HAL
archives-ouvertes.fr

Efficient object based streaming framework for web-based education

Ashraf M. A. Ahmad, Samir A. El-Seoud

▶ To cite this version:

Ashraf M. A. Ahmad, Samir A. El-Seoud. Efficient object based streaming framework for web-based education. Michael E. Auer. Conference ICL2007, September 26 -28, 2007, 2007, Villach, Austria. Kassel University Press, 19 p., 2007. <hal-00197210>

HAL Id: hal-00197210
https://telearn.archives-ouvertes.fr/hal-00197210

Submitted on 14 Dec 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

EFFICIENT OBJECT BASED STREAMING FRAMEWORK FOR WEB-BASED EDUCATION

Ashraf M. A. Ahmad and Samir A. El-Seoud
Department of Computer Science, IT colleague
Princess Sumaya University, Amman Jordan
Ashraf@psut.edu.jo

Abstract

An efficient moving object extraction algorithm suitable for real-time content-based multimedia streaming systems is proposed in this paper. A Motion Vector (MV) based object extraction is used to dynamically detect the objects. To utilize the bandwidth efficiently, the important object can be real time detected, encoded, and transmitted with higher quality and higher frame rate than those of background. In order to meet the real-time requirement, no computationally intensive operation is included in this framework. Moreover, in order to guarantee the highest speed, all the implementation is operating on the compressed domain without need for decompression. Good extraction performance is demonstrated by the experiment results.

Keywords: Object Detection, Video Streaming, MPEG1/2, Texture, Motion Vector, Web-based Education.

1. Introduction

[19] Uskov pointed out multimedia streaming as one of the active tools for advanced web-based education systems. Videos extraction, which extracts the shape information of moving object form the video sequence, is a key operation for object based video streaming [20], multimedia content description [3], [4], and intelligent signal processing. However, the shape information of moving objects may not be available from the input video sequences; therefore, extraction is an indispensable tool. In addition, many multimedia streaming applications have real-time requirement, and an efficient algorithm for automatic video extraction is very desirable. To achieve the objective of object based streaming system, a reliable object extraction mechanism is needed as a primary step. There are some sources of information in video that can be used to detect objects: visual attributes (such as color, texture and shape) and motion information (such as motion vector). Motion extraction complicates the object extraction problem by imposing the additional requirement of tracking an object's temporal position.

It also provides an additional information source that can be exploited for the purpose of object extraction by algorithms operating over the uncompressed [1] or compressed domains [2,3]. When using visual attributes for object extraction, we will often need to perform processing in the pixel domain, which includes the additional burden of attribute extraction. Performing the object extraction only based on the visual attributes in the pixel domain could be based on shape [4,5], color [6,7], or other visual features. Approaches using certain complex analysis in the pixel level are extremely computationally intensive and have other drawbacks compared with the approaches in the compressed domain. We concentrate on doing object extraction in compressed domain. Although processes in uncompressed domain give accurate results, the work in compressed domain has these advantages. Most videos are not provided in the form of image sequences, but rather as compressed formats. Implementation of the same manipulation algorithms in the compressed domain will be much cheaper than that in the uncompressed domain as the data rate is highly reduced in the compressed domain (e.g., a typical 20:1 to 50:1 compression ratio for MPEG). Compressed video data offers us additional information like DC coefficients and motion vectors.

2. Related Work

The motion information can be available from the compressed domain. In many cases, especially in the case of well-textured objects, the motion vector values reflect the movement of objects in the stream very well. Some approaches [8,9] utilize these motion vector values directly. Processing digital video directly in the compressed domain has reduced processing time, enhanced storage efficiency, speed, and video quality. Object extraction directly in compressed video without full-frame decompression is clearly advantageous, since it is efficient and can more easily reach real-time processing speeds. Motion vector information is an important cue for humans to perceive video content. Thus, the need for reliable and accurate motion vector information becomes clear for those approaches that are employing the motion information [2,3,8] as well as to get highly efficient extraction algorithms at the macroblock level. Motion vector information is sometimes difficult to use due to the lack of effective representation and due to the fact that it introduces large amounts of noise which make further processing of the data impractical. Besides, it is still far from ideal in performance, as the key motion estimation is carried out using a coarse area-correlation method that has proven its inefficiency in terms of accuracy. Some researchers [10] elaborate on the noise in motion vectors due to camera noise and irregular object motion.

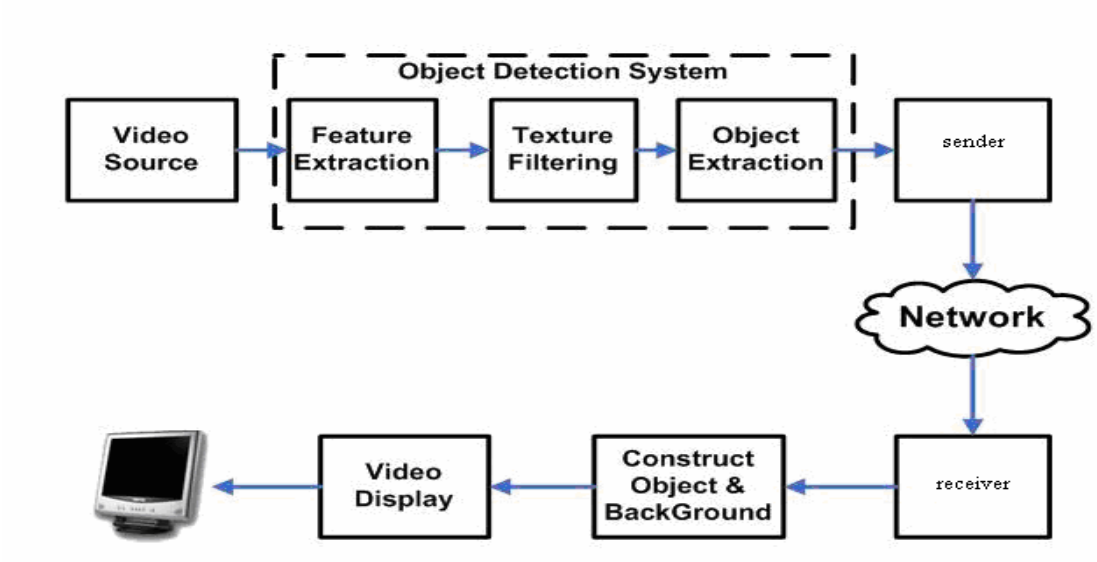
It is known [11] that the motion fields in MPEG streams are quite prone to quantization errors, especially in low-textured areas. However, typical samples in the motion vector field are usually inaccurate [14,15]. These defects can be combated with robust error recovery schemes that repair motion fields and reduce noise. Consequently we can produce a smoother shape a boundary, where the motion vectors are used to determine

object boundaries in object extraction. Therefore, in this paper, we introduce a technique that can overcome those defects and produce more reliable motion vector information and smoothed object boundaries for the object extraction technique. Initially, we pass the result of the features extraction into the texture based filter; which will be described later in detail. In this way, many situations that may cause trouble in conventional approaches can be handled properly without using complex operations. [16] used P,B frames. One [14] applied a median filter for the magnitude only, while we applied it for both magnitude and direction which resulted in a more accurate and reliable outcome in terms of object extraction. [11] used spatial confident measure which is Mean- Filter like, where authors in [14] proved this insufficient in terms of accuracy and unrealistic for real-time application. In that, [11] combined both the texture and spatial measure equally, which was proven insufficient in terms of accuracy and unrealistic for the real-time applications. [14] uses spatial filter without regarding the texture measure which resulted in a less realistic result.

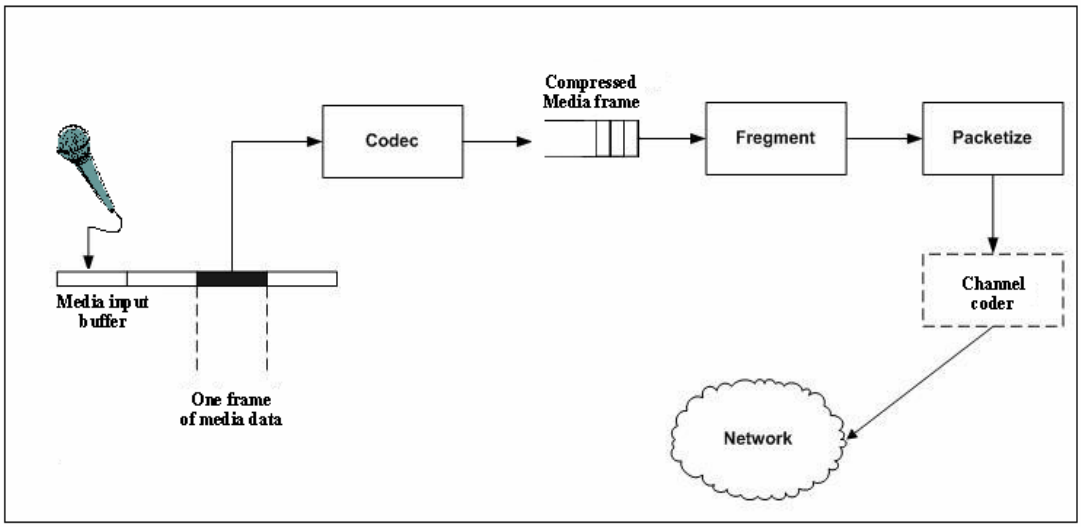
3. System Overview

In general, the object-based video streaming technique can be designed as stated in Fig. 1. Our proposed scheme is located in block which is surrounded by dashed lines. Initially, in this figure, we capture video stream in MPEG format, extract features, detect moving objects and transmit their encoded streams. Fig 1 illustrates the streaming system architecture, which covers four key modules, including the Object Extraction, Sender, Receiver, and Composer.

(a)



(b)



(c)

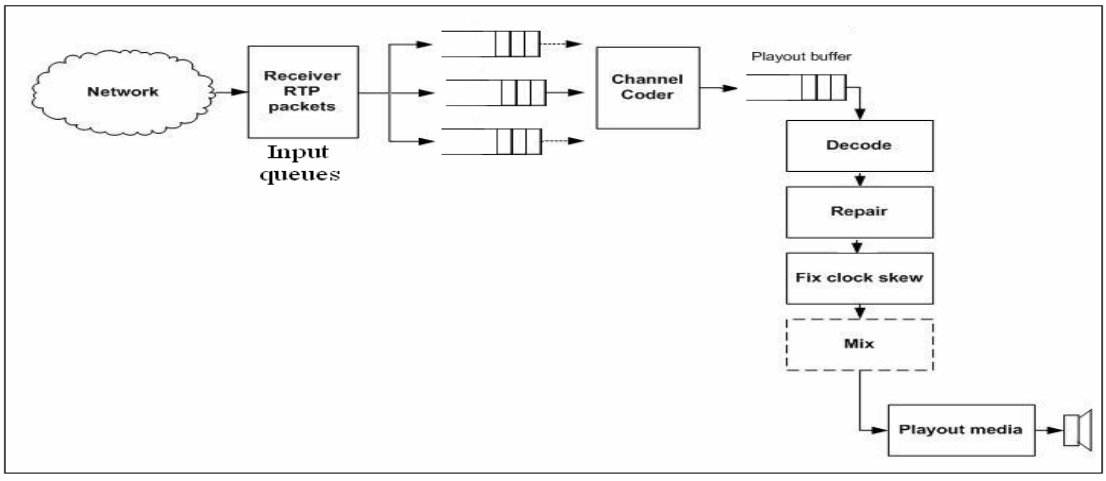


Fig. 1: Streaming System Architecture (A), Block Diagram of RTP Sender (B) and Block Diagram of RTP Receiver (C)

4. Object Extraction System

The MPEG compressed video provides one motion vector for each macroblock of size 16x16 pixels, which means that the motion vectors are quantized to 1 vector per 16x16 blocks. The motion vectors are not the true motion vectors of a particular pixel in the frame. Our object extraction algorithm requires motion vectors of each P-frame from the video streams. Our system takes the motion vectors from the compressed video stream as the only input. For the computational efficiency, only the motion vectors of P-frames are used for object extraction algorithm. Besides, we need to extract the DCT information from I-frames. This information is readily available in MPEG stream, thus too much time is not spent in decoding the MPEG stream. Hence, our approach fits for the real-time application environment.

4.1 Features Extraction

Now, we will present the following diagram which states an abstract overview of our object extraction proposed system, and then we will describe its components in detail. In our proposed approach we first take an MPEG video stream with the [IBBPBBPBBPBBPBB] structure. *Fig.2* shows the proposed system architecture. Next, we extract the motion vectors from P-frames only in order to reduce the computational complexity. Since, in general, in a video with 30 fps, consecutive P-frames separated by two or three B- frames are still similar and would not vary too much. It must be noted that B-frames are just "interpolating" frames that hinge on the hard motion information provided in P-frames and therefore using them for the concatenation of displacements would be redundant. It is sufficient to use the motion information of P-frames only to detect the objects. Meanwhile, we will extract the DCT coefficients from I frames, these coefficients include the DC coefficient, and the AC components as well. Then, we will pass the DCT coefficients into a module to calculate the energy values "texture" of each frame.

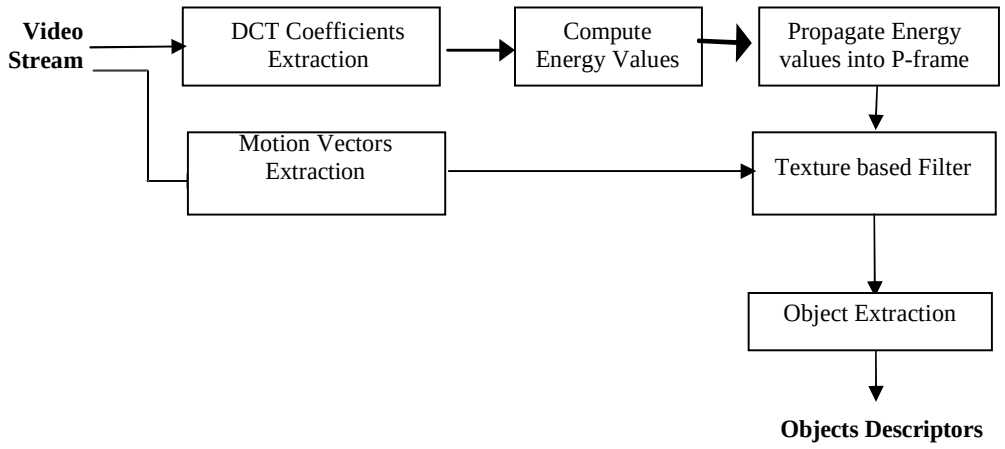


Fig. 2 : The Object Extraction System Overview

After which, we will propagate these "texture information" values into P frames. We pass these texturally filtered motion vectors into our object extraction algorithm to get a set of detected objects in each frame. Steps will be described in detail in the following sections.

4.2 Textures Filtering

In fact, in most cases motion vectors are not only inaccurate, but in some cases they are meaningless. This will not allow even a sturdy fitting stage to operate reliably. It is more robust if these low-textured macroblocks are not included in the fitting. By doing so, we analyze the AC components of the DCT coefficients and DCT coefficient, thus staying in the compressed domain.

4.2.1 Texture Energy Computation

Object regions are distinguished from non-object regions using their distinguishing texture characteristics. Unlike previously published methods which fully decompress the video sequence before extracting the desired object regions. This method helps in locating the candidate object regions directly in the compressed domain by using the intensity variation information encoded in the DCT domain. In some applications, researchers have used either horizontal intensity variation, or vertical intensity variation. We design *Directional Texture Energy Map* (DTME) in DCT domain *Fig.3*, by assigning a directional intensity variation indicator for each coefficient in DCT domain. The DC or AC components are indicated as the following:

H: Horizontal intensity variation.
V: Vertical intensity variation.
D: Diagonal intensity variation.

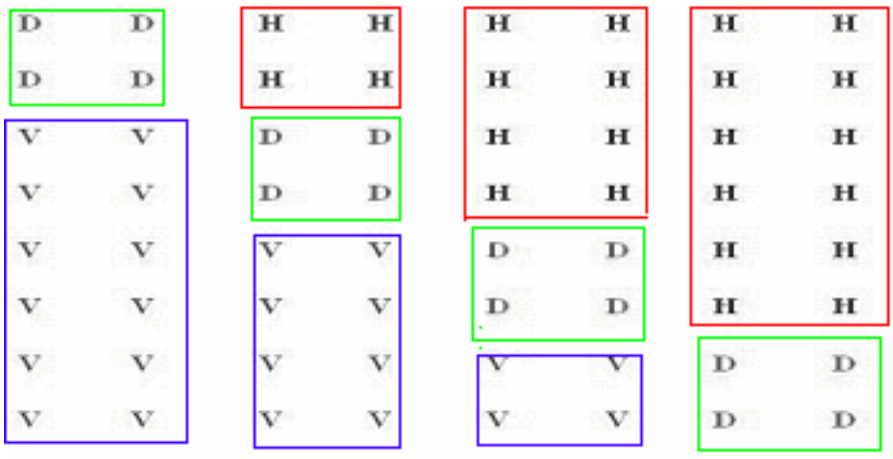


Fig.3: Directional Texture Energy Map in DCT

We are processing in the DCT domain to obtain the directional intensity variation or the so called directional texture energy using only the information in the compressed domain. Note that the operating units are the 8X8 blocks in I-frames. Blocks of Horizontal Spatial Intensity Variation for each 8X8 DCT block, we compute the Horizontal energy Eh by summing up the absolute amplitudes of the horizontal harmonics of the block: which have been marked as H in the DTME. Blocks of Vertical Spatial Intensity Variation for each 8X8 DCT block, we compute the Vertical energy Ev by summing up the absolute amplitudes of the vertical harmonics of the block: which have been marked as V in the DTME. Blocks of Diagonal Spatial Intensity Variation for each 8X8 DCT block, we compute the Diagonal energy Ed by summing up the absolute amplitudes of the Diagonal harmonics of the block: which have been marked as D in the DTME. Finally, we will calculate the average energy Ea. for each macroblock, which is the average value of the Vertical energy, Diagonal energy and Horizontal energy, as the following:

$$E_a = \frac{3 \times (E_h + E_v)}{8} + \frac{2 \times E_d}{} \qquad (1)$$

4.2.2 Texture Filter Process

Upon getting the average energy, these average energy values are then thresholded to obtain the blocks of large intensity variations. We will update motion vector values based on the Ea as described in the following procedure.

For every macrblock:

$$Motion\_Vector_{new} = \begin{cases} Motion\_Vector_{old} \times \frac{100 \times E_a}{E_t} \%, & E_a < E_t \\ Motion\_Vector_{old}, & E_a \geq E_t \end{cases} \quad (2)$$

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 frame.

4.3 Object Extraction

So far, we have obtained the fine motion vector. We started by eliminating undesired motion vectors before the process of extraction in order to achieve more robust performance. Motion vectors with magnitude equal to or approaching zero are recognized as undesirable and hence are not taken into consideration. On the contrary, motion vectors with larger magnitude are considered more reliable and therefore are selected.

4.3.1 Object Extraction Algorithm

An object extraction algorithm is used to detect potential objects in video shots. Initially, motion vectors that have similar magnitude and direction are clustered together and this group of associated macroblocks of similar motion vectors is regarded as a potential object. Details are presented in our previous published paper. *Fig. 3-4* show the extraction results of using our system and without using any kind of filtering respectively. *Fig.4* shows the result of object extraction using directly extracted motion vector, *Fig.3* shows the result using our system. All of these figures use the same frames with the same extracted motion vectors. The detected objects are marked over with thicker, darker lines in Fig 3-4. The reader can notice how precisely our system is able to detect the objects.

9

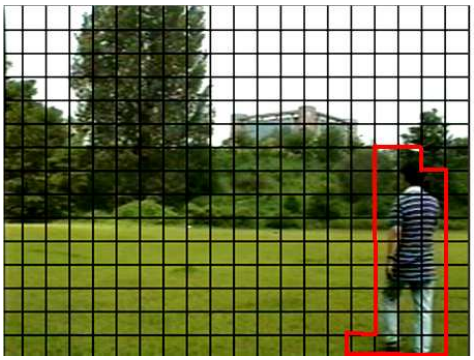


Fig.3: Object Extraction using our System

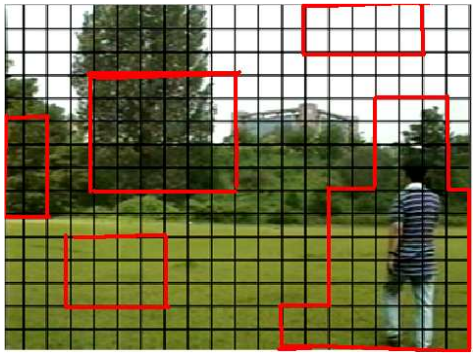


Fig.4: Object extraction without processing.

5. Video Streaming

To link the sender, receiver and transmission channel for real time displaying video sequences, a standard RTP based network streaming technique is used. The streaming module uses two categories of network protocols containing the network-layer protocol and transport protocol. Basing this on IP network, the network layer protocol uses a network address to serve the basic network support. The majority of transport protocols perform over an RTP stack, which is implemented on top of UDP/IP to provide an end-to-end network transport for video streaming. A sender is responsible for capturing and transforming audiovisual data for transmission, as well as for generation RTP packets, it may also participate in error detection and correction and congestion control by adapting the transmitted media stream in response to receive feedback. Uncompressed media data is captured into a buffer, from which compressed frames are produced. Frames may be encoded in several ways depending on the compression algorithm used, MPEG format and so forth and encoded frames may depend on both earlier and later data. Compressed Frames are loaded into RTP packets, ready for sending. If frames are large, they may be fragmented into several RTP packets: If they are small, several frames may be bundled into a single RTP packet. Depending on the error correction scheme in use, a channel coded may be used to generate error correction packets or to recorder packets before transmission. After the RTP packets have been sent, the buffered media data corresponding to those packets is eventually freed. The sender must not discard data that might be needed for error correction or for encoding process. This requirement may mean that the sender must buffer the data for some time after corresponding packets have been sent, depending on the codec and the error correction

10

scheme used. The sender is responsible for generating periodic status reports for the media streams it is generating, including those required for lip synchronization. It also receives reception quality feedback from other participants and may use the information to adapt its transmission. A receiver is responsible for collecting RTP packets from the network, correcting any losses, recovering the timing, decompressing the media, and presenting the result to the user. It also sends reception quality feedback, allowing the sender to adapt the transmission to the receiver, and it maintains a database of participants in the session. Implementations sometimes perform the operations in a different order depending on their needs. The first step of the receiver is to collect packets from the network, validate them for correctness, and insert them into a sender-specific input queue. Packets are collected from input queue and passed to an optional channel-coding routine to correct for loss. Following the channel coder, packets are inserted into a source-specific playout buffer. The playout buffer is ordered by timestamp, and the process of inserting packets into the buffer corrects any reordering induced during transport. Packets remain in the playout buffer until complete frames have been received, and they are additionally buffered to remove any variation in interpacket timing caused by the network. Calculation of the amount of delay to add is one of the most critical aspects in the design of RTP implementation. Each packet is tagged with the desired playout time for the corresponding frame. After their playout time is reached, packets are grouped to form complete frames, and any damaged or missing frames are repaired. Following any necessary repairs, frames are decoded (depending on the codec used, it may be necessary to decode the media before missing frames can be repaired). At this point there may by observable differences in the nominal clock rates of the sender and receiver. Such differences manifest themselves as drift in the value of the RTP media clock relative to the playout clock. The receiver must compensate for this clock skew to avoid gaps in the playout. Finally, the media data is played out to the user. Depending on the media format and output device, it may be possible to play each stream individually- for example, presenting several video stream for playout- for example, combining several audio sources for playout via a single set of speakers.As is evident from this brief overview, the operation of an RTP receiver is complex, and it is somewhat more involved than the operation of a sender. This increased complexity is largely due to variability of IP networks: Much of the complexity comes from the need to compensate for packet loss, and recover the timing of a stream affected by jitter.

8. Experimental Results and Discussion

We have designed an experiment in order to verify optimal performance. The experiment has been designed to test the proposed scheme on three video clips. These video clips are in MPEG format and are part of the MPEG7 testing dataset. Testing is performed using four types of related work which are, Group A using Gaussian filter only [18], group B using Median filter [14], Group C using Cascade Filter , and group **D** our system, finally without any kind of post processing. In order to compare the performance among these four system's results, we make the following configurations fix among all the experiments. The frame size is 320x240 which implies that we have 20X15 macroblocks in each P frame. Our testing dataset presents walking persons in different dimensions, positions, speed, and can vary slightly in object size. We choose the *recall* and *precision metrics* because they are most commonly used to evaluate object extraction system performance [12,13]. *Fig. 5-8* show the results of object extraction performance over the second video clip among the MPEG testing dataset. We show the precision metric and recall metric of our object extraction scheme for this video clip both with and without the filter being used, and we construct manually the ground truth of the video clip. *Fig.5-6* illustrate the values of the recall and precision metrics for each frame in the video clip. We note that the performance of our system is consistently superior to performance using others schemes. We show the average recall metric and average precision metric for the whole clip in *Fig.7-8.* Again, our system topped them all. Through our experiment we noticed that there is a weakness in the single Gaussian filter performance when the object location is in the frame border. This can be explained due to the lack of information in the neighborhood near the border.

In summary, the proposed system boosts the performance, while keeping the computational complexity low. Both the Gaussian and Median filters are available as a readily implemented component in both hardware and software. In Addition, the motion vectors, DCT coefficient, and AC component are readily available in MPEG stream. As we refine the motion vectors resulting in vectors that are easy to process, execution time of the object extraction algorithm after using the filter will be reduced significantly compared to that without using any kind of post processing.

12

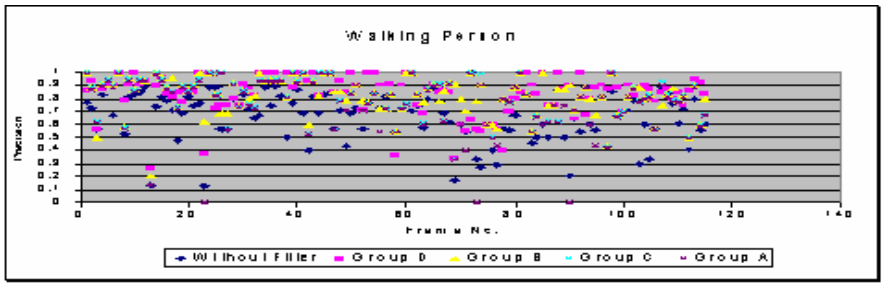


Fig.5: Precision for Object extraction in P frames

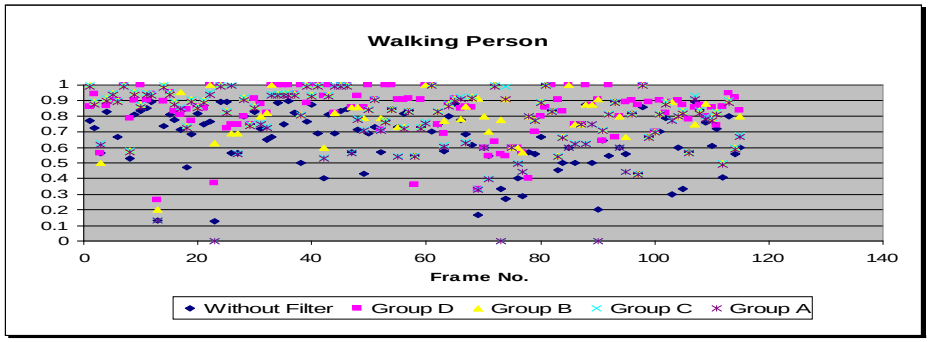


Fig.6: Recall for Object extraction in P frames

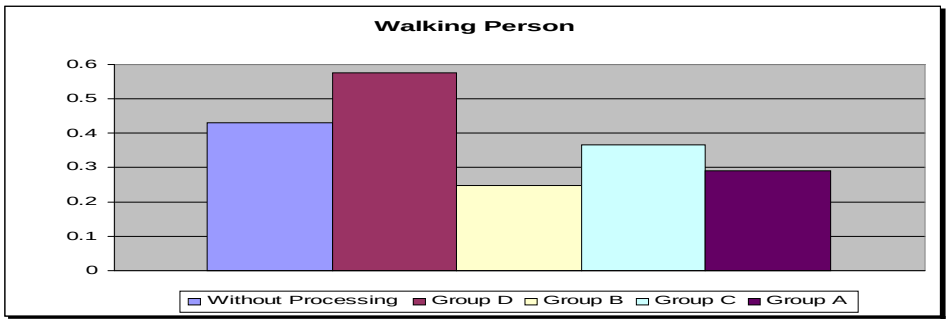


Fig.7: Average precision of object extraction for 2nd Video Clip

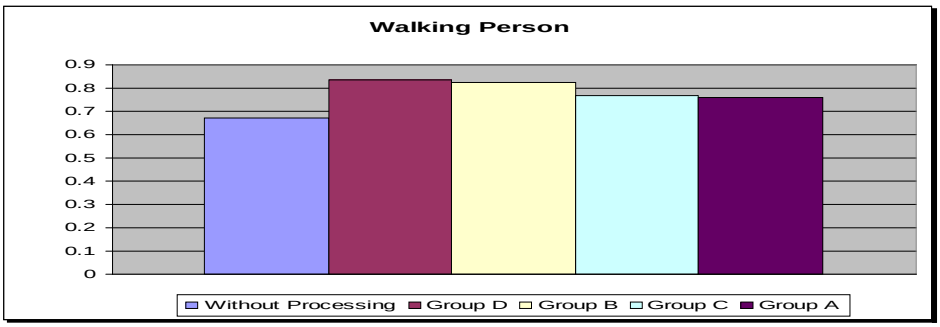


Fig.8: Average Recall of object extraction for 2nd Video

9. Conclusions and Future Work

In this paper, we present a novel object extraction scheme for the object-based streaming system. We are going to develop our streaming system to achieve QOS, and adopt error recovery scheme and RTCP control.

10. References

[1] N. Haering, R. J. Qian, and M. I. Sezan, "A Semantic Event-Detection Approach and Its Application to Detecting Hunts in Wildlife Video," IEEE Transactions on Circuits and Systems for Video Technology, Vol. 10, No. 6, 2000, pp. 857-868.

[2] H. L. Eng, and K. K. Ma, "Bidirectional Motion Tracking for Video Indexing," *Proc .Third IEEE Workshop on Multimedia Signal Processing*, 1999, pp. 153-158.

[3] L. Favalli, A. Mecocci, and F. Moschetti, "Object Tracking for Retrieval Applications in MPEG-2," *IEEE Transactions on Circuits and Systems for Video Technology, Vol. 10*, No. 3, 2000, pp. 427-432.

[4] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz, "Efficient and Effective Querying by Image Content," *Journal Intelligent Information Systems, vol. 3, no. 1*, 1994, pp. 231-262.

[5] A. Pentland, R. Picard, and S. Sclaroff, "Tools for Content-Based Manipulation of Image Databases.Storage and Retrieval of Image and Video Databases II," *Proc. SPIE,* 1994, 34-47.

[6] V. V. Vinod and H. Murase, "Video Shot Analysis using Efficient Multiple Object Tracking," *Proc. ICMCS*, 1997, pp. 501-508.

[7] P. Fieguth, "Color-Based Tracking of Heads and Other Mobile Objects at Video Frame Rates," *Proc. CVPR*, 1997, 21-27.

[8] R. C. Jones, D. DeMenthon and D. S. Doermann, "Building mosaics from video using MPEG Motion Vectors," *Proc. ACM Multimedia Conference*, 1999, pp. 29-32.

[9] J. I. Khan, Z. Guo and W. Oh, "Motion based object tracking in MPEG-2 stream for perceptual region discriminating rate transcoding," *Proc. ACM Multimedia Conference*, 2001, pp. 572-576.

[10] S. Chien, S. Ma, L. Chen, "Efficient Moving Object Segmentation Algorithm Using Background Registration Technique," *IEEE Transactions on Circuits and Systems for Video Technology, vol. 12*, no. 7, 2002, pp. 577-586.

[11] R. Wang, H.-J. Zhang and Y.-Q. Zhang, "A Confidence Measure Based Moving Object Extraction System Built for Compressed Domain," *Proc. ISCAS*, 2000, pp. 21-24.

[12] D.-Y. Chen, S.-Y. Lee, and H.-T. Chen, "Motion Activity Based Semantic Video Similarity Retrieval," *Proc. IEEE PCM*, 2002, pp. 319-327.

[13] S.-C. Chen, M.-L. Shyu, C. Zhang, and R.L. Kashyap, "Video Scene Change Detection Method Using Unsupervised Segmentation and Object Tracking," *Proc. ICME*, 2001, pp. 57-60.

[14] Ashraf M.A. Ahmad; Duan-Yu Chen and Suh-Yin Lee "ROBUST COMPRESSED DOMAIN OBJECT DETECTION IN MPEG VIDEOS" *Proc. 0f the 7th IASTED International Conference Internet and Multimedia System Applications*, 2003, pp. 706-712.

[15] M. Pilu, *On Using Raw MPEG Motion Vectors To Determine Global Camera Motion* (HPL-97-102, http://www.hpl.hp.com/techreports/97/HPL-97-102.pdf, 1997).

[16] R. V. .Babu, and K. R. Ramakrishnan , "Compressed Domain Motion Segmentation for Video Object Extraction", *Proc. of ICASSP*, 2002, pp. 3788-3791.

[17] N. W. Kim, T. Y. Kim, and J. S. Choi, "Motion analysis using the normalization of Motion Vectors on MPEG compressed domain," *Proc. ITC-CSCC2002*, 2002, pp. 1408-1411.

[18] Yu Zhong, Hongjiang, and Anil K. Jain "Automatic Caption Localization in Compressed Video", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000, 22(4): pp. 385

[19] V. Uskov, A. Uskov, "Blending Streaming Multimedia and Communication Technology in Advanced Web-Based Education" Jr. Advanced Technology for Learning, Vol. 1, No. 1, 2004, pp. 54-66.

[20] Ashraf M.A. Ahmad, and Suh-Yin Lee "FAST AND ROBUST OBJECT DETECTION FRAMEWORK FOR OBJECT BASED STREAMING SYSTEM," The Second International Conference on Advances in Mobile Multimedia, September 22-24 2004, pp. 332-339.

15