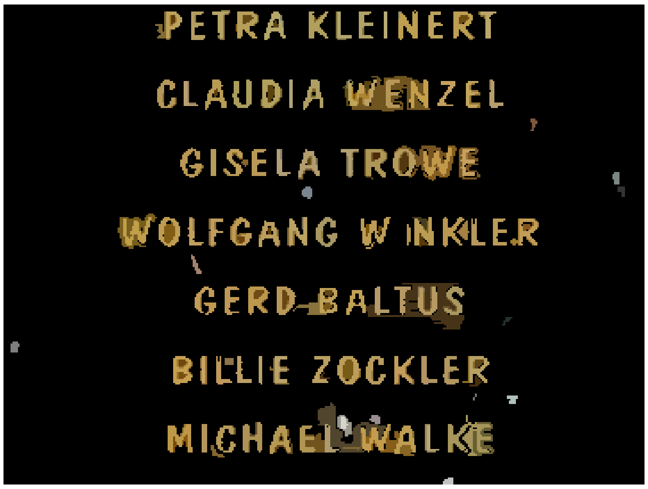


Figure 3: Result after the analysis of the regions' geometric features (246 regions left)

3.4 Texture Analysis

Text appearances are distinguished by a characteristic spatial pattern on the word and text line level: In at least one direction, the direction of writing, periodically strong contrast fluctuations can be notified. This periodicity is peculiar to words and text lines.

This particular texture pattern of text was already used in [8] to separate text columns from images, graphics and other non-text parts of a document. In this work, however, it is used to separate text within images. Also, and unlike in [22], it is not used as the first feature in the text segmentation process. This has several advantages. The color segmentation identifies many large color regions touching the characters. Thus, by simply eliminating those regions which do not comply with the geometric features of characters, parts of the characters' outlines can be cut out precisely. If, however, text were segmented first by means of its characteristic texture, we would lose this advantageous feature of the color segmentation. Especially the large non-character regions would be reduced to small left-overs which often could no longer be ruled out by their geometric features.

Against a (more or less) uniform background one can observe that the direction of the contrast fluctuations changes rhythmically. In most cases, this regular alternation of the contrast direction can also be observed in text superimposed on an inhomogeneous background since it is then often surrounded by a type of aura. This aura is usually added during video production to improve the legibility (see "Buttern" in Figure 4(a)). Exploiting this feature in a suitable manner enables edges between non-character regions to be distinguished from edges between a character and a non-character region.

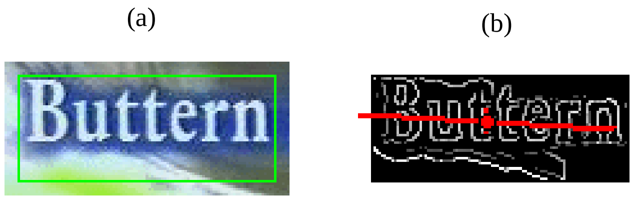


Figure 4: Example of text surrounded by a blue aura to improve readability (a) and its writing direction automatically determined (b)

In detail, the texture analysis consists of two steps:

1. The extraction of potential words or lines of text and the estimate of their writing direction.
2. The test whether the potential words and lines of text exhibit frequent contrast alternations in the estimated writing direction.

Potential words and lines of text can be determined by enlargement of the potential character regions of a suitable size. Due to the great proximity of the characters of a word and sometimes also between the words in a text line, their regions are merged into a so-called *potential word cluster.* Note that often a small number of non-character regions are merged into potential word clusters, and sometimes non-character regions merge into a cluster on their own. The amount of enlargement necessary depends on the average running width (i.e. the size of the character spaces), which in turn is influenced by the character size. In the experiments, the necessary expansion was determined experimentally to be 2 pixels.

Next, the writing direction within a potential word cluster is to be estimated. Unlike existing approaches, we do not assume that text is aligned horizontally [19][22]. Although this assumption is quite reasonable in many cases, it restricts the application domain of the text segmentation algorithm unnecessarily.

The previous segmentation steps having already separated the words from large non-character regions, the writing direction of a word cluster can then be estimated via the direction of its main axis. This is defined as the angle between the x axis and the axis, around which the word cluster can be rotated with minimum inertia (see Figure 4(b)). In accordance with [7], this direction is determined by

$$\phi = \frac{1}{2} arc\tan\frac{2m_{1,1}}{m_{2,0}-m_{0,2}}$$

with the moments

$$m_{p,q} = \sum_{x_1,x_2} (x_1-\mu_{x_1})^p(x_2-\mu_{x_2})^q .$$

Two cases are to be considered more precisely:

1. The moments of inertia of the main and secondary axis (designated J_1 and J_2) differ only insignificantly (J_1/ J_2 <1.5). This happens only in the case of very short words such as the word “IN”. Since in this case the estimated direction is likely to deviate greatly from the actual writing direction (e.g. diagonal for “IN”), clusters with J_1/ J_2 <1.5 are never rejected.
2. Errors as to the estimated writing direction may also appear for short words and J_1/ J_2 >= 1.5 if the word cluster takes on a crooked form due to an unfavorable character sequence such as “You”. However, the magnitude of the inaccuracy of the estimated writing direction keeps within a scope of ± 20 degree. Thus, it can be expected that most characters in a word cluster are still cut by the main axis. Together with the necessary tolerance in the selection of edges in writing direction in the subsequent texture analysis step, no special precautions are necessary.

Once the writing direction of a potential word cluster has been determined, the texture can be analyzed. Decisive for the exact parameters of the texture analysis of a cluster is the required minimal number N_{min} of characters which have to be in close vicinity to each other in writing direction. In general, we demand that at least $2N_{min}$ edges have to be present within a range of $2C_{maxDist}$. In the experiments the parameters were set to N_{min} = 2 and $C_{maxDist}$ = 1.5 * (max. character width) = 46. The edges and their directions were determined by means of the Canny operator, extended to color images.

Numerous non-character regions, which are difficult to identify in the preceding steps, can now be found and removed by means of texture analysis. The result for the sample video is depicted in Figure 5.

3.5 Motion Analysis

Another feature of artificial text occurrences is that they either appear statically at a fixed position on the screen or move linearly across the screen. More complicated motion paths are extremely improbable between the on- and dissassembly of text on the screen. Any other, more complex motion would make it much harder to track and thus read the text, and this would contradict the intention of artificial text occurrences. This feature applies both to individual characters and whole words. Note, however, that in commercials this rule is sometimes broken intentionally in order to carry an unconscious message to the spectators.

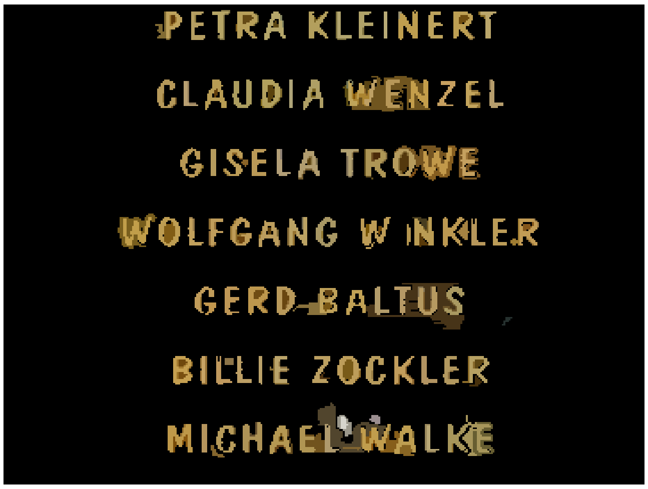


Figure 5: Result after texture analysis (242 areas)

It is the objective of motion analysis to identify regions which cannot be tracked or which do not move linearly, in order to reject them as non-character regions. Unlike in our previous system [10][11] and all other related work, the object here is to track the characters not only over a short period of time but over the entire duration of their appearance in the video sequence. This enables us to extract exactly one bitmap of every text occurring in the video - a feature which e.g. is needed by our video abstracting system [12]. Motion analysis can also be used to summarize the multiple recognition results for each character to improve the overall recognition performance.

In addition, a secondary objective of motion analysis is that the output should be suitable for standard OCR software packages. Essentially this means that a binary image must be created.

Formation of Character Objects

A central term in motion analysis is the *character object C*. It gradually collects from contiguous frames all those regions which belong to one individual character. Since we assume that a character consists of exactly one region per image after the color segmentation step, at most one region per image can be contained in a character object.

A character object C is described formally by the triple $(A, [a, e], v)$. A stands for the feature values of the regions which were assigned to the character object and which are employed for comparison with other regions, $[a, e]$ for the frame number interval of the regions’ appearance and v for the estimated and constant speed of the character in *pixels/ frame*.

In a first step, each region r_i in frame n is compared against each character object $C_j, j \in \{1, \ldots, J\}$ constructed from the frames 1 to n-1. To this are compared the mean color, size and position of the region r_i and the character objects $C_j, j \in \{1, \ldots, J\}$. In addition, if a character object consists of at least two regions, each candidate region r_i is checked whether it fits smoothly into a linear motion path of the character object.

If a region is sufficiently similar to the best-matching character object, a copy of that character object will be created, and the region added to the initial character object. We need to copy the character object before assigning a region to it since, at most, one region ought to be assigned to each character object per frame. Due to necessary tolerances in the matching procedure, however, it is easy to assign the wrong region to a character object. The falsely assigned region would block that character object for the correct region. By means of the copy, however, the correct region can still be matched to its character object. It is decided at a later stage in motion analysis, whether the original character object or one of its copies is to be eliminated.

If a region does not match to any character object existing so far, a new character object will be created and initialized with the region.

Also, if a region best fits to a character object that consists of fewer than three regions, a new character object is created and initialized with the region. This prevents a possible starting region of a new character object from being sucked up by a still shorter and thus unstable character object.

Finally upon the processing of frame *n*, all character objects which offend against the features of characters are eliminated. In detail

- all copies of a character object are discarded which were created in frame *n*-1 but not continued in frame *n* as well as
- all character objects which could not be continued during the last 6 frames or whose forecasted location lies outside the frame
  and
  - whose regions do not fit well to each other, (note that the requirements are less strict during the construction of the character object, becoming more restrictive once a character object is finished.),
  - which are shorter than 5 frames,
  - which consist of fewer than 4 regions or
  - whose regions move faster than 9 *pixels*/*frame*.

The values of the parameters were determined experimentally.

After processing all frames of the video sequence some character objects will represent a subset of some larger character objects. This peculiarity results directly from the design of the formation procedure for character objects. Whenever a region was added to a character object consisting of fewer than 3 regions, a new character object (initialized with that region) was created, too. Thus, two character objects C_1 and C_2 are merged if $[a_1, e_1] \subseteq [a_2, e_2]$, $v_1 \sim v_2$, and $A_1 \subseteq A_2$.

Formation of Text Objects

In order to, firstly, eliminate character objects standing alone which either represent no character or a character of doubtful importance, and secondly, to group character objects into words and lines of text, character objects are merged into so-called text objects. A *valid text object* $T_i = \{C_{i_1}, \ldots, C_{i_{n(i)}}\}$ is formed by at least three character objects which approximately

1. occur in the same frames,
2. show the same (linear) motion,
3. are the same mean color,
4. lie on a straight line and
5. are neighbors.

These grouping conditions result directly from the features of Roman letters.

We use a fast heuristics to construct text objects: At the beginning all character objects belong to the set of the character objects to be considered. Then, combinations of three character objects are built until they represent a valid text object. These character objects are moved from the set of the character objects into the new text object. Next, all character objects remaining in the set which fit well to the new text object are moved from the set to the text object. This process of finding the next valid text object and adding all fitting character objects is carried out until no more valid text objects can be formed or until all character objects are grouped to text objects.

To avoid splintering multi-line horizontal text into vertical groups, this basic grouping algorithm must be altered slightly. In a first run, only text objects are constructed whose characters lie roughly on a horizontal line. The magnitude of the gradient of the line must be less than 0.25. In a second run, character groups are allowed to run into any direction.

During our experiments we noticed that a character is sometimes described by two valid character objects which register the character in different but interleaved frames (see Figure 6). In a further processing step, such character objects are merged.

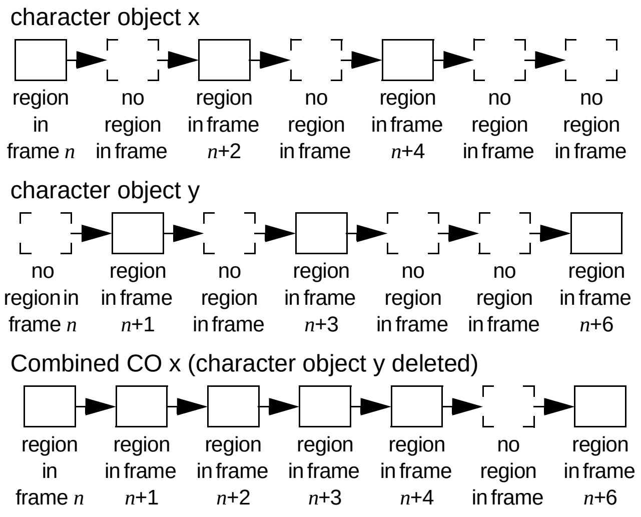


Figure 6: Merging two interleaved character objects

The text objects constructed so far are still incomplete. The precise temporal range $[a_i, e_i]$ of occurrence of each character object C_i of a text object are likely to differ somewhat. In addition, some character objects have gaps at frames in which, for various reasons, no appropriate region was found.

The missing characters are now interpolated. At first, all character objects are extended to the maximum length over all character objects of a text object, represented by $[min\{a_{i_1}, \ldots, a_{i_{n(i)}}\}, max\{b_{i_1}, \ldots, b_{i_{n(i)}}\}]$. The missing regions are interpolated in two passes: a forward and a backward pass. The backward pass is necessary in order to predict the regions missing at the beginning of a character object. This procedure is depicted in Figure 7.

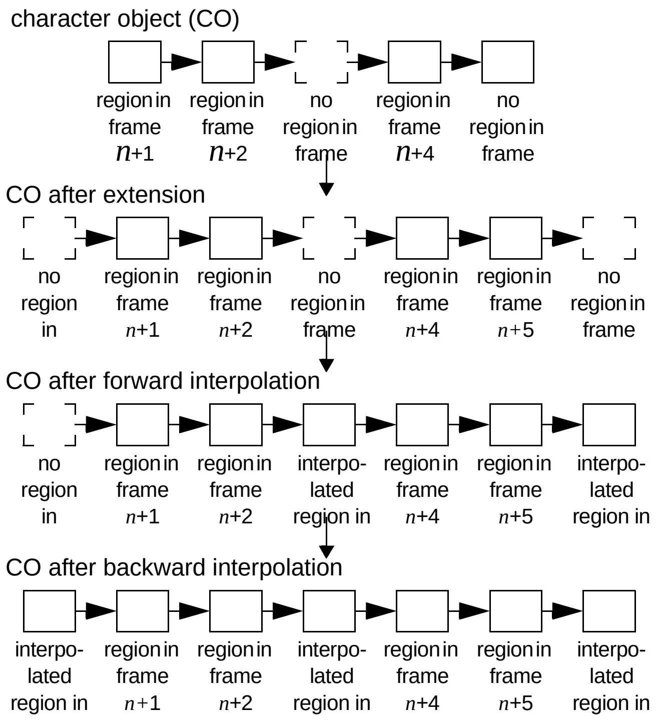


Figure 7: Completion of character objects (CO)

Visual Presentation of the Results

Motion analysis delivers detailed information about which text occurs when and where. In order to enable further processing in a most flexible way, three different kinds of output images are created:

1. A binary image per frame showing the extracted characters at their original location (see Figure 8)
2. A binary image per frame showing each extracted text object on a new line. The relative positions of the characters within a text object are preserved (see Figure 9).
3. A binary and color image showing all text objects extracted from the video sequence (see Figure 10). We call this representation a text summary.

4 TEXT RECOGNITION

For text recognition we incorporated the OCR-Software Development Kit Recognita V3.0 for Windows 95 into our

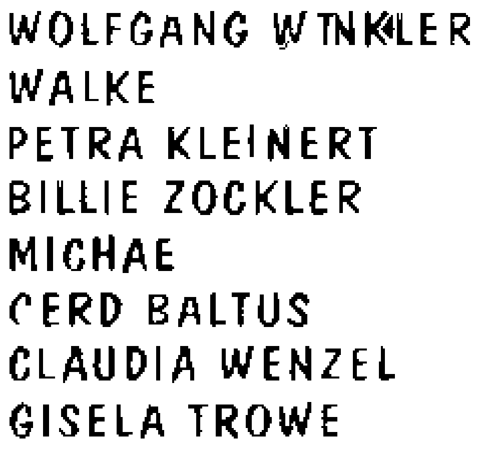


Figure 9: Rearranged binary image per frame

implementation. Two recognition modules are offered: one for typed and one for handwritten text. Since most artificial text occurrences appear in block letters, the OCR module for typed text was used to translate the rearranged binary text images into ASCII text. The recognized ASCII text in each frame was written out into a database file.

Due to the unusually small character size for such software packages, the recognition performance partially varied considerably, even from frame to frame as illustrated in Figure 11.

Principally, the recognition result can be improved by taking advantage of the multiple instances of the same text over consecutive frames, because each character in the text often appears somewhat altered from frame to frame due to noise, and changes in background and/or position. Combining their recognition results into one final character result might improve the overall recognition performance. However, as we will see in the next section, it is not needed by our indexing scheme.

5 INDEXING AND RETRIEVAL

The upcoming question is how to use the text recognition result to index and retrieve digital videos. A related question

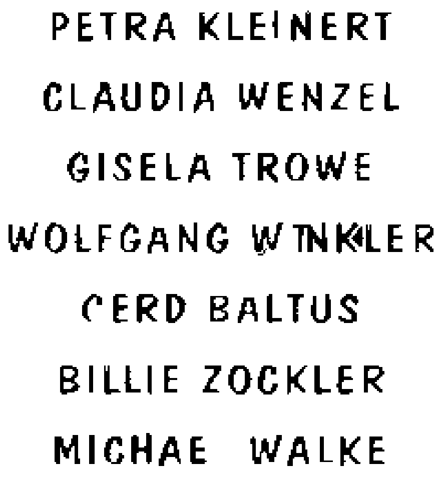


Figure 8: Binary image per frame at original location

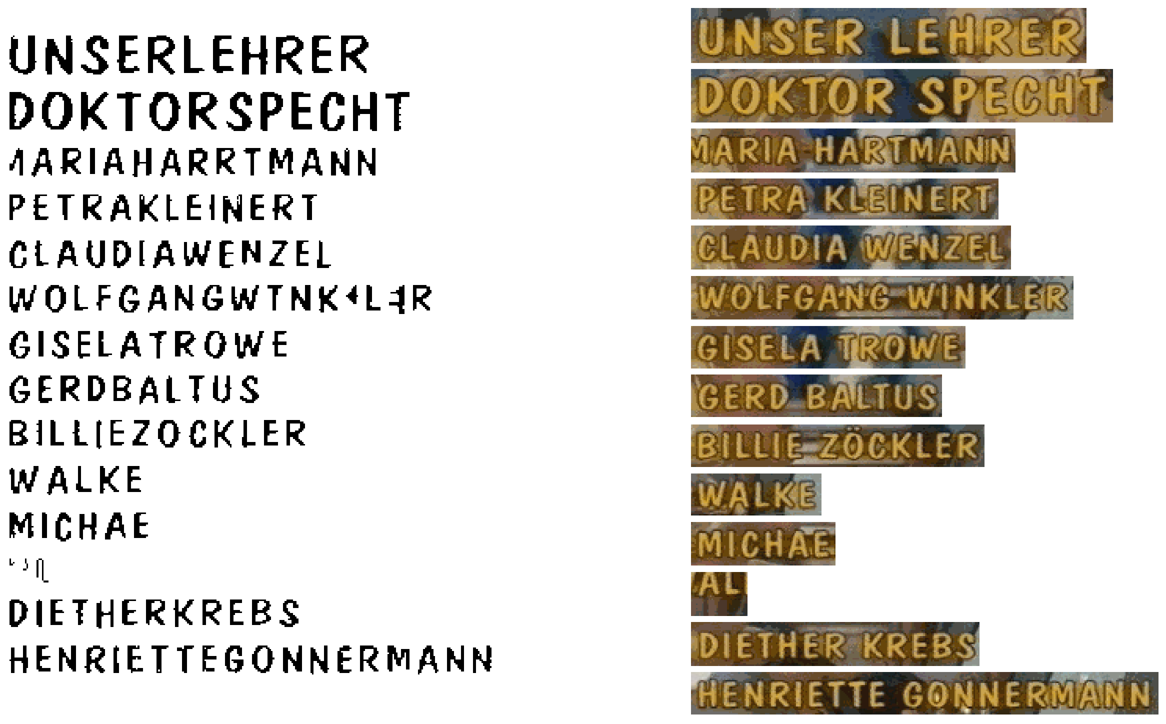
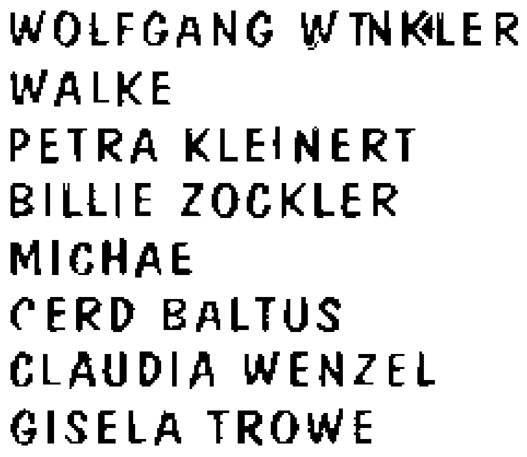


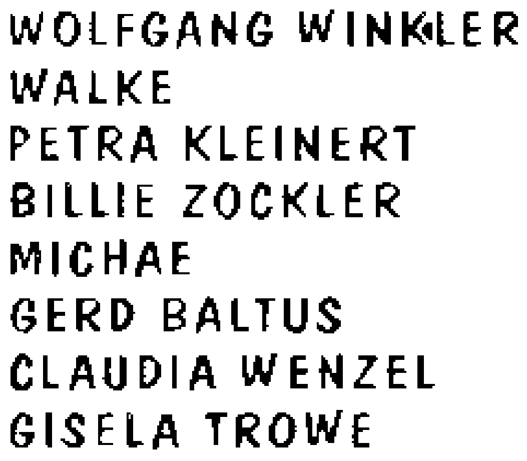
Figure 10: Text summary

frame *n*



```
WOLFCANO ISlTNKnLER
WALKE
PETRA KLEINERT
BILLIE ZOCKLER
MICHAE
fERD BALTUS
CLAUDIA WENZEL
OISELA TROWE
```

frame *n*+1



```
WOLFGANG WINKLER
WALKE
PETRA KLEINERT
BILLIE ZOCKLER
MICNAE
GERD BALTUS
CLAUDIA WENZEL
GISELA TROWE
```

Figure 11: Bildein- und Textausgabe zweier aufeinanderfolgender Videobilder

with significant impact on the answer to the original question is what minimal text recognition quality should we assume/demand?

Numerous different font families in all sizes, and sometimes even artistic fonts, are used in artificial text in digital videos. Therefore, OCR errors are very likely. We also have to deal with many garbage characters which result from non-character regions that could neither be eliminated by our system nor by the OCR module. Consequently, our indexing and retrieval scheme should deal well with a poor recognition quality.

Indexing

The indexing scheme is quite simple. The recognized characters for each frame are stored after deletion of all text lines with fewer than 3 characters. The reason for this deletion is that, as experience shows, text lines with up to two characters are produced mainly by non-character objects and, even if not, consist of semantically weak words such as “a”, “by”, “in”, “to”.

Retrieval

Video sequences are retrieved by specifying a search string. Two search modes are supported:

- exact substring matching and
- approximate substring matching.

Exact substring matching returns all frames with substrings in the recognized text that are identical to the search string. Approximate substring matching tolerates a certain number of character differences between the search string and the recognized text. For approximate substring matching we use the Levenshtein distance $L(A,B)$ between a shorter search string A and longer text string B. It is defined as the minimum number of substitutions, deletions and insertions of characters needed to transform A into a substring of B [20]. For each frame we calculate the minimal Levenshtein distance. If the minimal distance is below a certain threshold, the appearance of the string in the frame is assumed. Since it can be expected that long words are more likely to contain erroneous characters, the threshold value depends on the length of the search string A.

For instance, if a user is interested in commercials from Chrysler, he/she uses “Chrysler” as the search string and specifies the allowance of up to one erroneous character per four characters, i.e. the allowance of one edit operation (character deletion, insertion, or substitution) to convert the search string “Chrysler” into some substring of recognized text.

The retrieval user interface of our system is depicted in Figure 12. In the “OCR Query Window” the user formulates his/her query. The result is presented in the “Query Result Window” as a series of small pictures. Multiple hits within one second are grouped into one picture. A single click on a picture displays the frame in full resolution, while a double click starts the external video browser.

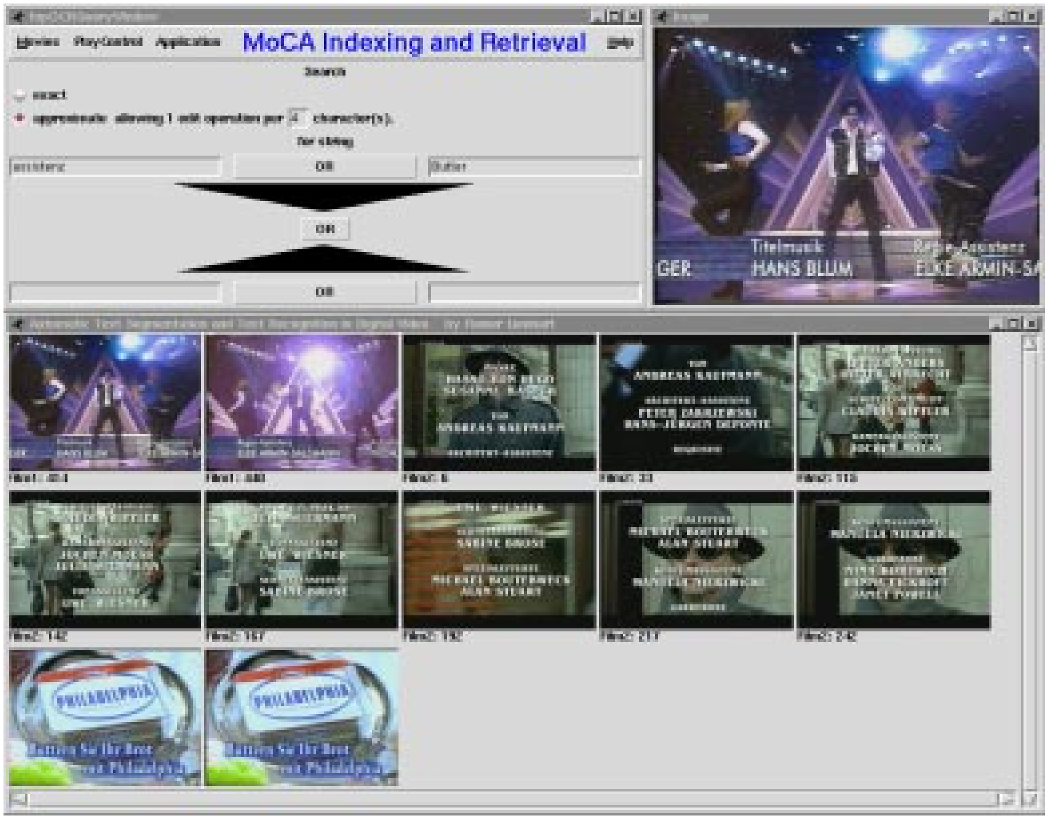

MoCA Indexing and Retrieval
Search
exact
approximate allowing 1 edit operation per 4 character(s).
for string
OR
OR
OR
Titelmusik
HANS BLUM
ELKE ARMIN-S
PHILADELPHIA

Figure 12: Retrieval user interface

6 EXPERIMENTAL RESULTS

In this chapter we discuss two things: Firstly, the performance of our text segmentation and recognition algorithms, and secondly their suitability for indexing and retrieval. Since text is used differently in different film parts and/or film genres, both issues are dealt with separately for three exemplary video genres:

- feature films (i.e. pre-title sequences, credit titles and closing sequences with title and credits),
- commercials, and
- newscasts.

Ten video samples for each class have been recorded, adding up to 22 minutes of video. They were digitized from several German and international TV broadcasts as 24-Bit JPEG images at a compression ratio of 1:8, a size of 384 by 288 pixels and at 25 fps. All JPEG images were decoded into 24-bit RGB images.

6.1 Text Segmentation

Before processing each video sample with our text segmentation algorithms, we manually wrote down the text appearing in the samples and the frame number range of its visibility. Then we processed all ten video samples with our segmentation algorithms and investigated whether or not a character had been segmented. To be more precise: we measured the quality of our segmentation with regard to the main objective not to discard character pixels. The results for each video genre are averaged and summarized in Table 2. The segmentation performance is high for title sequences or credit sequences and newscasts, ranging from 88% to 96%.

It is higher for video samples with moving text and/or moving background than for video samples where both are stationary. In the latter case our algorithms cannot profit from multiple instances of the same text in consecutive frames, since all instances of the same character have the same background. Moreover, motion analysis cannot rule out background regions. Thus, the segmentation performance is lower. Stationary text in front of a stationary scene can often be found in commercials. Therefore, segmentation performance in commercials is lower (66%).

The elimination of non-character pixels is measured by the reduction factor. It specifies the performance of the segmentation algorithms with regard to our secondary objective: the reduction of the number of pixels which have to considered during the recognition process. The amount of reduction has a significant impact on the quality of character recognition and on speed in the successive processing step. The reduction factor is defined as

$$\text{reduction factor}_{avg} = \frac{1}{\text{\# of frames in video}} \cdot \sum \frac{\text{\# of pixels left in frame f}}{\text{\# of pixels in original frame f}}$$

It ranges from 0.04 to 0.01, thus demonstrating the good performance of the text segmentation step. More details are given in [9].

	title sequences or credit sequences	commercials	newscasts
# of frames	2874	579	3147
# of characters	2715	264	80
thereof segmented	2596 (96%)	173 (66%)	79 (99%)
RF_{avg}	0.04	0.01	0.01

Table 2: Segmentation results

6.2 Text Recognition

The performance of the text recognition step is evaluated by two ratio measurements:

- the ratio of the characters recognized correctly to the total number of characters and
- the ratio of the additional garbage characters to the total number of characters.

We call the ratios *character recognition rate (CRR)* and *garbage character rate (GCR)*, respectively. However, their exact values have to be determined manually on a tedious basis, frame by frame. Thus, we approximate their values by the following formulas, whose values can be calculated automatically from the manually determined values of text appearances in the segmentation experiments and the calculated recognition result.

$$CRR_{avg} = \frac{1}{\text{\# of frames with text}} \cdot \sum_{\langle f \in video \rangle \wedge \langle f \text{ contains text} \rangle} CRR_f$$

$$GCR_{avg} = \frac{1}{\text{\# of frames with text}} \cdot \sum_{\langle f \in video \rangle \wedge \langle f \text{ contains text} \rangle} GCR_f$$

where

$$CRR_f \approx 1 - \frac{1}{|W_f|} \cdot \sum_{w \in W_f} \frac{L(w, t_f)}{|w|}$$

$$GCR_f \approx max\left\{0, \frac{\text{\# of recognized characters in frame f}}{\text{\# of actual characters in frame f}} - 1\right\}$$

for $\langle f \in video \rangle \wedge \langle f \text{ contains text} \rangle$, W_f the set of all words actually appearing in frame f and t_f the text recognized in frame f.

Note that the garbage character rate (GCR) is only defined for frames in which text occurs. For frames lacking text we cannot relate the garbage characters to the total number of characters. Thus, we just count their number per text-free frame and call it the garbage character count (GCC).

$$GCC_{avg} = \frac{1}{\text{\# of frames without text}} \cdot \sum \text{\# of recognized characters in frame f}$$

The measurements show that the recognition rate is fairly low, ranging from 41% to 76% (see Table 3). Also, the garbage count is quite high for frames without text, especially for our samples of newscasts and commercials due to their many stationary scenes with stationary text. This observation gives us a strong lead for future research: A computationally cheap detection method for text-free frames has to be developed that can reduce the GCC considerably.

OCR errors and misses originate from the narrowness of current OCR package software with respect to our domain. They are not adjusted to the very small font sizes used in videos nor to the specific mistakes of the text segmentation step.

video type	title sequences or credit sequences	commercials	newscasts
CRR	0.76	0.65	0.41
GCR	0.09	0.14	0.46
GCC	0	25.44	16

Table 3: Recognition results

A peculiarity of the Recognita OCR engine can be noticed when comparing the GCR and GCC values. While non-character regions are easily discarded by the OCR engine in frames with text, it has significant difficulties in text-free frames. Thus, the GCC exploded for commercials and newscasts, in which most of the frames were text-free in contrast to the title sequences or credit sequences.

6.3 Retrieval Effectiveness

Retrieval effectiveness is the ability of information retrieval systems to retrieve only relevant documents. Applied to our domain, we measure the effectiveness of finding all video locations depicting a query word while curbing the retrieval of false locations prompted by recognition errors or garbage strings generated from non-characters regions which survived both in the segmentation and recognition steps.

There exist two well-accepted measures for the evaluation of retrieval effectiveness. These have been adjusted to our purpose: recall and precision [17]. *Recall* specifies the ratio of the number of relevant video locations found to the total number of relevant video locations in the video database; *precision* specifies the ratio of the number of relevant retrieval results to the total number of returned video locations. We assume that a video location depicting the search text is retrieved correctly if at least one frame of the frame range has been retrieved in which the query text appears.

Table 4 depicts the measured average values for recall and precision. They are calculated from the measured recall and precision values, using each word that occurs in the video samples as a search string. The recall value for approximate substring matching ranges from 0.54 to 0.82, i.e we get 54% to 82% of the relevant material, which is quite high. Also the precision value is considerable. Thus, our proposed text segmentation and text recognition algorithms can be effectively used to retrieve relevant video locations. The retrieval application in Figure 12 gives an example.

video type	exact substring matching		approx. substring matching	
	recall	precision	recall	precision
title sequences or credit sequences	0.60	0.71	0.78	0.54
commercials	0.47	0.73	0.54	0.65
newscasts	0.64	0.95	0.82	0.60

Table 4: Retrieval results.

6.4 Availability

Code for running the text segmentation algorithms will be available at publishing time via FTP from the host *ftp.informatik.uni-mannheim.de* in the directory */pub/MoCA/*. In addition, readers interested in seeing some of the video clips can retrieve them from http://www.informatik.uni-mannheim.de/informatik/pi4/projects/MoCA/MoCA_TextRecognition/.

7 RELATED WORK

Numerous reports have been published about indexing and retrieval of digital video sequences, each concentrating on different aspects. Some employ manual annotation [5][2], others compute indices automatically. Automatic video indexing generally uses indices based on the color, texture, motion, or shape of objects or whole images [3][18][24]. Sometimes the audio track is analyzed, too, or external information such as story boards and closed captions is used [13]. Other systems are restricted to specific domains such as newscasts [24], football, or soccer [4]. None of them tries to extract *and* recognize automatically the text appearing in digital videos *and* use it as an index for retrieval.

Existing work on text recognition has focused primarily on optical recognition of characters in printed and handwritten documents in answer to the great demand and market for document readers for office automation systems. These systems have attained a high degree of maturity [14]. Further text recognition work can be found in industrial applications, most of which focus on a very narrow application field. An example is the automatic recognition of car license plates [21]. The proposed system works only for characters/ numbers whose background is mainly monochrome and whose position is restricted.

There exist some proposals regarding text detection in and text extraction drom complex images and video. In [19], Smith and Kanade briefly propose a method to detect text in video frames and cut it out. However, they do not deal with the preparation of the detected text for standard optical character recognition software. In particular, they do not try to determine character outlines or segment the individual characters. They keep the bitmaps containing text as they are. Human beings have to parse them. They characterize text as a “horizontal rectangular structure of clustered sharp edges” [19] and use this feature to identify text segments. Their approach is completely intra-frame and does not utilize the multiple instances of the same text over successive frames to enhance segmentation and recognition performance.

Yeo and Liu propose a scheme of caption detection and extraction based on a generalization of their shot boundary detection technique for abrupt and gradual transitions to locally restricted areas in the video [23]. According to them, the appearance and disappearance of captions are defined as a localized cut or dissolve. Thus, their approach is inherently inter-frame. It is also very cheap computationally since it operates on compressed MPEG videos. However, captions are only a small subset of text appearances in video. Yeo and Liu's approach seems to fail when confronted with general text appearance produced by video title machine, such as scroll titles, since these text appearances cannot just be classified by their sudden appearance and disappearance. In addition, Yeo and Liu do not try to determine the characters' outline, segment the individual characters and translate these bitmaps into text.

Zhong et. al. propose a simple method to locate text in complex images [26]. Their first approach is mainly based on finding connected monochrome color regions of certain size, while the second locates text based on its specific spatial variance. Both approaches are combined into a single hybrid approach.

Wu et. al. propose a four-step system that automatically detects text in and extracts it from images such as photographs [22]. First, text is treated as a distinctive texture. Potential text locations are found by using 3 second-order derivatives of Gaussians on three different scales. Second, vertical strokes coming from horizontally aligned text regions are extracted. Based on several heuristics, strokes are grouped into tight rectangular bounding boxes. These steps are then applied to a pyramid of images generated from the input images in order to detect text over a wide range of font sizes. The boxes are then fused at the original resolution. In a third step, the background is cleaned up and binarized. In the fourth and final step, the text boxes are refined by repeating steps 2 and 3 with the text boxes detected thus far. The final output produces two binary images for each text box and can be passed by any standard OCR software. Wu et. al. report a recognition rate of 84% for 35 images.

Another interesting approach to text recognition in scene images is that of Ohya, Shio, and Akamatsu [15]. Text in scene images exists in 3-D space, so it can be rotated, tilted, slanted, partially hidden, partially shadowed, and it can appear under uncontrolled illumination. In view of the many possible degrees of freedom of text characters, Ohya et al. restricted characters to being almost upright, monochrome and not connected, in order to facilitate their detection. This makes the approach of Ohya et al. feasible for our aim, despite their focus on still images rather than on video. Consequently they do not utilize the characteristics typical of text appearing in video. Moreover, we focus on text generated by video title machines rather than on scene text.

8 CONCLUSIONS

We have presented our new approach to text segmentation and text recognition in digital video and demonstrated its suitability for indexing and retrieval. The text segmentation algorithms operate on uncompressed frames and make use of intra- and inter-frame features of text appearances in digital video. The algorithm has been tested on title sequences of feature films, newscasts and commercials. The performance of the text segmentation algorithms was always high.

Unlike in our previous work [10][11], where individual characters still may consist of several regions of different colors after the text segmentation, the objective of the current text segmentation was to produce a binary image that depicted the text appearing in the video. Hence, this enables standard OCR software packages to be used to recognize the segmented text. Moreover, the tracking of characters over the entire duration of their appearance in a video sequence is a feature unique to our text segmentation algorithms that distinguishes them from all other related work.

The recognition performance of the OCR-Software Development Kit Recognita V3.0 for Windows 95 on our text-segmented video was sufficient for our simple indexing scheme. We demonstrated the usefulness of the recognition results for retrieving relevant video scenes.

Many new applications of our text segmentation algorithms are conceivable. For instance, they can be used to find the beginning and end of a feature film, since these are framed by title sequences (pre-title and closing sequence). Or they can be used to extract its title [12]. In addition, the location of text appearances can be used to enable fast-forward and fast-rewind to interesting parts of a video. This particular feature might be useful when browsing commercials and sportscasts. Together with automatic text recognition algorithms, the text segmentation algorithms might be used to find higher semantics in videos.

ACKNOWLEDGEMENTS

We thank Carsten Hund for helping us with the implementation of the character segmentation algorithms and the experiments.

REFERENCES

1. T. M. Cover and P. E. Hart. Nearest Neighbor Pattern Classification. *IEEE Trans. Information Theory*, Vol. IT-13, pp. 21-27, January 1967.
2. Marc Davis. Media Streams: Representing Video for Retrieval and Repurposing. *Proc. ACM Multimedia 94*, pp. 478-479, San Francisco, CA, USA, October 15-20, 1994.
3. M. Flickner, H. Sawney, et al. Query by Image and Video Content: The QBIC System. *IEEE Computer*, Vol. 28, No. 9, pp. 23-32, September 1995.
4. Y. Gong, L. T. Sin, C. H. Chuan, H. J. Zhang, and M. Sakauchi. Automatic Parsing of TV Soccer Programs. In *Proc. International Conference of Multimedia Computing and Systems*, pp. 167-174, May 1995.

5. R. Hjelsvold, S. Langørgen, R. Midstraum, and O. Sandstå. Integrated Video Archive Tools. *Proc. ACM Multimedia 95*, San Francisco, CA, Nov. 1995, pp. 283-293.
6. S. L. Horowitz and T. Pavlidis. Picture Segmentation by a Traversal Algorithm. *Comput. Graphics Image Process*. 1, pp. 360-372, 1972.
7. B. Jähne. *Digital Image Processing. Concepts, Algorithms, and Scientific Applications*. Springer-Verlag Berlin Heidelberg New York, 1995.
8. A. K. Jain und S. Bhattacharjee. Text Segmentation Using Gabor Filters for Automatic Document Processing. *Machine Vision and Applications*, Vol. 5, No. 3, S. 169-184, 1992.
9. R. Lienhart. *Methods Towards Automatic Video Analysis, Indexing and Retrieval*. Ph.D. thesis, University of Mannheim, June 1998. in German.
10. R. Lienhart and F. Stuber. Automatic Text Recognition in Digital Videos. In *Image and Video Processing IV 1996*, Proc. SPIE 2666-20, Jan. 1996.
11. R. Lienhart. Automatic Text Recognition for Video Indexing. In *Proceedings of the ACM Multimedia '96*, (Boston, MA, USA, November 11-18, 1996), S. 11-20, November 1996.
12. R. Lienhart, S. Pfeiffer, and W. Effelsberg. Video Abstracting. *Communications of the ACM*, Vol. 40, No. 12, pp.55-62, December 1997.
13. C. J. Lindblad, D. J. Wetherall, and W. Stasior. View-Station Applications: Implications for Network Traffic. *IEEE Journal on Selected Areas in Communications*, Vol. 13, 1995.
14. S. Mori, C. Y. Suen, and K. Yamamoto. Historical Review of OCR Research and Development. *Proceedings of the IEEE*, Vol. 80, No. 7, pp. 1029-1058, July 1992.
15. J. Ohya, A. Shio, and S. Akamatsu. Recognizing Characters in Scene Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 2, pp. 214-220, 1994.
16. C. A. Poynton. Frequently Asked Questions about Colour. <ftp://ftp.inforamp.net/pub/users/poynton/doc/colour/>, 1995.
17. G. Salton and M. J. McGill. Introduction to Modern Information Retrieval. *McGraw Hill*, New York, 1983.
18. J. R. Smith and S. F. Chang. Visualseek: A Fully Automated Content-based Image Query System. In *ACM Multimedia*, pp. 87-98, November 1996.
19. M. A. Smith and T. Kanade. Video Skimming for Quick Browsing Based on Audio and Image Characterization. Carnegie Mellon University, Technical Report CMU-CS-95-186, July 1995.
20. G. A. Stephen. *String Searching Algorithms*. World Scientific Publishing Co. Pte. Ltd., Singapore, 1994.
21. M. Takatoo et al., Gray Scale Image Processing Technology Applied to Vehicle License Number Recognition System. In *Proc. Int. Workshop Industrial Applications of Machine Vision and Machine Intelligence*, pp. 76-79, 1987.
22. V. Wu, R. Manmatha and E. M. Riseman. Finding Text in Images. In *Proceedings of Second ACM International Conference on Digital Libraries*, Philadelphia, PA, S. 23-26, July 1997.
23. B.-L. Yeo and B. Liu. Visual Content Highlighting via Automatic Extraction of Embedded Captions on MPEG Compressed Video. In *Digital Video Compression: Algorithms and Technologies*, Proc. SPIE 2668-07 (1996).
24. H.J. Zhang, Y. Gong, S. W. Smoliar, and S. Y. Tan. Automatic Parsing of News Video. *Proc. IEEE Conf. on Multimedia Computing and Systems*, pp. 45-54, 1994.
25. H. J. Zhang, C. Y. Low, S. W. Smoliar, and J. H. Wu. Video Parsing, Retrieval and Browsing: An Integrated and Content-Based Solution. *Proc. ACM Multimedia 95*, San Francisco, CA, pp. 15-24, Nov. 1995.
26. Y. Zhong, K. Karu and A. K. Jain. Locating Text in Complex Color Images. *Pattern Recognition*, Vol. 28, Nr. 10, S. 1523-1535, October 1995.
27. S. Zucker. Region Growing: Childhood and Adolescence. *Computer Graphics and Image Processing*, Vol. 5, pp. 382-399, 1976.