Fast and robust object-extraction framework for object-based streaming system

Ashraf M.A. Ahmad* and Suh-Yin Lee

Department of Computer Science and Information Engineering,
National Chiao-Tung University,
1001 Ta-Hsueh Rd, Hsinchu, Taiwan
Fax: 886-3-5724176
E-mail: ashraf@csie.nctu.edu.tw
E-mail: sylee@csie.nctu.edu.tw
*Corresponding author

Abstract: Video streaming poses significant technical challenges in the quality of service guarantee and efficient resource management. In this paper, we propose an efficient moving object-extraction algorithm suitable for real-time content-based multimedia streaming systems. A motion vector based object extraction is used to dynamically detect the objects. To utilise the bandwidth efficiently, the video objects can be detected in real time, encoded, and transmitted with higher quality and a higher frame rate than those in the background. To meet the real-time requirement, no computationally intensive operation is included in this framework. Moreover, to guarantee the highest speed, the entire implementation is operating in the compressed domain without any need for decompression.

Keywords: object extraction; video streaming; MPEG1/2; texture; motion vector.

Reference to this paper should be made as follows: Ahmad, A.M.A. and Lee, S-Y. (2008) 'Fast and robust object-extraction framework for object-based streaming system', *Int. J. Virtual Technology and Multimedia*, Vol. 1, No. 1, pp.39–60.

Biographical notes: Ashraf M.A. Ahmad obtained his BSc from Princess Sumya University for Technology in Jordan. He is currently researcher and member in Multimedia and Information System Lab, at National Chiao Tung University, as he is pursuing his PhD study in Computer Science and Information Engineering Department at NCTU in Taiwan. His research interests include MPEG video features extraction, object detection in MPEG video and multimedia retrieval and communication. He has published many conference and journal papers in his field of expertise. He has co-edited several conference proceedings and books. He chaired and organised many special issues in international conferences.

Lee-Suh Yin obtained PhD (Computer Science) from the Institute of Electronics, National Chiao Tung University. Her interest area includes multimedia information system, computer network, image/spial database, data mining. Since August 1991, she has been a Professor in the Department of Computer Science and Information Engineering, National Chiao Tung University. She has published many conference and journal papers in her field of expertise. She has co-edited several conference proceedings and books. She chaired and organised many special issues in international conferences.

1 Introduction

Video has been an essential element in communications and entertainment for many years. Initially, video was captured and transmitted in analogue mode. The emergence of digital integrated circuits and computers led to the digitisation of video, and digital video enabled a revolution in the compression and communication of video. Video compression (ISO/IEC, 1995; Information Technology, 1993) and transmission have become an important area of research in the last two decades. It enabled a variety of applications, including video storage on DVD and Video-CD, video broadcasting over digital cable, satellite and terrestrial digital television (DTV), High-Definition TV (HDTV), video conferencing and videophone over circuit-switched networks H.234M network.

With the rapid development of the internet, systems using video through the internet are rapidly increasing. Video over best-effort packet networks is complicated by a number of factors, including unknown time-varying bandwidth, delay and packet losses. In addition, many additional issues, such as how to fairly share the network resources amongst many flows and how to efficiently perform one-to-many communication for popular content 'congestion control'. The internet disseminates enormous amounts of information for a wide variety of applications all over the world. As the number of active users on the internet has increased, so has the tremendous volume of data that is being exchanged between them, resulting in periods of transient congestion on the network. In regard to data transmitted over the internet, some researchers' estimates (Chandra and Ellis, 1999; Ortega et al., 1997) and majority of the data bytes accessed on the web are in the form of multimedia objects.

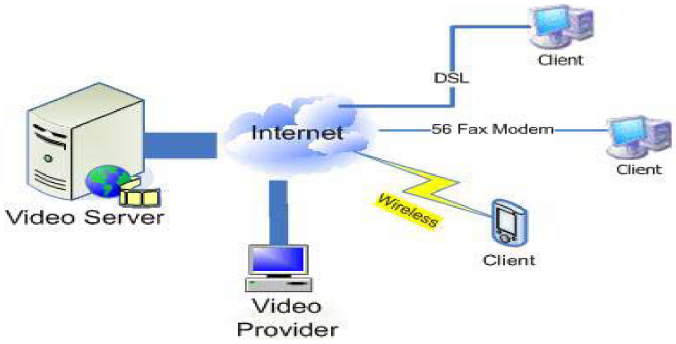
Generally, the most straightforward approach for video delivery in the internet is using an approach similar to a file download. We refer to this as a video download. Keep in mind that it is a video and not a general file type. Specifically, video download is similar to a file download, but it is a very large file. This scheme allows the use of established delivery mechanisms, for example TCP as the transport layer, FTP, HTTP, or HTTPS at the application layers. However, this scheme has a number of drawbacks. Generally, videos correspond to very large files, the download approach usually requires long download times and large storage spaces. These are all crucial practical limitations. In addition, the entire video file must be downloaded before viewing can start. This requires patience on the client part and also reduces flexibility in certain scenarios. In one scenario, if the client is unsure of whether he wants to view the video, he must still download the entire video before viewing it and making a decision. In another scenario, the user may not be aware of the exact disk space on his machine; therefore, he might start to download a large video file, which takes few hours, then an error message would pop up stating disk insufficiency. The user wasted hours but for nothing. These scenarios and other scenarios cause great obstacles in the video file download scheme. Video delivery by video-streaming attempts to overcome the problems associated with the video file download scheme, and also provides a significant amount of additional capabilities in 'viewing flexibility'.

The basic idea behind video streaming is to make "simultaneous delivery and playback". This splits the video into portions, transmits the portions in succession and enables the receiver to decode and playback the video as these parts are received, without spending time waiting for the entire video to be delivered. Video streaming enables simultaneous delivery and playback of the video. This is in contrast to file download where the entire video must be delivered before playback can begin. In video streaming,

there is usually a short latency (usually 10–15 s) between the start of delivery and the beginning of playback at the client. Video streaming provides a number of advantages, including low delays before viewing starts, and low storage requirements since only a small portion of the video is stored at the client at any point in time.

A general architecture for video streaming is presented in Figure 1. You may notice that the streamed video can be transmitted, through different paths, among different users in different environments, such as a mobile user in wireless network, a video client in DSL network and modem.

Figure 1 General video-streaming architecture



There are a number of basic problems that afflict video streaming over the internet as the internet only offers best-effort service. That is, it provides no guarantees on bandwidth, delay jitter, or loss rate. Moreover, these characteristics are unknown and dynamic. Therefore, a key goal of video streaming is to design a system, which can reliably deliver high-quality video over the internet when dealing with unknown and dynamic bandwidth, delay jitter and loss rate.

The bandwidth available between two points in the internet is generally unknown and time-varying. If the server transmits faster than the available bandwidth, then congestion occurs, some packets are lost, and there is a severe drop in video quality. If the server transmits slower than the available bandwidth, then the receiver produces suboptimal video quality. To overcome the bandwidth dilemma, one has to estimate the available bandwidth and then match the transmitted video bit rate to the available bandwidth. Additional considerations that make the bandwidth problem very challenging include accurately estimating the available bandwidth, matching the pre-encoded video to the estimated channel bandwidth, transmitting at a rate that is fair to other concurrent flows in the internet, and solving this problem in a multicast situation wherein a single sender streams data to multiple receivers while each may have a different available bandwidth.

The end-to-end delay that a packet experiences may propagate from packet to packet. This variation in end-to-end delay is referred to as the delay jitter. Delay jitter is a problem because the receiver must receive, decode and display frames at a constant rate, and any late frames resulting from the delay jitter can produce problems in the reconstructed video. This problem is typically addressed by including a playout buffer at the receiver. While the playout buffer can compensate for the delay jitter, it also introduces an additional delay. The third fundamental obstacle is packet losses. A number of different types of losses may occur, depending on the particular network under consideration. For example, wired packet networks such as the internet are afflicted by

packet loss, where an entire packet is lost. Although wireless channels are typically afflicted by bit errors or burst errors, losses can have a very destructive effect on reconstructed video quality. To overcome the effect of losses, a video-streaming system is designed with error control.

Thus, an efficient and robust video-streaming system is highly desirable. Many of the traditional video-streaming systems, which are trying to overcome the aforementioned limitations and constraints, consider videos as low-level bit streams, ignoring the underlying visual content. Unfortunately, current video applications adapt to fit the available network resources without regard to the video content. To overcome the aforementioned problems and to achieve the efficient robust video streaming, we propose object-based video streaming. Object-based video streaming is a new framework that explores the strong correlation between video content, video data (bit rate) and quality of service. Such a framework facilitates new ways of quality modelling, and resource allocation in video streaming.

We refer video content to the high-level multimedia features so that it can be analysed by the computer. For instance, our target high-level feature is the video object. This feature can be systematically analysed. The video objects can be used for controlling the video generation to facilitate the network-wise scalability. It can be used in selecting the optimal transcoding architecture and content filtering.

The object-based video-streaming framework is based on the recognition of strong correlation among video content, required network resources (bandwidth) and the resulting video quality. Such correlation between the video content and the traffic has been reported in our prior work (Ahmad et al., 2004, 2005; Ahmad and Lee, 2004) in which a conceptual model for content-aware based video-streaming has been proposed. Closer schemes to our approach are joint source-channel coding (Ortega and Khansari, 1995), adaptive media scaling and resilience coding (Ortega et al., 1997; Hur and Hong, 2006; Buchinger and Hlavacs, 2006) and object- and texture-aware video streaming (Ahmad et al., 2004, 2005; Ahmad and Lee, 2004).

The object-based streaming system dynamically detects the objects in the video stream, to utilise the bandwidth efficiently. As important objects can be real-time detected, encoded and transmitted with a higher bandwidth and higher frame rate than those in the background, to achieve the objective of the object-based streaming system, a reliable object-extraction mechanism is needed as a primary step. Therefore, we present a novel and reliable object detection scheme suitable for object-based video streaming. Our approach processes entirely in the compressed domain, thus saving great computation time and providing efficient video streaming. Moreover, we have designed a robust video-streaming mechanism, as we refine the motion vector through a sophisticated filter scheme in order to get fine motion vectors. Thus, our approach offers an accurate and fast object-based video streaming as proved by the conducted experiments.

2 Related work

Our approach is related to three levels of work. The first level is object-based streaming work; the second is the feature extraction process for the object detection; the third is refining the extracted feature for object detection.

On the object-based streaming-related work, researchers (Hsiao et al., 2004) have developed an object-based streaming system to be implemented in the intelligent transportation system. Unfortunately, their approach was fully conducted in pixel domain. Instead, our proposed approach was fully conducted in compressed domain. Therefore, their approach results in a higher computation time when compared with the one in compressed domain. In addition, their scheme is only dedicated for intelligent transportation system. Although they claim that their approach is portable for different platforms, the portability issue was not defined. Our scheme supports general application deployment with easy customisation process. Hence, we have no portability issues. Chang et al. (2001) has proposed a new object-based adaptive streaming approach for sports videos. Their approach was operated in the so-called semi-compressed domain, as they need to partially refer to the pixel domain. Thus, our scheme outperforms their approach as it fully operates in compressed domain. The robustness issue was not considered in their system, as they processed compressed features directly with no post-processing or refinement. Robustness issues were a big concern in our approach and have been fully stated and taken care of.

Concerning the second level of related work, we first will describe the features extraction approaches and justify our choosing for motion vector as the main clue for our approach.

Some sources of information in video can be used to detect objects: visual attributes (such as colour, 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. Motion extraction also provides an additional source of information that can be exploited for the purpose of object extraction by algorithms operating in the pixel domain (Haering et al., 2000) or compressed domain (Eng and Ma, 1999; Favalli et al., 2000). Perform processing in the pixel domain includes the additional burden of attribute extraction, i.e., extracting the pixel information when compared with extracting macroblock information. Performing the object extraction only in the pixel domain could be based on shape (Faloutsos et al., 1994; Pentland et al., 1994), colour (Vinod and Murase, 1997; Fieguth, 1997), or other visual features. Approaches using certain complex analysis in the pixel level are extremely computationally intensive and have other drawbacks when 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 been proven to offer comparable results for uncompressed domain. Most videos are not provided in the form of decoded 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 : 1–50 : 1 compression ratio for MPEG). Compressed video data offers us additional information like DC coefficients and motion vectors.

According to our earlier discussion, 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 (Jones et al., 1999; Khan et al., 2001) utilise these motion vector values directly. Processing digital video directly in the compressed domain has reduced processing time, enhanced storage efficiency, video data processing speed, and video quality. Object extraction directly in compressed video without full-frame decompression

as our proposed approach 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 (Eng and Ma, 1999; Favalli et al., 2000; Jones et al., 1999) as well as to get highly efficient extraction algorithms at the macroblock level.

The third level of related work addresses the extracted feature refining. Apparently, motion vectors are our main clue to detect object but we address the main problem regarding motion vector robustness. Motion vector information is sometimes difficult to use due to the lack of effective representation and due to the fact that it introduces large amount of noise, which makes 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 (Chien et al., 2002) elaborate on the noise in motion vectors due to camera noise and irregular object motion. It is known (Wang et al., 2000) that the motion fields in MPEG streams are quite prone to quantisation errors, especially in low-textured areas. However, typical samples in the motion vector field are usually inaccurate (Ahmad et al., 2003a, 2003b; Pilu, 1997).

These defects can be combated with robust error recovery schemes that repair motion fields and reduce noise Ahmad et al. (2003a, 2003b). Consequently, we can produce a smoother shape 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 a 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. Researcher Babu and Ramakrishnan (2002) used P-, B-frames in their approaches to extract the motion information; therefore, they increase the system complexity and add some processing cost. We only use P-frames to extract the motion vector; still we have accurate and robust results. This will be proven in the result section. One Ahmad et al. (2003a, 2003b) 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 accuracy. Wang et al. (2000) used spatial confident measure, which is Mean-Filter like, where the authors in Ahmad et al. (2003a, 2003b) proved using this metric is insufficient in terms of accuracy and unrealistic for real-time application. In our approach, we deploy adaptive metric based on the texture of video frames. Moreover, Wang et al. (2000) combined both the texture and spatial measure equally, which was proven insufficient in terms of accuracy and unrealistic for the real-time applications. Ahmad et al. (2003a, 2003b) use only spatial filter without regarding the texture measure, which resulted in a less realistic object detection result. Our approach accommodates all these issues by using adaptive scheme based on texture of each frame and spatial distribution by further filtering the motion vector values using pre-defined Gaussian filter cascaded by median filter to fix the gaps occurring due the hard cutting by the Gaussian filter.

3 Object-based video streaming

First, we need to discuss the point of view of video communication protocols. To overcome short-term network condition changes and avoid long-term congestion collapse, various network condition changes control strategies that have been built into the Transmission Control Protocol (TCP). For video traffic, TCP is not video protocol of choice. Unlike traditional data flows, video flows do not necessarily require a completely reliable transport protocol because they can absorb a limited amount of loss without significant reduction in perceptual quality (Claypool and Tanner, 1999). On the other hand (Sakazawa et al., 2005), some researchers try to exploit TCP protocol to deliver video content by setting up many TCP connections at the same time. Thus, they can accomplish video content delivering, but have inefficient network performance. According to our earlier discussion, video flows have fairly strict delay and delay jitter requirements. Video flows generally use the User Datagram Protocol (UDP). This is significant since UDP does not have a network condition changes control mechanism built in; therefore, most video flows are unable to respond to network congestion and adversely affect the performance of the network as a whole.

While proposed multimedia protocols like that of Floyd et al. (2000) respond to congestion by scaling the bit rate, they still require a mechanism at the application layer to semantically map the scaling technique to the bit rate. In times of network condition changes, the random dropping of frames by the router Floyd et al. (2000) and Lin and Morris (1997) may seriously degrade multimedia quality since the encoding mechanisms for multimedia generally bring in numerous dependences between frames (Mitchell and Pennebaker, 1996).

For instance, in MPEG encoding (ISO/IEC, 1995; Information technology, 1993), dropping an independently encoded frame will result in the following dependent frames being presented as useless since they cannot be displayed and would be better off being dropped, rather than occupying unnecessary bandwidth. A multimedia application that is aware of these data dependences can drop the least important frames much more efficiently than can the router (Hemy et al., 1999; Ahmad et al., 2004; Ahmad and Lee, 2004). Such application-specific data rate reduction is classified as content-aware video streaming.

Clearly, object-based video streaming is a combination of the video object extractor, network condition estimator and a scaling mechanism to respond for the network conditions after studying the video content. The estimator part is clearly stated in the literatures of computer networks (Miyabayashi et al., 2000; Rejaie et al., 1999; Bocheck et al., 1999). The video object extractor is proposed herein. It has been shown that the content of the stream can be an important factor in influencing the video-streaming mechanism. Video scaling or transcoding techniques for video to be used in the object-based video-streaming systems can be broadly categorised as follows (ISO/IEC, 1995; Tripathi and Claypool, 2002; Claypool and Tanner, 1999):

- *Spatial scaling.* In spatial scaling, the size of the frames is reduced by transmitting fewer pixels and increasing the pixel size, thereby reducing the level of detail in the frame.

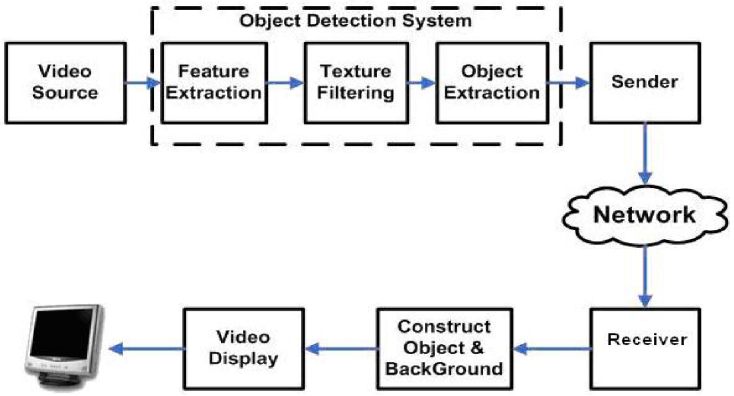
- *Temporal scaling.* In temporal scaling, the application drops frames. The order in which the frames are dropped depends upon the relative importance of the different frame types. In the case of MPEG, the encoding of the I-frames is done independently and they are therefore the most important and are dropped last. The encoding of the P-frames is dependent on the I-frames and the encoding of the B-frames is dependent on both the I-frames and the P-frames, and the B-frames are least important since no frames are encoded based upon the B-frames. Therefore, B-frames are most likely to be the first ones dropped.
- *Quality scaling.* In quality scaling, the quantisation levels are changed, chrominance is dropped or DCT and DWT coefficients are dropped. The resulting frames are of a lower quality and may have fewer colours and details.

In summary, it has been shown that the content of the stream can be an important factor in influencing the choice of the scaling scheme for under processing video (ISO/IEC, 1995; Ahmad et al., 2004, 2005; Ahmad and Lee, 2004; Tripathi and Claypool, 2002).

In sum, the object-based video-streaming technique can be designed as stated in Figure 2. Our proposed object detection is in the block surrounded by dashed lines. Initially, we capture video stream in MPEG format, extract features, detect moving objects and transmit their encoded streams. Figure 2(a) illustrates the streaming system architecture, which covers four key modules, including the Object Extraction, Sender, Receiver and Composer. Figure 2(b) presents a typical RTP sender side. Figure 2(c) present a typical RTP receiver side.

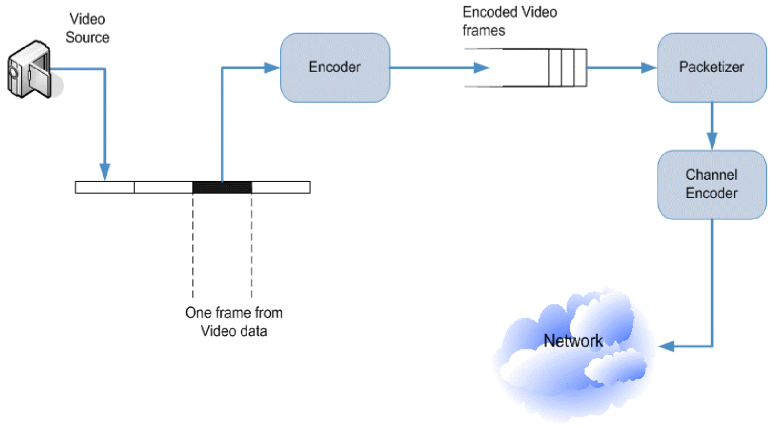
We are going to discuss many different approaches relevant to object-based video streaming. A fine-grained, content-based, packet-forwarding mechanism (Shin et al., 2000) has been developed for differentiated service networks. This mechanism assigns relative priorities to packets based on the characteristics of the macroblocks contained within it. These characteristics include the macroblock encoding type, the associated motion vectors, the total size in bytes and the existence of any picture level headers. Shin et al. (2000) proposed scheme requires some mechanisms for queue management and weighted fair queuing to provide the differentiated forwarding of packets with high priorities and therefore will not work in today's internet.

Figure 2 (a) Streaming system architecture; (b) block diagram of RTP sender and (c) block diagram of RTP receiver

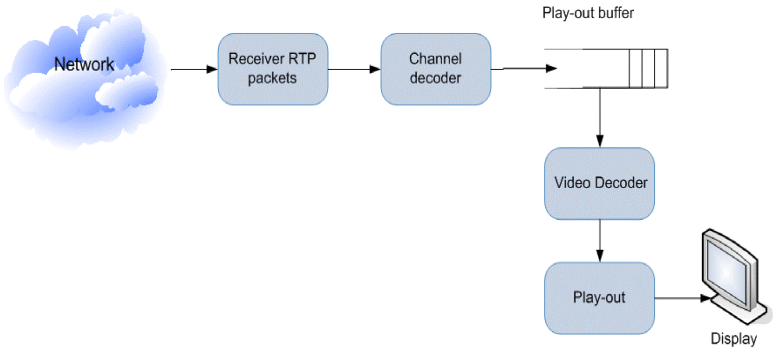


(a)

Figure 2 (a) Streaming system architecture; (b) block diagram of RTP sender and (c) block diagram of RTP receiver (continued)



(b)



(c)

A basic mechanism that uses temporal scaling for MPEG streams is suggested in Chung and Claypool (2000). In case of network condition being changed, the frame rate is reduced by dropping frames in a pre-defined precedence (first B-frames and then P-frames) until the lowest frame rate, where only the I-frames are played out, is reached or the minimum bandwidth requirement matches the availability.

An adaptive MPEG streaming player based on similar techniques was developed (Walpole et al., 1997). These systems have the capabilities for dynamic rate adaptation but do not support real-time, automatic content detection and analysing. Automatic adaptive content-based scaling may significantly improve the perceptual quality of their played out streams. The above mechanisms, while considering the specific characteristics of streaming flows, do not take into account the content of the video flows when scaling bandwidth.

Based on the following phenomena, if a movie shot has quick motion and had to be scaled, then it would look better if all the frames were played out albeit with lower quality, some expert designs their streaming systems. That would imply the use of either quality or spatial scaling mechanisms. On the other hand, if a movie scene has little motion and needed to be scaled, it would look better if a few frames were dropped but the

frames that were shown were of high quality. Such a system has been suggested in Tripathi and Claypool (2002). Relevant approach (Yeadon et al., 1996) has developed a filtering mechanism for video applications capable of scaling video streams. Using these filters, it is possible to change the characteristics of video streams by dropping frames, dropping colours, changing the quantisation levels, etc. Tripathi and Claypool (2002) and Information technology (1995) utilise these filtering mechanisms in conjunction with a real-time content analyser that measures the motion in an MPEG stream to implement a content-aware scaling system. Tripathi and Claypool (2002) conducted a user study where the subjects rate the quality of video clips that are first scaled temporally and then by quality to establish the optimal mechanism for scaling a particular stream. They find that the content-aware system can improve perceptual quality of video by as much as 50%.

Protocol-related solutions have various limitations and capabilities. Various mechanisms have been proposed for video protocols to respond to network condition changes on the internet (Tripathi and Claypool, 2002; Floyd et al., 2000). It is a mechanism for equation-based network condition changes control for unicast traffic. Unlike TCP, it refrains from reducing the sending rate in half in response to a single packet-loss. Therefore, traffic such as best-effort unicast streaming multimedia could find use for this TCP-friendly congestion control mechanism. A TCP-friendly protocol (Miyabayashi et al., 2000) was implemented and evaluated for fairness in bandwidth distribution among the TCP and its flows. Rejaie et al. (1999) have shown a TCP-friendly Rate Adaptation Protocol, which employs an additive increase and a multiplicative decrease scheme. Its primary goal is to be fair and TCP-friendly while separating network congestion control from application level reliability. Object-based video streaming can make the most effective use of available bandwidth from these protocols.

Another approach to media scaling uses a layered source-coding algorithm (McCanne et al., 1997) with a layered transmission system (McCanne et al., 1996). By selectively forwarding subsets of layers at constrained network links, each user may receive the best quality signal that the network can deliver. In the Receiver-driven Layered Multicast scheme suggested, multicast receivers can adapt to the static heterogeneity of link bandwidths and dynamic variations in network capacity. However, this approach may encounter problems with excessive use of bandwidth for the signalling that is needed for hosts to subscribe or unsubscribe from multicast groups and fairness issues in that a host might not receive the best quality possible on account of being in a multicast group with low-end users.

A semi-reliable protocol that uses a TCP congestion window to pace the delivery of data into the network has also been suggested to handle video network condition changes (Jacobs and Eleftheriadis, 1998). However, other TCP algorithms, like retransmissions of dropped packets, etc., that are detrimental to real-time multimedia applications have not been incorporated. This solution is closed to SVFTP (Sakazawa et al., 2005) solution and has the same limitations.

4 Object-extraction system

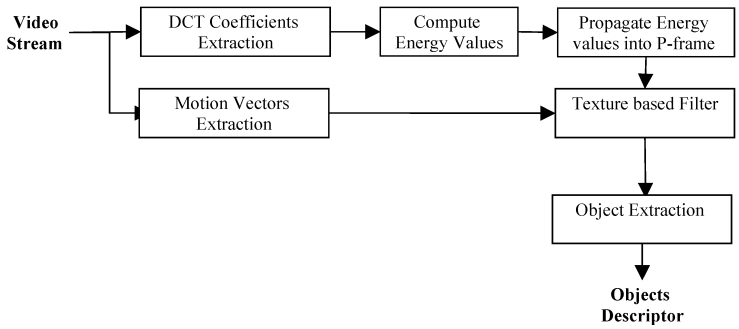
The MPEG compressed video provides one motion vector for each macroblock of size 16×16 pixels, which means that the motion vectors are quantised to 1 vector per 16×16 block. The motion vectors are not the true motion vectors of a particular pixel in the

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, then we will describe its components in detail. In our proposed approach, we first take an MPEG video stream with the [IBBPBBPBBPBBPBB] structure. Figure 3 shows the proposed system architecture. Next, we extract the motion vectors from P-frames only 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. 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.

Figure 3 The object-extraction system overview



4.2 Texture 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 analyse the AC components of the DCT transformation and DCT coefficient, thus staying in the compressed domain.

4.2.1 Texture energy computation

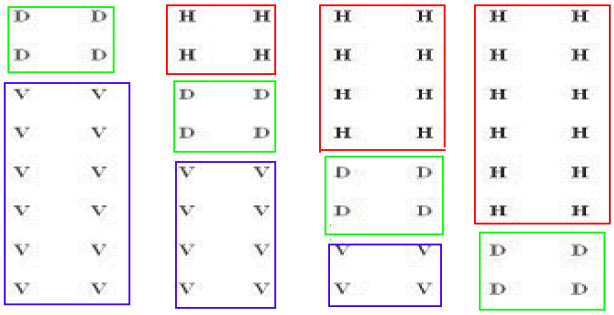
Object regions are distinguished from non-object regions using their distinguishing texture characteristics. Unlike previously published methods, which fully decompress the video sequence before extracting the desired object regions, this method helps in locating the candidate object regions directly 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 designed the Directional Texture Energy Map (DTEM) in DCT domain as shown Figure 4, 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.

Figure 4 Directional Texture Energy Map (DTME) 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 8×8 pixels per block in I-frames. We compute the Horizontal energy E_h by summing up the absolute amplitudes of the horizontal harmonics of the block, which have been marked as H in the DTME. We compute the Vertical energy E_v by summing up the absolute amplitudes of the vertical harmonics of the block, which have been marked as V in the DTME. We compute the Diagonal energy E_d 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 E_a for each macroblock, which is the average value of the Vertical energy, Diagonal energy and Horizontal energy, as in equation (1)

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

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. For every macroblock, we will update motion vector values based on the E_a as described in equation (2).

For every macroblock:

$$\text{Motion\_Vector}_{\text{new}} = \begin{cases} \text{Motion\_Vector}_{\text{old}} \times \dfrac{100 \times E_{\text{a}}}{E_t}\% \ , E_{\text{a}} < E_t \\ \text{Motion\_Vector}_{\text{old}}, E_{\text{a}} \geq E_t . \end{cases} \tag{2}$$

Empirically, we have used an adaptive threshold value, which is 1.5 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 to achieve more robust performance. Motion vectors with magnitude equal to or approaching zero are recognised 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 early works Ahmad et al. (2003a, 2003b). Figures 5 and 6 shows the extraction results of using our system and without using any kind of filtering, respectively. Figure 5 shows the result of object extraction using directly extracted motion vector, Figure 6 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 Figures 5 and 6. The reader can notice how precisely our system is able to detect the objects.

Figure 5 Object extraction using our system

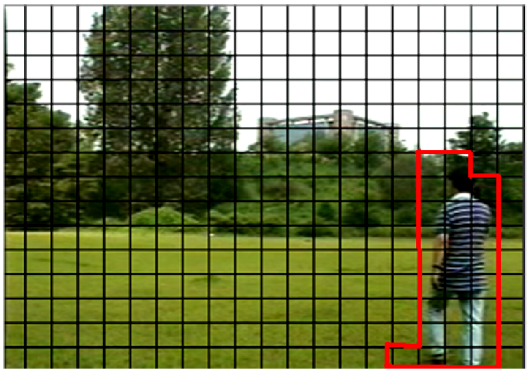
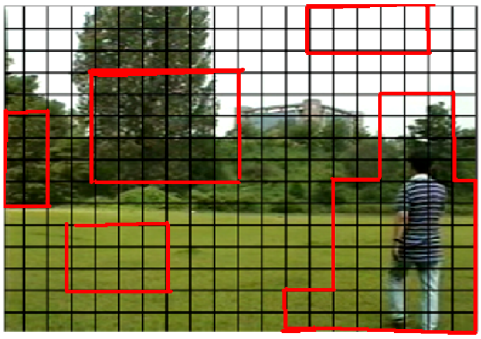


Figure 6 Object extraction without processing



5 RTP sender and receiver

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 of RTP packets, sender 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, which in our case is MPEG format. 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 the 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 scheme used. The sender is responsible for generating periodic status reports for the media streams, which the sender is generating, including those required for lip synchronisation. It also receives reception quality feedback from other participants and may use the information to adapt its transmission.

The 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. The receiver also sends reception quality feedback, allowing the sender to adapt the transmission to the receiver, and receiver 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 inter-packet 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, the frames are decoded (depending on the code used, it may be necessary to decode the media before missing frames can be repaired). At this point, there may be 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, combining several audio sources for playout via a single set of speakers. As is evident from this 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 delay jitter, as the receiver needs to deploy some buffering mechanism to overcome delay jitter.

6 Experimental results and discussion

We have designed an experiment 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 (Zhong et al., 2000), Group **B** using Median filter Ahmad et al. (2003a, 2003b), Group **C** using Cascade filter, and Group **D** our system, finally without any kind of post-processing. To compare the performance among these four system's results, we make the following configurations fix among all the experiments. The frame size is 320×240, which implies that we have 20×15 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 (Chen et al., 2001, 2002; So, 2005). Figures 5–8 shows 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. Figures 7 and 8 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 other schemes. We show the average recall metric and average precision metric for the whole

clip in Figures 9 and 10. 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 owing to the lack of information in the neighbourhood 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 when compared with that without using any kind of post-processing.

Figure 7 Precision for object extraction in P-frames

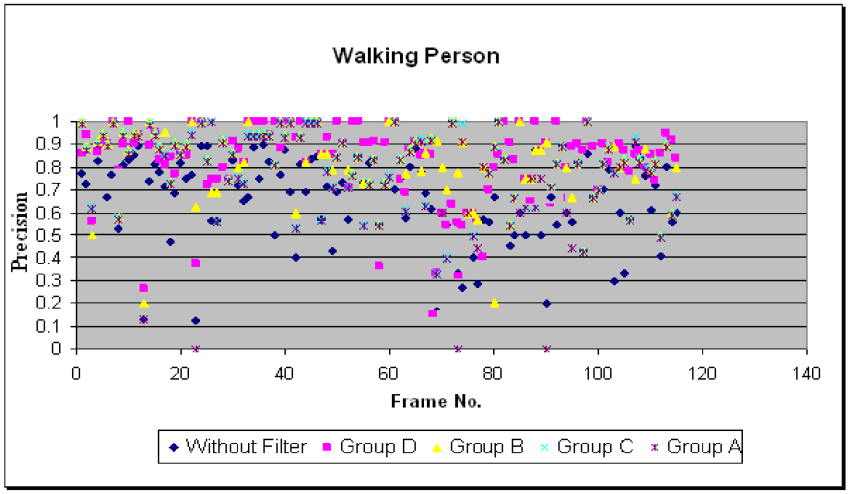


Figure 8 Recall for object extraction in P-frames

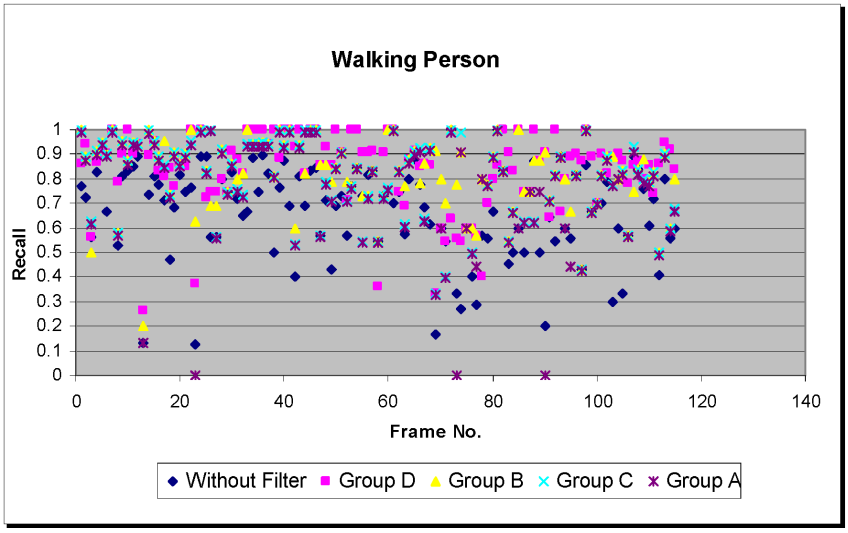


Figure 9 Average precision of object extraction for second video clip

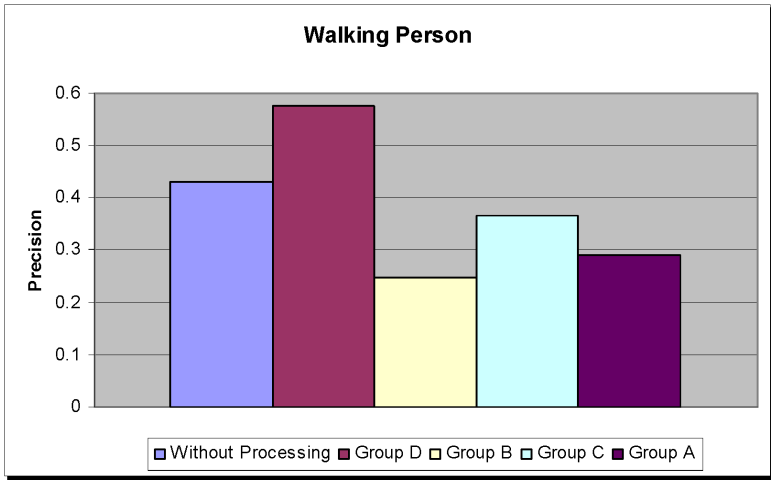
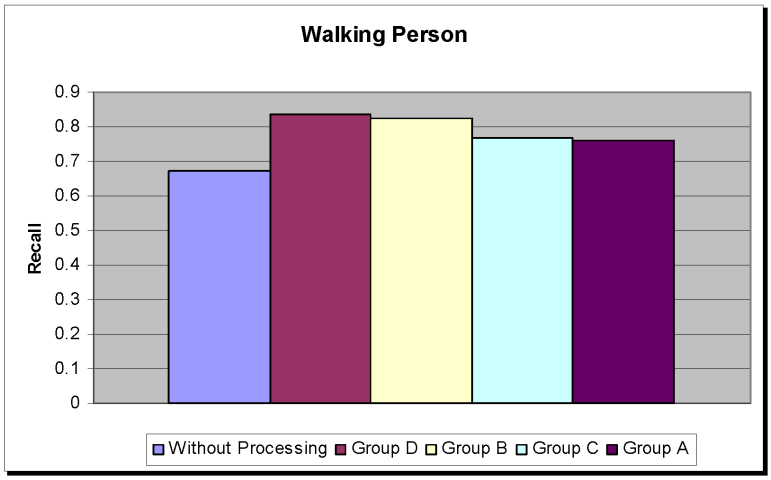


Figure 10 Average recall of object extraction for second video



To verify the above-mentioned fact about run-time efficiency, we measure the performance of the object-extraction performance in terms of run time. We use Pentium4 with 2 Giga Hertz CPU speed, and 30 Giga Byte Hard disk capacity. This result was prepared on the second video clip in our testing dataset. The experimental result will be presented in the following table. In Table 1, we show the run time in milliseconds (ms) for the object-extraction module using our system, other related work, and without any post pre-processing. Also, we described the I/O time needed to perform the object-extraction task. Finally, the total spent time. The reader may notice the run time difference among our system, other related work and without post-processing, where our system's performance is so efficient in terms of run time and I/O and certainly in terms of the total time spent. This result shows compatibility with our expectation and initial idea.

Table 1 Run time for object extraction in millisecond

	Group A	*Group B*	*Group C*	*Group D*	*Without post-processing*
Object-extraction run time	585	579	575	545	605
I/O processing	104	96	89	70	124
Total time	689	675	664	615	729

7 Future work

In this section, we will describe future work for both the deployment of the proposed approach and enhancements that can be added to it. Several future work applications can be anticipated based on our proposed approach.

First of all, Application Level Gateway is a very suitable future work application that can deploy our scheme, as the function of application level gate is to respond the application requirements while it is transmitting video throughout the network. Therefore, our scheme provides very high quality of service in object level in response to system requirements and network conditions.

Second, mobile video communication is a very promising and hot research topic in academia and industry. To allow very smooth and attractive mobile video communication, streaming video in mobile network with high level features is crucial; therefore, adapting our scheme could result in very promising outcomes, further investigation need to be taken.

Third, online video broadcasting such as game broadcasting, news broadcasting and so forth, have cretin requirements, real-time processing and display. In addition, these applications offer some presumptions about the video contents such as in game broadcasting context, we have court shape, players' number, games rules, and so on. Using our scheme, these applications can make use of the presumed environment parameters and map them as detected objects. For example, game players can be decided as moving objects and court areas as static scene along the streaming presentation time. Knowing it would provide good results in the real-time processing, our scheme is expected to fit the requirements of these applications.

Fourth application is video conferencing. As video conferencing has several known presumptions, such that we have background area and people talking in the foreground area. This may relax the procedure and make use of our approach by defining talking heads as moving objects and separating it from the background area. Also our approach will fit for the real-time requirement, which is imposed by video conferencing. Also e-learning and web distance education may take advantage of our scheme in the same manner as video conferencing will do. But this time, lecturer or the expressions of the lecturer will be assumed as a moving object.

On enhancing the proposed scheme future work, we still think that there are issues, on the quality of service in RTP stack level, needed to be further addressed. We anticipate that further quality of service can be attained by co-operating the object detection results with the RTP stack definition.

In addition, the motion vector sets we are processing in the current status are non-uniform as they belong to different frame types of P- and B-frames. Hence, for future work on this issue, we believe that by informing the motion vector values in one set, no matter what frame type they may belong to, will result in a more efficient performance.

8 Conclusions

In this paper, we present a novel object-extraction scheme for the object-based video-streaming system. Object-based video streaming overcomes the significant technical challenges in quality of service guarantee and efficient resource management for video streaming. We conclude that end-to-end quality requirements of video-streaming application can be reasonably achieved only by integrative study of advanced networking and content processing techniques.

However, most existing integration techniques stop at the bit stream level, ignoring a deeper understanding of the media content. Yet, the underlying visual content of the video stream contains a vast amount of information that can be used to stream video in a semantic manner. We explore different approaches that are thought to be content-aware video streaming. Some approaches were more network-centric like approaches to solving the problems of unresponsiveness in video flows. Others were classified as either protocol-related solution or based on its features, i.e., solution used object to do the scaling was classified as object-based video-streaming system. We believe that object-based video streaming is a very promising field for both video and communication societies. Still it has a lot of room for investigation and developing.

Acknowledgement

The authors express their heartfelt thanks to Mr. Ahmad Ma for his proofreading of this work.

References

Ahmad, A.M.A. and Lee, S-Y. (2004) 'Novel object-based video streaming technique', *2nd International Conference on Computing, Communications and Control Technologies*, Texas, USA, pp.255–300.

Ahmad, A.M.A., Chen, D-Y. and Lee, S.Y. (2003a) 'Robust object detection using cascade filter in MPEG videos', *IEEE Fifth International Symposium on Multimedia Software Engineering*, Taichung, Taiwan, 10–12 December, pp.78–84.

Ahmad, A.M.A., Chen, D-Y. and Lee, S-Y. (2003b) 'Robust compressed domain object detection in MPEG videos', *Proc. the 7th IASTED International Conference Internet and Multimedia System Applications*, Hawaii, USA, pp.706–712.

Ahmad, A.M.A., Ahmad, B.M.A., Talat, T.S. and Lee, S.Y. (2004) 'Fast and robust object detection framework for object based streaming system', *Proceeding The Second International Conference on Advances in Mobile Multimedia*, Kuala, Malaysia, pp.77–86.

Ahmad, A.M.A., Ahmad, B.M.A. and Talat, S. (2005) 'A novel approach for improving the quality of service for mobile video transcoding', *Proceeding The Third International Conference on Advances in Mobile Multimedia*, Bali, Indonesia, pp.255–266.

Babu, R.V. and Ramakrishnan, K.R. (2002) 'Compressed domain motion segmentation for video object extraction', *Proc. ICASSP*, Orlando, Florida, pp.3788–3791.

Bocheck, P., Campbell, A., Chang, S-F. and Lio, R. (1999) 'Utility-based network adaptation for MPEG-4 systems', *Proceedings of Ninth International Workshop on Network and Operating System Support for Digital Audio and Video*, Basking Ridge, New Jersey, USA, pp.279–288.

Buchinger, S. and Hlavacs, H. (2006) 'Subjective quality of mobile MPEG-4 videos with different frame rates', *Journal of Mobile Multimedia*, Vol. 1, No. 4, January, pp.327–341.

Chandra, S. and Ellis, C. (1999) 'JPEG compression metric as a quality aware image transcoding', *Proceedings of Second Usenix Symposium on Internet Technologies and Systems*, Boulder, Colorado, USA, Vol. 2, pp.81–92.

Chang, S-Fu., Zhong, D. and Kumar, R. (2001) *Real-Time Content-Based Adaptive Streaming of Sports Video*, July, ADVENT Technical Report #121 Columbia University, IEEE Computer Society, Washington DC.

Chen, D-Y., Lee, S-Y. and Chen, H-T. (2002) 'Motion activity based semantic video similarity retrieval', *Proc. IEEE PCM*, Hsinchu, Taiwan, pp.319–327.

Chen, S-C., Shyu, M-L., Zhang, C. and Kashyap, R.L. (2001) 'video scene change detection method using unsupervised segmentation and object tracking', *Proc. ICME*, Tokyo, Japan, pp.57–60.

Chien, S., Ma, S. and Chen, L. (2002) 'Efficient moving object segmentation algorithm using background registration technique', *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 12, No. 7, pp.577–586.

Chung, J. and Claypool, M. (2000) 'Better-behaved, better-performing multimedia networking', *Proceedings of Euromedia Conference*, Belgium, pp.245–252.

Claypool, M. and Tanner, J. (1999) 'The effects of jitter on the perceptual quality of video', *Proceedings of ACM Multimedia Conference*, Orlando, Florida, USA, pp.115–118.

Eng, H.L. and Ma, K.K. (1999) 'Bidirectional motion tracking for video indexing', *Proc. Third IEEE Workshop on Multimedia Signal Processing*, Copenhagen, Denmark, pp.153–158.

Faloutsos, C., Barber, R., Flickner, M., Hafner, J., Niblack, W., Petkovic, D. and Equitz, W. (1994) 'Efficient and effective querying by image content', *Journal Intelligent Information Systems*, Vol. 3, No. 1, pp.231–262.

Favalli, L., Mecocci, A. and Moschetti, F. (2000) 'Object tracking for retrieval applications in MPEG-2', *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 10, No. 3, pp.427–432.

Fieguth, P. (1997) 'Color-based tracking of heads and other mobile objects at video frame rates', *Proc. CVPR*, San Juan, Puerto Rico, pp.21–27.

Floyd, M., Handley, J. Padhye, J. and Widmer, J. (2000) 'Equation-based congestion control for unicast applications', *Proceedings of ACM the Special Interest Group on Data Communication Conference*, Stockholm, Sweden, pp.43–56.

Haering, N., Qian, R.J. and Sezan, M.I. (2000) '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, pp.857–868.

Hemy, M., Hangartner, U., Steenkiste, P. and Gross, T. (1999) 'MPEG system streams in best-effort networks', *Proceedings of Packet Video Workshop*, New York, USA, pp.84–96.

Hsiao, M-Ho., Kuo, H-P., Wu, H-C., Chen, Y-K. and Lee, S-Y. (2004) 'Objected-based video streaming technique with application to intelligent transportation systems', *Proceedings of IEEE International Conference on Networking, Sensing and Control*, Taipei, Taiwan, Vol. 1, pp.315–320.

Hur, H-S. and Hong, Y-S. (2006) 'An improvement of multimedia data downstream with PDA in an infrastructure network', *Journal of Mobile Multimedia*, Vol. 2, No. 1, March, pp.81–96.

Information Technology (1993) *Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to About 1.5 Mbit/S*-Part II:Video, ISO/IEC 11 172-2.

Information Technology (1995) *Generic Coding of Moving Pictures and Associated Audio Information*, International Standard ISO/DIS 13818, ISO.

Jacobs, S. and Eleftheriadis, A. (1998) 'Streaming video using dynamic rate shaping and TCP congestion control', *Journal of Visual Communication and Image Representation, Special Issue on Image Technology for World Wide Web Applications*, Vol. 9, No. 3, pp.211–222.

Jones, R.C., DeMenthon, D. and Doermann, D.S. (1999) 'Building mosaics from video using MPEG motion vectors', *Proc. ACM Multimedia Conference*, Florida, USA, pp.29–32.

Khan, J.I., Guo, Z. and Oh, W. (2001) 'Motion based object tracking in MPEG-2 stream for perceptual region discriminating rate transcoding', *Proc. ACM Multimedia Conference*, Ottawa, Canada, pp.572–576.

Lin, D. and Morris, R. (1997) 'Dynamics of random early detection, *Proceedings of ACM the Special Interest Group on Data Communication Conference*, Cannes, France, pp.127–137.

McCanne, V., Jacobsen, V. and Vetterli, M. (1996) 'Receiver-driven layered multicast', *Proceedings of ACM the Special Interest Group on Data Communication Conference Conference*, California, USA, pp.117–130.

McCanne, S., Vetterli, M. and Jacobson, V. (1997) 'Low-complexity video coding for receiver-driven layered multicast', *IEEE Journal on Selected Areas in Communications*, Vol. 16, No. 6, pp.983–1001.

Mitchell, J. and Pennebaker, W.(1996) *MPEG Video: Compression Standard*, Chapman and Hall, ISBN 0412087715, London, UK.

Miyabayashi, M., Wakamiya, N., Murata, M. and Miyahara, H. (2000) 'Implementation of video transfer with TCP-friendly rate control protocol', *Proceedings of International Technical Conference on Circuits/Systems, Computers and Communications*, Pusan, Korea, pp.117–120.

Ortega, A. and Khansari, M. (1995) 'Rate control for video coding over variable bit rate channels with applications to wireless transmission', *Proceedings IEEE International Conference on Image Processing*, Washington DC, USA, pp.3388–3393.

Ortega, A., Carignano, F., Ayer, S. and Vetterli, M. (1997) 'Soft caching: web cache management techniques for images', *Proceeding IEEE Signal Processing Society Workshop on Multimedia Signal Processing*, NJ, USA, pp.475–480.

Pentland, A., Picard, R. and Sclaroff, S. (1994) 'Tools for content-based manipulation of image databases storage and retrieval of image and video databases ii', *Proc. SPIE*, Orlando, Florida, USA, pp.34–47.

Pilu, M. (1997) *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).

Rejaie, R., Handley, M. and Estrin, D. (1999) 'RAP: an end-to-end rate-based congestion control mechanism for realtime streams in the internet', *Proceedings of IEEE Infocom*, New York, USA, pp.1337–1345.

Sakazawa, S., Takishima, Y., Kitatsuji, Y., Nakajima, Y., Wada, M. and Hashimoto, K. (2005) 'Video data transmission protocol 'SVFTP' using multiple TCP connections and its application', *IEICE Transaction on Information and Systems*, Japan, pp.180–188.

Shin, J., Kim, J. and Kuo, C.J. (2000) 'Content-based video forwarding mechanism in differentiated service networks', *Proceedings of International Packet Video Workshop*, Sardinia, Italy, pp.340–355.

So, S. (2005) 'Efficient image indexing and retrieval over mobile devices', *Journal of Mobile Multimedia*, Vol. 1, No. 2, pp.149–160.

Tripathi, A. and Claypool, M. (2002) 'Improving multimedia streaming with content-aware video scaling', *Proceedings of the Second International Workshop on Intelligent Multimedia Computing and Networking*, North Carolina, USA, pp.1021–1024.

Vinod, V.V. and Murase, H. (1997) 'Video shot analysis using efficient multiple object tracking', *Proc. ICMCS*, Ontario, Canada, pp.501–508.

Walpole, J., Koster, R., Cen, S., Cowan, C., Maier, D., McNamee, D., Pu, C., Steere, D. and Yu, L. (1997) 'A player for adaptive mpeg video streaming over the internet', *Proceedings of 26th Applied Imagery Pattern Recognition Workshop*, Washington DC, USA, pp.249–258.

Wang, R., Zhang, H-J. and Zhang, Y-Q. (2000) 'A confidence measure based moving object extraction system built for compressed domain', *Proc. ISCAS*, Geneva, Switzerland, pp.21–24.

Yeadon, N., Garcia, F. and Hutchinson, D. (1996) 'Filters: QoS support mechanisms for multipeer communications', *IEEE Journal on Selected Areas in Communications*, Vol. 4, No. 7, pp.1245–1262.

Zhong, Y., Zhang, H. and Jain, A.K. (2000) 'Automatic caption localization in compressed video', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 4, pp.385–392.