ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2, Issue 12, December 2013

Text Detection and Extraction in Video Sequences

P.JAYAPRIYA [1] **V.GOPI**[2] **C.NARASIMHAN** [3]

***Abstract*—The objective of the paper is to In the video text detection stage, text locations in each frame are found via an unsupervised clustering performed on the connected components produced by the stroke width transform (SWT). Since SWT needs an accurate edge map, Efficient indexing and retrieval of digital video data is an important aspect of video database management. Video indexing involves content analysis of video sequences, which is usually a computationally intensive process. Since most video data are stored in compressed format, processing directly in the compressed domain offers the possibility of computationally more efficient algorithms. We present here an efficient algorithm for detecting and extracting text directly in MPEG video sequences. Most existing methods for text detection in video usually operate on raw pixel data and only output the location of the detected text regions in a single frame. Our algorithm makes use of the features encoded in compressed data to perform fast text detection. Motion information and the characteristics of text region continuity in multiple frames are then used to fine-tune the detected candidate text regions. Furthermore, characters are reliably extracted by an adaptive thresholding method after applying some noise reduction filtering in multiple frames. Such extracted characters can be directly fed into a conventional OCR system for recognition. Experimental results on several video sequences show that the proposed algorithm is able to detect and extract text in MPEG video sequences with various scene complexities..**

P.JAYAPRIYA M.E In Applied Electronics. Of Bharathidasan Engineering College, Nattarampalli, Vellore Dist. *Tamil Nadu, Mobile No.,+919566355488*

***V.GOPI**, M.E In Embedded Systems, P.S.V.college of Engineering & Technology, Krishnagiri, Tamil Nadu, Mobile No.,9487326748.*

C.NARASIMHAN is an Assistant professor in the HOD of ECE in Bharathi dasan Engineering College, Nattarampalli, Vellore Dist.,*Tamil Nadu, Mobile No.,+919626138258*

***Index Terms*— SWT, video inpainting, MPEG video, framework, OCR ,restoration process**

Introduction:-

Digital video now plays an important role in entertainment, education, and other multimedia applications. With the ever-increasing amount of archival videos, there is an urgent demand for tools that will allow efficient browsing and retrieval of video data. In response to such needs, various video content analysis techniques using images, audio, and textual information present in video have been proposed to parse, index, and annotate a massive amount of data [3, 11, 14]. Among these information sources, text appearing in a video sequence plays an important role in understanding the content of a raw video sequence.

Text in a video can be broadly categorized into two kinds: *scene text* and *artificial text*. Scene text is part of the environment and is captured by the camera along with the rest of the scene. Artificial text is produced separately from the video shooting and overlaid over the scene in a post-processing stage. We are interested in methods of detecting artificial (or embedded) text in video since it carries important information on the semantics of the video content. For example, caption text present in news broadcasts and documentaries usually annotate information on *where, when, and who* of the reported events. Also in sports programs, scores and player or team names are often superimposed on the video in textual form, instead of being spoken aloud.

In summary, text in video provides highly condensed information about video content and can be used for video indexing, browsing, and retrieval in large video databases. It is not an easy task to reliably detect and extract text appearing in images and videos. In a single frame, the size of characters can vary within a large range (from very small to very big). They can also have multiple colors and font styles. Besides, text can appear over a very cluttered background. For

video sequences, the background can be moving/changing, independent of the text. Nevertheless, a number of algorithms to detect embedded text from still images and video sequences have been published in recent years [1, 4, 8, 9, 11, 12]. These algorithms are based on the following properties of text:

- characters are bounded in size;
- characters contrast strongly with their background; and
- characters appear in clusters of sharp edges confined in a rectangular region. Most of the published methods for text detection can be categorized as either *component-based* or *texturebased*. For component-based methods, text regions are detected by analyzing the geometrical arrangement of edges or homogeneous color/greyscale components that belong to characters. For example, Smith and Kanade [11] located text as horizontal structures of clustered sharp edges. Similarly, Lienhard [8] identified text as connected components which are of same color and have corresponding matching components in successive video frames. Jain and Yu [4] decomposed the video frames into subimages of different colors and then examined if each subimage contains text components which satisfy some specified heuristics.

In this paper, we extend our previous method of detecting text regions directly in MPEG video sequences [3] by adding the text region consistency check in multiple frames and enhancing detected text regions by multiple frame integration. Characters are then extracted using an adaptive thresholding method for recognition by commercial OCRs.

The rest of paper is organized as follows: The details of the proposed method are described in Section 2, which includes text detection, in particular the text region consistency check in multiple frames, and text extraction. Experimental results are presented in Section 3. Section 4 concludes the paper and presents future work.

Proposed Method:-

The MPEG compression scheme [6] uses a suite of techniques to reduce both spatial and temporal redundancies in a video. There are three basic frame types in an MPEG stream: I-, P-, and B-frames. I-frames are compressed using Discrete Cosine Transform (DCT) of local blocks (8*8). B- and P-frames are introduced to reduce temporal redundancies, where a P-frame is predictively coded with motion compensation from an I- or P-frame preceding it, and a B-frame is bi-directionally interpolated using the two I- or P-frames before and after it. Our method consists of two main steps: text detection in MPEG compressed domain and text extraction in decoded text regions. Text detection operates initially on I-frames and is achieved by capturing intensity variations, which are characterized by the AC coefficients of each block.

The motion vector information of the following P-frame is then used to remove some moving blocks. This is followed by the post-processing, including morphological operations, region labeling, and text region consistency checking in multiple frames. The text extraction step consists of noise reduction filtering and thresholding operations. Summarizes the major steps of the proposed method.

Text Detection:
Detection of Sharp Edge Blocks

Since characters usually contrast strongly with the background, a text region will contain blocks of sharp edges, characterized by large intensity variations. The AC coefficients of each 8*8 luminance block in an I-frame capture intensity changes within the block in different directions at different scales. For each block (*i*, *j*) in an I-frame, we can use the average AC energy *Eac*(*i*, *j*) as its edge-ness measure, calculated as follows

$$E_{ac}(i,j) = \left(\sum_{u=0}^{7} \sum_{v=0}^{7} |C_{uv}(i,j)| \right) / 63 \qquad (u,v) \neq (0,0) \qquad (1)$$

where *Cuv* is the AC coefficient at the horizontal and vertical frequencies *u* and *v* respectively. For each I-frame in an MPEG video, we calculate this average AC energy for all the luminance blocks. These AC energy values are then thresholded to obtain the blocks of sharp edges. A block is declared as a candidate text block if its AC energy value exceeds some threshold. This simple thresholding procedure detects all blocks with

a high AC energy value and thus also picks up some non-text blocks of sharp edges.

Removal of Moving Background Blocks:-

Since embedded text do not move from frame to frame, some above detected candidate text blocks, which move between frames and belong to the background, can be removed. More specifically, it is observed that the corresponding.block of a static text block in the following P-frame will not be encoded at all (simply skipped) or have a zero motion vector for coding efficiency. As a result, if a candidate block is skipped or has a zero motion vector in the following P-frame, it remains as a candidate text block. Otherwise it corresponds to a moving background block and is removed. This procedure is particularly effective in eliminating moving background with lot of textures, such as in sports video where lot of motion is involved. This step can be skipped if there are no P-frames following the I-frames or when moving text is expected.

Refining Candidate Text Blocks:-

The above processing considers only the edge-ness within each block and the results tend to be noisy. A text region usually consists of characters connected together horizontally to form a row of one or more words while non-text blocks occur generally random and rarely merge collectively into rows. Considering the nature of text appearing
horizontally, we use a structuring element of size blocks to apply a morphological *closing* operation followed by an *opening* operation to the binary image obtained from the above two steps.

This morphological filtering basically removes most of the isolated candidate text blocks, fills in the holes, and merges the nearby detached text blocks in the horizontal direction into coherent regions.The resulting candidate text blocks are further processed to form text regions. Specifically, an 8-neighbor connectivity region labeling process is applied to the above binary image to form text regions from the connected blocks. Each region has a unique label. Small regions and thin vertical regions are removed.

For each text region, its rectangular bounding box is used to represent it. Regions, whose bounding boxes overlap, are merged to form a large region with a new bounding box. As a result, each I-frame will have a set of bounding boxes to represent the detected text regions.

Consistency Checking in Multiple Frames:-

Most existing methods usually perform text detection in a single frame and information present in multiple frames is not used. Since a text region usually lasts at least two seconds in order to be read comfortably, each potential text region detected in the I-frames is checked for this continuity to see whether it has a matching region at a similar location in the following I-frames. If a text region appears only in one of the I-frames, it is considered as random and removed. Otherwise, the duration of each text region is checked and text regions, which appear for less than two seconds, are removed. The start frame and end frame for each remaining text region are marked and recorded for later processing. This simple checking effectively removes random candidate regions containing sharp edges.

Text Region Change Detection:-

From the above processing, each potential text region is associated with a start I-frame and an end I-frame. In some video sequences such as videos with subtitles, text appears at some particular location in every frame and the bounding boxes in these I-frames are similar to each other. As a result, the above procedure tends to detect them as the same text region although there are abrupt changes of text within the whole duration.

We need to detect these abrupt changes in order to separate and detect all the text regions. A change is detected if the text region difference in two successive I-frames is large. The percentage of pixels with a large grey level difference is used as a measure for this difference as it emphasizes more on large changes than other measures such as the average difference. If this percentage is larger than a threshold, a change is declared and a new

text region is formed at the same location with the same bounding box. The difference detection continues until the last I-frame of each text region.

Text Extraction:-

The above text detection procedure only outputs a set of regions in a video, which indicate where in a frame the text appears and its duration. Most existing methods for text detection usually end here. To annotate a video using the detected text, it must be extracted and recognized. Since text in video often appears over a complex background and its resolution is low, especially in MPEG-1 videos, it is difficult to generate a binarized image with large black characters over a clean white background, as required by commercial OCRs, from a detected text region by thresholding. Although there are some efforts to enhance text in videos [7, 9], most of these methods are effective only in some applications and operate only on raw pixel data. In the following, an efficient method for extracting characters from the detected text regions is described. For each text region, its pixel data from all the I-frames between the start frame and the end frame, which can be quickly decoded from an I-frame, are used in the following processing.

Text Region Enhancement:-

As mentioned above, text often appears over a complex background. This makes the task of character segmentation and recognition difficult. However, the same text usually appears in multiple frames and this data redundancy in video can be used to enhance the text region. Firstly we can reduce white noise by averaging the pixel values of a text region in all I-frames where it appears. Secondly, this averaging process also smooths the background pixels. Especially when the background is moving or changing, the averaged background pixels will be mostly uniform and easy to be separated from the characters. As a result, for each pixel in a text region we calculate the average value of the corresponding pixels in all the I-frames containing the text region, as following:

$$\bar{g}(i) = \left(\sum_{f=1}^{N} g_f(i) \right) / N \qquad (2)$$

where $g\,f\,(i)$ is the grey level value of the pixel in frame I and N is the total number of I-frames containing the text region.

Text Line Separation:-

Characters within a text region can be in multiple colours (or intensity values) and it is therefore difficult to extract characters from the text region using a global threshold. It is observed that characters in a single text line are however mostly monochrome for its uniformity and coherence. As a result, if a text region is separated into individual
text lines, global thresholding can be applied to each text line to extract the characters. Based on the property that text lines are arranged from top to bottom with a small gap between them to form paragraphs, we calculate the projection of the measure *Di* on to the Y-axis, defined in the following:

$$D_i = \begin{cases} 1 & if \quad \sigma_i > 25 \\ 0 & otherwise \end{cases} \qquad (3)$$

where s*i* is the standard deviation of the grey-level values *p*(*y*) of the pixel within a small window (3x3). Character pixels have high standard deviations while non-text background pixels have low standard deviations (see [7]). The projection value at the location is simply the total number of pixels with high standard deviation values in the horizontal direction.

Text lines show high values on the projection curve while the small gap between two text lines is a valley on the curve. By detecting regions of high values on this projection curve, text lines can be exactly located, along with their heights. False detection can also be eliminated by this procedure as no such characteristics will be present for non-text regions.

Text Line Binarization:-

Once the text region is enhanced and text lines are separated, character extraction in each line can be performed. Since characters of a text line are usually monochrome, either darker or brighter than the background, an optimal global threshold value for segmenting the characters in each text line can be found. Of course, this threshold must

be adaptive, i.e. its value depends on the region to be processed. There are several techniques available for selecting an optimal threshold in the literatures. Here we adopt the iterative selection method developed by Ridler and Calvard [10] and extend it to handle multiple (more than 2) subregions. For each region *R* (in our case, a text line), the original iterative selection method determines the optimal threshold *To* by repetitively partitioning the region into two subregions using a threshold, which changes iteratively until it reaches a stable value. This threshold is optimal only if the considered region is formed by two major subregions, e.g. white characters over a mainly dark background. However, text can appear over a complex background in a video. Figure 2 shows one example of text appearing over a background of one dark region and one bright region. In such a case, the text region consists of three major subregions. Segmentation using the above threshold will result in either splitting of one subregion into two or two subregions still being included in the same region. In order to find an optimal threshold to correctly segment characters from a background containing multiple groups of grey level values, the following extension is proposed. For each of the two subregions partitioned by using the above threshold, the grey level values of the majority pixels can be calculated. These two values correspond to the maxima in the two regions separated by the threshold on the histogram curve. The distances from the maxima to the threshold on the histogram curve, i.e., the grey level differences between the maxima and the threshold, are indications of how well each subregion is segmented. If the maximum of a subregion is very close to the threshold, there is a large probability that this subregion contains more than one significant subregions. Therefore, we calculate the following for each subregion to measure the initial segmentation quality:

$$S_i = \frac{|M_i - T_o|}{g_{avg}} \qquad i = 1, 2 \qquad (4)$$

The next step is to decide which subregion is the actual characters as it is unknown beforehand whether the characters are darker or brighter than the background. Here we use the number of pixels in each subregion to complete this task. Statistically, the number of character pixels only occupies about 20-30% of the total number of pixels in a text line no matter how the font and size of the characters vary. As a result, the character region is the subregion with the smaller number of pixels. The final result from this step will be a binarized image of black characters over a white background for each text region, which is the standard format required by conventional OCRs, no matter in what mode (inverse or normal) the original text appears in video.

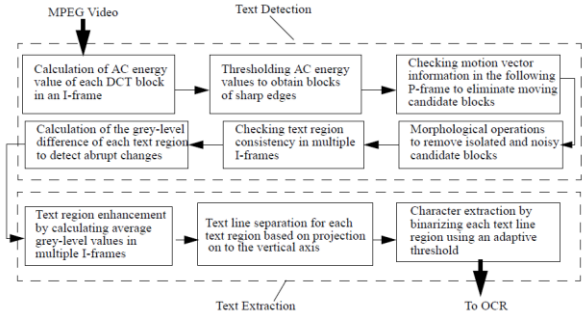

Figure 1. Major steps of the algorithm for text detection and extraction.

Experimental Results:-

The above algorithm has been implemented in MATLAB on top of a general MPEG parsing and decoding library. For each I-frame in an MPEG video sequence, the AC coefficients of their luminance blocks are extracted for calculating the average AC energy values. For P-frames following an I-frame, information on their motion vectors is also extracted from the encoded MPEG stream. results of the proposed text detection algorithm applied on two video sequences at various stages. shows the AC energy images of the first I-frames, in which text regions are detected. The candidate text blocks detected by thresholding the AC energy values in these two I-frame are shown in Figure 3b. It can be seen that using the AC energy values alone has also picked up non-text blocks of sharp edges. The use of motion vector information in P-frames eliminates moving non-text blocks. Random and isolated candidate text blocks are removed by the morphological operations.The detected text regions in the last I-frames.

The consistency check in multiple I-frames effectively removes random non-text regions. shows the results of text enhancement and extraction from the above detected text regions.
shows the grey level images of the text regions in the first I-frames while the same text regionsafter the average operation, which reduces noise and smooths the background. shows the segmentation results using a global threshold for each of the whole text regions.
The segmentation results using different thresholds for different text. By comparing the images of these two columns, it is obvious that dividing a text region into individual text lines before applying the binarization process is necessary. The adaptive threshold found by the iterative selection method works well for text lines with two major subregions.

However, it tends to include some background pixels into the character subregion for complex background
regions. The use of measure effectively identifies whether further splitting is necessary and the
final optimal threshold found leads to a correct segmentation. summarizes the test results in several video sequences under various scene complexities. Our algorithm achieves 99% detection rate and an average false detection rate of 27% in these sequences. This is very promising as almost all of text regions are correctly detected and extracted. Most false positives in the test sequences correspond to static textured background regions with strong contrast. They can be further reduced by applying criteria, such as the edge direction distribution in a text region. They will not be able to be recognized by OCR systems anyway.

Since only compressed data are used for the initial text detection and partially decoded data in I-frames are used for some post-processing and text extraction, our algorithm is much faster than most of the pixel-domain methods. Yet, the results have shown that our algorithm is very effective in text detection and extraction from videos
with various background complexities.

Text in video screen:-

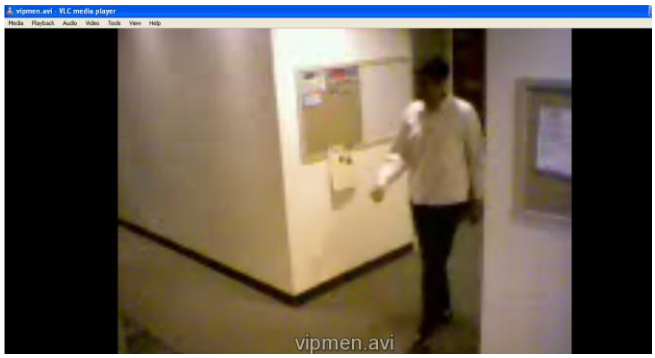

Figure:1 Text in video screen

Extraction in Video:-

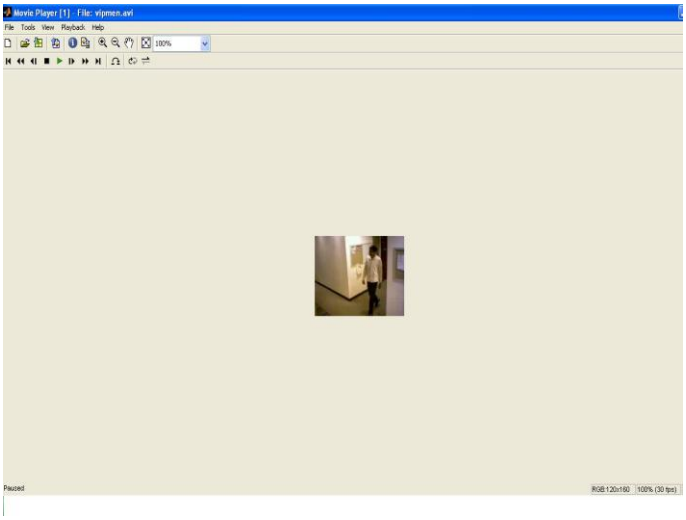

Figure:2 Extraction In Video

Conclusions:-

In this paper, an algorithm for detecting text regions and extracting characters directly from MPEG video sequences has been presented. Our algorithm uses the AC coefficients in I-frames to detect potential text blocks and motion vectors in P-frames to eliminate moving non-text blocks. Post-processing is applied to candidate text blocks to further group them into horizontally aligned text regions. The text region consistency check in multiple frames can effectively remove random

non-text regions. For text extraction, data redundancy in video is explored to reduce noise and smooth background pixels by averaging the grey level values of each text region in multiple frames. Adaptive thresholding is performed in each text line to extract characters. The original iterative thresholding method has been extended to handle complex regions by introducing the quality measure of the subregion segmentation. Experimental results have shown that the extracted characters can be directly fed into conventional OCRs for recognition.

Currently, we are exploring the use of color information to achieve better segmentation when the background contains pixels with a similar brightness value to those of characters. Also some morphological filtering can be performed on the binarized text region to further enhance the extracted characters for better recognition results.

References:-

[1] L. Agnihotri and N. Dimitrova, Text detection for video analysis, *IEEE Workshop on CBAIVL*, Colorado, 1999.
[2] U. Gargi, S. Antani, and R. Kasturi, Indexing text events in digital video databases, *Proc. Int. Conf. on Pattern Recognition(ICPR)*, pp 916-918, 1998.
[3] L. Gu, Scene analysis of video sequences in the MPEG domain, *Proc. Int. Conf. on Signal and Image Processing (SIP)*,pp 486-490, Las Vegas, 1998.
[4] A.K. Jain and B. Yu, Automatic text location in images and video frames, *Pattern Recognition*, vol. 31, no. 12, pp 2055-2076, 1998.
[5] A.K. Jain and Y. Zhong, Page segmentation using texture analysis, *Pattern Recognition*, vol. 29, no. 5, pp 743-770. 1996.
[6] D. LeGall, MPEG: A video compression standard for multimedia applications, *Commun. ACM*, vol. 34, no. 4, pp 46-58,
1991.
[7] H. Li and D. Doermann, Text enhancement in digital video using multiple frame integration, *Proc. ACM Multimedia*, pp.19-22, Orlando, Florida, 1999.
[8] R. Lienhart, Indexing and retrieval of digital video sequences based on automatic text recognition, *Technical Report 6/96,*University of Mannheim, 1996.
[9] M.A. Shim and C. Dorai, and R. Bolle, Automatic text extraction from video for content-based annotation and retrieval,*Proc. Int. Conf. Pattern Recognition (ICPR)*, pp. 618-620, 1998.
[10] T. Ridler and S. Calvard, Picture thresholding using an iterative selection method, *IEEE Transactions on Systems, Man,and Cybernetics*, vol. SMC-8, no. 8, pp 630-632, 1978.
[11] M. A. Smith and T. Kanade, Video skimming and characterization through the combination of image and language understandingtechniques, *Technical Report CMU-CS-97-111*, Carnegie Mellon University, Feb. 1997.
[12] V. Wu, R. Manmatha, and E.M. Riseman, TextFinder: an automatic system to detect and recognize text in images, *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 21, no.11, pp 1224-1229, Nov. 1999.
[13] B.L. Yeo and B. Liu, Visual content highlighting via automatic extraction of embedded captions on MPEG compressedvideo, *SPIE Digital Video Compression: Algorithms and Technologies*, February 1995
[14] 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.
[15] Y. Zhong, H.J. Zhang, and A.K. Jain, Automatic caption localization in compressed video, *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 22, no.4, pp 385-392, April 2000.

P.JAYAPRIYA Received B.E. degree from University of periyar, India in 2003. At Studying IIYear M.E In Applied Electronics. Of Bharathidasan Engineering College, Nattarampalli, Vellore Dist. And Doing Research In images processing.

V.GOPI Received The B.E Degree From Anna University, In 2010.At Studying I Year M.E In Embedded Systems Technologies. Of P.S.V.College Of Engineering & Technology. And Doing Research In Wireless Communcations

C.NARASIMHAN is an Assistant professor in the HOD of ECE in Bharathi dasan Engineering College, Nattarampalli, Vellore Dist.India.. He received his Master of Engineering in Applied Electronics in 2005. He is doing his research work in Embedded systems**.**