# Automatic Caption Localization in Compressed Video

Yu Zhong, Hongjiang Zhang, and
Anil K. Jain, *Fellow*, *IEEE*

**Abstract**—We present a method to automatically localize captions in JPEG compressed images and the I-frames of MPEG compressed videos. Caption text regions are segmented from background images using their distinguishing texture characteristics. Unlike previously published methods which fully decompress the video sequence before extracting the text regions, this method locates candidate caption text regions directly in the DCT compressed domain using the intensity variation information encoded in the DCT domain. Therefore, only a very small amount of decoding is required. The proposed algorithm takes about $0.006$ second to process a $240 \times 350$ image and achieves a recall rate of $99.17$ percent while falsely accepting about $1.87$ percent nontext DCT blocks on a variety of MPEG compressed videos containing more than $2,300$ I-frames.

**Index Terms**—Caption extraction, text location, texture, compressed video, segmentation, multimedia.

## 1 Introduction

Digital video now plays an important role in entertainment, education, and other multimedia applications. With hundreds of thousands of hours of archival videos, there is an urgent demand for tools that will allow efficient browsing and retrieving of video data [1], [20], [21]. In response to such needs, various video content analysis techniques using one or a combination of image, audio, and textual information present in video have been proposed to parse, index, and abstract massive amounts of data [1], [3], [15], [20]. Among these information sources, caption text present in the video frames plays an important role in understanding the content of a raw video sequence. For example, captions in news broadcasts and documentaries usually annotate information on *where*, *when*, and *who* of the reported events. More importantly, a sequence of frames with caption text is often used to represent *highlights* in documentaries. Also, captions are widely used to depict titles, producers, actors, credits, and sometimes, the context of a story. Furthermore, text and symbols that are presented at specific locations in a video image can be used to identify the TV station and program associated with the video. In summary, captions in video frames provide highly condensed information about the contents of the video and can be used for video skimming, browsing, and retrieval in large video databases.

Although embedded text/captions provide important information about the image, it is not an easy problem to reliably detect and localize text/captions embedded in images. In a single frame, the size of characters can change from very small to very big. The font of text can be different. Text present in the same image can have multiple colors. Text can occur in a very cluttered background. For video sequences, the text can be either still or moving in an arbitrary direction. The same text may vary its size from frame to frame, due to some special effects. The background can also be moving/changing, independent of the text. Fig. 1 illustrates some examples of video frames containing captions (Fig. 1a, Fig. 1b, Fig. 1c, Fig. 1d, Fig. 1e, Fig. 1f, and Fig. 1g), where the text is embedded in the video using video editing techniques, or scene text (Fig. 1d, Fig. 1e, Fig. 1h, and Fig. 1i), where the text is part of the environment and captured by the camera along with the rest of the scene. The objective of our algorithm is to locate the caption text, although it also works on some scene text as well.

A number of algorithms to extract caption texts from still images and video have been published in recent years [2], [5], [7], [9], [10], [13], [14], [17], [22]. These methods utilize the following properties of text:

1. Characters are bounded in size;
2. a text line always contains a cluster of characters which are aligned horizontally; and
3. text usually has a good contrast from the background.

Most of the published methods for text location can be categorized as either *component-based* or *texture-based*. For component-based text extraction methods, text regions are detected by analyzing the geometrical arrangement of edges or homogeneous color/grayscale components that belong to characters. For example, Smith and Kanade [14] located text as horizontal rectangular structures of clustered sharp edges. Zhong et al. [22] extracted text as those connected components of monotonous color which follow certain size constraints and horizontal alignment constraints. In a similar manner, Lienhart and Stuber [9] identified text as connected components which are of the same color, fall in some specified size range, and have corresponding matching components in consecutive video frames. Shim et al. [13] used chain-codes to segment text components from video images and used temporal information to refine the extraction based on the assumption that captions stay static in a video sequence. Jain and Yu [5] decomposed the video frames into subimages of different colors and then examined if each subimage contained text components that satisfy some prespecified heuristics.

It is generally agreed that text regions possess a special *texture* because text usually consists of character components which contrast the background and, at the same time, exhibit a periodic horizontal intensity variation due to the horizontal alignment of characters. In addition, character components form text lines with approximately the same spacing between them [4], [6]. As a result, text regions can be segmented using texture features. Jain and Zhong [4], [6] have used the distinguishing texture present in text to determine and separate text, graphics, and halftone image regions in scanned grayscale document images. Zhong et al. [22] further utilized the texture characteristics of text lines to extract text in grayscale images with complex backgrounds. For each pixel, the text energy is defined as the horizontal spatial variation in a $1 \times n$ neighborhood window; a rectangular region of high energy is identified as a text region. This texture-based method was applied to a variety of still images with an acceptable performance.

All the above text detection methods were applied on *uncompressed* images, even though they are designed for digital images and video. None of the previously published algorithms utilized features present in the compressed domain to locate the text directly in compressed images. However, digital video and many still images are usually stored in compressed form for efficient storage and transmission. For example, the MPEG video compression standard applies DCT coding to reduce spatial redundancies within a video frame (same as in the JPEG image compression standard) and motion compensation to reduce temporal redundancies between consecutive frames [8]. One has

- *Y. Zhong is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213. E-mail: zhongyu@cs.cmu.edu.*
- *H. Zhang is with the Internet Systems and Applications Lab, Hewlett-Packard Company, Palo Alto, CA 94040. E-mail: hjzhang@hpl.hp.com.*
- *A.K. Jain is with the Deptpartment of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824. E-mail: jain@cse.msu.edu.*

*Manuscript received 12 Dec. 1998; Accepted 22 Oct. 1999*
*Recommended for acceptance by T.K. Ho.*
*For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 108434.*

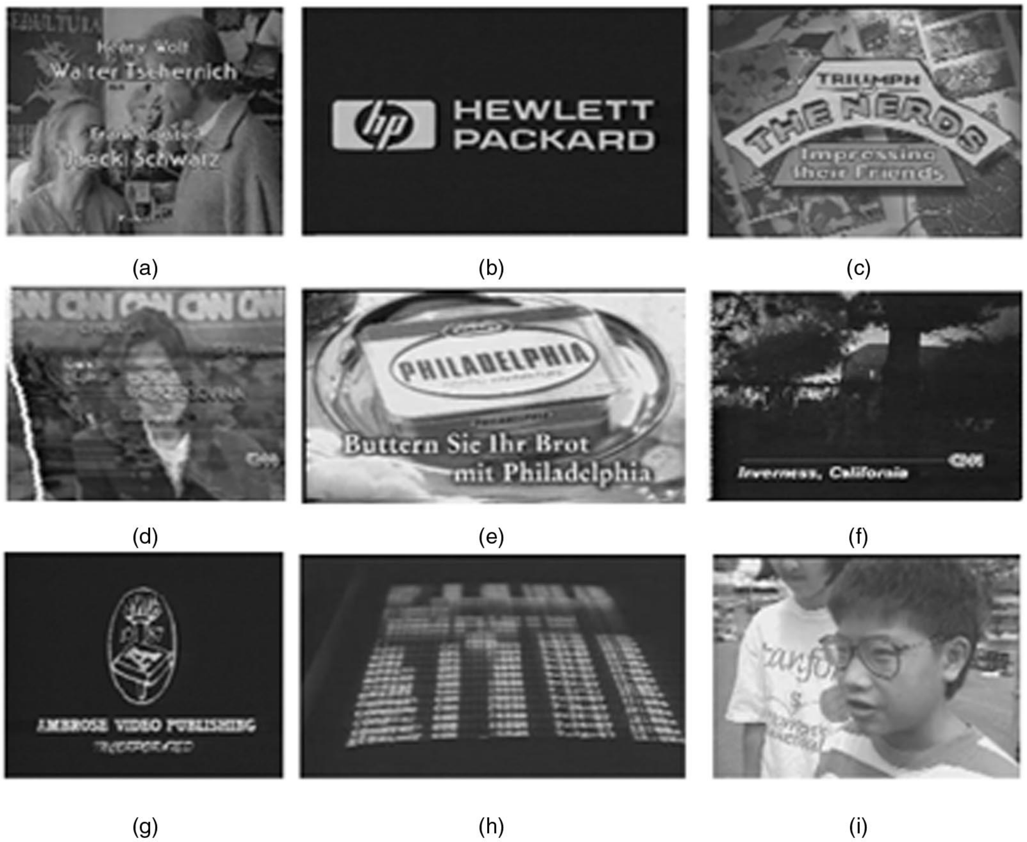


Fig. 1. Examples of video frames with caption text or scene text.

to first decompress these images and video in order to apply the currently available text detection methods.

As compression techniques are becoming more efficient and cost effective, an increasing proportion of images and video are being stored in the compressed form. Therefore, there is an emerging trend to extract features directly in the compressed domain [11], [12], [19]. By manipulating features directly in the compressed domains, we can save the resources (computation time and storage) needed for decompressing video sequences. Another reason, which is often ignored is that many compressed domain features such as DCT coefficients and motion vectors in MPEG video are actually very effective in several applications, including text detection.

There has been very little effort to utilize compressed domain features to localize text in videos. Yeo and Liu [18] proposed to extract embedded captions in partially uncompressed MPEG video, where reduced resolution video frames were reconstructed from original MPEG sequences using either the DC components or the DC components plus two AC components. Text regions in these frames were then detected wherever a large interframe difference was observed, indicating the appearance and disappearance of the captions in video frames. However, this algorithm was only able to detect some captions that abruptly appeared or disappeared, assuming that changes resulting from other sources can be ignored. Therefore, Yeo and Liu's method would not be able to handle captions that gradually enter or disappear from the frames. It is also vulnerable to fast moving objects in a video. As the image resolution is reduced during compression by a factor of 64 (DC sequence only) or 16 (DC+2AC), a considerable amount of information is lost, resulting in a lower accuracy of the method. Gargi et al. [2] proposed to filter the video stream for the text-appearance events using the number of Intracoded blocks in P- or B-frames based on the assumption that when captions appear and disappear, the corresponding blocks are usually Intracoded. This approach, however, is vulnerable to abrupt scene change and motion, so its application is limited to relatively static and smooth video segments between shot changes. It was only applied to the P- or B-frames and does not handle captions that appear in the I-frames.

In this paper, we propose a texture-based caption text localization method which operates directly in the DCT domain for MPEG video or JPEG images. The DCT coefficients in JPEG images [16] or MPEG video [8], which capture the directionality and periodicity of local image blocks are used as texture measures to identify text regions. Each unit block in the compressed images is classified as either text or nontext based on local horizontal and vertical intensity variations. In addition, postprocessing procedures including morphological operations and connected component analysis are performed to refine the detected text. This algorithm is extremely fast due to the fact that: 1) it requires very little decoding of the compressed image/video stream; and 2) the refinement and postprocessing proceed on images of reduced sizes. Our algorithm can be used as a prefilter for information retrieval systems to quickly signal the potential image regions with caption text and, thus, reduce the amount of data that needs to be processed by more sophisticated but relatively slower algorithms to extract "OCR-ready" text components.

The rest of the paper is organized as follows: We describe the details of the proposed method in Section 2, which includes texture feature extraction from MPEG compressed domain and the refinement of the text candidate blocks. The experimental results and performance evaluation are presented in Section 3. Section 4 summarizes the paper and presents future work.

## 2 Proposed Method

The MPEG compression scheme utilizes the spatial redundancy in one frame and the temporal redundancy between successive frames to achieve a low-bit rate compression. An MPEG compressed video consists of a sequence of I-frames (Intracoded) with a number of B- and P-frames in between. I-frames are compressed using Discrete Cosine Transform (DCT) of local blocks to reduce spatial redundancy. B- and P-frames are introduced to reduce temporal redundancies, where a P-frame is predicted from the I- or P-frame immediately preceding it, and a B-frame is bi-directionally interpolated using the two I- or P-frames before and after it.

The proposed method operates on the I-frames of MPEG video or JPEG images, using texture features captured by the DCT coefficients of image blocks. It consists of two basic steps: Detection of candidate caption regions in the DCT compressed domain and the postprocessing to refine the potential text regions. The diagram shown in Fig. 2 summarizes the major steps of the proposed method.

### 2.1 Selecting Texture Features

Our approach is texture-based, which utilizes the fact that text regions present a special kind of texture pattern that makes them different from other image regions. Text regions possess a unique texture: Each text region consists of text lines of the same orientation with approximately the same spacings in between and each text line consists of characters of approximately the same size, placed next to each other.

We propose to use the DCT coefficients directly from compressed images and video as texture features to localize text regions. DCT compressed images encode a two-dimensional image using the DCT coefficients $\{c_{uv}\}$ of an $N \times N$ (N is usually a power of two) image region $\{I_{xy},\ 0 \le x < N, 0 \le y < N\}$:

$$c_{uv} = \frac{1}{N}\mathcal{K}_u\mathcal{K}_v\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} I_{xy}\cos\frac{\pi u(2x+1)}{2N}\cos\frac{\pi v(2y+1)}{2N}, \tag{1}$$

where $u$ and $v$ denote the horizontal and vertical frequencies $(u, v = 0, 1, \ldots, N-1)$ and $\mathcal{K}_w = \frac{1}{\sqrt{2}}$, $(w \in \{u, v\})$, for $w = 0$ and $\mathcal{K}_w = 1$, otherwise. The AC components ($c_{uv},\ u \neq 0$ or $v \neq 0$) capture the spatial frequency (characterized by $u$ and $v$) and directionality (by tuning the $u$ and $v$ values) properties of the $N \times N$ image block.

This approach is justified for the following reasons:

- The DCT coefficient values, which are computed based on the $8 \times 8$ spatial input, capture *local* image features.
- The values of DCT coefficients, which are amplitudes of harmonic waves, denote the relative amount of various 2D spatial frequencies contained in $8 \times 8$ blocks. Therefore, they can be used as measures of spatial periodicity and directionality, when frequencies in the x and y dimensions are properly tuned.
- The quantized DCT coefficients can be readily extracted from a video stream and JPEG data. Although they are quantized, the rank information is preserved and we can use them to compute texture features without any decoding procedure.

In summary, values of DCT coefficients in compressed domain images capture the local periodicity and directionality features in the spatial image domain. This is the basis of our text localization approach.

To gain some insight into the DCT spectrum, Fig. 3a shows an input image and Fig. 3b shows the absolute values of the DCT coefficients directly extracted from the compressed domain of the intensity image. Each subimage in Fig. 3b represents one DCT channel of the input image. Each pixel in the subimage is the energy in this channel for the corresponding DCT block of the input image, where the magnitude is proportional to the brightness of the pixel. The channels, from top to bottom, indicate horizontal variations, with increasing frequencies; and from left to right, indicate vertical variations, with increasing frequencies. In particular, the subimage (channel) at the top left corner corresponds to the DC component, which is the averaged and subsampled version of the input image and the subimages on the top row, from left to right, correspond to channels of zero vertical frequency and increasing horizontal frequencies. This figure shows that the top left channels, which represent the low frequency components, contain most of the energy, while the high frequency channels, which are located at the bottom right corner of each subimage, are mostly blank. It also indicates that the channel spectrums capture the directionality and coarseness of the spatial image; for all the vertical edges in the input image, there is a corresponding high frequency component in the horizontal frequencies, and vice versa. Furthermore, diagonal variations are captured by the channel energies around the diagonal line. This example illustrates that the DCT domain features do characterize the texture attributes of an image.

The proposed algorithm is based on the observation that a text line which consists of characters should have a high response to the horizontal harmonics because of the rapid changes in intensity introduced by characters in a text line. At the same time, we expect a text region to have a high response in vertical harmonics because of the changes in intensity due to the spacing between different text lines. In particular, we use the AC coefficients of the horizontal harmonics ($c_{0v},\ v > 0$) to capture the horizontal intensity variations caused by characters within a text line and the amplitude of the vertical harmonics ($c_{u0},\ u > 0$) to capture the vertical intensity variations caused by the spacings between text lines.

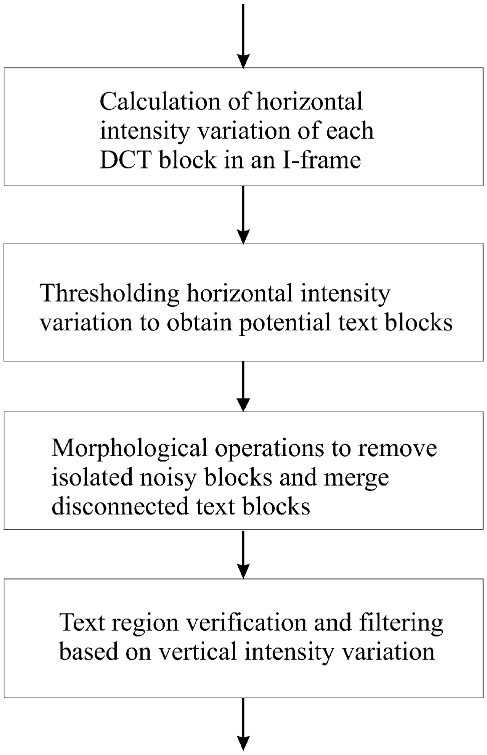

Calculation of horizontal intensity variation of each DCT block in an I-frame
Thresholding horizontal intensity variation to obtain potential text blocks
Morphological operations to remove isolated noisy blocks and merge disconnected text blocks
Text region verification and filtering based on vertical intensity variation

Fig. 2. Algorithm for locating text in compressed video.

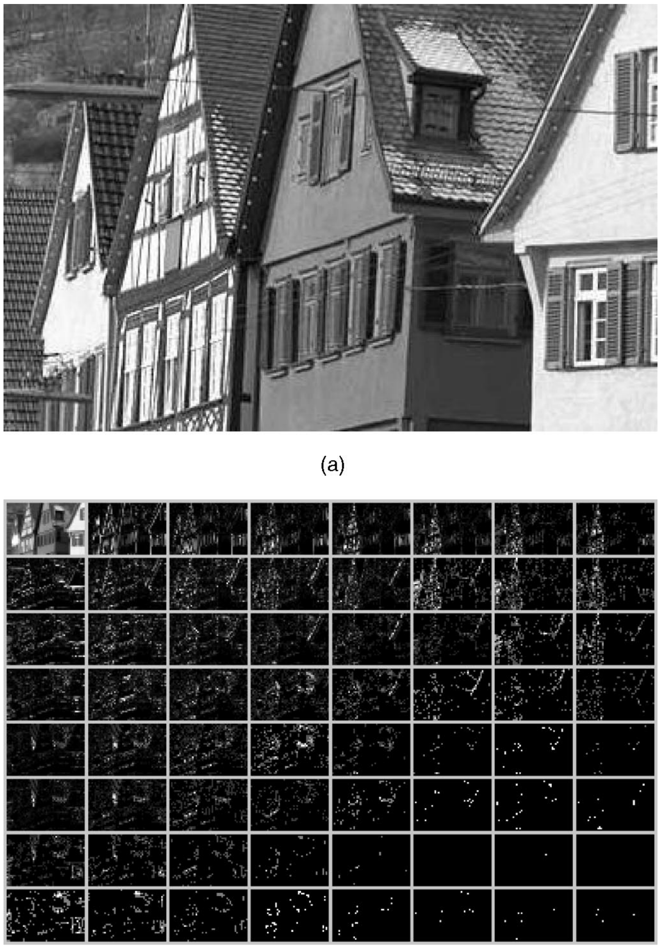


Fig. 3. Feature extraction from DCT coefficients for text location. (a) 250 x 384 imput image and (b) DCT features from the intensity image.

## 2.2 Processing in the Compressed Domain

To obtain candidate text regions using *only* the information in the compressed domain, we perform the following operations. Note that the operating units are the $8 \times 8$ blocks in I-frames.

### 2.2.1 Detecting Blocks of High Horizontal Spatial Intensity Variation

For each $8 \times 8$ DCT block $(i, j)$, we compute the horizontal text energy $E_{hor}(i, j)$ by summing up the absolute amplitudes of the horizontal harmonics $c_{0v}(i, j)$ of the block:

$$E_{hor}(i, j) = \sum_{v_1 \leq v \leq v_2} |c_{0v}(i, j)|, \tag{2}$$

where $v_1$ and $v_2$ are parameters of the algorithm. They should be selected based on the size of the characters to be located. We have used $v_1 = 2$ and $v_2 = 6$ in our algorithm. These horizontal text energy values are then thresholded to obtain the blocks of large horizontal intensity variations. A block is a text candidate if its horizontal text energy is above some threshold. We have used an adaptive threshold value which is $1.45$ times the average texture energy of the corresponding DCT channel for all the blocks in the image. This simple thresholding procedure identifies most of the text blocks. But at the same time, it also picks up some nontext blocks containing high vertical intensity variations. Furthermore, depending on the spacings between characters or words, the detected text candidate blocks may be detached or disconnected due to wide spacing, low contrast, or large fonts. Therefore, results using the simple thresholding tend to be noisy and need further refinement and verification.

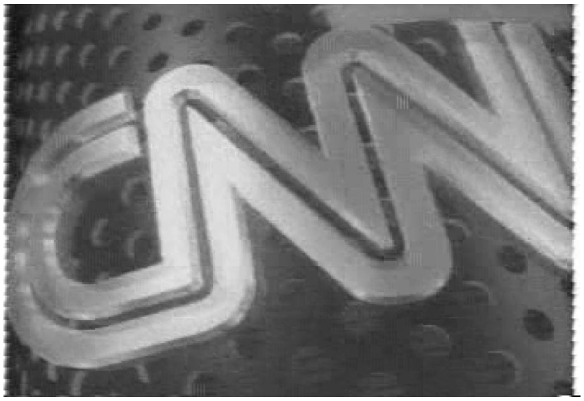


Fig. 4. The text in the image presents no local texture.

### 2.2.2 Refining Text Candidate Blocks Using Spatial Constraints

The noise introduced in the previous stage is removed by applying morphological operations to the thresholded image (with $8 \times 8$ pixel block units, the image size is $1/8$th of the original image size in each dimension). Although nontext blocks may respond high to horizontal harmonics, their occurrences are generally random and they seldom merge collectively into rows as text blocks do. We applied a *closing* operation followed by an *opening* operation to the thresholded image to remove false detections. Currently, we use a structured element of size $1 \times 3$. This particular size allows the text blocks to merge into horizontal lines. This processing step basically removes most of the isolated noisy blocks and merges the nearby detached text blocks into coherent regions.

## 2.3 Segmentation of Potential Caption Text Regions

The resulting text candidate blocks are further analyzed to form caption text regions. Based on the observation that characters are arranged next to each other horizontally to form text lines and text lines are arranged from top to bottom with a small gap between them to form paragraphs, we segment the text blocks into individual text regions. Disconnected blocks form separate text regions. Moreover, a connected text region is divided into two disconnected text regions at the row (text line) where the width of the region is less than a fraction of the maximum width of text lines in the text region. We have used a factor of $0.5$ in our system.

### 2.3.1 Refining Text Candidate Regions

The segmented candidate text regions are further filtered using the third property of text regions listed in Section 2.3: A text line that contrasts the background should possess a large vertical intensity variation at its top and bottom borders. As a result, a text region contributes to local vertical harmonics. Therefore, we expect that for a horizontal text line, there will be a corresponding row of blocks with high vertical spectrum energy.

The vertical text energy $E_{ver}(i, j)$ for each block $(i, j)$ is calculated by summing up the absolute DCT coefficient values $c_{u0}(i, j)$

$$E_{ver}(i, j) = \sum_{u_1 \leq u \leq u_2} |c_{u0}(i, j)|, \tag{3}$$

where again, $u_1$ and $u_2$ are the parameters. We have used $u_1 = 1$ and $u_2 = 6$ in our system. The average vertical text energy $\mathcal{E}^R_{ver}(i)$

TABLE 1
System Performance Results

| Total no. of blocks tested (# of text blocks) | Correctly detected text blocks | Falsely detected text blocks | Missed text blocks | Miss rate (%) |
|---|---|---|---|---|
| 3,206,936 (141,680) | 140,983 | 50,759 | 697 | 0.83 |

for row $i$ in a detected candidate text region $R$ is computed as follows:

$$\mathcal{E}^R_{ver}(i) = \sum_{j:(i,j)\in R} E_{ver}(i,j) / \sum_{j:(i,j)\in R} 1. \qquad (4)$$

If a candidate region does not contain a row with high average vertical text energy, it is discarded as a nontext region.

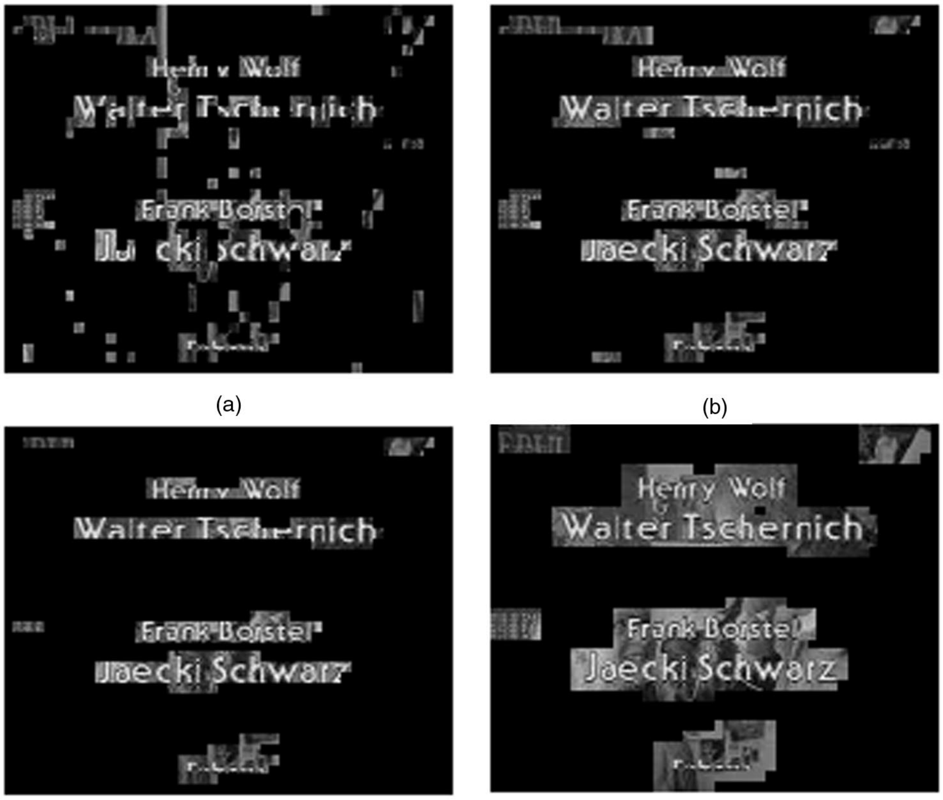
### 2.4 Discussion

The proposed caption localization method to extract text from compressed images utilizes the unique texture and geometrical arrangements presented in a text line/block which is captured by DCT coefficients. It is relatively robust to the font of the text and the image resolution. It is also relatively insensitive to the background complexity and presence of moving objects. However, since this algorithm uses local texture measure in DCT blocks, it is vulnerable to the size of characters and the spacing between them. If the characters are very big or sparsely spaced such that very little image texture is present, the approach will not work properly (e.g., see Fig. 4). However, for the commonly used caption sizes in video, the proposed method works well.

## 3 Experimental Results

To evaluate the performance of the proposed method, we have tested it on a number of MPEG video clips, including TV programs (news broadcasts, advertisements), home videos, and movie clips. The "Triumph of the nerds" video (471MB), a documentary on the successful high tech companies and their founders, contains various embedded captions and credits. We also used 1) a CNN video clip (3897 total frames, 268 I-frames), which covered a variety of events including outdoor and newsroom news programs, and weather forecast and 2) 6 video clips that were used in [9] (267 I-frames total), which included home video, movie clips, and commercial advertisements.

When evaluated on the $2,360$ I-frames from $8$ different video sequences, the proposed method achieved a false reject rate of 0.83 percent among all the caption text present in the test videos, and falsely accepted 1.58 percent of the DCT blocks which is not text (as shown in Table 1.). The major reasons for the false rejects are: 1) The font size of characters or the gap between the characters in the text is too big such that there is no strong texture present in a MPEG block and 2) the contrast between the background and the text is too weak so that the text energy is not sufficiently high. The



(a)

(b)

(c)

(d)

Fig. 5. Illustration of intermediate results for caption extraction for the image in Fig. 1a: (a) DCT blocks with high horizontal intensity variation $E_{hor}$; (b) merged regions after applying morphological operations to the blocks with high horizontal text energy; (c) potential caption regions after the region-based horizontal/vertical text energy test; (d) dilating the previous result by one block.

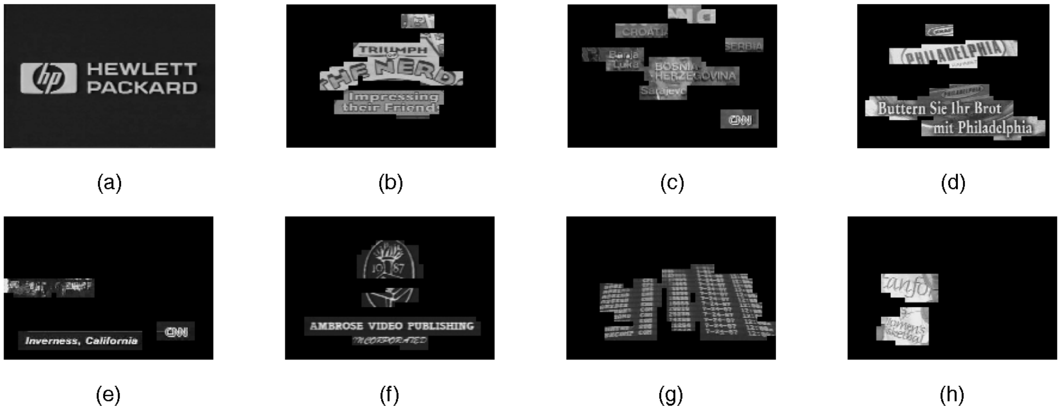


Fig. 6. Caption extraction results for the image in Fig. 1b, Fig. 1c, Fig. 1d, Fig. 1e, Fig. 1f, Fig. 1g, Fig. 1h, Fig. 1i (results shown are dilated by one DCT block).

false positive detection is mostly caused by nontext regions with multiple vertical structures whose spatial arrangements resemble that of text regions.

The processing speed of the proposed method is very fast since it does not require a fully decompressed MPEG video. The DCT information which is used is readily available from the video stream. Furthermore, the processing unit for the algorithm is a DCT block. So, once the feature vectors are extracted for each block, the segmentation and refinement processes operate on an image which is $1/8$th of the original image size in each dimension when $8 \times 8$ DCT blocks are used. Currently, it takes about $0.006$ to $0.009$ second on a Sun Sparc workstation for each I-frame (with a size in the range of $240 \times 350$ to $288 \times 384$). Therefore, our algorithm can be used to filter out text captions in MPEG videos in realtime.

Fig. 5 illustrates various intermediate processing steps for the image in Fig. 1a. Fig. 5a, Fig. 5b, Fig. 5c, and Fig. 5d, show the DCT blocks with high horizontal text energy $E_{hor}$ (2), refined blocks after applying morphological operations, potential caption regions

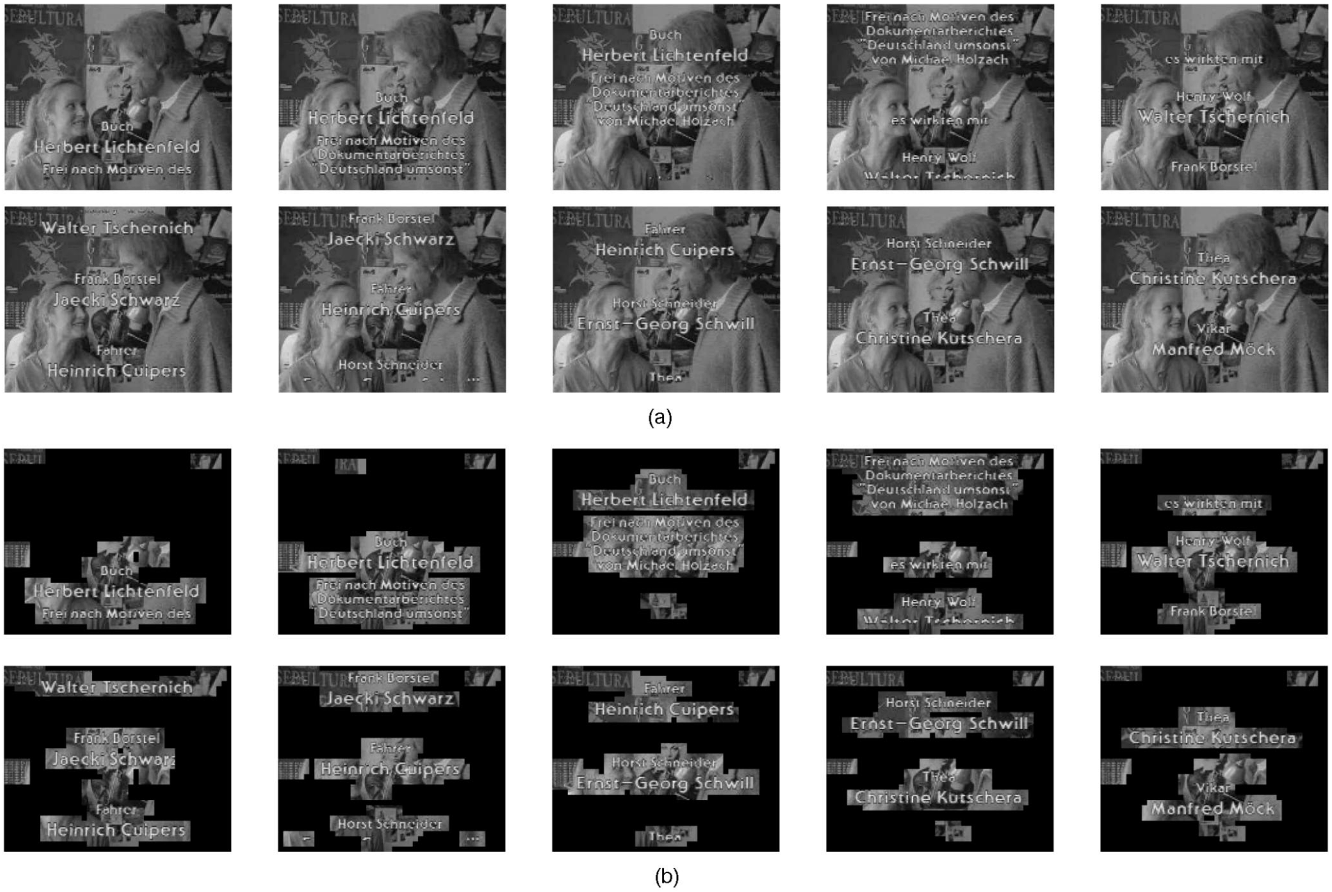


Fig. 7. Results of caption extraction on a video sequence with text moving upward in a static background. (a) input sequence; (b) computed potential caption text regions (results shown are dilated by one DCT block).

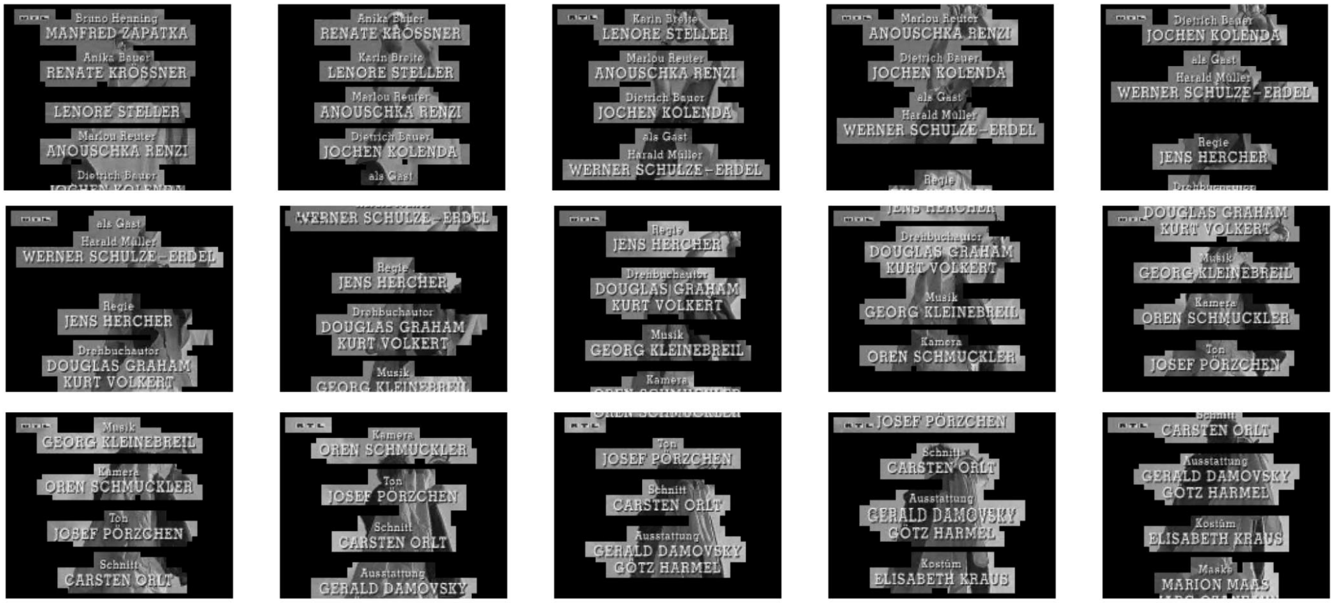


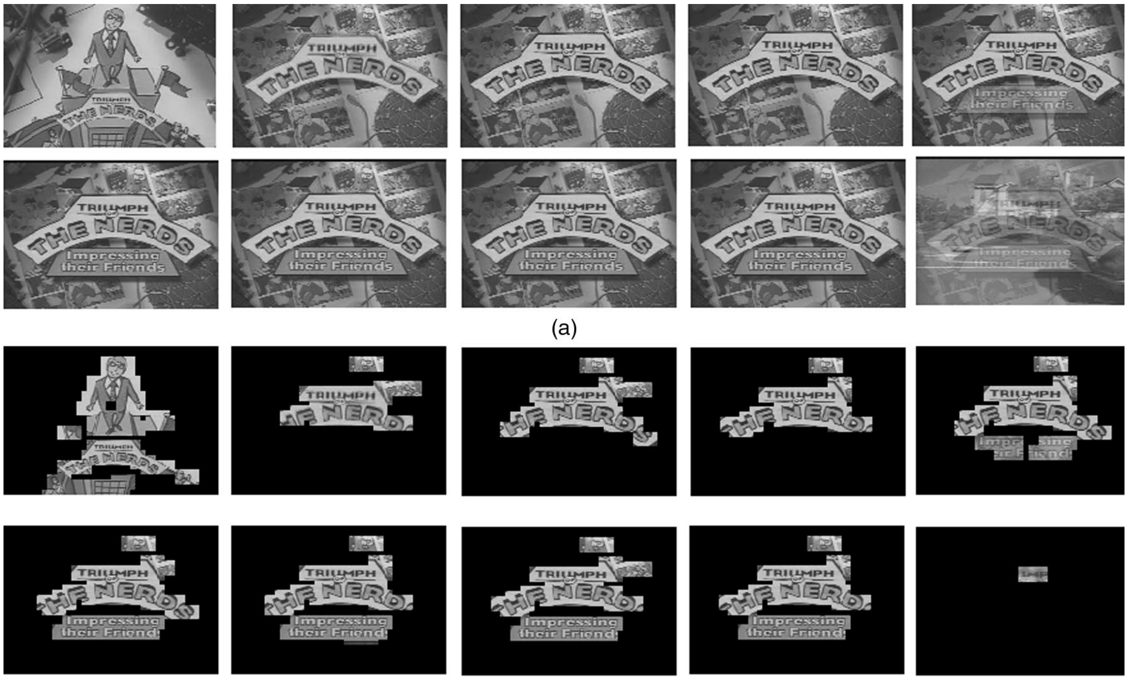
Fig. 8. Results of caption extraction on a video sequence with text moving upward in a changing background (results shown are dilated by one DCT block).

after the vertical text energy $\mathcal{E}^{R}_{ver}$ (4) test and recovered candidate regions by dilating the previous results by one block, respectively.

Fig. 6 illustrates the results of text location for the images in Fig. 1b, Fig. 1c, Fig. 1d, Fig. 1e, Fig. 1f, Fig. 1g, Fig. 1h, Fig. 1i. Fig. 6a, Fig. 6b, Fig. 6c, and Fig. 6d show correctly localized caption text. The half-emerging captions for a map in Fig. 1d are correctly located. The scene text in Fig. 6d, Fig. 6g, and Fig. 6h are also detected. Fig. 6e and Fig. 6f illustrate some occurances of falsely detected text. Fig. 7 and Fig. 8 show caption extraction results on three sequences with moving captions. Fig. 7 shows the results on a sequence with text moving upwards in a complicated static background. Fig. 8 shows the results on a sequence with text moving upwards in a changing background.

## 4 Summary

We have proposed a filter to detect text regions that directly operates in the DCT domain of JPEG images and MPEG video sequences. It utilizes the unique texture characteristics of the text and locates the caption text as regions of high horizontal intensity variations. The texture features are extracted directly from the DCT



(a)

(b)

Fig. 9. Detection of a caption event. (a) Input sequence and (b) computed potential caption text regions (results shown are dilated by one DCT block).

domain using the quantized DCT coefficients, which capture the local frequency information. Although our algorithm has been implemented for DCT compressed domain, we believe that it can be modified to work in other transform-based compressed domains, including wavelets and subband compressed videos, because of the fact that the transformed data capture the spatial frequency and orientation which correspond to texture features.

The proposed algorithm is very fast due to the fact that the texture features are readily available in the compressed domain and all the postprocessing/refinements are performed in reduced-resolution images. As shown in the performance evaluation data, although the DCT information is effective in detecting the presence of caption regions (more than 99 percent of the presented caption text is localized), there is still a significant percentage (1.58 percent) of noncaption areas that are falsely accepted. The texture-based method, based on local intensity statistics, alone may not be adequate to discriminate between text components and noncharacter regions with spatial variations similar to text regions. However, we believe that the high recall rate of our algorithm together with its fast speed makes it a promising prefilter for information retrieval systems where there is a need to quickly filter the video stream to obtain a very small portion of data which contains all the caption text information. This small set of potential text containing area can then be reconstructed and examined using a more elaborate and sophisticated approach such as component-based analysis to extract OCR-ready text components.

Future works on this topic include the following:

1. Currently, we only use the intensity information to locate the text; color information is not utilized. Since captions could have a large color contrast from their background, while the intensity contrast could be small, the use of color information can improve the performance of the proposed method. We may be able to apply the same algorithm to the compressed domains of the two color frames (Cr and Cb) to extract text with high color contrast.
2. The texture features are currently being extracted from local $8 \times 8$ blocks which capture image texture at a relatively fine scale. In order to segment text with a larger font size, we need to investigate a combination of neighboring DCT blocks to compute texture features at a larger scale.
3. The algorithm has been developed for DCT compressed domain and has been applied only to I-frames of the MPEG videos which are DCT compressed. In general, the detection in the I-frames should be sufficient since the duration of a text caption in a sequence is usually longer than the time interval between two I-frames. However, there may be a need to extend the algorithm to extract text from B-frames and P-frames in MPEG videos, to capture text which appears for a very short duration. This extension can be achieved by tracking the text blocks detected in I-frames in P- and B-frames.

## Acknowledgments

The authors would like to thank Aditya Vailaya and the anonymous reviewers for their many constructive comments.

## References

[1] M. Christel, S. Stevens, and H. Wactlar, "Informedia Digital Video Library," *Proc. ACM Multimedia Conf.*, pp. 480–481, Oct. 1994.

[2] U. Gargi, S. Antani, and R. Kasturi, "Indexing Text Events in Digital Video Databases", *Proc. 14th Int'l Conf. Pattern Recognition (ICPR)*, pp. 916–918, 1998.

[3] A. Hauptmann and M. Smith, "Text, Speech, and Vision for Video Segmentation: The Informedia Project," *AAAI Symp. Computational Models for Integrating Language and Vision*, 1995.

[4] A. K. Jain and S. Bhattacharjee, "Text Segmentation Using Gabor Filters for Automatic Document Processing," *Machine Vision and Applications*, vol. 5, no. 3, pp. 169-184, 1992.

[5] A.K. Jain and B. Yu, "Automatic Text Location in Images and Video Frames," *Pattern Recognition*, vol. 31, no. 12 , pp. 2,055–2,076, 1998.

[6] A.K. Jain and Y. Zhong, "Page Segmentation Using Texture Analysis," *Pattern Recognition*, vol. 29, no. 5, pp. 743–770, 1996.

[7] S.W. Lee, D.J. Lee, and H.S. Park, "A New Methodology for Grayscale Character Segmentation and Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 18, no. 10, pp. 1,045–1,050, Oct. 1996.

[8] D. LeGall, "MPEG: A Video Compression Standard for Multimedia Applications," *Comm. ACM*, vol. 34, no. 4, pp. 46–58, Apr. 1991.

[9] R. Lienhart and F. Stuber, "Automatic Text Recognition in Digital Videos," *Proc. Praktische Informatic IV*, pp. 68–131, 1996.

[10] J. Ohya, A. Shio, and S. Akamastsu, "Recognizing Characters in Scene Images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, pp. 214–220, 1994.

[11] I. K. Sethi and N. Patel, "A Statistical Approach to Scene Change Detection," *SPIE Storage and Retrieval for Image and Video Databases III*, pp. 329–338, Feb. 1995.

[12] B. Shen and I.K. Sethi, "Convolution-Based Edge-Detection for Image/Video in Block DCT Domain," *J. Visual Comm. and Image Representation*, vol. 7, no. 4, pp. 411–423, 1996.

[13] J.C. Shim, C. Dorai, and R. Bolle, "Automatic Text Extraction from Video for Content-Based Annotation and Retrieval," *Proc. 14th Int'l Conf. Pattern Recognition*, pp. 618–620, 1998.

[14] M.A. Smith and T. Kanade, "Video Skimming and Characterization through Language and Image Understanding Techniques," technical report, Carnegie Mellon Univ. 1995.

[15] H.D. Wactlar, T. Kanade, M. Smith, and S. Stevens, "Intelligent Access to Digital Video: The Informedia Project," *IEEE Computer*, pp. 46–52, 1996.

[16] G.K. Wallace, "The JPEG Still Picture Compression Standard," *Comm. ACM*, vol. 34, no. 4, pp. 31–44, 1991.

[17] V. Wu, R. Manmatha, and E. Riseman, "Finding Text in Images," *20th Int'l ACM SIGIR Conf. Research and Development in Information Retrieval*, pp. 3–12, 1997.

[18] B.L. Yeo and B. Liu, "Visual Content Highlighting via Automatic Extraction of Embedded Captions on MPEG Compressed Video," *SPIE Digital Video Compression: Algorithms and Technologies*, Feb. 1995.

[19] H.J. Zhang, C.Y. Low, and S.W. Smoliar, "Video Parsing and Browsing Using Compressed Data," *Multimedia Tools and Applications*, pp. 89–111, 1995.

[20] 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*, pp. 15–24, Nov. 1995.

[21] H.J. Zhang and S.W. Smoliar, "Developing Power Tools for Video Indexing and Retrieval," *Proc. SPIE Conf. Storage and Retrieval for Image and Video Databases*, pp. 140–149, 1994.

[22] Y. Zhong, K. Karu, and A.K. Jain, "Locating Text in Complex Color Images," *Pattern Recognition*, vol. 28, no. 10, pp. 1,523–1,536, Oct. 1995.