# 國立交通大學

## 資訊工程系


## 博 士 論 文

# 移動向量精煉新方法運用於視訊處理和視訊轉換編碼的研究

# Study on Motion Vector Refinement in Video Processing and Transcoding Applications


研究生：艾斯拉夫

指導教授：李素瑛 教授



中華民國九十五年六月

# 移動向量精煉新方法運用於視訊處理和視訊轉換編碼的研究

# Study on Motion Vector Refinement in Video Processing and Transcoding Applications

研究生：艾斯拉夫　　Student：Ashraf M. A. Ahmad

指導教授：李素瑛 教授　　Advisor：Dr. Suh-Yin Lee

國 立 交 通 大 學

資 訊 工 程 學 系

博 士 論 文

A Dissertation
Submitted to
Department of Computer Science and Engineering
College of Electrical Engineering and Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in
Computer Science and Information Engineering
June 2006
Hsinchu, Taiwan, Republic of China

中華民國九十五年六月

# 移動向量精煉新方法運用於視訊處理和視訊轉換編碼的研究

研究生:　艾斯拉夫　　指導教授:李素瑛 教授

國立交通大學資訊工程學研究所

## 摘要

在這篇論文中，我們提出了修正視訊串流中移動向量特徵的新方法，並且實作幾個以移動向量為基礎的應用。我們將討論與驗證在 MPEG 視訊串流中的物件偵測與擷取，以物件為基礎的視訊串流，以及在有限資源環境中的視訊轉換編碼。我們對直接由壓縮域中的視訊串流擷取出的位移向量做處理，以降低移動向量的雜訊並獲得更多的物件資訊。透過我們所提出的系統，包含空間資訊過濾元件、時間資訊過濾元件以及紋路過濾元件，可以使這些物件資訊變得更精確。此外我們針對特定的應用提出了適應性移動向量修正的方法。因此，我們提出的視訊處理與通訊演算法可以準確地取得使用者想要的結果，在執行時間上也更有效率。我們也針對系統效能與其他常用的相關研究及技術作比較。

我們針對 MPEG7 測試資料與額外的視訊測試序列(超過 200 小時)來進行實驗，並用標準的指標評估系統效能，實驗結果顯示我們提出的系統效能明顯的優於其他技術。我們也介紹了新的效能評估指標來正確地追蹤我們的系統效率。

除此之外，本篇論文中也描述了我們發展的使用者介面，讓使用者可以維護與監控演算法的執行過程。

# Study on Motion Vector Refinement in Video Processing and Transcoding Applications

**Student: Ashraf M. A. Ahmad** **Advisor: Prof. Suh-Yin Lee**

**Department of Computer Science and Information Engineering**

**National Chiao Tung University**

## Abstract

In this thesis we propose novel approaches for refining motion vector features in video streams. Several motion vector based applications have been proposed and implemented. Object detection and extraction in MPEG video streams, object based video steaming, and video transcoding with resource limited environment are all discussed, designated and verified. Rather than processing the extracted motion vector fields directly extracted from video streams in the compressed domain, we perform several operations over the extracted motion vectors, in order to reduce the noise within the motion vector content and to obtain more robust object information. The information is refined through our proposed system which is composed of a *spatial filter component*, *temporal filter component* and *texture filter component*. In addition, an adaptive motion vector refinement has been proposed for specific applications. As a result, our proposed

video processing schemes are more capable of accurately gaining desired results with more efficient performance in terms of runtime. We compare the performance of our proposed system with other popular and commonly related work and techniques.

Based on the experimental results performed over the MPEG7 testing dataset and additional video testing sequences –more than 200 hours of videos-, the performance measured by standard performance metrics using our proposed system is superior to the alternative techniques. Moreover, we introduce new performance metrics to trace the efficiency of the proposed systems accurately.

A user system interfaces is also presented, where users can maintain and monitor the process of the proposed algorithms.

# Acknowledgment

I am very thankful for Allah, who has been helping me all the time since I was born till I die. Allah has been with me all the time. Allah helps me whenever I ask his help and assistance. Allah always puts the nice people in my way. And, he gives me some tests; I hope I did pass them. I refer all reasons behind my success for Allah.

I do appreciate the kind guidance of my advisor, Prof. Lee Suh-Yin. In virtue pf her fruitful suggestions and leads, I accomplish this work. In addition, thanks are extended to my friends and lab mates. I am also grateful to my great younger brother “de de” Bashar for his endless support and help. Bashar is very great brother, friend and partner. May Allah help him to get his PhD. degree so soon. Finally, most important, I wish to express my heartfelt thanks and grateful feeling to my parents for their cares and invaluable support. I wish them all the best and happiest life. This dissertation is dedicated to them.

# Table of Content

# List of Tables

# Table of Figures

# Chapter 1

# Introduction

Digital imaging and video streams are becoming prevalent. It is essential to have effective methods and paradigms to search, filter and retrieve visual contents. As the proliferation of compressed video sequences in MPEG formats continues, the ability to perform video analysis directly in the compressed domain becomes increasingly attractive.

To achieve the goal of identifying and selecting desired information, reliable semantic features need to be extracted and deployed as a primary step. Although it has been studied for many years, reliable video features extraction remains an open research problem. A robust, accurate and high performance approach is still a great challenge today.

With significant increases in desktop computer performance and storage comes the development of various multimedia compression standards. And the widespread exchange of multimedia information is becoming a reality. Video is the most popular means of communication and entertainment. With this popularity comes an increase in the volume of video data. Therefore, we need the ability to sift through and search for relevant material stored in large video databases automatically.

A first consideration therefore is an attempt to increase speed when using existing compression standards. Performing analysis in the compressed domain reduces the amount of effort involved in decompression.

## 1.1 Processing in Compressed Domain

In general, there are two sources of information in video signals: visual features (such

as color, texture and shape) and motion information (such as motion vector). Motion vectors provide a source of information for those who are using spatial-temporal analysis in uncompressed [1] or compressed domain [2,3].The visual features in the pixel domain could be based on shape [4, 5] or color [6, 7] or others.

A conventional solution to the problem of processing video streams, shown in the Fig. 1-1, involves either compressed domain or pixel domain processing. Approaches performed on raw images or decoded/re-constructed images at the pixel level are extremely computationally intensive and have other drawbacks compared to staying in the compressed domain.

Video compression algorithms are being used to compress digital video for a wide variety of applications, including video delivery over the Internet, advanced television broadcasting, video streaming, video conferencing, and video storage and editing. The end-to–end compressed digital video systems motivate the need to develop efficient algorithms for handling compressed digital video.

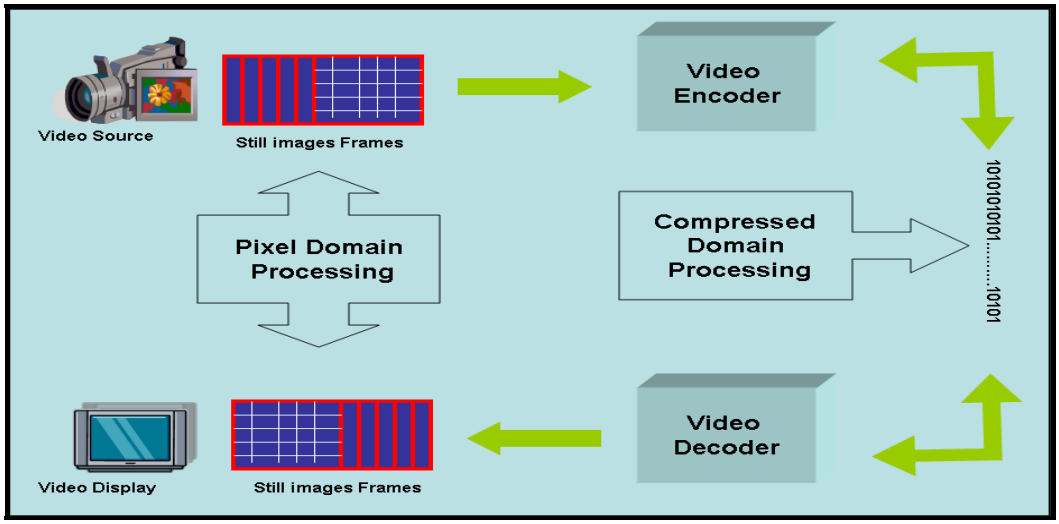



Fig. 1-1. The pixel domain processing steps

Algorithms are needed to adapt compressed video streams for playback on different devices and for robust delivery over different types of networks. Algorithms are needed

for performing video processing and editing operations, including VCR functionalities, on compressed video streams. These algorithms are applicable to a number of predominant image and video coding standards including JPEG, MPEG-1,MPEG-2, MPEG-4, H.261, H.263, and H.264/MPEG-4 AVC.

The analysis of compressed video can proceed in one of two fundamental ways. The first way is by decompressing some or all of the video and using the individual frames to gather information about the video content. The second way involves exploiting encoded information contained in the compressed representation without incurring the overhead of decompression. In this thesis, most of the proposed techniques are conducted in the second approach. Although processing in pixel domain can generally get accurate result, but the work, in compressed domain, has the following advantages:

1. Most videos are as compressed formats.
2. Implementation of the manipulation algorithms in the compressed domain will be much cheaper than that in the uncompressed domain because the data rate is highly reduced in the compressed domain (e.g., a typical 20:1 to 50:1 compression ratio for MPEG).
3. Full decoding and re-encoding of video are not necessary. Thus, we can avoid the extra quality degradation that usually occurs in the re-encoding process.
4. Compressed video data offer us additional information like DC coefficients and motion vectors for various applications.

## 1.2 Motion Vector based Video Processing and Applications

Motion vector based applications addressed in this thesis, are object detection, video streaming and video transcoding. In addition, motion vector feature have been deployed

and exploited in many video processing and communication applications, as depicted in Fig. 1-2.

First video processing using motion vector is video object detection. Video object detection is a primary step for several video processing applications. Video object detection has two forms, either stand alone application or intermediate level form. Stand alone video object extraction applications include:

1. Video surveillance
2. Vision-based control
3. Human-computer interfaces
4. Medical imaging
5. Robotics

Intermediate layer object extraction applications include:

1. High level content based video retrieval.
2. Video event Detection.
3. Video Summarization and abstraction.
4. Video tracking.

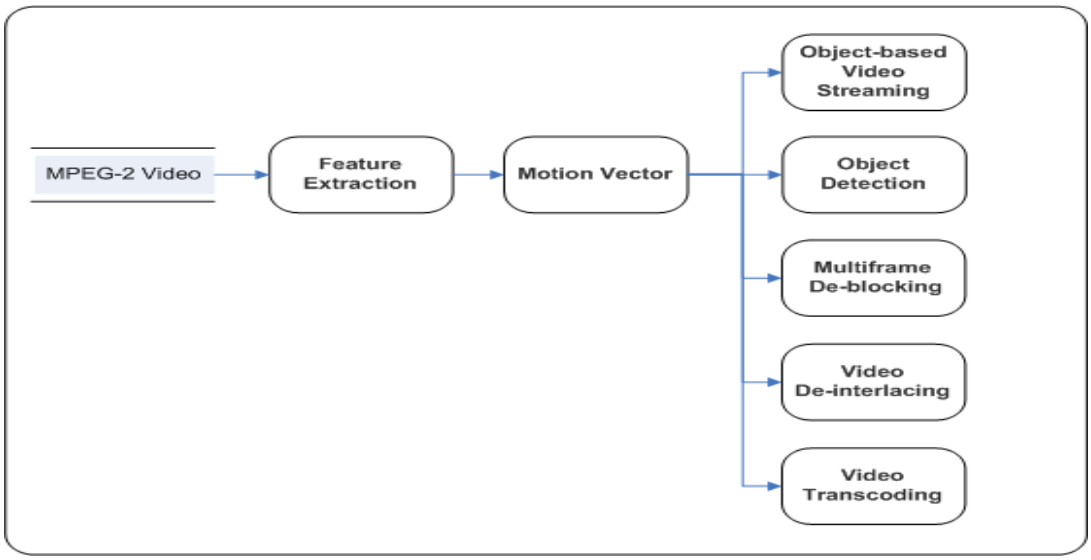



Fig. 1-2. Video processing and communication applications using motion vectors

Although, much work in pixel domain [8, 9, 10, 11], deploys the visual features and motion information, very little work has been carried out in the area of compressed domain video object extraction. Motion detection in pixel domain is performed based on the motion information at each pixel location like optical flow estimation [12], which is computationally very demanding.

In many cases, especially in the case of well-textured objects, the motion vector values reflect the movement of objects in the scene very well. In this thesis, object detection work has been carried out based on motion vectors to keep the processing in the compressed domain.

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 in the start of delivery and the beginning of playback at the client. Therefore, deploying information available in the compressed domain will make the real-time video streaming task possible and efficient. In this thesis, an efficient motion vector based approach has been investigated on video streaming.

Video transcoding application is also one of the most recent topics in video processing. It touches several newly established technologies such as 3G mobile network and Mobile TV. Mobile access to multimedia contents requires video transcoding functionality at the edge of the mobile network for interworking with heterogeneous networks and services.

Under certain conditions, the bandwidth of a coded video stream needs to be drastically reduced. Converting a previously compressed video bit stream to a lower bit rate through transcoding can provide finer and more dynamic adjustments of the bit rate of the coded

video bit stream to meet various channel situations [18]–[24]. Depending on the particular strategy adopted, the transcoder attempts to satisfy network conditions or user requirements in various ways. Since the delivery system must accommodate various transmission and load constraints, it is sometimes necessary to further convert the already compressed bitstream before transmission. This thesis addresses all the aspects of video transcoding that deals with motion vectors.

In sum, motion vector features provided by video in compressed domain is an important cue for analysis and communication applications. Thus, the need becomes clear for reliable and accurate motion vector information for those approaches deploying the motion information [2,3,13-27].

However, it is sometimes hard to use motion vectors due to the noise which makes further processing of the data almost impossible. Besides, motion vectors are still far from ideal, as the key motion estimation is carried out using coarse area-correlation method that has proven inefficient in terms of accuracy. Some researchers [28] elaborate on the noise in motion vector due to the camera noise and irregular object motion. It is realized [32] that the motion fields in MPEG streams are quite prone to quantization errors, and at the encoding steps might be not correctly matched in low-textured area. However, typical samples in the motion vector field are usually inaccurate [33,34].

These defects can be combated with robust error recovery schemes that repair motion fields and eliminate the noise. Consequently, we can produce reliable motion vector features to be used in video processing and communication applications.

Therefore, in this thesis, we introduce techniques that can overcome those defects and produce more reliable motion vectors. These techniques perform processing on the extracted motion vector fields from motion frames. Specific refinement techniques

remove noise and smooth motion vectors. It makes the resulting filtered data more representative of the original motion vectors and more reliable for use in further compressed domain video processing and communication applications.

## 1.3 Organization of the thesis

The rest of the thesis is organized as follows. Chapter 2 presents the background and related work. Chapter 3 analyzes the motion vectors and their limitations. Chapter 4 describes the proposed motion vector schemes. Chapter 5 presents the proposed object detection approach. Chapter 6 describes object based video streaming technique. Chapter 7 discusses and provides video transcoding mechanism. Chapter 8 presents the experimental results. Chapter 9 draws the conclusion and suggests the future directions.

# Chapter 2

# Background and Related Work

Since we want to make use of motion vectors and DCT coefficients, we have to understand the MPEG video compression standard. In addition, good explanation for the motion vector production should be stated.

## 2.1 Background information for MPEG and motion vectors

With recent improvements in processing and storage technologies, many personal computing systems have the capacity to receive, process, and render multimedia objects (e.g., audio, graphical and video content). While MPEG-1 and MPEG-2 can significantly reduce the number of bits needed to represent a video sequence without appreciable degradation of image quality, the compressed format does not lend itself to easy video processing.

The MPEG coding includes information in the bit stream to provide synchronization of audio and video signals, initial and continuous management of coded data buffers to prevent overflow and underflow, random access start-up, and absolute time identification. The coding layer specifies a multiplex data format that allows multiplexing of multiple simultaneous audio and video streams as well as privately defined data streams.

The basic scheme of MPEG is to predict motion from frame to frame in the temporal direction and then to use Discrete Cosine Transform (DCT) coefficients to organize the redundancy in the spatial directions.

MPEG uses two inter-frame coding techniques: predictive and interpolative. This results in three basic picture types in an MPEG stream: I-, P- and B-pictures. An I-picture is completely intra-coded. It provides access points for random access but only

with moderate compression. A P-picture is predictably coded with reference to a past picture, which can be either an I- or a P-picture. A P-picture will in general be used as a reference for future prediction. A B-picture is bi-directionally coded. It is similar to a P-picture, but requires both a past and a future reference picture for prediction. B-pictures provide the highest amount of compression. The relation between the three picture types is illustrated in Fig. 2-1.

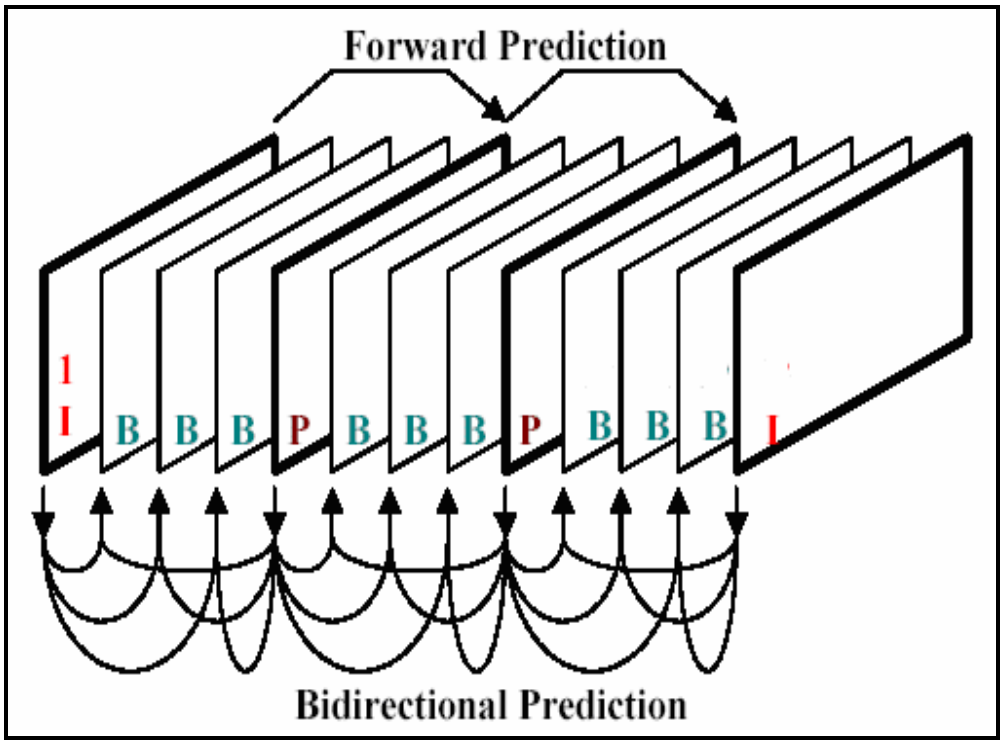



Fig. 2-1: A Group Of Pictures (GOP) structure

GOP is based on a random access requirement. Typically, a starting point is needed at least once in every 0.4 seconds. Providing an I frame every twelve frames correlates to starting with an I frame in every 0.4 seconds. To decode the bit stream, the I frame is first decoded followed by the first P frame. The two B frames in between the I frame and the P frame are then decoded. The primary purpose of the B frames is to reduce noise in the video by filling in or interpolating, between the I and the P frames, typically over a 33 or 25 millisecond picture period without contributing to the overall signal quality.

The B frames and P frames contain the motion information. The I frame has no motion values and stores the DCT information of the original frame of the video sequence. An interesting aspect of the MPEG-4 standard, necessary to support such interactive features, is that a video frame is actually defined as a number of video objects, each assigned to their own video object plane (VOP). When the compressed multimedia object is accessed for use, it is decompressed in accordance with the compression scheme used to compress the multimedia object, e.g., using the Inverse Discrete Cosine Transform (IDCT). Once decompressed further analysis of the multimedia objects may then be performed.

The difficult challenge in the design of the MPEG compression algorithm is the following. On one hand the quality requirements demand a very high compression ratio not achievable with intra-frame coding alone. On the other hand, the random access requirement is best satisfied with pure intra-frame coding. Inter-frame coding can achieve high compression while it does not promise random access. This requires a delicate balance between intra-frame and inter-frame coding. Fig. 2-2 shows the hierarchical structure of an MPEG video stream, which is represented by the following six layers:

**Video Sequence**: A sequence is the top level of the MPEG video coding. It is composed of several groups of pictures (GoPs) used as a random context access unit. For video analysis, information, such as frame rate, picture width and height, aspect ratio, and video bit rate, can be obtained from the video sequence layer.

**Group of Picture (GoP)**: A GoP consists of a series of pictures and is used as a random access unit for video coding. Information, such as the number of pictures in the GoP, is available from the GoP layer. A picture is the primary coding unit and consists of several

slices. Information, such as picture type (I, P and B) and motion vector resolution, is available for video analysis from this layer. A slice is used as a re-synchronization unit and consists of several macroblocks.

**Macroblock**: A macroblock contains a 16×16 pixel region of luminance component and the spatially corresponding 8 × 8 pixel region of each chrominance component since chrominance components are sampled at half the luminance resolution. It thus has four 8×8 luminance blocks and two 8 × 8 chrominance blocks. For video analysis, the macroblock layer provides the type of coding (intra versus non-intra) and the motion vector.

**Block**: A block is 8×8 pixels in size and is the unit of subsequent Discrete Cosine Transform (DCT). It provides 64 DCT coefficients (either of original pixel values or of the residues after the motion compensation).

MPEG uses a component color representation for each color pixel, namely one luminance (Y) and two chrominance components (Cb and Cr). The conversion, from YCbCr to conventional RGB space, can be carried out by a linear mapping (a 3×3 matrix). Since the human visual system (HVS) is most sensitive to luminance component, the Y values are encoded at the full resolution. The HVS is less sensitive to the chrominance information. As a result, the two chrominance components are encoded at half the resolution of their luminance counterpart. This considerably reduces the amount of information to be compressed.

### 2.1.1 Motion Compensation

Motion-compensated prediction assumes the current picture can be modeled as a translation of the picture at some previous time. The local unit used in MPEG is macroblock. This is the result of a trade-off between the coding gain provided by the

motion information and the cost associated with coding the motion information. Each macroblock in a P-picture is matched to the most similar group of 16 × 16 pixels in its reference picture. This process is called motion estimation.

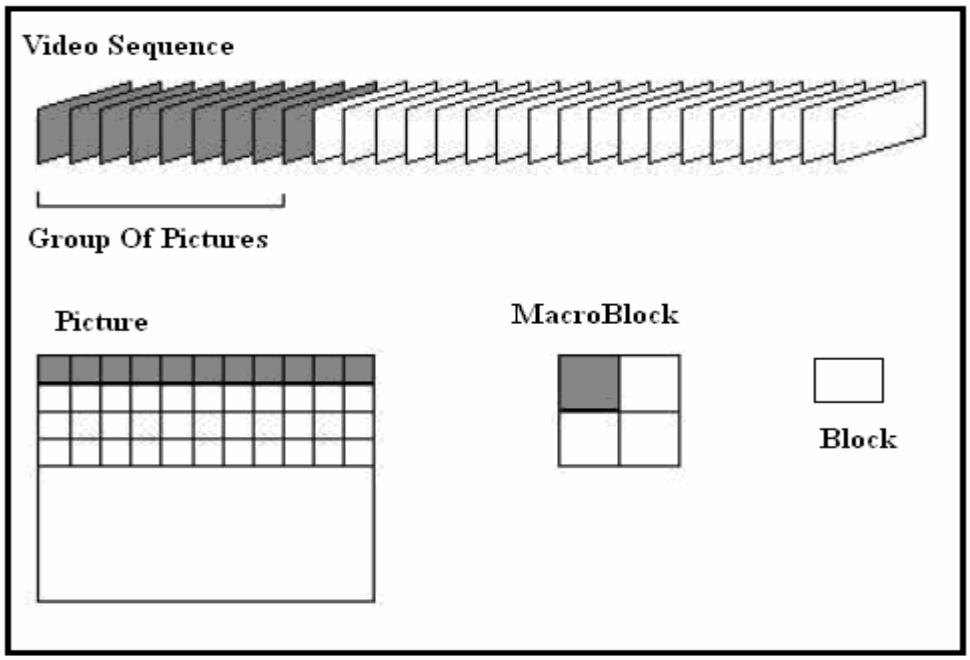



Fig. 2-2 Stream structure for MPEG video

Motion estimation obtains the motion vector, which is the displacement between a macroblock and its predictor candidate, by minimizing a cost function measuring the mismatch between the two macroblocks. If no match is found within a specified search range, a macroblock will be intra-coded. A macroblock in a P-picture can also be skipped, meaning it is exactly the same as the macroblock at the same location in the reference picture.

As a result, a skipped macroblock will be motion vector of zero. For each macroblock in a B-picture, it can be forward-predicted, backward predicted, or bi-directionally predicted. Consequently, its motion information consists of one forward motion vector, one backward motion vector, or both of the forward and backward motion vectors. Once the motion vector for each macroblock is estimated, the prediction error or residue, the

difference between a macroblock and its matched candidate, is calculated. The residue will then be intra-coded by the DCT transform method.

### 2.1.2 DCT Transform

Both still-image and difference image (residue) signals have a very high spatial redundancy. Because of the block-based nature of the motion compensation process and a relatively straightforward implementation, the two dimensional Discrete Cosine Transform (DCT) is chosen as the basis of compression of each I-picture and of the residue images from P- and B-pictures. The Forward and Inverse DCT are defined as follows:

$$c(i,j) = \frac{1}{4}k(i)k(j)\left[\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y)\cos\frac{(2x+1)i\pi}{16}\cos\frac{(2y+1)j\pi}{16}\right] \quad (2.1)$$

$$f(i,j) = \frac{1}{4}\left[\sum_{x=0}^{7}\sum_{y=0}^{7} k(x)k(y)c(x,y)\cos\frac{(2i+1)x\pi}{16}\cos\frac{(2j+1)y\pi}{16}\right] \quad (2.2)$$

where $c(i,j)$ is the DCT coefficient, $f(i,j)$ is the original pixel value, $i,j = 0,1,...7$ and

$$k(i) = \begin{cases} \frac{1}{\sqrt{2}} & i = 0; \\ 1 & \text{otherwise.} \end{cases} \quad (2.3)$$

Among the 64 DCT coefficients, c(0, 0) is the weighted value for the DCT basis function, referred to as the DC term. The other 63 DCT coefficients, c(i, j), for i, j = 0, 1, ..7 are referred to as the AC coefficients. From the Forward DCT, we have 64 coefficients for each block. These coefficients are quantized, zig-zag ordered, run-length and then Huffman coded to reduce spatial redundancy. Fig. 2-3 presents a typical video encoder/decoder including all the described stages for decoding and encoding. In addition, the motion vector flowing path is stated.

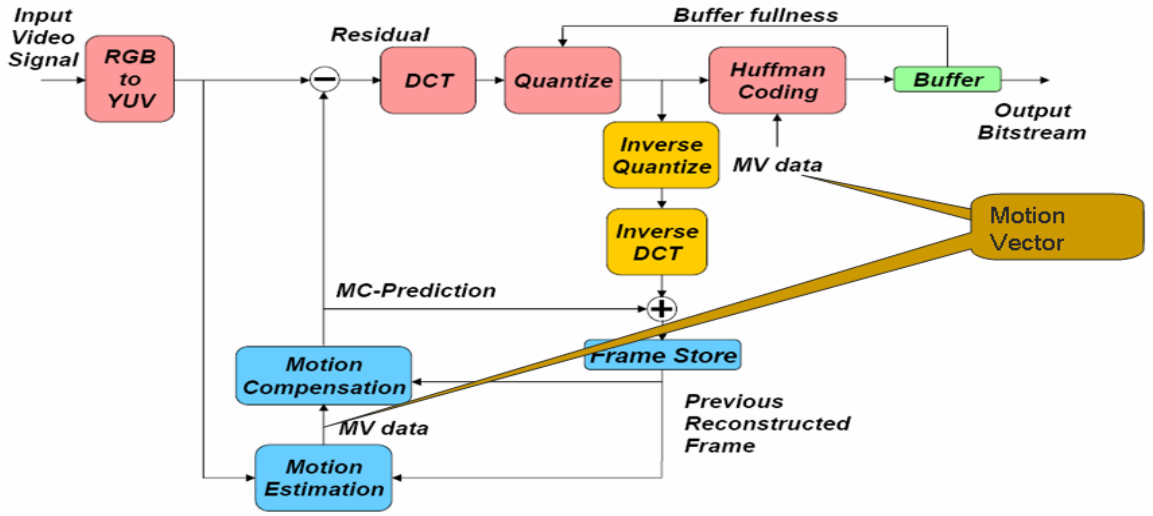



Fig. 2-3 Typical video encoder/decoder flowchart

## 2.2 Related Work

The need for reliable and accurate motion vector information is clear for those approaches that are employing the motion information [2,14-16,27-29,33,35,36]. However, a refining process has to be performed on the motion vectors.

Some researchers applied the motion vector refinement in the pixel domain. This implies they were actually filtering the visual features. Some [1] matched the filtered visual feature with the motion vector. Others used filtered visual features as a post-processing step [29]. Although [1,29] claimed that the motion vector refining results can be improved significantly, drawbacks remain. These drawbacks include the increment in computational complexity, as well as the other already mentioned drawbacks of doing video processing in the pixel domain.

Median filter and modified median filter [33,37] has been used for refining motion vectors. However, results in [59,61] prove the usage of median filter is insufficient and unreliable for high level applications. In these approaches [33,37], the time consumption is high due to the computation complexity resulting from partially processing in the pixel

domain. Approach [37] used P,B frames in order to extract motion vectors. [33] applied refining for the motion vector magnitude only, while in our proposed scheme, refinement is applied on both magnitude and direction, which will result in a more accurate and reliable outcome.

The approach in [38] is based on an assumption which consider motion vector refining results from median filter is the closest to true motion vector. In addition a fuzzy set membership function and some statistical approach were used to decide the motion vector robustness and filter it out accordingly.

However their basic assumption is not accurate according to what has been discussed earlier regarding the weakness and the limitation of median filter in smoothing motion vectors. In addition, the proposed post processing operations after using the motion vectors smoothing operation are sophisticated and time consuming. Hence it is not suitable for real time application.

Scheme in [39] performs motion vector refinement by using simple thresholding for both motion vector magnitude and direction. For general purpose, applying simple thresholding to the magnitude and the direction of the motion vector is not effective enough to remove random noisy vectors in the background region. In addition, the thresholding process based on experimental results only is not deterministic.

# Chapter 3

# Motion Vector Analysis

## 3.1 Motion Information Extraction from MPEG video

The sparse motion vectors from the compressed video stream are extracted. For the computation efficiency, only the motion vectors of P-frames are used for object detection algorithm 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. Besides, it must be noted that B frames are just 'interpolating' frames that hinge on the motion information provided in P frames and therefore using them for the concatenation of displacements would be redundant. Therefore, it is sufficient to use the motion information of P frames only in several video processing applications.

## 3.2 Motion Vector Noise Analysis

Due to the complexity of motion vector calculation in the encoding process, the raw motion vectors extracted from an MPEG or H.26x video stream may contain incorrect motion vectors and noise as will be proved later on in this section. Therefore, the realization of any approach which utilizes motion vector has to contain a refining mechanism to eliminate the incorrect motion vectors and noise.

Fig. 3-1 state the relation between the processing-time and the information level. Traditional methods start from the bottom level to detect raw motion information from the original video. In contrast, our proposed scheme starts from the middle level by extracting the motion vectors directly from the MPEG videos. The top level operates on specified and semantically filtered motion information. This represents a significant saving in computational cost in comparison to traditional approaches. This will be proven

in the result and discussion section.

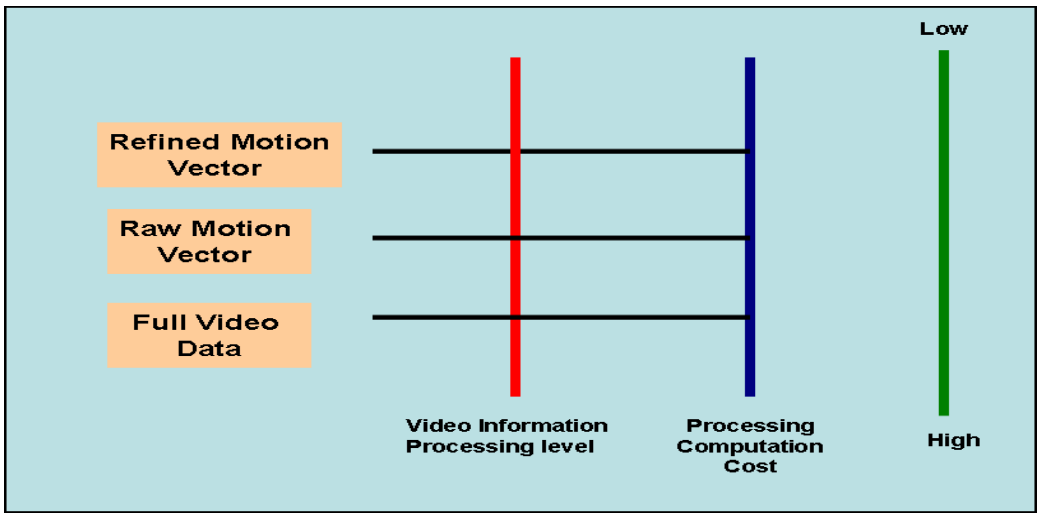



Fig. 3-1 Video processing level and cost effect

## 3.2 Motion vector limitations

Motion estimation can be seen as an optimization problem. The brute force methods simply try all possible vectors, in a predefined range, to be sure to obtain the global optimum of the criterion function. Also, there are efficient approaches that test only the most likely motion vectors. This likelihood is usually determined by spatial or temporal proximity, and, consequently temporal and spatial prediction vectors have been popular in the efficient motion estimation algorithms. Depending on the motion estimation algorithm used, the quality of the resulting motion vector is different.

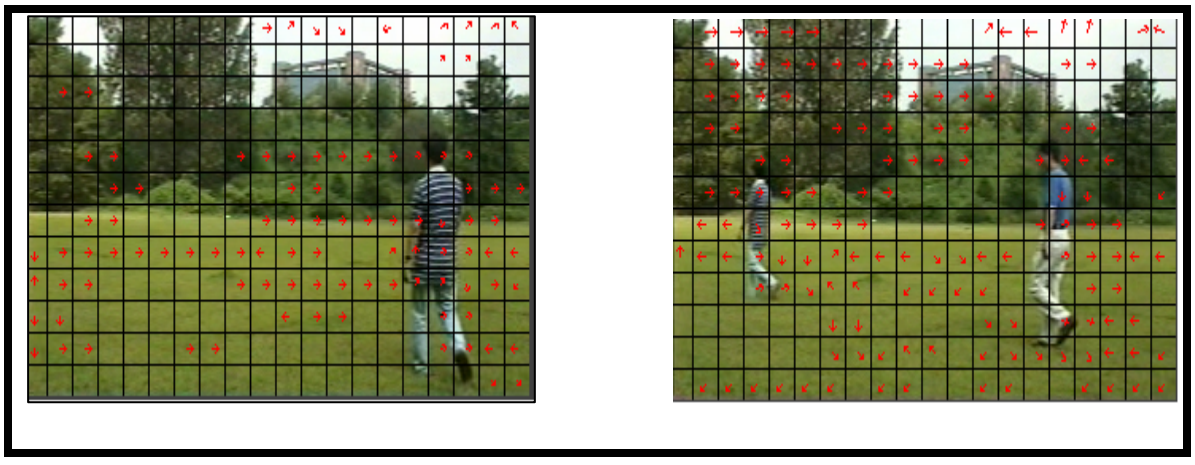
At the decoder/receiver side, it is unknown what type of motion estimation algorithm was used at the encoder/transmitter side, so one must assume, as a worst case situation, that the MPEG motion vectors are optimized for an efficient compression, and that they do not represent true motion vectors.

In sum, motion vectors in compressed domain suffer form the following limitations:

1. A homogeneous background could produce strange and long inconsistent vectors when small change of light happens in a heterogeneous way. More specifically,

periodical structures and noise in picture areas with little detail may cause such inconsistent vectors.

2. In MPEG data streams, it is uncertain that all motion vectors are transmitted within the data stream.
3. The motion vectors lack effective representation and its strong noise makes further processing almost impossible.
4. Researchers [28] elaborate on the noise in motion vector due to the camera noise and irregular object motion. However, it is known that the motion vectors in MPEG-1/2 may not represent the true motion of a Macro-block.
5. The macroblock scheme makes the usage of motion vector hard. For example, in object detection application, objects that are too small are simply ignored and object contours are distorted [29,30,31].
6. It is realized [32] that the motion fields in MPEG streams might have incorrectly matched in low-textured area. Fig. 3-2 and Fig. 3-3 show examples of extracted motion vectors from "walking person" and "interview" video streams respectively. Some of these motion vectors are either irrelevant or false. For example, in background areas or static object there exist 'motion vectors,' which contradict with basic idea of motion vector.



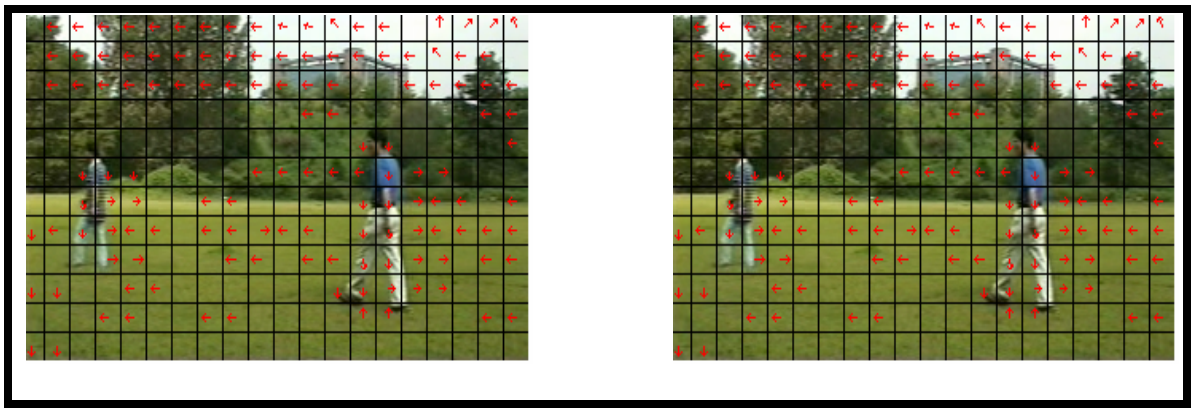


Fig. 3-2 extracted motion vectors from video stream “walking persons”

Fig. 3-3 extracted motion vectors from video stream “interview”

## 3.3 Motion Vector Noise Model

These described limitations above do not exclude the motion vectors entirely from use in high quality video processing applications. When an appropriate post-processing is applied, the motion vectors can be made useful. More specifically, when the receiver is able to determine the quality of the motion vectors, and when it is able to improve the quality of the motion vectors so that they meet certain criteria for the intended processing, the motion vectors can be used.

Now, we describe the motion vector noise model. A motion vector can be represented in Eq. (3.1). $v(i,j)$ is the sum of the true reliable motion vector $t(i, j)$ and the noise $n(i, j)$, where $i,j$ are the indices of the corresponding macroblock in each frame.

$$v(i, j) = t(i, j) + n(i, j) \qquad (3.1)$$

Noise within the content of the motion vector fields can generally be expressed as independent noise, which can often be described by an additive noise model.

# Chapter 4

# Motion Vectors Refinement Approaches

Several motion refinement approaches have been proposed and conducted. These approaches are developed in different domains, such as spatial domain, temporal domain, texture domain. Spatial domain filter concerns the motion vector refinement based on the spatial correlations among the motion vectors in each motion frame, such as P-frame.

## 4.1 Gaussian Based Motion Vector Refinement

Noise within the content of the motion vector fields can generally be expressed as independent noise, which can often be described by an additive noise model. Additive noise is evenly distributed over the frequency domain, whereas a reliable motion vector exists at low frequencies. Hence the noise is dominant for high frequencies and its effects can be reduced by using some kind of low pass filter. This can be done either with a frequency filter or with a spatial filter, but a spatial filter is preferable as it is computationally less expensive than a frequency filter.

Fig. 4-1 states the general scheme for Gaussian based motion vectors refinement. First step extracts the motion vectors from the MPEG video stream. Then, noise elimination based on Gaussian filter controlled by the so called configuration box is conducted. Configuration box is the module in charge of allowing users to interact and adjust the Gaussian filter parameters to optimize the noise elimination process.

In the literature there are many spatial filters such as Gaussian, median and mean filters. We use the Gaussian filter due to its desirable characteristics. The Gaussian filter has the significant characteristic of its step response containing absolutely no overshoot. The Gaussian filter uses a kernel that represents the shape of a Gaussian distribution. Fig.

4- 2 illustrates the Gaussian distribution. Gaussian functions are separable. As a result, a Gaussian convolution can be implemented by a 1D horizontal convolution followed by a 1D vertical convolution. Using this decomposition, the number of operations decreases significantly. Hence the Gaussian filter can fit real time applications.

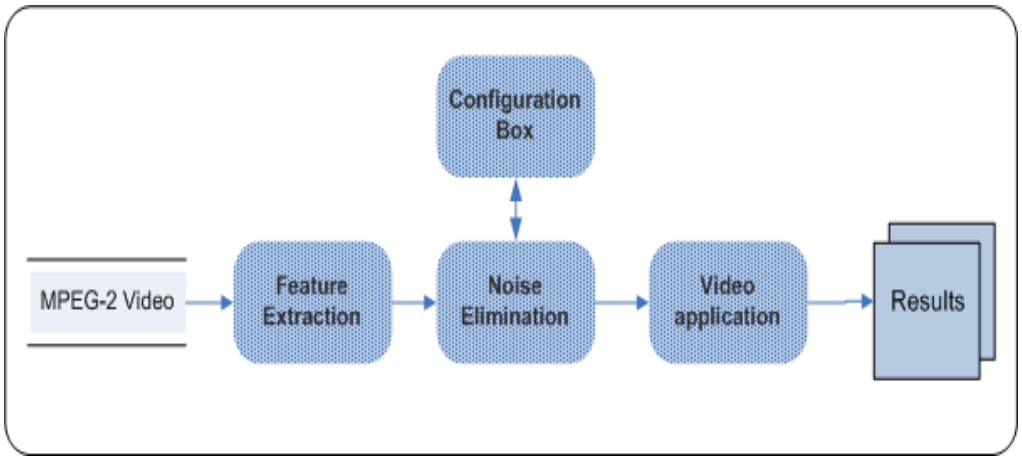



Fig. 4-1 General scheme for Gaussian based motion vector refinement

The Gaussian function is uni-modal. The Gaussian filter provides gentler smoothing and preserves the crucial motion vector value better than mean filter. Gaussian functions are rotationally symmetric in two dimensions. Thus, the amount of smoothing is independent of the direction. This property implies that no bias is introduced.

The degree of smoothing is parameterized by the standard deviation of the filter $\sigma$. We can maintain this value interactively to control the result of motion vector filtering. Hence, using the Gaussian filter gives some flexibility, which makes the refinement scheme possible for a wide range of applications.

### 4.1.1 Smoothing of Motion Vector Field

Up to this stage we have the motion vectors extracted from motion frames in MPEG. Then, we will pass the motion vector magnitude and direction values to the Gaussian filter where using two values instead of only one makes the refinement process more robust and meaningful. First we need to configure Gaussian filter by setting the

parameters such as $\sigma$. Such parameters have a crucial effect in the smoothing process. Thus we implement a user interface "configuration box" as demonstrated in Fig. 4-3 where users can interactively change the parameters for filtering until the optimal filter performance is obtained.

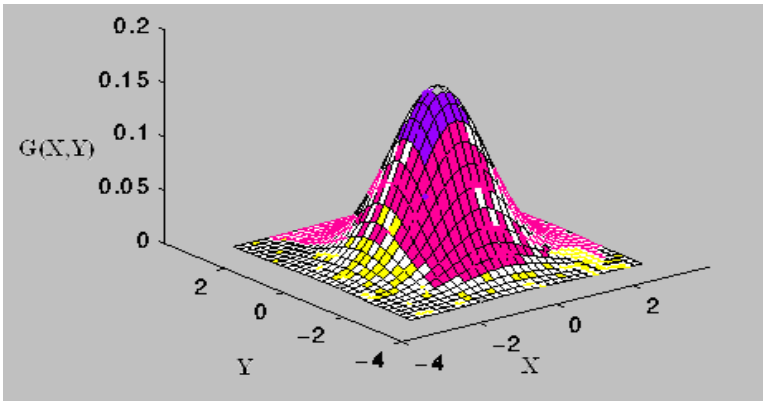



Fig. 4-2. 2-D Gaussian distribution with mean (0,0) and $\sigma$=1

First, the value of $\sigma$ is chosen to maximize the smoothing. By experiment, the value $\sigma$=1.2 gives us the best performance for Gaussian filter. [36] explored the use of a Gaussian filter, and excellent results can be achieved with a $\sigma$ value between 1.0 and 1.5. Smaller values of $\sigma$ (values less than 1.0) tend to leave slightly inhomogeneous cluster patterns as shown in Fig. 4-4. While larger values tend to form regularly spaced clusters patterns as shown in Fig. 4-5. This proved to be in agreement with our experiment over the motion vector smoothing.

Concerning other parameters, the kernel size was chosen to be 3X3 since the window search in our object detection is 3X3 as well. Moreover, the kernel size is recommended to be 3X3 in the interest of reducing the cost of computation. The last parameter to determine is the iteration, the number of times to repeat the convolution step. This will affect the degree of enhancement and the accuracy of the filter. Empirically we find the value 5 to be a suitable value, taking into consideration the performance and the execution time.

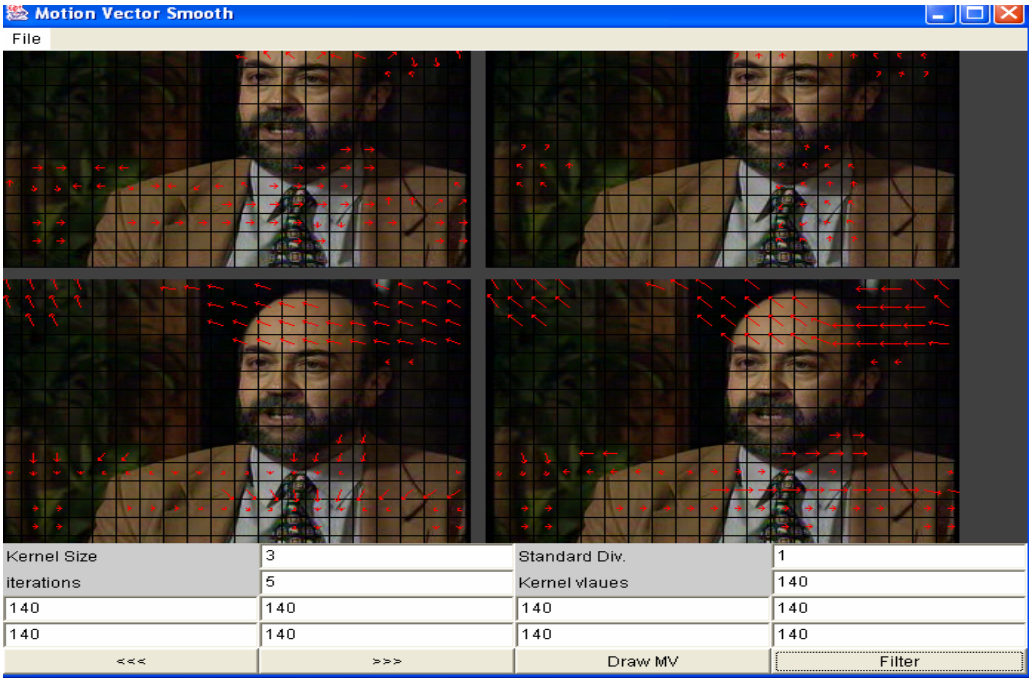



Fig. 4-3. The user interface for motion vector smoothing

Fig. 4-4 The effect of using small $\sigma$ value, $\sigma$=0.002

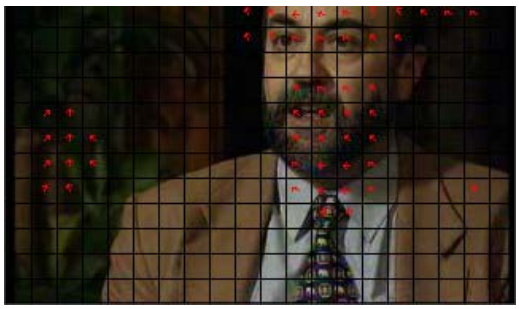


Fig. 4- 5. The effect of using large $\sigma$ value, $\sigma$=3.5

Fig. 4-6(a)(b)(c)(d) show the performance of motion vector refinement using no filter, Gaussian filter, median filter and mean filter, respectively. The value of $\sigma$ has been chosen to be 1.2 and kernel size to be 3X3. The presented results show clearly the advantage of using Gaussian filter comparing to the usage of mean and median filters

Fig. 4-7 represents the same idea for the different video clip “walking person”.

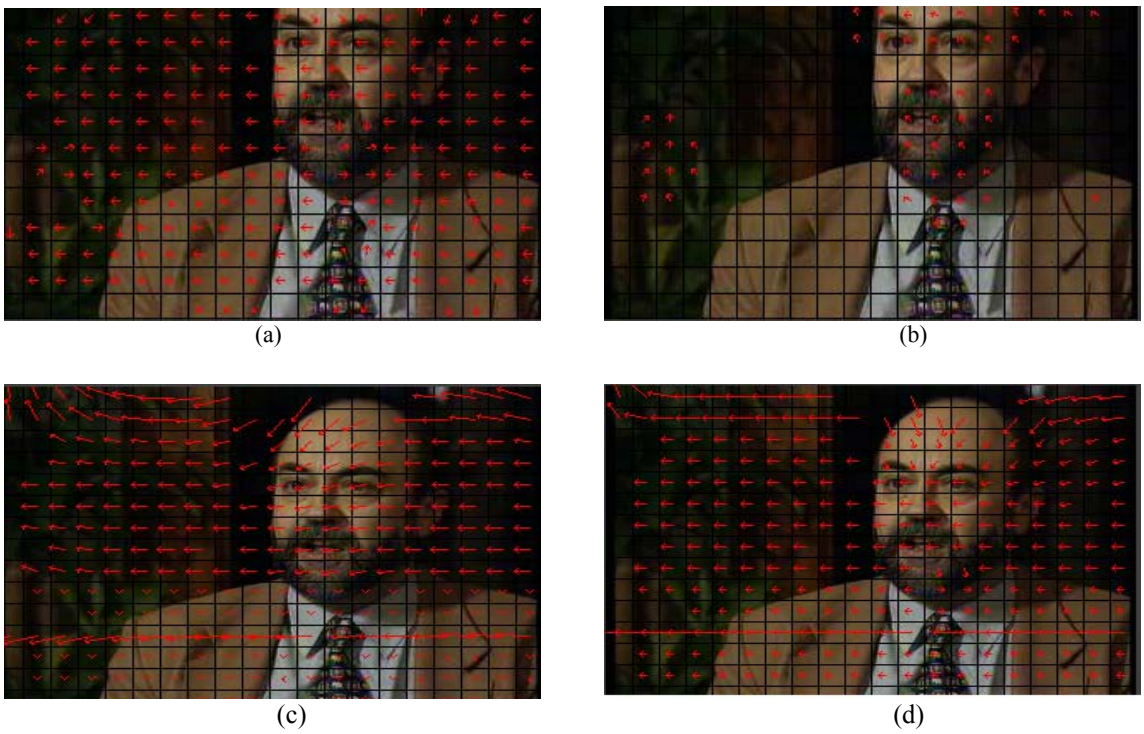



Fig. 4-6 Extracted motion vectors from “interview video” after applying (a) Without processing (b) Gaussian filter (c) Mean filter (d) Median filter.

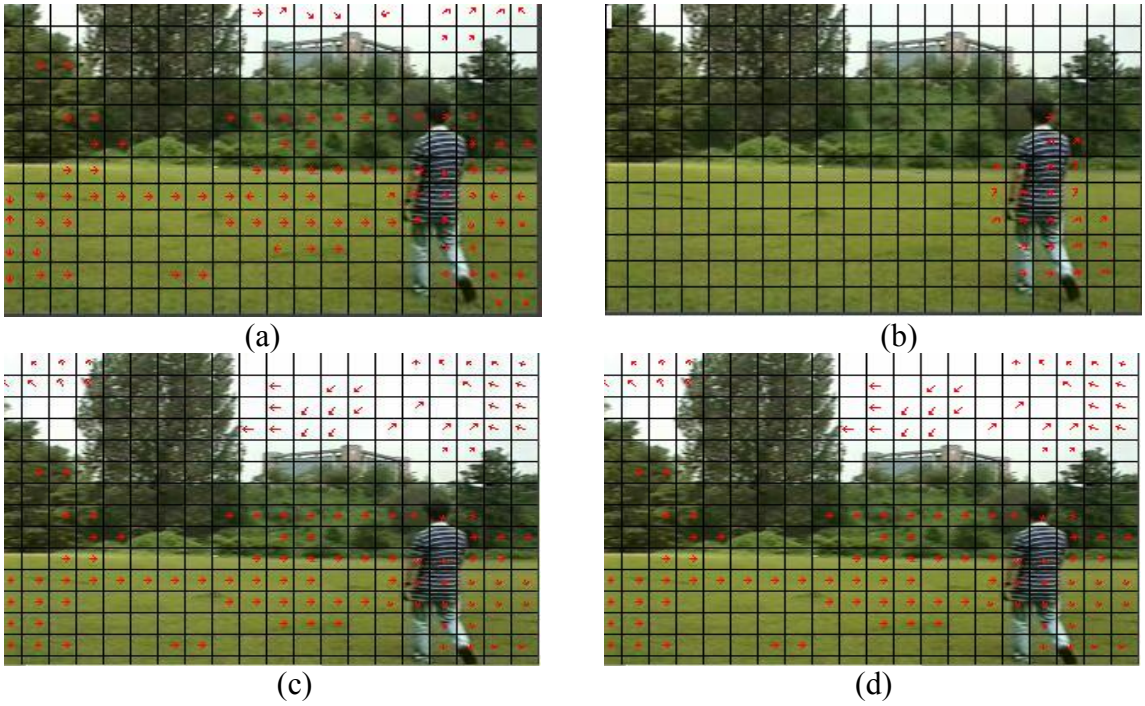



Fig. 4-7 Extracted motion vectors from “walking person video” after applying (a) Without processing (b) Gaussian filter (c) Mean filter (d) Median filter.

## 4.2 Cascade Filter Based Motion Refinement

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. We use a *cascade filter* which is composed of a Gaussian filter followed by median filter to improve the performance. Fig. 4-8 and 4-9 schematically illustrates the cascade filtering.

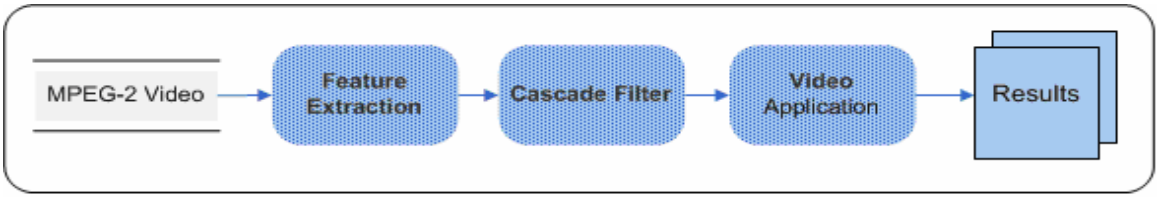



Fig. 4-8 Refinement scheme overview

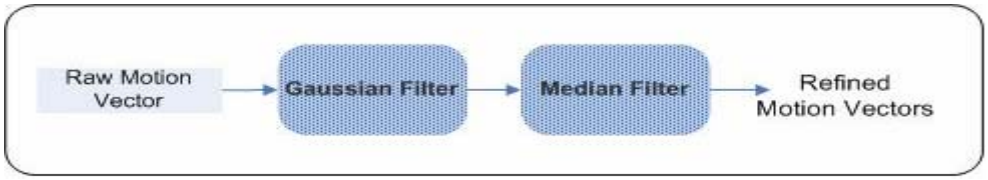



Fig. 4-9 Cascade filter design

The deficiency in frame border will disappear due to the median filter's characteristic of rearranging the motion vectors value to become more representative of the true motion vector and better aligned. The proposed cascade filter boosts the performance. In addition, the computational complexity is low. Both the Gaussian and median filters are available as a readily implemented component in both hardware and software.

The resultant motion vectors resulting after filtering are less noisy. Execution time of deploying the refined motion vectors will be reduced significantly compared to that without using a filter. Although we are adding another block for filtering, the efficiency is almost the same or even better in terms of execution time for the entire object detection process.

## 4.3 Temporal based Motion Vector Refinement

Temporal filter component is designed based on the temporal adjacent neighborhood of a macroblock. The main idea is that a 'fine' motion vector should not have its direction altered in a drastic manner.

Fig. 4-10 states the relation among the current, successive and precedent frames. Motion vectors in these frames are temporally correlated. Each frame is affected by its successive and precedent frames. The closer the frame is, the more correlated and contributing to its neighbor, so $MV_{N+1}$ is twice important than $MV_{N+2}$ to current motion vector $MV_N$. Eq(4.1) states the temporal refinement.

$$MV_{new} = (2(MV_{N+1} + MV_{N-1}) + (MV_{N+2} + MV_{N-2}))/6 + \tfrac{1}{2}(MV_N) \qquad (4.1)$$

where $MV_{new}$ is the refined motion vector in temporal domain, and N-1,N-2,N,N+1,N+2 are frame numbers in the video sequence.

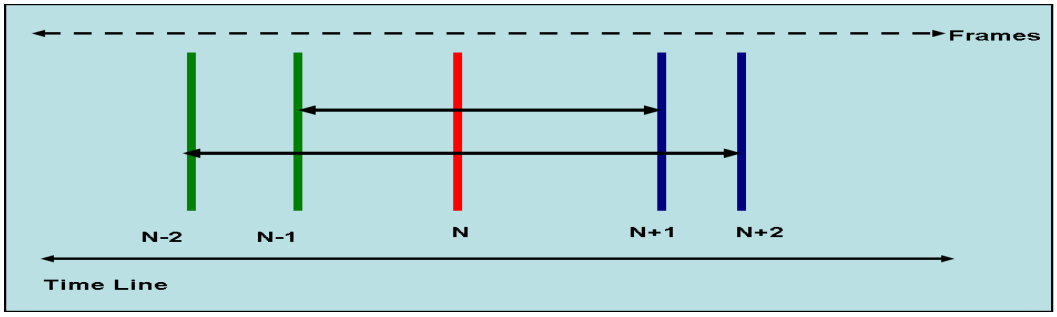



Fig. 4-10 Relation among the current fame and other frames in temporal domain.

## 4.4 Texture Based Motion Vector Refinement

Up to this stage, motion vectors have been refined temporally and spatially. Important domain is the texture domain due to the fact that well textured motion vector has good motion vector value. This will lead us to adaptively refine the motion vectors based on texture. To calculate the texture for each motion vector we analyze the AC components of the DCT coefficients, thus staying in the compressed domain. The proposed scheme was inspired by this fact.

The MPEG compressed video provides one motion vector for each macroblock The

DCT information from I-frames are readily available in MPEG stream. Thus we need to spend too much time in decoding the MEPG stream. Hence our approach can fit the real time application environment.

### 4.4.1 Texture energy computation

Important regions are distinguished from background using the distinguishing texture characteristics. The video analysis is performed directly in the DCT compressed domain using the *intensity variation* encoded in the DCT domain. Therefore, only a very small amount of decoding is required. In some applications, researchers do use either the horizontal intensity variation or vertical intensity variation. For example, in the text detection, it is generally approved that text regions possess a special texture because text usually consists of character components which contrast the background and, at the same time, show a periodic horizontal intensity variation due to the horizontal alignment of characters. In addition, character components form text lines with approximately the same spacing [42,43]. As a result, text regions can be segmented using texture features.

In this thesis we propose a texture-based motion vector filter which operates directly in the DCT domain in video. The DCT coefficients in MPEG video [45], which capture the directionality and periodicity of local image blocks, are used as measures to identify high texture regions. Therefore, we will be able to treat each motion vector accordingly. Each unit block in the compressed images is classified based on local horizontal, vertical and diagonal intensity variations.

In summary, DCT coefficients in compressed domain images capture the local periodicity and directionality features in the spatial domain. We examine the coefficients of each DCT block of a frame. By noting the correlation of frequency distribution in each block and the corresponding spatial features (see Fig. 4-11(a)), we may choose to

process further only those blocks which meet certain criteria. For example, to detect edges, only the medium and high frequency components are needed [10], and only the blocks containing coefficients in that range are considered for decompression.

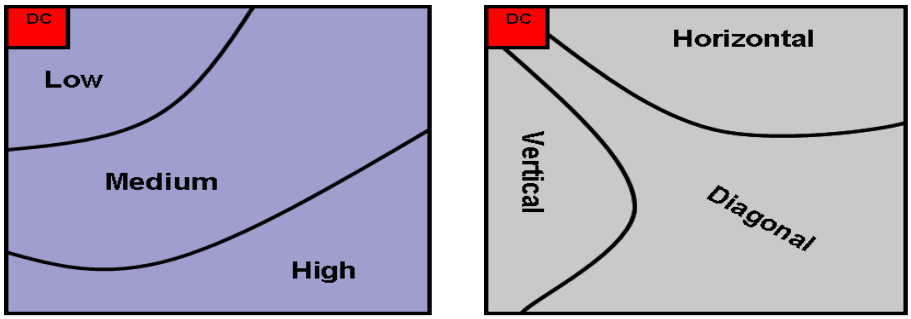
In addition, we may use the feature distribution patterns of DCT coefficients to choose subregions of a representative video frame (see Fig. 4-11(b)). Blocks containing high and medium frequencies are selected, and only the set of blocks that correspond to a 'large' region in the spatial domain are selected. The computational savings that result from this simple step are many folds. The DCT processing need not be applied to the entire image, resulting in additional savings in time. Since a smaller image area is analyzed, all subsequent steps, such as detecting straight edges, or detecting long edges can be completed more effectively.

To gain some insight into the DCT spectrum, Fig. 4-12(a) shows an input image and Fig. 4-12(b) shows the absolute values of the DCT coefficients directly extracted from the compressed domain of the intensity image. Each subimage in Fig. 4-12(b) represents one DCT block of the input image.

The blocks, from top to bottom, indicate horizontal variations, with increasing frequencies; and from left to right, indicate vertical variations, with increasing frequencies. The blocks on the top row, from left to right represents zero vertical frequency and increasing horizontal frequencies. Top left blocks represent the low frequency components, contain most of the energy, while the high frequency channels, which are located at the bottom right corner of each subimage, are mostly blank.

These observations indicate that the blocks spectrums capture the directionality and coarseness of the spatial image. For all the vertical edges in the input image, there is a corresponding high frequency component in the horizontal frequencies, and vice versa.

Furthermore, diagonal variations are captured by the energies around the diagonal line. This example illustrates that the DCT domain features do characterize the texture attributes of an image. The basis images are placed with increasing horizontal frequency from left to right, and increasing vertical frequency from top to bottom.




(a) (b)

Fig. 4-11 (a) frequency distribution (b) block distribution [11]

Therefore, we can design *Directional Texture Energy Map* in DCT domain as shown in Fig. 4-13, by assigning a directional intensity variation indicator for each AC coefficient as the following.

**H: Horizontal intensity variation**

**V: Vertical intensity variation**

**D: Diagonal intensity variation**

We are processing in the DCT domain to obtain the directional intensity variation, called directional texture energy, using only the information in the compressed domain. Note that the operating units are the 8X8 blocks in I-frames.

Fig. 4-12-a: Input image

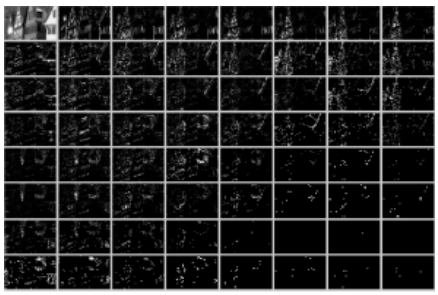


Fig. 4-12-b: The absolute values of the DCT coefficients directly extracted from the compressed domain.

For each DCT block, we compute the Horizontal energy Eh by summing up the absolute amplitudes of the horizontal harmonics of the block: which are marked as H in the directional texture energy map. For each DCT block, we compute the vertical energy Ev by summing up the absolute amplitudes of the vertical harmonics of the block: which are marked as V in the directional texture energy map. For each DCT block, we compute the diagonal energy Ed by summing up the absolute amplitudes of the diagonal harmonics of the block: which are marked as D in the directional texture energy map. Finally, we will calculate the average energy Ea for each block, which is the average value of the Vertical energy, Diagonal energy and Horizontal energy as in Eq.( 4.2).

$$E_a = \begin{cases} \dfrac{12*E_h + 5*E_d + 12*E_v}{29} \\ \dfrac{11*E_h + 7*E_d + 11*E_v}{29} \\ \dfrac{11*E_h + 6*E_d + 12*E_v}{29} \\ \dfrac{12*E_h + 6*E_d + 11*E_v}{29} \end{cases} \qquad (4.2)$$

We will update Motion vector values based on the Ea as described in the following procedure.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | H | H | H | H | H | H |
| 0 | 0 | [D,H] | H | H | H | H | H |
| V | [V,D] | D | D | 0 | 0 | 0 | 0 |
| V | V | D | D | D | 0 | 0 | 0 |
| V | V | 0 | 0 | D | D | 0 | 0 |
| V | V | 0 | 0 | D | D | 0 | 0 |
| V | V | 0 | 0 | 0 | 0 | 0 | 0 |
| V | V | 0 | 0 | 0 | 0 | 0 | 0 |

Fig. 4-13: Directional Texture Energy Map in DCT

$$For\ every\ macrblock\ :$$

$$Motion\_Vector_{new} = \begin{cases} Motion\_Vector_{old} * \dfrac{100 * E_a}{E_t}\% & , E_a < E_t \\ Motion\_Vector_{old} & , E_a \geq E_t \end{cases}$$

We have used an adaptive threshold value. The following diagram in Fig. 4-14 describes the process for texture filtering in the frame level. The I-frame has no motion values and it stores DCT information of the original frame. Though I-frame provides no motion information, we still could grasp the frame texture, and propagate that information to the P frames as described in next section.

Fig. 4-14 clearly states the operation in algorithmic way, where, after knowing the frame type we send this frame for its specific module. If this frame is an I-frame, then we know it contains the DC value and AC components. Thus, we can calculate the texture energy as stated before according to the previous procedure and texture energy map. After we propagate those values into P-frames then we perform the texture filter on the motion vector values.

Alternative procedure for texture and edge calculation has been proposed for providing additional texture information. The details of the approach are presented herein. First of all the energy map is redefined according to Fig. 4-15. New energy detentions are introduced.

$A_{tot}$ : Total Energy

$A_D$ : Diagonal Energy

$A_H$ : Horizontal Energy

$A_V$ : Vertical Energy

$A_{Fin}$ : Final Energy

Energy calculation is based on the redefined texture energy map as show in Fig. 4-15.

$$A_V = C_{16\text{-}17}^2 + C_{24\text{-}25}^2 + C_{32\text{-}33}^2 + C_{40\text{-}41}^2 + C_{48\text{-}49}^2 + C_{56\text{-}57}^2$$
$$A_{Fin} = C_{57}^2 + C_{45}^2 + C_{15}^2$$
$$A_{tot} = A_H + A_V + A_H$$
$$A_D = C_{45}^2 + C_{44}^2 + C_{36}^2 + C_{37}^2 + C_{28}^2 + C_{27}^2 + C_{26}^2 + C_{19}^2 + C_{18}^2$$
$$A_H = C_{2\text{-}7}^2 + C_{10\text{-}15}^2$$

To make a decision regarding the texture of each macroblock the following procedure is deployed.

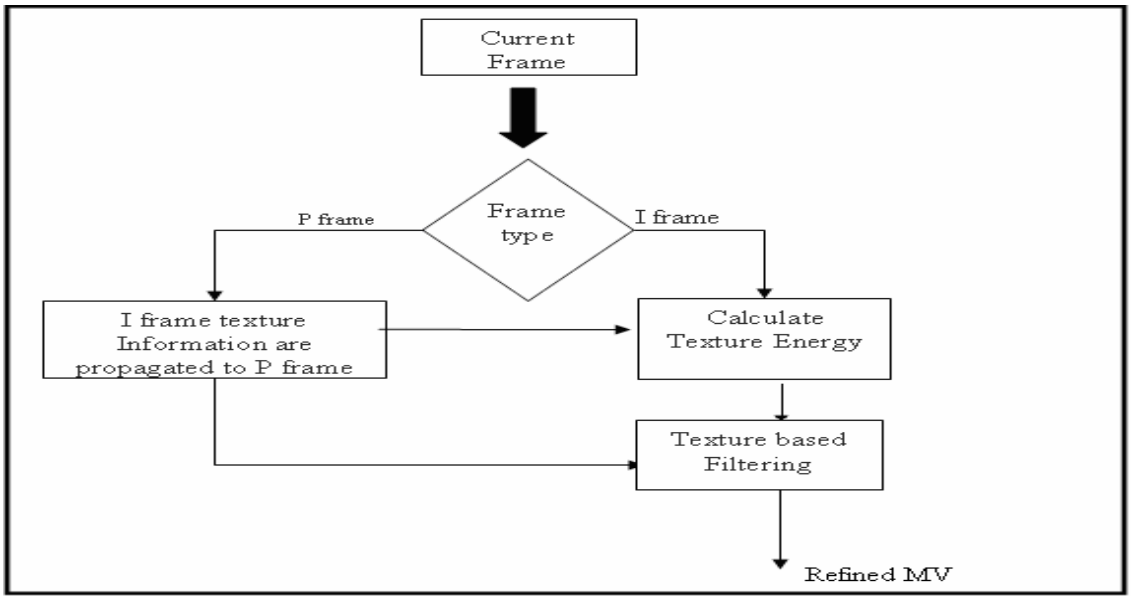



Fig. 4-14: Flowchart of texture based motion vectors refinement.

(1) IF $(A_D/A_{tot}) > Threshold_1$ Then
High contrast → Fine Else
Low Contrast →Texture End IF
(2) IF $(A_H/(A_H + A_V) > Threshold_2$ Then
Vertical Edge Else
Edge or "texture or coarse area" End IF

(3) IF $(A_V/(A_V + A_H) > Threshold_3$ Then
Horizontal Else
Texture or Coarse Area End IF
(4) IF $(A_{Fin}/A_{tot}) > Threshold_4$ Then
Coarse Else
Texture End IF
$Threshold_{1\text{-}4}$ are decided by experiments

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $C_0$ | $C_1$ | $C_2$ | | | | | $C_7$ |
| $C_8$ | $C_9$ | $C_{10}$ | | | | | $C_{15}$ |
| $C_{16}$ | $C_{17}$ | $C_{18}$ | | | | | |
| | | | $C_{27}$ | $C_{28}$ | | | |
| | | | | $C_{36}$ | | | |
| | | | | | $C_{45}$ | | |
| | | | | | | $C_{54}$ | $C_{55}$ |
| $C_{56}$ | $C_{57}$ | | | | | $C_{62}$ | $C_{63}$ |

Fig. 4-15 Redefined Texture Energy Map

## 4.4.2 Reconstruction of DC Images

To propagate the texture information into the p frames we need to reconstruct the DC images for P frames. In the following, only the reconstruction of the luminance DC images is discussed. Chrominance DC images can be similarly reconstructed. For intra-coded I-pictures, reconstruction of such DC images is trivial since the DCT DC value of each block can be directly obtained from an MPEG stream.

The DC values of an intra-coded macroblock in a P-picture can be similarly obtained as those in an I-picture. Extraction of exact DC values for motion compensated macroblocks in a P-picture is given in [32] and is computationally expensive.

Here we describe an approximation method proposed by [112]. A motion compensated macroblock in a P-picture has a motion vector and four blocks of DCT coded motion compensated errors. The motion vector allows us to trace back the macroblock to its matching counterpart in the previous reference picture. Each of the four luminance blocks will be matched in a location in the reference picture as shown in Fig. 4-16. The matching block may overlap as many as four blocks in the reference picture. Assume that the DC values of the reference picture are available and the

luminance variance within each block is small. Then the DC value of the motion compensated block in a P-picture can be approximated by taking the overlapping area of the four blocks in the reference picture pointed by the motion vector plus its DC value of the residues as in Eq.(4.3).

$$DC(b) = \frac{1}{64}\sum_{i=1}^{4} h_i w_i DC(b_i) + DC(b_{residue}), \qquad (4.3)$$

Where $DC(b_i)$ is the DC value of block *i* in the reference picture, and $w_i$ and $h_i$ are the overlapping width and height respectively. Their values are related to the motion vector (u,v) as follows: w1=w3=u, w2=w4=8-u, h1=h1=v and h3=h4=8–v. The term DC(bresidue ) is the residue DC values of the current block. bi is block number .

For those motion compensated macroblocks with both forward and backward motion vectors, their DC values can be calculated as the average of those reconstructed from the previous reference picture and the future reference picture plus the DC values of their residues. Using the above method, we can reconstruct a DC image sequence from an MPEG stream, no matter what picture types (I, P or B) it contains.

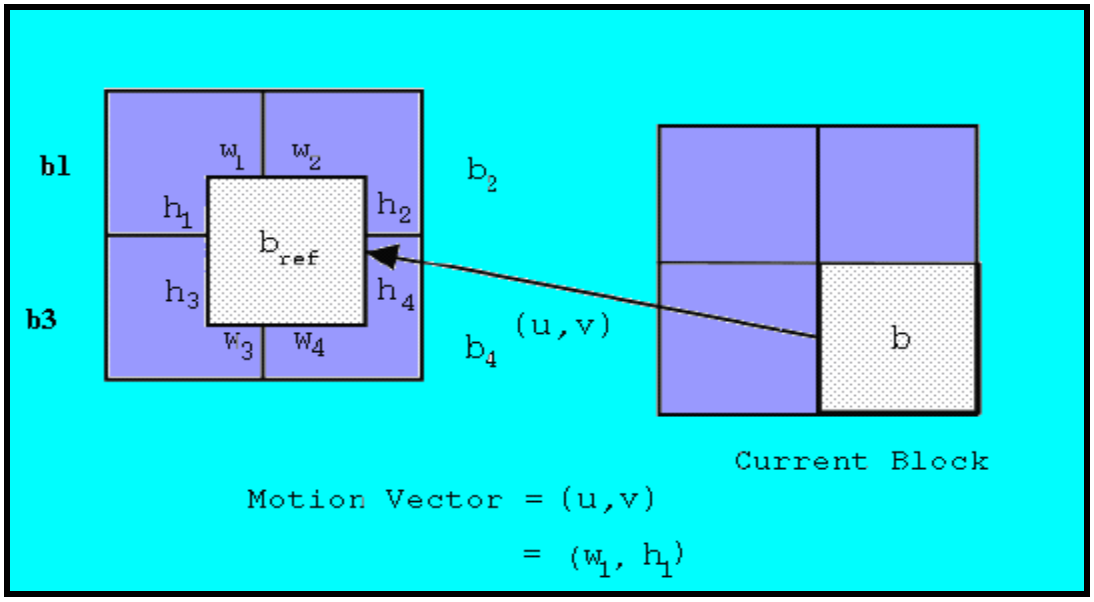



Fig. 4-16 illustrations on the relation among the current block, reference block and motion vector.

## 4.5 Overview of the Motion Vector Refinement System

The proposed system architecture for motion vector refinement now is shown in Fig. 4-17. For an input video stream, we extract the motion vectors from inter-frames such as P-frame. In practice, P-frames only are suggested to be processed in order to reduce the computational complexity. Meanwhile, we will extract the DCT coefficients from I frames, including the DC coefficient, and the AC components as well. Then, we will pass the DCT coefficients into a module to calculate the texture of each frame. Later, we will propagate this texture information into the inter-frames using the DC image reconstruction technique described early. After that, we will filter each motion vector based on its texture value.

Then, we pass the filtered motion vectors to temporal filter component. This filter is derived from the temporal adjacent neighborhood of a macroblock. After obtaining the motion vector field's magnitude and direction values, we pass these values through the Gaussian filter. Meanwhile, filter parameters such as standard deviation and kernel size are initialized to obtain the optimal performance. We then pass these filtered motion vectors into a specific application.

Furthermore, we use the median filter because it does not alter motion vector values. Rather, it simply rearranges motion vectors, not altering the values contained within any motion vector. Hence the median filter is used to repair potential irregularities introduced by the previous filter processing and in order to straighten up some single motion vector which has been influenced.

## 4.6 Adaptive Motion Vector Refinement

This section introduces for adaptive motion vector refinement technique in spatial domain. Previously, the adaptation process is supported in texture domain, but in certain

cases, neither the texture domain nor the temporal domain techniques are applicable. Therefore, a spatial domain adaptive motion vector refinement is proposed. Some application like video transcoding puts hard limitation on the refinement techniques and resources. Thus we develop a strict and very resource limited technique which lies only in spatial domain. In [1,12,41] the analysis showed that the differential reconstruction error causes incoming motion vectors to deviate from optimal values.

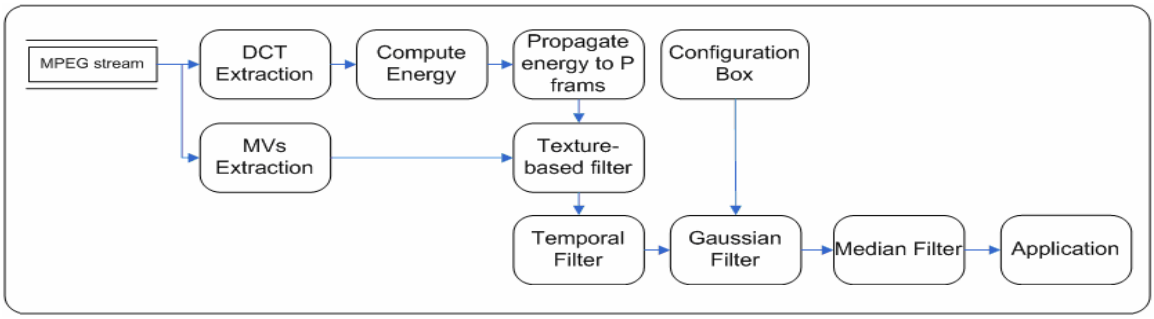



Fig. 4-17. The proposed motion vector refinement system architecture

Therefore, a fine motion vector can be obtained by refining the incoming motion vectors according to the proposed scheme. Depending on the application, various types of noise can be distinguished. Gaussian additive noise is a commonly applied model to represent the noise as obtained in motion vector signals. However, impulsive noise originating from uncertain defection can be recognized and requires an individual approach in order to get optimal performance. In this chapter we will focus on the Gaussian additive noise, and optionally try to defeat the impulsive noise effects.

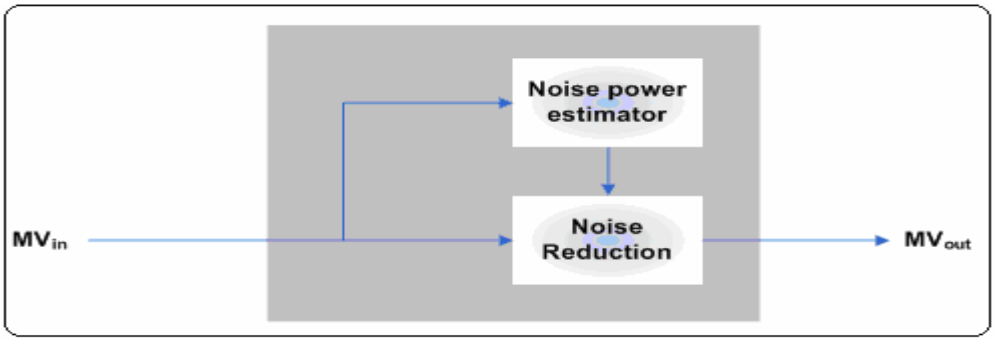



Fig. 4-18 The overview architecture of the proposed adaptive motion vector refinement scheme

In Fig. 4-18, we introduce a general view of the proposed motion vector refinement scheme. Our scheme consists of two parts; the first part is the noise level estimator. It is used to estimate the noise power for every frame processed. This parameter will be used in the noise reduction block to generate the filter parameters. The second part is the noise reduction block. It is a general spatial filter which uses the correlation between the current motion vector and its neighbourhoods. The spatial filter is shown in Fig. 4-19. It is a recursive spatial filter and includes a kernel and weighting module. The kernel as shown in Fig. 4-12 consists of a 2D window of motion vectors MV (i) (i=0, 1...8) where MV(0) is the central motion vector that will be filtered. MV(1)-MV(8) are the neighbourhoods of MV(0).

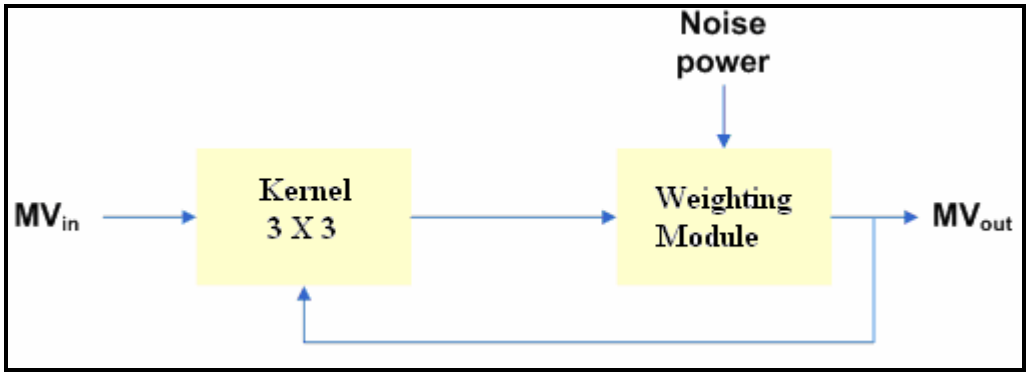



Fig. 4-19. Block diagram of the noise reduction block

| | | |
|---|---|---|
| $MV_1$ | $MV_2$ | $MV_3$ |
| $MV_4$ | $MV_0$ | $MV_5$ |
| $MV_6$ | $MV_7$ | $MV_8$ |

Fig. 4-20. Kernel of 3X3 motion vectors

In the weighting module, the output motion vector can be calculated as in Eq.(4.4).

$$MV_{out} = \frac{\sum_{i \in N_1} W_1(i) * MV'(i) + \sum_{i \in N_2} W_2(i) * MV(i)}{\sum_{i \in N_1} W_1(i) + \sum_{i \in N_2} W_2(i)} \quad (4.4)$$

where N1 = {1, 2, 3, 4} and N2 = {5, 6, 7, 8} are sets of neighbourhood vectors. And the weights of each motion vector are defined as:

$$W_1(i) = z_1 K(i) \qquad i \in N_1 \quad (4.5)$$

$$W_2(i) = z_2 K(i) \qquad i \in N_2 \quad (4.6)$$

where z1 and z2 are the parameters related to the position of the weighted Motion vector. To use more information of refined motion vector, we have z1 =3* z2. K(i) is a parameter related to the local noise power and the absolute difference between the weighted neighbourhood motion vector (MV(i)) and the current input motion vector (MV(0)). K(i) function is defined in Eq.(4.7).

$$K(i) = e^{-\left(\frac{MV(i)-(MV(0)+\lambda)}{Noise\ \ Power}\right)^2}, \quad (4.7)$$

where λ is a parameter related to the local noise power of current input motion vector. It is ranged within $-Noise\,Power \le \lambda \le Noise\,Power$. λ can be estimated by Eq.(4.8).

$$\lambda = \arg\min_{\lambda_j}\left\{\sum_{i \in N_1}\left|MV'(i) - (MV(0) + \lambda_j)\right| + \sum_{i \in N_2}\left|MV(i) - (MV(0) + \lambda_j)\right|\right\} \quad (4.8)$$

The weighting function K(i) indicates that the contribution of a neighborhood motion vector to the current centre motion vector is exponentially related to the difference between them. The smaller the difference, the more contribution the neighbourhood motion vector has. This property can eliminate the Gaussian noise effectively.

### 4.6.1 Noise Power Estimator

We introduce a method for noise power estimation. This noise estimation algorithm allows refinement at varying motion vector conditions. To increase the adaptation of the concepts, a simple but effective new noise power estimation algorithm was designed that controls the parameters of the noise reduction. The Noise Power Estimator is used to estimate the noise power for every frame processed. Noise power values are used in the noise reduction. The difference of every motion vector with its previous and next motion vector is calculated and the accumulated differences are delayed for four motion vectors and further accumulated into a variable entitled AC_DIFF. The Noise Power of each frame is updated as in Eq.(4.9).

$$Noise\ Power(f) = \begin{cases} Noise\ Power(f) - 1 & \text{NPI} > Thershold_t\ \Lambda Noise\ Power(f) \\ Noise\ Power(f) + 1 & \text{NPI} \le Thershold_t\ \Lambda Noise\ Powe(f) \end{cases} \quad (4.9)$$

Noise Power (f) is the noise power of frame $f$ and the $\text{Threshold}_t$ is an experimentally optimized constant defined as:

$$Threshold_t = (Number of\ Motion Vector * 60)/100 \quad (4.10)$$

The number of Motion vector is defined as the total number of motion vectors in each frame.

The Noise Power Indicator (NPI) is defined

$$\text{NPI} = \sum_{\text{MV}} \text{B(MV)} \quad (4.11)$$

where B(MV) is a binary variable assigned to every motion vector.

$$B(MV) = \begin{cases} 1, & Noise\ \ Power\ (f) < AC\ \_\ DIFF\ (MV) < 1.5\, Noise\ \ Power\ (f) \\ 0, & else \end{cases} \quad (4.12)$$

and AC_DIFF is the local noise power estimator calculated as a sum of Absolute Differences

$$AC\ \_\ DIFF\ (MV) = \sum_{n=1}^{4} \left| F(MV + n) + F(MV + n - 1) \right| \quad (4.13)$$

# Chapter 5

# Object Detection in Video Streams

## 5.1 Introduction

Over the last decade, there has been growing interest in content representation of video sequences with rapid developments in multimedia and internet applications. New representation of video sequences needs to be constructed not only in compact forms, but also as semantic entities for content-based functionalities such as retrieval and manipulation. Video semantic object is a perfect content representation.

To bridge the semantic gap and achieve a high level of content based representation, some previous works [46] select video object plane supported by the MPEG-4 standard as the underlying video patterns for video content representation and feature extraction. However, the major problem is that semantic video object extraction in general does not perform automatically. A good performance still needs human's interaction at the current stage. Recently, many researches on human visual system show that moving objects can be easily distinguished and have more attraction of visual attention.

Approaches [47,48] extract moving objects as semantic objects for content representation. Moving objects can provide a good pattern for motion-related high-level semantic analysis.

Studies on visual attention and eye movements [49,50] have shown that humans generally can only attend to a few areas in an image. Even with unlimited viewing time, attention will continue to focus on these few areas rather than scan the whole image. An increasing number of researchers are now exploring the intermediate-level processing, shifting the focus of attention away from pixel-based indicators to high-level processing.

In [51], authors use motion information to construct salient map for video sequence. Salient region extraction based on saliency map [31] provides a good starting point for semantic-sensitive content representation. However, perceived salient region extraction for image or video is still an unsolved problem. One reason is that video sequence has more context information than single image. Hence, low-level features are often not enough to classify some regions unambiguously without the incorporation of high-level and human perceptual information into the classification process. Another reason for the problems is perception subjectivity. Different people can differ in their perception of high-level concepts. Thus a closely related problem is that the uncertainty or ambiguity of classification in some regions cannot be resolved completely.

A basic difference between perceptions and measurements is that, in general, measurements are crisp whereas perceptions are fuzzy [52]. Moreover moving object detection is a useful tool for intelligent video browsing/analysis and video surveillance systems. In order to meet the real-time requirement, no computationally intensive operation is included. Video object detection is a key operation for content-based video coding multimedia content description, and intelligent signal processing.

New functionalities like object manipulation and scene composition can be achieved because the video bitstream contains the object shape information. However, the shape information of moving objects may not be available from the input video sequences; therefore, segmentation is an essential task.

In addition, many multimedia communication applications have real-time requirement, and an efficient algorithm for automatic video segmentation is very desirable. Motion estimation has been proposed [54] to solve this problem. If both ends of a motion vector are inside the frame difference mask, then the corresponding area is

part of the object. Otherwise, that area is assumed to be background.

This approach has several drawbacks. First, the motion estimation is not very accurate near the object boundary where highest accuracy is required. Second, motion estimation can deal with the translation type of motion only. If other forms of movement are involved, motion vectors may fail to track the object motion. Also, motion estimation is a computationally intensive operation and this process will dramatically increase the complexity of the segmentation system.

Motion-based video object detection has been explored, with approaches generally based on the analysis of optical flows. Compressed videos require the decompression of the sequences and the computation of optical flows, two steps computationally heavy. In this chapter we propose some methods for motion-based video object detection by motion features (mainly related to motion vector) and by motion-based spatial segmentation of frames, in a fully automatic way.

Our idea is to use motion vectors as an alternative to optical flows. It does not require a decompression of the stream and saves us from computing optical flows. Additional computational economy comes from having one motion vector each macroblock. This makes the algorithms faster than those that work with dense optical flows. Experimental results reported at the end of this chapter show that MPEG motion compensation vectors are suitable for this kind of applications.

## 5.2 Related Work and Background

Moving object detection techniques have been studied extensively [53-56] for purposes such as video content analysis as well as remote surveillance. For example, in [53] optical flow method is employed in pixel domain and a moving object is detected

when similar optical flow is found in a certain area. In [54], a moving object is detected using inter-frame difference after compensating camera work. As for the compressed data domain processing, the [55] proposes the detection method by finding objects with similar motion vectors in a picture after compensating global motion.

However, from experimental point of view, flat background was often falsely detected as a moving object since random motion vectors appear in the flat background [56]. Previously the authors have proposed a method to detect moving object area on MPEG coded data domain [56] by analyzing the motion vectors and DCT coefficients. In P- and B- pictures, moving objects are detected by analyzing motion vectors and spatial-temporal correlation of motion. In addition, by analyzing coding characteristics of intra macroblocks (MBs) in P- and B-pictures and by investigating temporal motion continuity in I-pictures, moving objects in these situations have been also detected in intra MBs.

## 5.3 Object Detection and Extraction

We started by thresholding motion vectors before the detection process 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 are therefore selected.

### 5.3.1 Object Detection Algorithm

An object detection algorithm is used to detect potential objects in video shots. Initially, undesired motion vectors are eliminated. Subsequently, 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 the object detection algorithm. Fig. 5-1 shows the graphical user interface for the moving object detection module, as video clips can be traced and monitored while this module detects each object in real time.

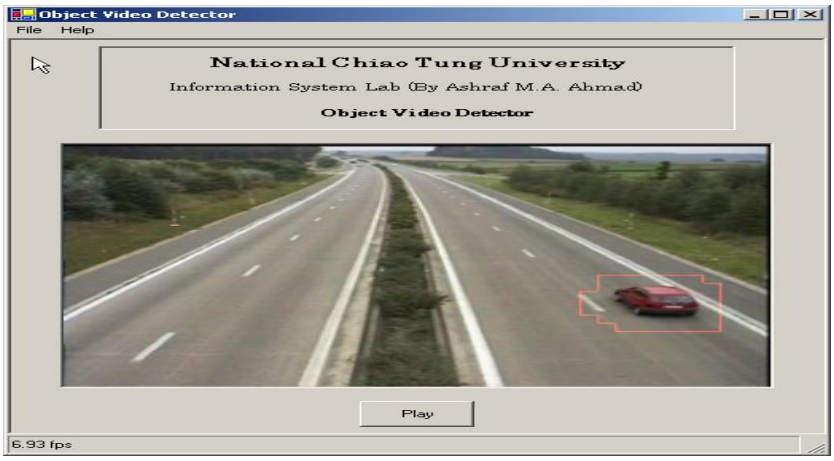



Fig. 5-1 Graphical user interface for object detection

**The Object Detection Algorithm**

**Input:** P-frames of a video clip

**Output:** object sets {$Obj_1$ , $Obj_2$ , … $Obj_N$ } where N is the total number of regions in P-frame and $Obj_N$ means the $N^{th}$ object of the P-frame. Each object size is measured in terms of number of macroblocks.

1. Cluster motion vectors that are of similar magnitude and direction into the same group with region growing approach.
   1.1 Set search windows (*W*) size 3×3 macroblocks.
   1.2 Search all macroblocks (*MB*) within *W*, and compute the difference ($diffMag_k$ and $diffAng_k$) of motion vector (MV) magnitude ($|MV|$) and direction ($\angle MV$) between center $MV_{center}$ and its neighboring eight motion vectors $MV_k$ within *W*.

1.2.1 diffMag = abs( $|MV_{center}|$ - $|MV_k|$ )

1.2.2 diffMag = abs( $\angle MV_{center}$ - $\angle MV_k|$ )

1.2.3 $EDGE_k$ = detect_edge($MB_k$) ,

*where* $k \in [1,8]$ and $MV_{center}$ *is the MV in the center position of W*

$MV_k \in$ *MVs within W except* $MV_{center}$

1.2.4 *For all* $1 \leqq k \leqq 8$, *flag*

$$F_k = \begin{cases} 1, & diffMag_k < T_{Mag} \text{ and } diffAng_k < T_{Ang} \text{ and } EDGE_k \\ 0, & otherwise \end{cases}$$

*where* $T_{Mag}$ *is the predefined threshold for MV magnitude and* $T_{Ang}$ *is the threshold for MV direction*

1.2.5 *If* $\sum_{k=1}^{8} F_k \geq 6$ , *mark* $F_{center}$ *of* $MV_{center}$ *as 1, where* $F_{center}$ *is the flag of the center MV within W.*

*Otherwise, set all flags within W to 0.*

1.3 Go to step 1.2 until all macroblocks are processed.

1.4 Group macroblocks that are marked as 1 into the same cluster.

1.5 Compute each object center and record its associated macroblocks.

## 5.3.2 Edge Detection using AC coefficients

We use the predefined two edge features [28] to derive edges. The two horizontal and vertical edge features can be formed by two dimensional DCT of a block.

In the DCT domain, the edge pattern of a block can be characterized with only one edge component, which is represented by projecting components in the vertical and horizontal directions, respectively. The edge features from the DCT basis images is

shown in Fig. 5-2. In order to extract the edge features the following conditions in Eq(5.1) and Eq(5.2) are re-checked.

$$\sum_{\mathbf{i}=0}^{7} \left| H_i \right| \geq Threshold\_Horizontal \qquad (5.1)$$

$$\sum_{j=0}^{7} \left| V_j \right| \geq Threshold \quad \_Vertical \qquad (5.2)$$

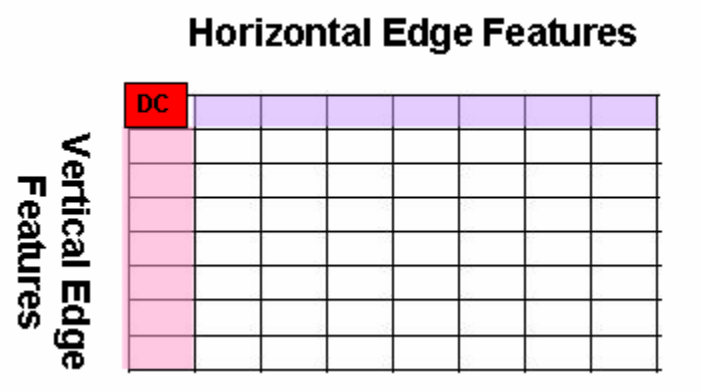



Fig. 5-2 Horizontal and vertical edge feature in DCT domain

If the test results for the Eq.(5.1) and Eq.(5.2) conditions are 'true' and 'false', respectively, it is defined that the block contains a vertical edge. For a horizontal edge in the block, the converse is true, *i.e.* the tests have to be 'false' and 'true', respectively. If both tests are 'true', the block contains a diagonal edge and it is further tested to determine its orientation using the polarities of the first coefficients: *H*1 and *V*1. That is, the coefficients have the same polarities (*V*1 & *H*1 = *positive*, or *V*1 & *H*1 = *negative*) for a 45-degree diagonal edge, and different polarities (*V*1 = *positive* and *H*1 = *negative*, or *V*1 = *negative* and *H*1 = *positive*) for a 135-degree diagonal edge. Therefore, we are now able to detect the edge block in each frame to decide is edge block or not.

## 5.4 Object detection overview

After all, we can introduce the whole object detection system including the motion vector refinement and our proposed video object detection. Then we use the proposed

object detection and edge information to finally detect the moving objects and its descriptors.

Fig. 5-4 states the structure of the extracted object descriptors. For instance an object descriptor can contain the appearance time field which state the time when the extracted object appears in the video sequence. It may contain the time length for the object being active in the video sequence. Velocity of the object can be added to object descriptor structure. Object shape attribute may be included in the object descriptor. This attribute gives the analyzer great sight regarding the information of the object shape.

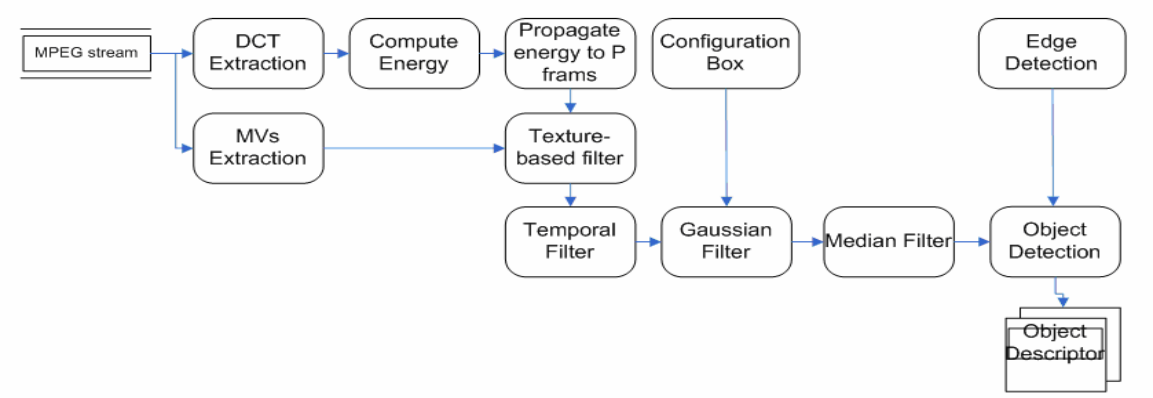



Fig. 5-3 Overview of the object detection system

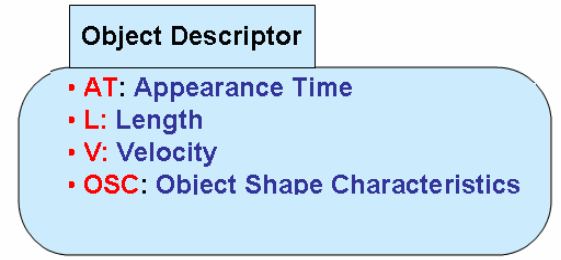



Fig. 5-4 Object descriptor structure

The experimental results show that each moving object is successfully identified for several sequences. Figures 5-5 through 5-7 show the detection results of using the filtered motion vectors and without using filtered motion vectors respectively. The detected objects are marked over with thicker and darker lines in the figures. Further

results are discussed in the result and discussion chapter.

Fig. 5-5 object detection results for Miss America

Fig. 5-6 object detection results for Speed way

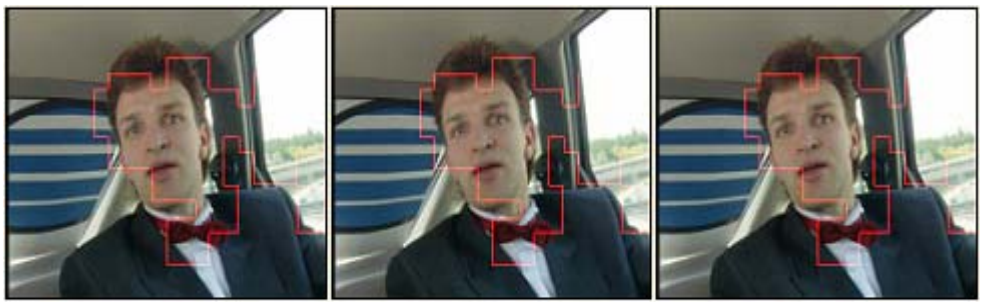


Fig. 5-7 object detection results for Car Phone

# Chapter 6

# Fast and Robust Object Extraction Framework for Object Based Streaming System

Video streaming poses significant technical challenges in the quality of service guarantee and efficient resource management. Generally, it is recognized that end-to-end quality requirements of video streaming application can be reasonably achieved only by integration of advanced networking and content processing techniques.

However, most existing integration techniques focus on 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 predict the bit-rate or quality more accurately. In the object based video streaming framework, video object is extracted automatically and used to control video quality under various manipulations and network resource requirements.

In this chapter, 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 utilize 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. 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, the entire implementation is operating in the compressed domain without need for decompression. Remarkable extraction performance is demonstrated in the experimental results.

## 6.1 Introduction

Video has been an essential element in communication and entertainment for many years. Initially video was captured and transmitted in analog shape. The emergence of digital integrated circuits and computers led to the digitization of video, and digital video enabled a revolution in the compression and communication of video. Video compression [71,72] and transmission became 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. 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 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" are increasing the overhead of running video over best-effort networks. 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, resulting in periods of transient congestion on the network. In regards to data transmitted over the internet, majority of the data bytes accessed on the Web are in the form of multimedia objects [64, 82].

Generally the most straightforward approach for video delivery in the Internet is using an approach similar to a file download, referred to as a video download. 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, since videos are 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 which lead us to long response time on the client part and 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 about the exact disk space on his machine, therefore he might start to download a large video file which takes a few hours, then an error message might pop up stating disk insufficiency. The user wasted hours but for nothing. These 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 capability in "viewing flexibility". In video streaming there is usually a short latency (usually 10-15 seconds) 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 Fig. 6-1. You may notice 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.

A number of basic problems afflict video streaming over the Internet as the Internet

only offers best effort service. Internet provides no guarantees on bandwidth, delay jitter, or loss rate. 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 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

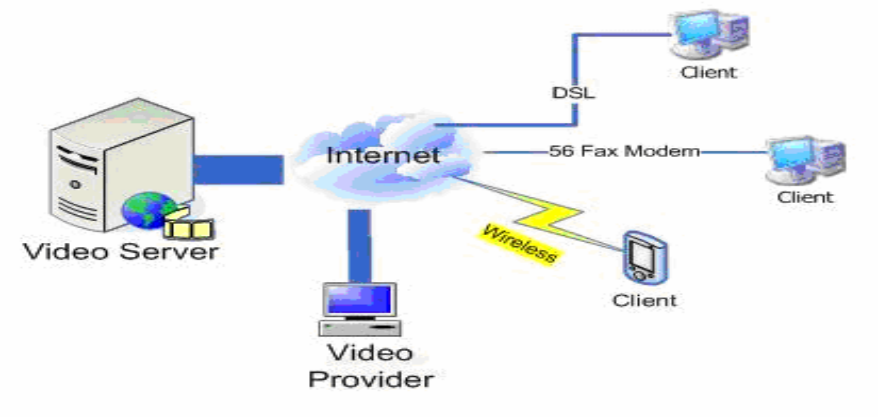



Fig. 6-1. General Video Streaming Architecture

. 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 concerned because the receiver must receive, decode and display frames at a constant rate. Any late frames resulting from the delay jitter can produce problems in the reconstructed video. These problems are 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. 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.

Many of the traditional video streaming systems, 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 correlation between video content, video data (bit rate) and quality of service. Such a framework facilitates new ways of quality modeling, and resource allocation in video streaming. For instance video objects can be used for controlling the video generation. It can be used in selecting the optimal transcoding architecture and content filtering.

The correlation between the video content and the traffic has been reported in [65-67] in which a conceptual model for content-aware based video streaming has been proposed. Closer schemes to our approach are joint source-channel coding [63], adaptive media scaling and resilience coding [64], and object and texture aware video

streaming [65-67]. The object based streaming system dynamically detects the objects in a video stream in real time, objects are 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, which contains a component to refine the motion vector through a sophisticated filter scheme in order to get fine motion vectors. Our approach offers an accurate and fast object based video streaming as proved by the conducted experiments.

## 6.2 Related Work

Researchers have developed an object based streaming system implemented in the intelligent transportation system [58]. Unfortunately, their approach was fully conducted in pixel domain and results in a higher computation time. Although they claim their approach is portable for different platforms, the portability issue was not defined. Scheme in [57] proposes object based adaptive streaming approach for sports videos, and is operated in the so called semi-compressed domain, as they need to partially refer to the pixel domain. However, typical samples in the motion vector field are usually inaccurate [34,61]. These defects can be combated with robust error recovery schemes that repair motion fields and reduce noise [59,61]. Consequently we can produce a smoother shape boundary, where the motion vectors are used to determine object boundaries in object extraction. This refinement mechanism has been described in Chapter 4.

## 6.3 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 control strategies have been built into the Transmission Control Protocol (TCP). For video traffic, TCP is not the 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 [77]. On the other hand, SVFTP [85] tries to exploit TCP protocol in order to deliver video content by setting up many TCP connections at the same time. Thus, video content delivering can be accomplished, but with 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 [83] and [84] 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 [69,84] may seriously degrade multimedia quality since the encoding mechanisms for multimedia generally bring in dependencies between frames [74]. For instance, in MPEG encoding [71,72] dropping an independently encoded frame will result in the following dependent frames being presented as useless since they cannot be displayed

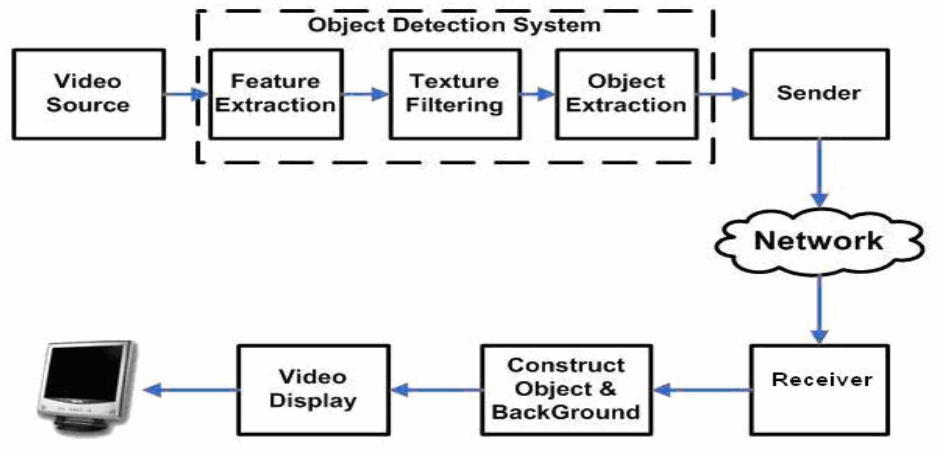
and would be better off being dropped, rather than occupying unnecessary bandwidth.

A multimedia application that is aware of these data dependencies can drop the least important frames much more efficiently than can the router [65,66,78]. 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 to the network conditions after studying the video content. The estimator part is clearly stated in the literatures of computer networks [79,81,89]. 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 in the object based video streaming systems can be broadly categorized as follows [68,71,76]:

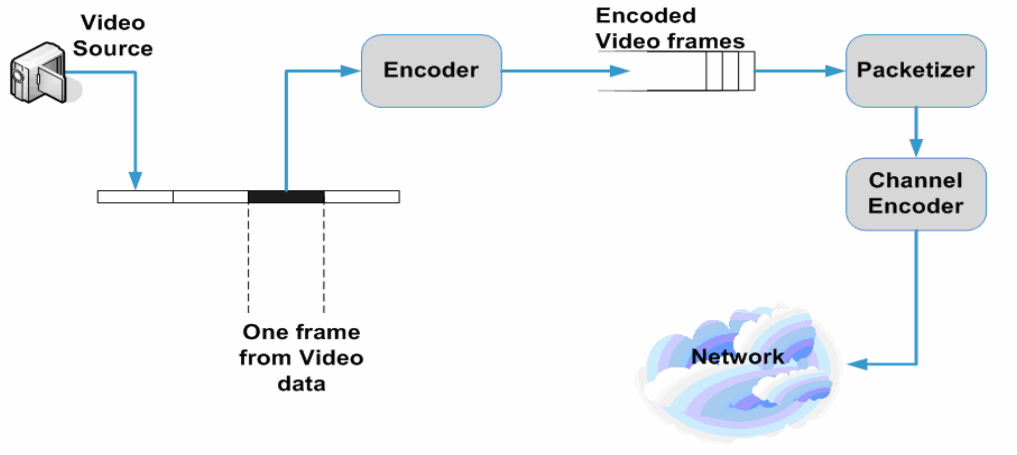
1. Spatial scaling: In spatial scaling, the size of the frames is reduced by transmitting fewer pixels thereby reducing the level of detail in the frame.
2. Temporal scaling: In temporal scaling, the application drops frames. The order in which the frames are dropped depends upon the relative importance 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 dropped first.
3. Quality scaling: In quality scaling, the quantization 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 colors and details.

In summary, it has been shown that the content of the video stream can be an important factor in influencing the choice of the scaling scheme [65-68,71].

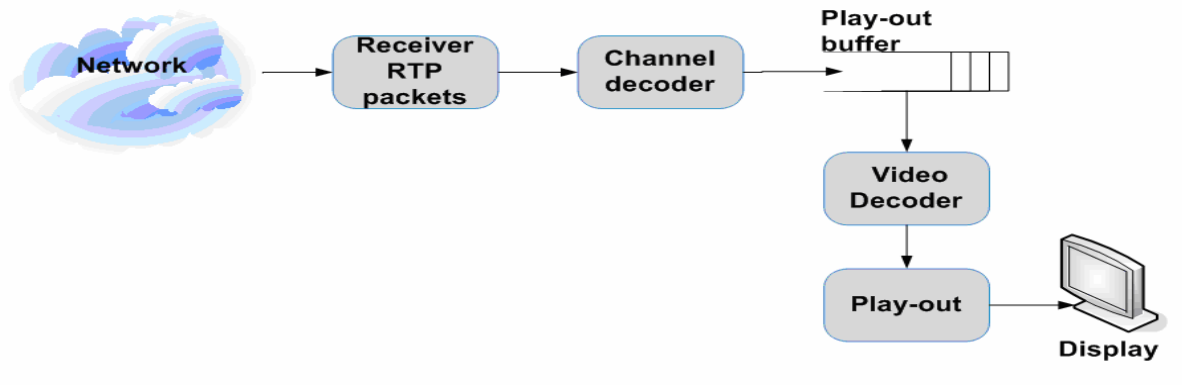
In sum, the object-based video streaming technique is designed as stated in Fig. 6-2. Our proposed object detection is in the block surrounded by dashed lines. Initially, we capture video stream in MPEG format, extract features, detect moving objects and transmit their encoded streams. Fig. 6-2(a) illustrates the streaming system architecture, which covers four key modules, including the Object Extraction, Sender, Receiver and Composer. Fig. 6-2(b) presents a typical RTP sender side. Fig. 6-2(c) presents a typical RTP receiver side.




(a)




(b)




(c)

Fig. 6-2: (a) Streaming System Architecture, (b) Block Diagram of RTP Sender and (c) Block Diagram of RTP Receiver

We are going to discuss different approaches relevant to object based video streaming. A fine grained, content-based, packet forwarding mechanism [75] 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. [75] proposed mechanisms for queue management and weighted fair queuing to provide the differentiated forwarding of packets with high priorities.

A basic mechanism that uses temporal scaling for MPEG streams is suggested in [73]. In case of network condition change, the frame rate is reduced by dropping frames in a predefined precedence (first B-frames and then P-frames) until the lowest frame rate, (only the I-frames) or minimum bandwidth. An adaptive MPEG streaming player based on similar techniques [76] has the capabilities for dynamic rate adaptation but does not support real-time, automatic content detection. Automatic adaptive content-based scaling may significantly improve the perceptual quality of the played out streams.

If a movie scene has little motion and need to be scaled, it would look better if a few

frames were dropped but the frames shown were of high quality [68]. Relevant approach [80] has developed a scaling mechanism for video applications capable of scaling video streams.

Using these scaling mechanisms, it is possible to change the characteristics of video streams by dropping frames, dropping colors, changing the quantization levels. [68,71] utilize these scaling mechanisms in conjunction with a real-time content analyzer that measures the motion in an MPEG stream in order to implement a content- aware scaling system. [68] conducts a pilot study on video scaling where the subjects rate of video clips that are first scaled temporally and then by quality in order to establish the optimal mechanism for scaling a particular stream. They find the content aware system can improve perceptual quality of video by as much as 50%.

Various mechanisms have been proposed for video protocols to respond to network condition changes on the Internet [68,82]. Their mechanism is equation-based network condition changes control for unicast traffic. Unlike TCP, the control refrains from reducing the sending rate in half in response to a single packet-loss.

A TCP-friendly protocol [79] was implemented and evaluated for fairness in bandwidth distribution. [89] is 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.

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 [88] with a layered transmission system [87]. By selectively forwarding subsets of layers at constrained network links, each user may receive the best quality signal that the network can deliver. The receiver-driven layered multicast

scheme suggested that 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 signaling that is needed for hosts to subscribe or unsubscribe from multicast groups. Also fairness issues exist 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 [86]. However other TCP algorithms, like retransmissions of dropped packets. that are detrimental to real time multimedia applications have not been incorporated.

## 6.4 Object Extraction System

Our system takes the motion vectors from the compressed video stream as input. Besides, we need to extract the DCT information from I-frames. This information is readily available in MPEG stream, thus not too much time is spent in decoding the MPEG stream. Hence, our approach fits for the real-time application environment. Now, we will present the following diagram which states an abstract overview of our object extraction proposed system, which is similar to the one described in Chapter 5. Fig. 6-3 shows the proposed system architecture. Next, we extract the motion vectors from P-frames 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.

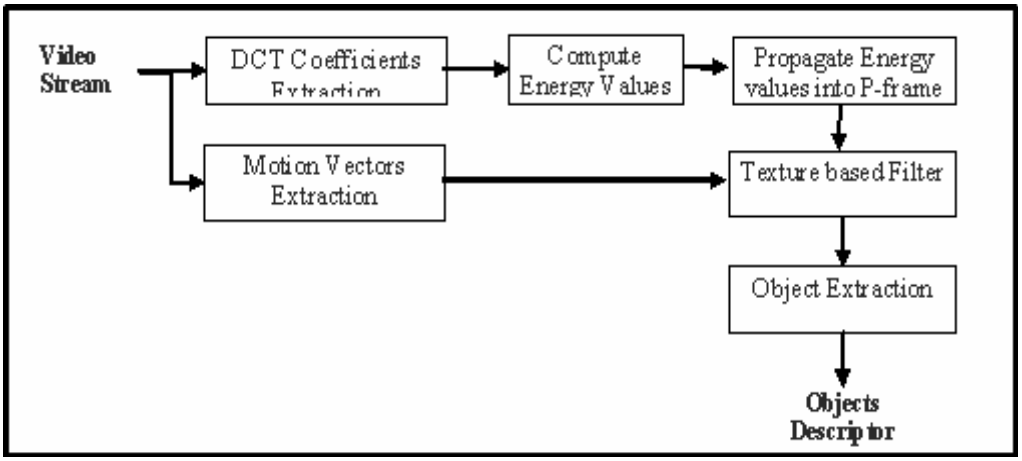



Fig. 6-3 The Proposed Object Extraction System Overview.

## 6.5 RTP Sender and Receiver

To link the sender, receiver and transmission channel for real time video displaying sequences, a standard RTP based network streaming technique is used. The streaming module uses two categories of network protocols 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, 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 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 decoder may be used to generate error correction packets or to reorder 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 synchronization. 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 codec 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.

## 6.6 Experimental Results and Discussion

In order to verify the performance of the proposed scheme, the experiment has been designed to test 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 [62], group **B** using Median filter [61], Group **C** using Cascade Filter , group **D** our system, and finally without any kind of post processing.

In order to compare the performance among these four systems, we fix the configurations of 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* to evaluate object extraction system performance [41,60].

Figures 6-4 through 6-6 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. Figures 6-4 through 6-5 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 Fig. 6-6.

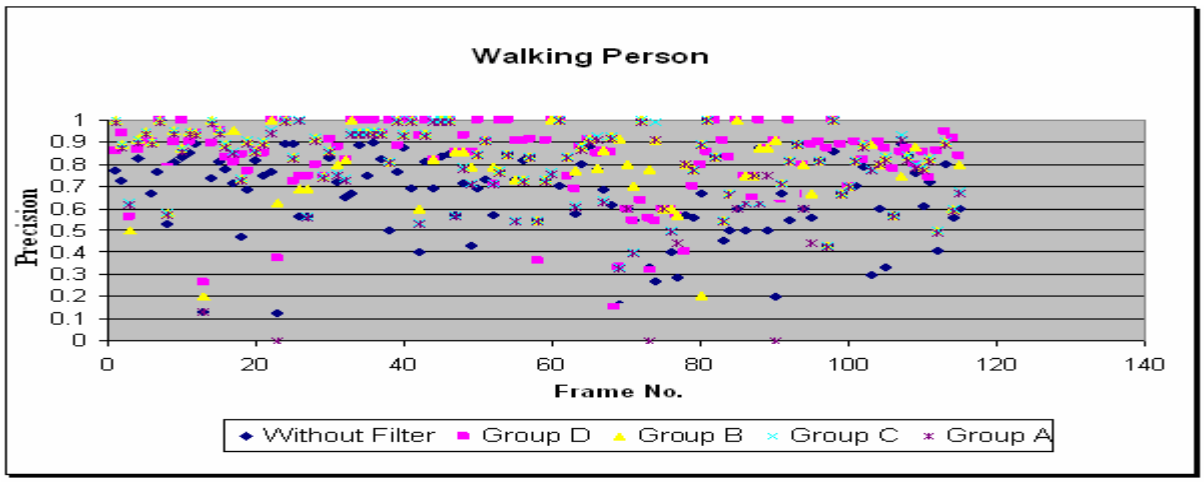



Fig. 6-4: Precision for Object extraction in P frames

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.

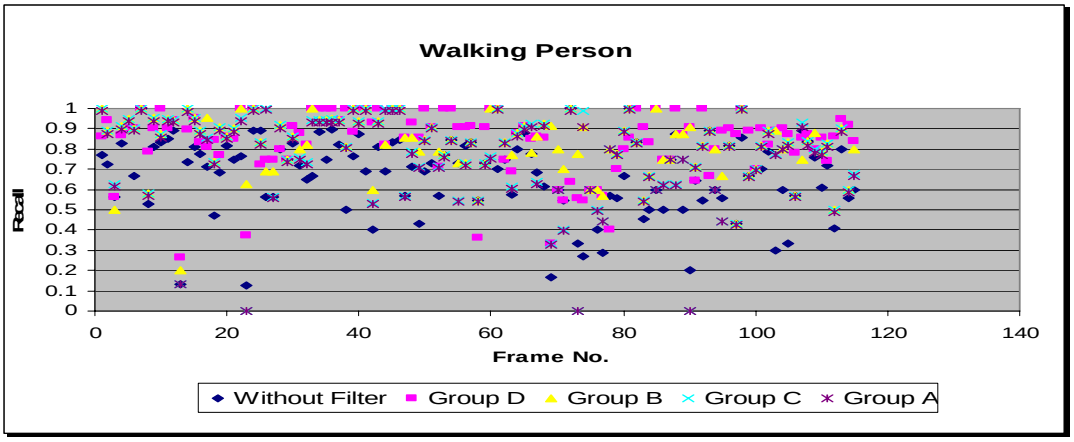



Fig. 6-5: Recall for Object extraction in P frames

In order to verify the previously 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 walking person testing dataset. The experimental result is presented in Table 6-1. We show the run time in milliseconds (ms) for the object extraction module using our system, other related work, and without any post preprocessing. Also, we describe the I/O time needed to perform the object extraction task.

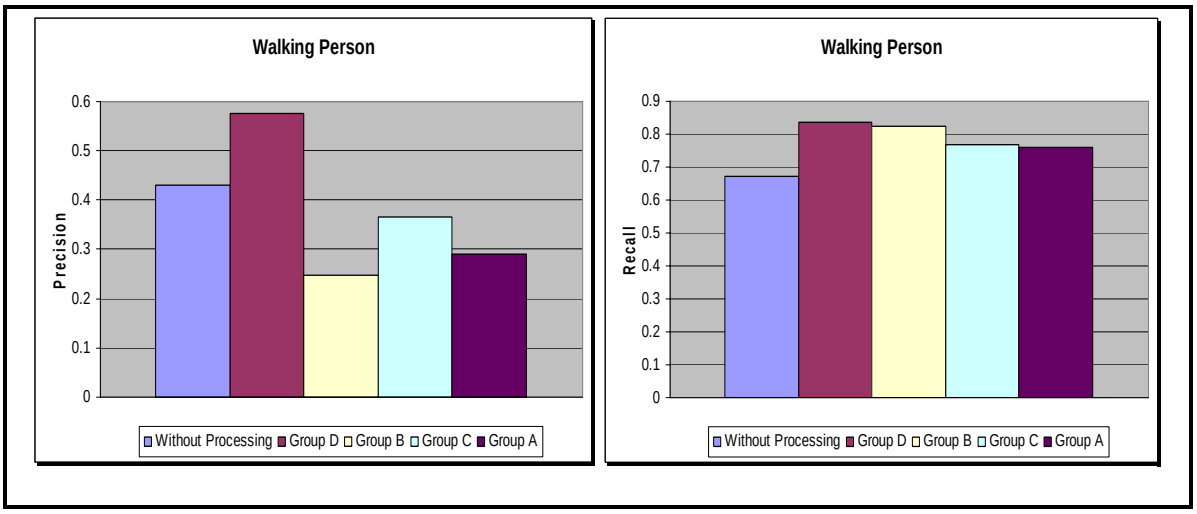



Fig. 6-6: Average Precision and Recall of object extraction for 2nd Video Clip of walking person sequences

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 comes compatibly with our expectation and initial idea.

Table 6-1: Run time for object extraction in millisecond

| | *Group A | †Group B | ‡Group C | •Group D | Without Post Processing |
|---|---|---|---|---|---|
| Object Extraction Run time | 585 ms. | 579 ms. | 575 ms. | 545 ms. | 605 ms. |
| I/O processing | 104 ms. | 96 ms. | 89 ms. | 70 ms. | 124 ms. |
| Total Time | 689 ms. | 675 ms. | 664 ms. | 615 ms. | 729 ms. |

*Group A using Gaussian filter only [62] †Group B using Median filter [61]

‡Group C using Cascade Filter •Group D the proposed system

## 6.7 Future Work

In this section, we will describe future work for both the deployment of the proposed approach and enhancements possible. Several future applications can be anticipated based on our proposed approach. First of all, Application Level Gateway 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. Secondly, mobile video communication is 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. Online video broadcasting such as game broadcasting, news broadcasting and so forth, have real-time processing and display requirements.

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. For example, game players can be decided as moving objects and court areas as static scene along the streaming presentation time. Fourth application is video conferencing. As video conferencing applications have several known presumptions, such that we have background area and people talking in the foreground area. Talking heads as moving objects can be separated from the background area.

Also our approach will fit for the real-time requirement 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.

## 6.8 Conclusions

In this chapter, 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 video streaming 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 object based video streaming system. We believe object based video streaming is a very promising field for both video and communication societies. Still there is a lot of room for investigation and development.

# Chapter 7

# A Novel Approach for Improving the Quality of Service for Wireless Video Transcoding

To deliver streaming video over wireless networks is an important component for most interactive multimedia applications running on wireless handset devices. Such devices should be inexpensive, compact, and lightweight. Wireless channels have limited bandwidth and a high channel bit error rate. Delay variation of packets due to network congestion with the high bit error rate lessens the quality of video at the handheld devices.

Therefore, mobile access to multimedia content requires video transcoding functionality at the edge of the mobile network for interworking with heterogeneous networks and services. Under certain conditions, the bandwidth of a coded video stream needs to be drastically reduced.

We propose a cost-efficient mechanism for improving the quality of service (QoS) delivered to the mobile user, by introducing a robust and efficient transcoding scheme as proven by extensive experiments. The proposed approach refines the motion vectors value without the need to re-perform the motion estimation process. Then the transcoding mechanism will be preformed using the new fine motion vectors. Thus, great amounts of computing resources are saved. To verify and prove the robustness of the proposed approach, experimental results demonstrate the exceptional performance.

## 7.1 Introduction

Recent advances in mobile communications and portable client devices enable us to access multimedia content ubiquitously. However, when multimedia content becomes

richer, including video and audio, it becomes more difficult for wireless access due to many practical restrictions. Most important of all, wireless connections usually have a much lower bandwidth compared to wired ones and communication conditions change dynamically due to the effect of fading [90]. Another practical factor is that portable client devices equipped with limited computing and display capabilities.

Most portable devices are not suitable for high quality video decoding and displaying. Concerning the heterogeneity issue, the previous era has seen a variety of developments in the area of multimedia representation and communication. In particular, we are beginning to see delivery of various multimedia data for all types of users and conditions. In a diverse and heterogeneous world, the delivery path for multimedia content to a multimedia terminal is not straightforward especially in the mobile communication environment. Access networks vary in nature, sometimes limited, and differ in performance.

The characteristics of end user devices vary increasingly, in terms of storage, processing capabilities, and display qualities, also the natural environment, e.g., position, elucidation or temperature changes. Finally, users are different by nature, showing dissimilar preferences, special usage, disabilities, etc. However, the major traffic component in multimedia services is undoubtedly due to visual information encoded and delivered either as video frames or visual components [91]. In order to cope with the current heterogeneous communication infrastructure and the diversity of services and user terminals, different transcoding mechanisms are necessary at internet working nodes [92,93]. Whenever a client terminal or its access channel does not comply with the necessary requirements, media transcoding must be triggered to allow interoperability. This is basically an adaptation function operating on coded streams such as MPEG1/2 [94]

for matching a set of new constraints, different from those assumed when the signals were originally encoded. Since many multimedia services are not specifically meant for mobile systems, in general the channel bandwidth required for transmission, as well as the coded signal format, do not match mobile applications [95-98]. Because of traffic characteristics such as high bit rate, video will be the dominant traffic in multimedia streams, hence it needs to be managed efficiently.

Obviously for efficient utilization of network resources, video must be compressed to reduce its bandwidth requirement. A wireless handset device, for instance personal data assistant, can integrate voice, video, and data in one device. In contrast to solely text information, multimedia data can tolerate a certain level of error and fading. Therefore, although a wireless network has a high bit error rate when compared to a wireline one, it is possible to be in cost effective manner transmit multimedia over wireless networks with an acceptable quality [99].

As mentioned earlier, although the constraints imposed by the heterogeneous nature of the communication network are quite different from those arising from the diversity of user terminals and the problem of fading and error in wireless channels, all of them may be dealt with the transcoding mechanism. In this work we address the problem of MPEG stream video transcoding, where the bandwidth of a coded video stream must be drastically reduced in order to cope with a highly constrained transmission channel.

Particularly, we work on the MPEG-2 compressed digital video content, as MPEG-2 is being used in a number of products including the DVDs, camcorders, digital TV, and HDTV. In addition, tons of MPEG2 data has already been stored in different accessible multimedia servers. The ability to access this widely available MPEG-2 content on low-power end-user devices such as PDAs and mobile phones depends on effective

techniques for transcoding the content to a more appropriate, low bit-rate video.

In the subject where video is concerned, transcoding may be needed to convert pre-coded high quality videos into lower quality ones to display on handheld devices. Video transcoding deals with converting a previously compressed video signal into another one with a different format, such as different bit rate, frame rate, frame size, or even compression standard. Diversity of multimedia applications and the present communication infrastructure comprise of different underlying networks and protocols.

Therefore, there is a growing need for inter-network multimedia communications over heterogeneous networks and devices. Especially in applications where pre–encoded videos are spread to users through different connections, such as video on demand or streaming of pre-encoded videos, the end transmission channel conditions are generally unknown when the video is originally encoded. By means of transcoding, pre-encoded videos can be converted on the fly as they are transmitted.

Similar to source encoders, video transcoders can modulate the data they produce by adjusting a number of parameters, including quality, frame rate, or resolution. However, using transcoders gives us another chance to dynamically adjust the video format according to channel bandwidth and end devices. This is particularly useful when there are time variations in the channel characteristics.

## 7.2 Related Work

Converting a previously compressed video bit stream to a lower bit rate through transcoding can provide finer and more dynamic adjustments of the bit rate of the coded video bit stream to meet various channel situations [18-24]. Depending on the particular strategy that is adopted, the transcoder attempts to satisfy network conditions or user requirements in various ways.

In the context of video transmission, compression standards are needed to reduce the amount of bandwidth. Since the delivery system must accommodate various transmission and load constraints, it is sometimes necessary to further convert the already compressed bitstream before transmission. Depending on these constraints, conventional transcoding techniques can be classified into three major categories: bit-rate conversion or scaling, resolution conversion and syntax conversion [21,100].

Bit-rate scaling can accommodate deficiency in available bandwidth. Resolution conversion [18,19,96] can also accommodate bandwidth limitations, but is primarily used to account for limitations in the user devices, such as processing power, display constraints or memory capability. To ensure adaptability across hybrid networks, syntax conversion [101,102] at the protocol layer is required. Syntax conversions may also be considered at the compression layer to ensure receiver compatibility.

The simplest way to develop a video transcoder is by directly cascading a source video decoder with a destination video encoder, known as the cascaded pixel domain transcoder [102]. Without using common information, this direct approach needs to fully decode input video and re-encode the decoded video by an encoder with different characteristics as described in Fig. 7-1. Obviously, this direct approach is usually computationally intensive. The architecture is flexible, because the compressed video is first decoded into raw pixels, hence a lot of operations can be performed on the decoded video.

However, as we mentioned earlier, the direct implementation of the Cascaded Pixel Domain Transcoder is not desirable because it requires high complexity of implementation.

The alternative architecture for transcoding is an open-loop transcoding in which the

incoming bitrate is downscaled by modifying the discrete cosine transform (DCT) coefficients. For example, the DCT coefficients can be truncated, requantized, or partially discarded in the optimal sense [18,102] to achieve the desirable lower bitrate. In the open-loop transcoding, because the transcoding is carried out in the coded domain where complete decoding and re-encoding are not required, it is possible to construct a simple and fast transcoder.

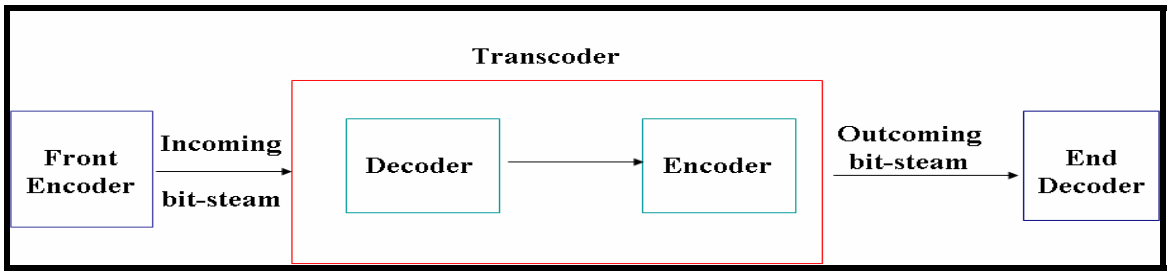



Fig. 7-1. A typical transcoder architecture

However, open-loop transcoding can produce "drift" degradations due to mismatched reconstructed pictures in the front-encoder and the end-decoder, which often result in an unacceptable video quality. Drift-free transcoding is possible by the direct cascade of a decoder and an encoder. Although this transcoder has a higher complexity than the open-loop transcoder, some information extracted from the incoming video bit stream after the decoding can be used to significantly reduce the complexity of the encoder.

Thus, the complexity may not be as bad as it looks. In transcoding, full motion estimation is usually not performed in the transcoder because of its computational complexity. Instead, motion vectors extracted from the incoming bit stream are reused.

Since a great deal of bit rate reduction is required, traditional transcoding methods based on simply reusing the motion vectors extracted from an incoming video bit stream [104,105,106] are not adequate. Fig. 7-2 states the basic scheme of these approaches. They would produce an unacceptable texture distortion in the reconstructed signals. Although an optimized motion vector can be obtained by full-scale motion estimation

[107,108], this is not desirable because of its high computational complexity.

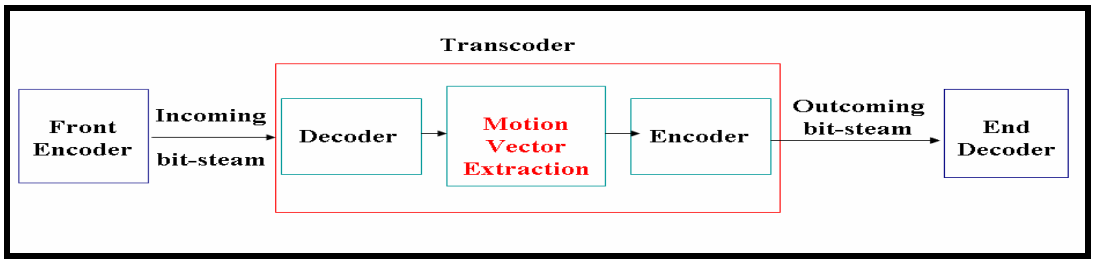



Fig. 7-2. General scheme for transcoding video based on motion vector reuse

Another related work [109], proposed to partially estimate the motion vectors based on predefined search area. Their constructed images quality was relatively good. The performance of video transcoding was boosted, but on the other hand they still have high computation complexity as they need to perform motion estimation even partially. For further description, basic scheme of there proposed approach is drawn in Fig. 7-3.

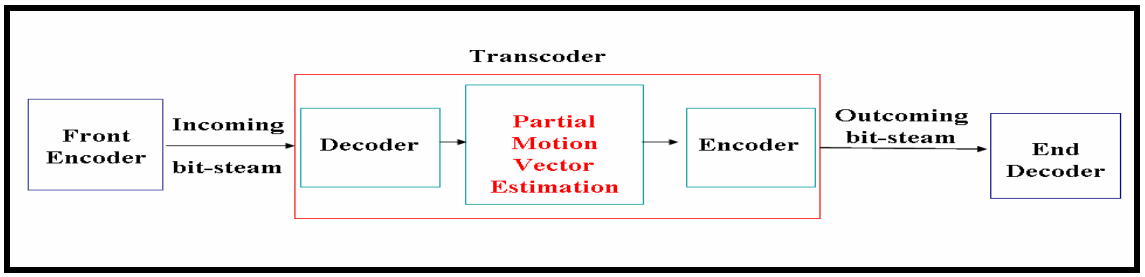



Fig. 7-3. Video transcoding scheme based on partial motion vector estimation

In this chapter, we consider the cascaded architecture as the framework for high-performance transcoding. The cascaded transcoder is very flexible and easily extendible to various types of transcoding, such as temporal or spatial resolution conversions. We will investigate techniques which can reduce the complexity while maintaining the same level of video quality. In transcoding, motion estimation is usually not performed in the transcoder because of its computational complexity. Instead, motion vectors extracted from the incoming bit stream are reused.

However, this simple motion-vector reuse- scheme may introduce considerable quality degradation in many applications [106,109,110]. Although an optimized motion

vector can be obtained by full-scale motion estimation, this is not desirable because of its high computational complexity.

Therefore, in this chapter, we propose a new motion vector refinement scheme for a transcoder using the MPEG1/2 [94] encoded bit streams. We propose an adaptive Gaussian motion vector refinement scheme that is capable of providing comparable video quality to those which can be achieved by performing a new full-scale motion estimation or partial-scale motion estimation. Our scheme requires considerably less computation. The reuse of incoming motion vectors has been widely accepted because it is almost as good as performing a new full scale motion estimation and was assumed in many transcoder architectures [21- 23].

However, as proved in [106,109,110], simply reusing the incoming motion vectors is not optimal. Their simulation results show that its performance may be considerably worse than the one which can be achieved with new motion estimation. [109] showed that the differential reconstruction error causes incoming motion vectors to deviate from optimal values. In most macroblocks the deviation is within a small range and the position of the optimal motion vector will be close to that of the incoming motion vector. These defects can be combated with the robust motion vector refinement scheme that repairs motion fields. Subsequently, the refined motion vectors can be reused in the transcoder side efficiently.

## 7.3 Mobile and Wireless Video Transcoding System Architecture

In this section, a full description for video transcoding in mobile and wireless network will be presented. The proposed transcoding scheme is drawn in Fig. 7-4. In this figure,

we capture video stream in MPEG format from the front encoder. Front encoder is to be transcoded video source encoder.

Usually this encoder will be located on the content provider or server side. The first part of transcoder is typical decoder. The second component is a module in charge of extracting motion vector after the decoding process has been started in the transcoder. The third module is motion vectors refining. The function of these modules is to refine motion vectors. Finally, transcoder encodes the decoded images using the refined motion vectors according to the transcoding parameters. Then transcoded video data will be transmitted to end users or clients. End users are supposed to have end decoder to be able to render new video.

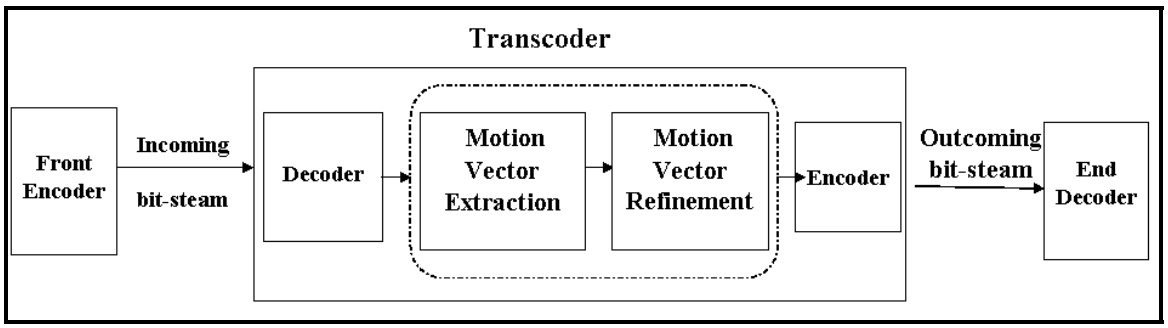



Fig. 7-4. The proposed video transcoding system based on refined motion vectors

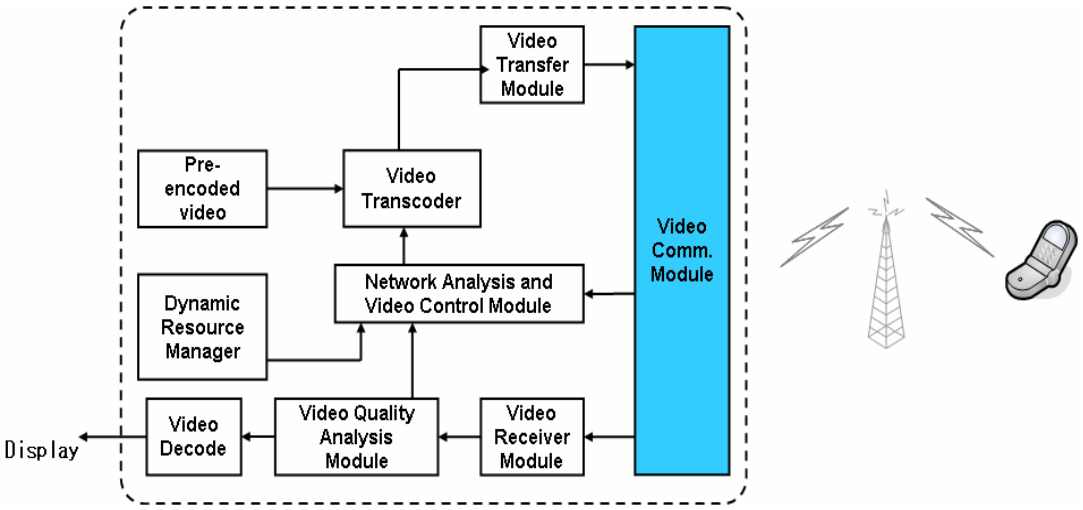



Fig. 7-5. Overview of Mobile and Wireless Video Transcoding System Architecture

In order to enable users to access video through wireless network and handheld devices, we propose using a transcoder as an intermediate node to dynamically convert the video according to user devices and network connections. In this approach, a content provider provides only one video stream, which is pre-encoded at high quality. The video transcoding performs transcoding dynamically for each user and provides converted video streams.

The architecture of the video transcoding system is illustrated in Fig. 7-5. The proposed system contains four important parts: user profile, video transcoder, control module and pre-encoded video content. In this system, the transcoder is integrated into the video streaming server. It can also be placed as an intermediate node along the transmission path. The user profile maintains several profile modules. The handheld device profile includes hardware and software information on the devices, such as display size, processing power, storage capability and decoder information. The user preference profile includes user preference information such as user preferred display size, user preferred video presentation and user preferred video player behavior.

The communication profile will measure the download throughput and update the communication conditions. The information included in the device profile and user profile will be transmitted to the video stream server before the start of the video transmission session. The information included in the transmission profile will be sent to the stream server periodically to control the bit rate adaptation in the transcoder. All information in the user profile is used for parameters in the transcoding process. Table 7-1 is a clear example for user profile information.

The video transcoder is the actual conversion engine of a video stream. It decodes a video stream, which is pre-encoded at high quality and stored in the video source, and

then performs transcoding according to our proposed scheme. According to the result presented in section 6 in this chapter, our proposed scheme has a very high performance in terms of visual quality. They are comparable to results which can be achieved by full-scale motion estimation based transcoding. But our proposed scheme outperforms the full-scale motion estimation based transcoding in terms of processing speed as proven by experiment results. When fast transcoding architectures are used, it is possible to execute transcoding in real time. Thus we can provide the handheld device user a smooth, online video presentation.

Table 7-1: User profile Information used by control module

| Frame rate |
|---|
| Pixel size |
| Monitor resolution |
| Graphic engine capability |
| CPU Usage |
| Memory usage |
| Visual objects summary |
| Visual combination report |
| Audio object summary |
| Scene description level |

The control module is responsible for creating a transcoding scheme according to the user profile Table 7-1 and other information. In order to decide appropriate transcoding parameters, decisions must be made by considering all of the factors adaptively. For example, when connection throughput is low, the bit rate of the video needs to be converted. At the same time, in order to ensure video quality, the frame rate of the video also needs to be reduced. This way, each frame will have enough bit budgets to maintain tolerable visual quality.

## 7.4 Motion Estimation in Transcoding

Current video compression techniques [94] exploit mainly two types of redundancies in the uncompressed video signal to achieve the desired compression gain. First, preserving only significant DCT coefficients can considerably eliminate the spatial redundancy between pixels within a single frame because of the energy compaction property of the DCT.

Furthermore, the motion-compensated predictive coding scheme is used to remove the temporal redundancy between frames. In other words, a motion-compensated block in the previous reconstructed reference frame is subtracted from the current macroblock. The residual signal is encoded using DCT to further remove the spatial redundancy. To find the motion vector for a macroblock in the current frame, a best matching macroblock is searched within a predefined search window in the previous reconstructed reference frame. The motion vector is defined as the displacement of the best matching block from the position of current macroblock. According to the aforementioned description for video compression techniques, cascaded pixel domain transcoder can be presented as in Fig. 7-6.

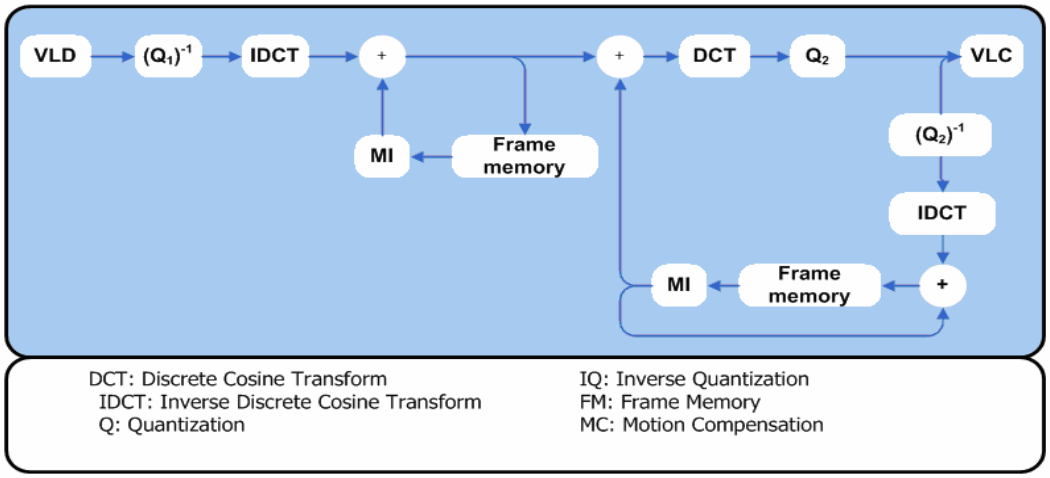



Fig. 7-6. Cascaded pixel domain transcoder

This transcoder will receive the encoded bitstream from the front encoder then decode the bitstream according to the encoding parameters provided from the front encoder. One of the most important parameters is quantization step 'Q1' used by the inverse quantizer in the transcoder. Then the transcoder will encode the decoded images according to a new encoding profile, for example new quantization step 'Q2.' Since the output bitrate is lower than the input bitrate, naturally, the quantization step size in Q2 in the transcoder is usually much coarser than the quantizer step size in Q1 in the front encoder. According to [109], there are non-zero probabilities that the quantization errors may cause the incoming motion vector to be non optimal i.e., a better motion vector can be found.

## 7.5 Motion Vector Refinement

Although the optimized motion vector can be obtained through new motion estimation, it is not desirable because of its high computational complexity. The reuse of the incoming motion vectors has been widely accepted because it was generally thought to be almost as good as performing a new full-scale motion estimation and was assumed in many transcoder architectures [21-23].

Fig. 7-7 shows the performance of motion vector refinement. The quality degradation introduced by reusing incoming motion vectors is about 0.40 dB on average comparing the full-scale search motion estimation with our approach. However, refinement of incoming motion vectors using the proposed motion vector refinement increases the performance close to that of the full-scale and partial-scale search motion estimation. Detailed simulation environment and coding parameters used in the simulations are described in the results section.

The subjective effectiveness of the spatial noise filter is therefore increased, by taking the noise weighting curve into account. Table 7-2 provides some subjective results of applying motion vector refinement scheme in transcoding against the simple reuse of motion vectors and no transcoding. It is evident from these frames that the motion vector refinement process eliminates a significant amount of noise in the reconstructed output.

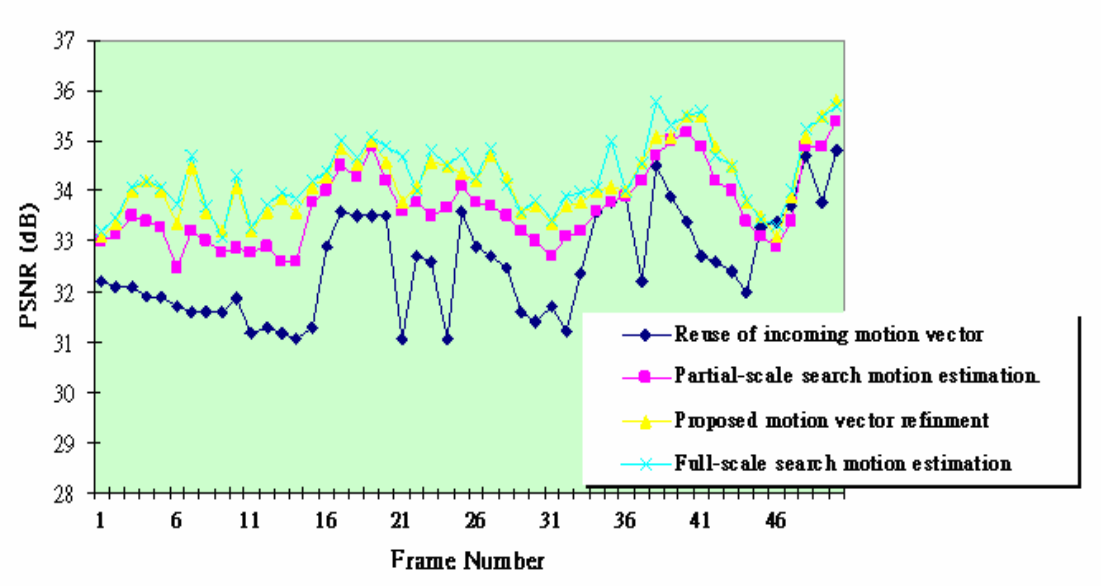



Fig. 7-7. Performance of motion vector refinement ("Claire of QCIF format. Process 50 frames) Incoming but rate at 128 Kb/s was transcoded t0 32 Kb/s with 30 frames/s.

Table 7-2 subjective results of applying motion vector refinement on Carphone Sequence

| **Frame number** | **Original Image** | **Transcoded by reuse motion vectors** | **Transcoded by the proposed motion vector scheme** |
|---|---|---|---|
| **Frame No. 59** |  |  |  |

## 7.6 Result and Discussion

We have designed an experiment to test the proposed scheme on several video clips in order to verify the performance. These video clips are part of the MPEG7 testing dataset. In all the simulations presented in this chapter, test sequences of QCIF (176X144) were encoded at high bitrate using a fixed quantization parameter.

At the front-encoder, the first frame was encoded as an intraframe (I-frame), and the remaining frames were encoded as interframes (P-frames). These picture-coding modes were preserved during the transcoding. In our simulations, bidirectional predicted frames (B-frames) were not considered, as it is sufficient to use the motion information of P-frames only in our experiment design.

Comparison is held among four types of work: transcoding scheme using full-scale motion estimation (FSME), using proposed motion vector refinement (PMVR), using re-use motion vector (RUMV) and finally using partial-scale motion estimation (PSME). The peak signal-to-noise ratio (PSNR) measures the video quality, because it is most commonly used to evaluate such system performance [23,24,109,111].

Assume we are given a source image f(x, y) that contains N by N pixels and a reconstructed image $\tilde{f}(x,y)$ where $\tilde{f}$ is reconstructed by decoding the encoded version of $f(x,y)$. First, we compute the mean squared error (MSE) of the reconstructed image as in Eq.(7.1).

$$MSE = \frac{\sum \left[ f(x,y) - \tilde{f}(x,y) \right]^2}{N^2} \tag{7.1}$$

Based on mean-square error (MSE), the PSNR for each colour component (Y, Cb, Cr) luminance and chrominance component is calculated separately. The equation for PSNR calculation in decibels (dB) is computed as in Eq.(7.2).

$$PSNR = 20 \log_{10}\left(\frac{255^2}{\sqrt{MSE}}\right) \tag{7.2}$$

In our result we present the PSNR values for Y components due to the space and readability. As we are emphasizing the processing performance and its importance to the handheld devices, we measure the speed-up according to Eq.(7.3)

$$Speed-up = \frac{Excution-Time(Y)}{Excution-Time(x)} \tag{7.3}$$

For comprehensive description sake, we provide simulation environment Table 7-3. The CPU, memory and hard-disk capability are listed under Hardware. The operating system and programming environment are listed under Software.

Table 7-3. Simulation Environment Hardware and Software Component

| **Hardware** | | | **Software** | |
|---|---|---|---|---|
| **CPU** | **Memory** | **Hard Disk** | **Operating System** | **Programming Language** |
| Pentium-3 1GHz | 256 SDRAM | 10025 RPM | Windows XP professional | Visual Studio 2005 |

Fig. 7-8 and Fig. 7-9 show the simulation results of different motion vector refinement schemes at different frame-rates for "Mother and daughter" and "Coast guard" sequences. The performances of the FSME transcoding, PMVR transcoding, RUMV transcoding and transcoding using PSME were compared. Based on our simulations, PMVR has about the same performance as FSME and PSME but requires less computation. The proposed adaptive refinement scheme has eliminated most of the untrue incoming motion vectors were this elimination process has computational savings which are significant and demonstrate the effectiveness of the proposed adaptive refinement scheme.

According to the presented results, one can infer the advantages of deploying our scheme in described applications in both wireless platform and future work. The results in Fig. 7-8 are conducted on mother and daughter video clip. The PSNR values will range from frame to frame based on the motion vector refining. Our approach is gaining in average around 1 to 2 DBs, while keeping remarkable resources utilization as proved in the rest of this section. Similarly, the presented results in Fig. 7-9 state the clear advantage from deploying our scheme in video transcoding for mobile and wireless networks. The performances in terms of processing speed are summarized in Table 7-4, and Table 7-5 as the performances of the FSME, PMVR, RUMV and PSME transcoding were compared. According to Eq.(7.3), speed up is computed.

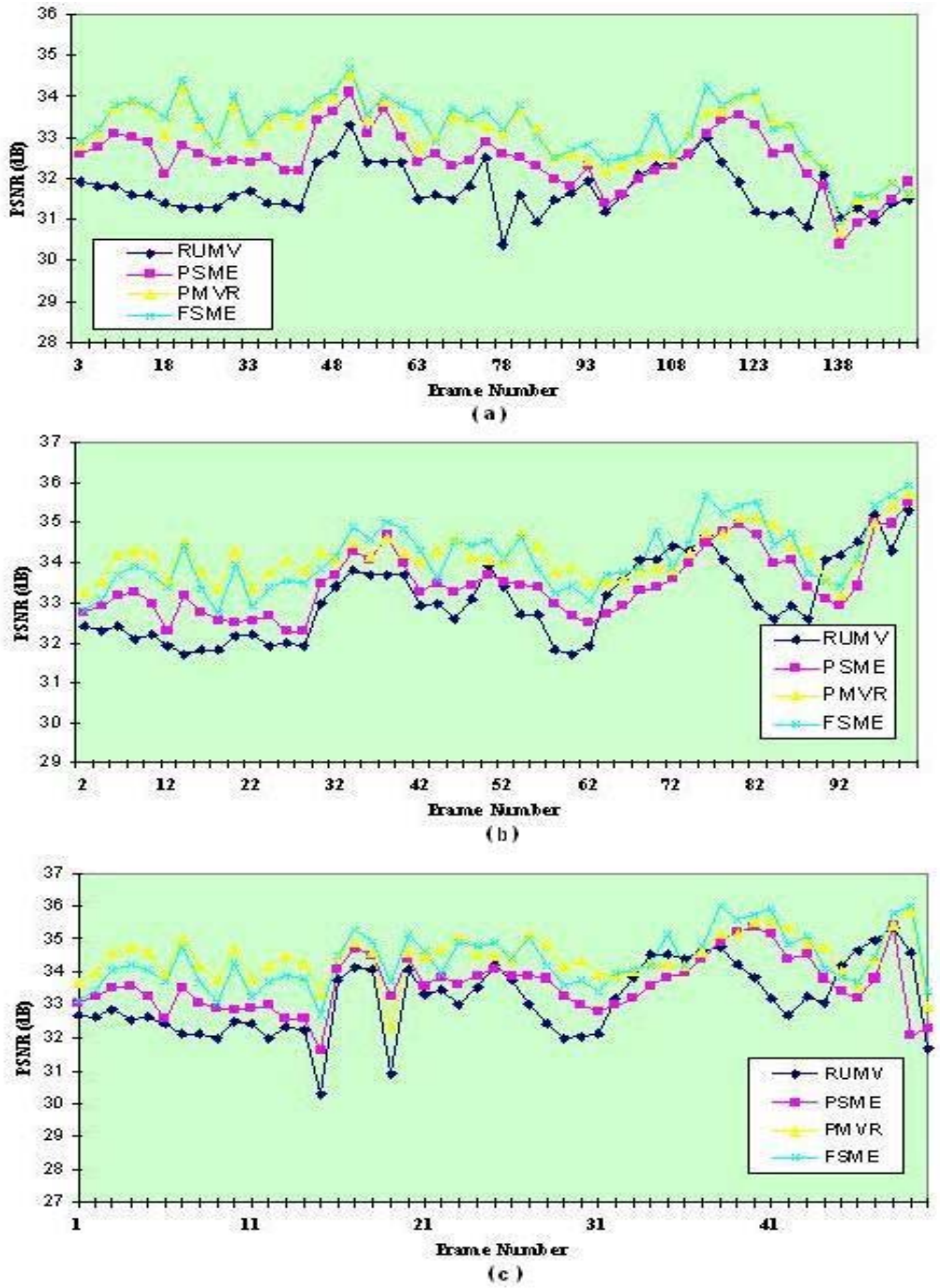



Fig. 7-8. Performance of the proposed motion vector refinement against related works on Mother and daughter. (a) outgoing frame-rate: 30 frames/s. (b) outgoing frame-rate: 15 frames/s and (c) outgoing frame-rate: 10 frames/s.

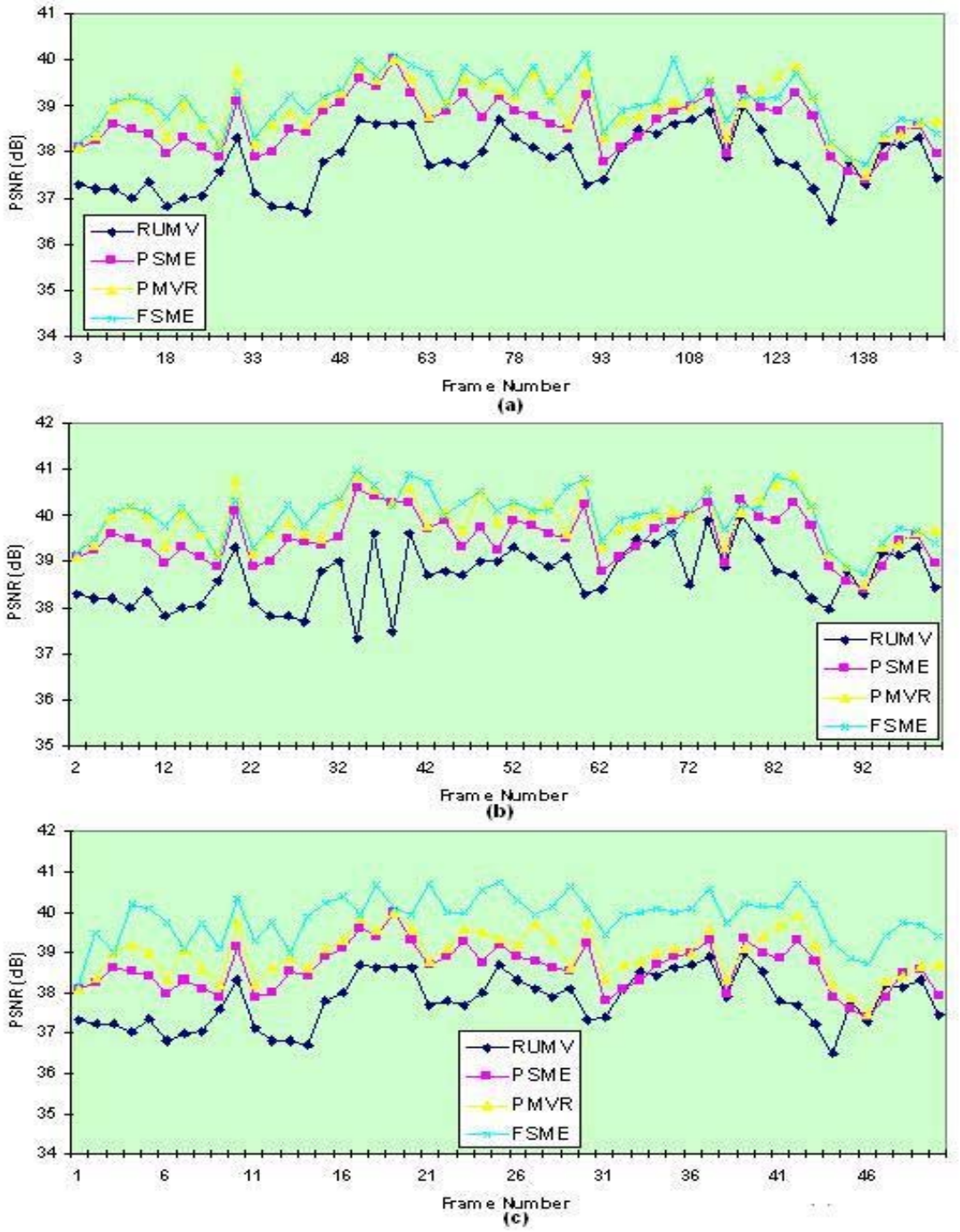



Fig. 7-9 Performance of the proposed motion vector refinement against related works on Coast Guard sequence. (a) outgoing frame-rate: 30 frames/s. (b) outgoing frame-rate: 15 frames/s and (c) outgoing frame-rate: 10 frames/s.

Clearly, the proposed motion vector refinement scheme has outperformed the re-use motion vector, spatial and full scale motion estimation approaches. This is due to the fact that refining motion vector values consumes few operations as opposed to the full or

partial scale motion estimation, as motion estimation is very time and power consuming. Even, PMVR outperforms RUMV in terms of processing time, due to the PMVR ability to eliminate non fine motion vector before re-decoding process in the transcoder, and thus PMVR saves a lot of memory and CPU power usage. These results are very important for real-time and interactive applications such as pre-encoded mobile video streaming. In summary, the proposed system boosts the performance, while keeping the computational complexity low.

Table 7-4 Execution time speed up table for Flower and garden sequence

| | **RUMV** | **PSME** | **PMVR** | **FSME** |
|---|---|---|---|---|
| **Execution Time (S)** | 144.32 | 1555.09 | 120.16 | 1866.4 |
| **Speed-up** | 12.93237 | 1.200188 | 15.53262 | 1 |

Table 7-5 Execution time speed up table for Claire sequence

| | **RUMV** | **PSME** | **PMVR** | **FSME** |
|---|---|---|---|---|
| **Execution Time (S)** | 184.32 | 1999.33 | 158.77 | 2250.33 |
| **Speed-up** | 12.20882 | 1.125542 | 14.17352 | 1 |

## 7.7 Conclusion and Future work

In this chapter, we have discussed motion vector refinement for high performance transcoding. Since a great deal of bit rate reduction is required, traditional transcoding methods based on simply reusing the motion vectors extracted from an incoming video bit stream are not adequate. They would produce unacceptable texture distortion in the reconstructed signals. However, by applying a new full-scale or partial-scale motion estimation quality of the transcoding mechanism is boosted the in favor of the high computation power consumed to realize these mechanisms.

In this chapter, we present a novel transcoding scheme, of which the quality can be significantly improved by refining the incoming motion vectors as opposed to applying a new full-scale motion estimation to find the optimal motion vectors or partial-scale motion estimation. We propose a new motion vector refinement scheme for a transcoder using the MPEG1/2 CODEC. Through extensive simulations we have showed that the proposed motion vector refinement scheme does improve the video quality to the level achieved by using the full-scale motion or partial-scale estimation, with minimal computational complexity.

In addition, our transcoding mechanism has been applied only in MPEG domain videos. We believe further researches should be conducted in different video domains such as H.263 and so forth. Actually, extra investigation would result in good description for generic transcoding mechanism in any video processing domain. Currently, a good trend towards the mobile TV has been brought to the academia and industries. Therefore, video transcoding will be an interesting issue to address for mobile TV field.

# Chapter 8

# Experimental Results and Discussion

We have designed several experiments in order to verify the performance, to test the noise model assumptions, and to prove the goodness of our proposed techniques. Intensive experiments have been conducted in many subjective and objective manners. More than three hundred hours of video sequences have been tested. Numerous Standard measurement metrics have been used.

## 8.1 Objective Evaluation

The proposed object detection approach are applied to various kinds of videos including the Spanish news in MPEG7 testing sequences, CNN news, CTTV news in Taiwan, baseball game in Taiwan. The content of testing sequences mainly includes shots of the anchor person, interview, football games, baseball games, bicycle racing, entertainment education and scenery, etc. The testing videos are of resolution of 320 X 240 in MPEG1/2 format in most of the cases.

The spread of video content features are so various in order to verify the proposed approach against different video contents. For example the spread of edge features vary from scatter form, such as the full court view of the football came, to centralization form such as anchor person in the news. The temporal edge variation is different from high, such as players in bicycle racing, through medium, such as the pitching view of baseball game, to low, such as the anchor person in the news. Table 8-1 contains the description of a subset of the used video sequences.

These details include the video sequence category, short description of each video sequence, and source. These videos are subset of MPEG7 testing dataset. In addition

several video sequence captured by us, or worldwide used in video research have been used. For instance, video test sequences Mother Daughter, Akiyo, Claire, Miss America, Hall Monitor, have been used. The simulation was not restricted to MPEG-7 test sequences. The proposed algorithm was also applied to an indoor scene video captured by a low-end camera.

### 8.1.1 Testing dataset configurations

The frame size is 320X240 which implies that we have 20X15 macroblocks in each P frame. Our testing dataset presents objects in different positions, speed, and object size. Objects in scattered form, such as the full court view of the football game and centralization form such as anchor person in the news. The temporal variation is different from high, such as players in bicycle racing, through medium to low, such as the pitching view of baseball game to the anchor person in the news. In addition, typical videoconference sequences (Mother Daughter, Akiyo, Claire) and sequences containing objects with straight forward motion (Hall Monitor, Container Ship) have been tested.

The metric used in the experiments are precision and recall, which are used together to measure the accuracy of the object detection system. We choose the *recall* and *precision metrics* as defined in Eq(8.1) and Eq(8.2) as they are commonly used in performance evolution [36,28,41,60].

In each frame, the *number of hits* is the number of macroblocks that contain an object and this object is correctly detected. The *number of false alarms* is the number of macroblocks, which contain no object yet are falsely identified as containing objects. The *number of misses* is the number of macroblocks that contain an object but yet the detection algorithm fails to detect it. We use the macroblock as the unit of measurement because we are doing the object detection in the compressed domain. The aim of the

experiment is to evaluate the objective performance of the proposed system.

Table 8-1: video sequence clip database

| | | |
|---|---|---|
| **News** | Two complete TV news programs | Portuguese TV, RTP & SIC |
| | Universal newsreel collection. B&W video. | National Archives at Maryland |
| | Daily TV news program | Spanish TV, RTVE |
| | Weekly TV news program | Spanish TV, RTVE |
| **Drama/ Movie** | "Art" movie: Hallo | Christoph Rodatz, GMD |
| | Movie: "La sombra de un cipres es alargada" | Spanish TV, RTVE |
| | TV Drama series: "Pepa y Pepe" | Spanish TV, RTVE |
| | Sitcom (1 and 2) | Portuguese TV, RTP & SIC |
| **Documentary** | "Science Eye": Bridge construction | NHK |
| | 5 clips of scientific documentaries | SFRS |
| | Documentary about buildings | Lancaster Television |
| | Basic Ophthalmic Exam | Univ. of Tennessee |
| | Documentary about a village: "Santillana del Mar" | Spanish TV, RTVE |
| **Sport** | 3 Sport Clips: Soccer, Cycling, Basketball | Spanish TV, RTVE |
| | 2 Sport clips: Basketball, Golf | Korean Broadcasting Station |
| | Soccer sequence | Samsung |
| **Commercial** | 14 items of commercials in Korean | Samsung |
| **Music video and games** | Korea's pop singers' live music Show | Korean Broadcasting Station |
| | TV quiz program: "Saber y ganar" | Spanish TV, RTVE |
| | Music program: "Musica si" | Spanish TV, RTVE |
| | Variety Show. First 30 minutes of complete program | Portuguese TV, SIC |

$$\text{Precision} = \frac{\text{Number of Hits}}{\text{Number of Hits} + \text{Number of False Alarms}} \tag{8.1}$$

$$\text{Recall} = \frac{\text{Number of Hits}}{\text{Number of Hits} + \text{Number of Misses}} \tag{8.2}$$

Figures 8-1 through 8-4 show the results of object extraction over the anchor person video clip from 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.8-1 and 8-2 illustrate the value of the recall and precision metrics for each frame in the video clip. We note that the performance of object extraction using the Gaussian filter is consistently superior to that using other filters or no filter. We show the average recall metric and average precision metric for the whole clip in Fig. 8-3 and Fig. 8-4. Again, the Gaussian filter topped them all for both precision and recall. The high recall metric in the mean filter with low precision is due to the fact that the mean filter creates unrealistic motion vectors, generating a high recall value, but with many false alarms.

Besides, we can infer from Figure 8-1 through 8-4 that the median filter due to its nature does not adjust motion vector values. Rather it is just rearranging motion vectors, not adjusting the content of those values and not eliminating the noise within them. Hence, the precision is almost like that not using a filter and only the recall metric increases slightly. In summary, the Gaussian filter boosts the object detection performance. In addition, the computational complexity is low as discussed previously. The Gaussian filter is available as a readily implemented component in both hardware and software, which demonstrates the flexibility and extendibility of the proposed scheme. Testing is performed using four types of related work which are, Group **A** using Gaussian filter only [37], group **B** using Median filter [28], Group **C** using Cascade Filter , and group **D** our system, finally without any kind of post processing.

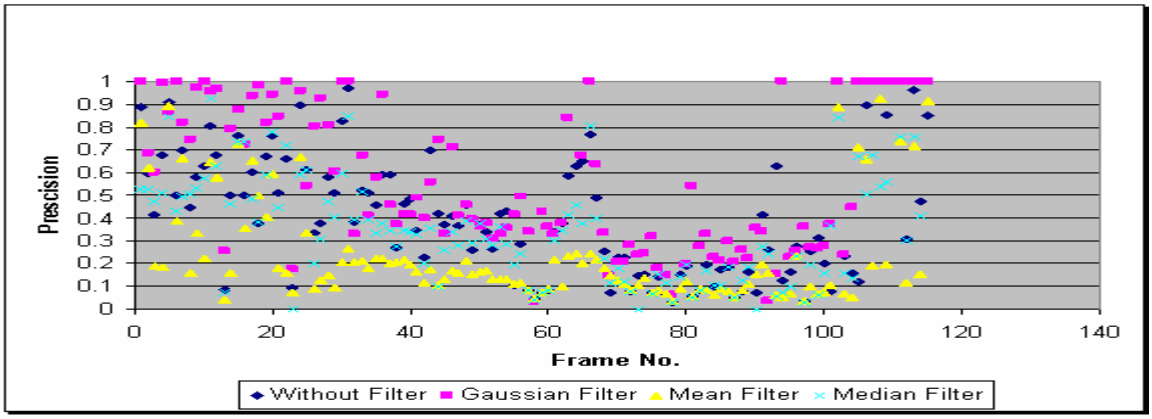



Fig. 8-1: Precision for Object extraction in P-frames of Anchor person

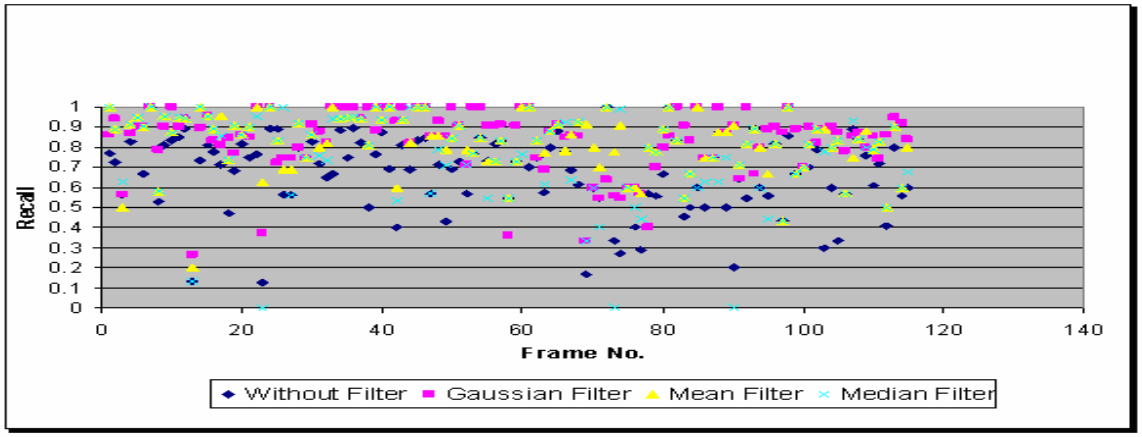



Fig. 8-2: Recall for Object extraction in P-frames of Anchor person

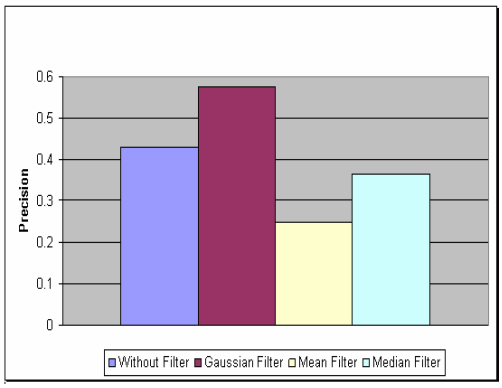



Fig. 8-3 Average Precision for object detection for anchor person Video

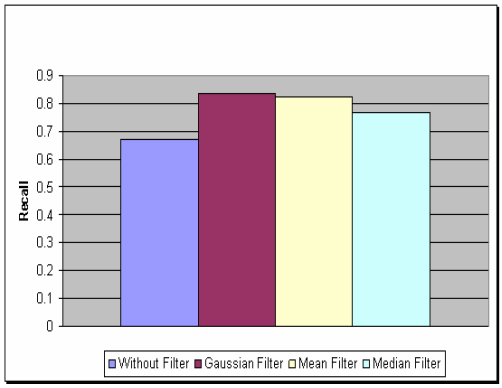



Fig. 8-4 Average Recall for object detection for anchor person Video

Figures 8-5 through 8-8 show the performance results of object extraction over the second video clip among the MPEG7 walking person 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. 8-5 and 8-6 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 other schemes. We show the average recall metric and average precision metric for the whole clip in Fig. 8-7 and 8-8. Again, our system topped them all. Through the experiment, we noticed that there is a weakness in the single Gaussian filter when the object location is in the frame border. This is 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 stable, 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.

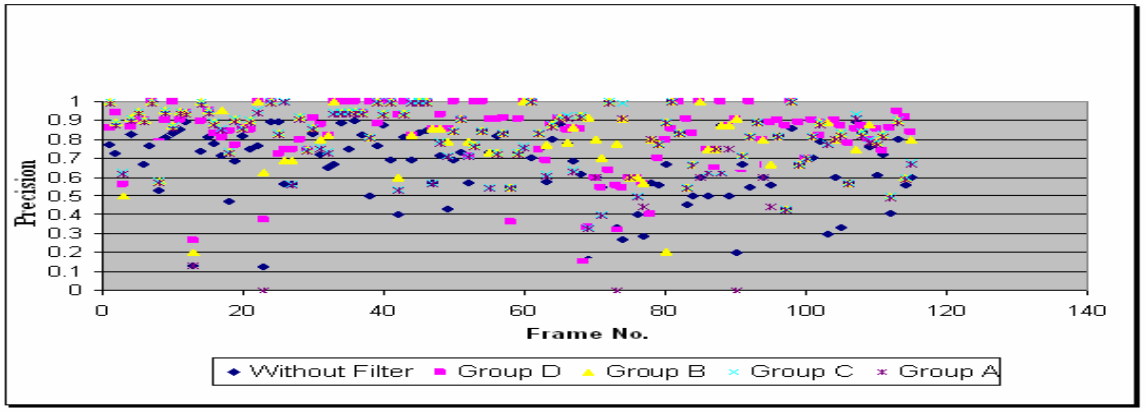



Fig. 8-5: Precision for Object extraction in P frame2nd sequence In walking person testing dataset

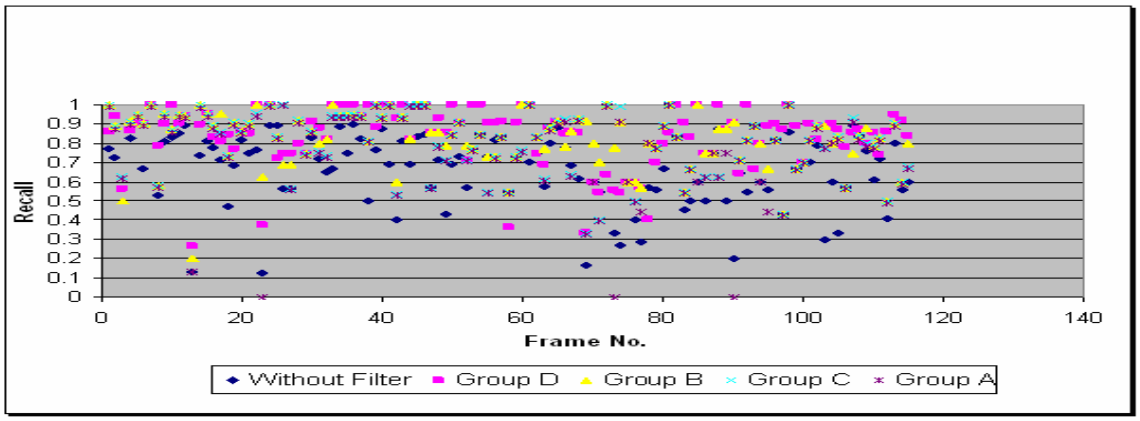



Fig. 8-6: Recall for Object extraction in P frames 2nd sequence In walking person testing dataset

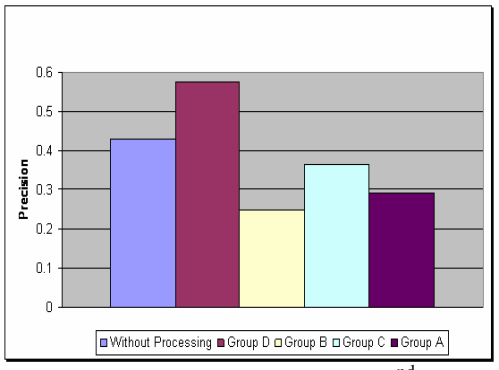



Fig. 8-7 Average precision of 2nd sequence In walking person testing dataset

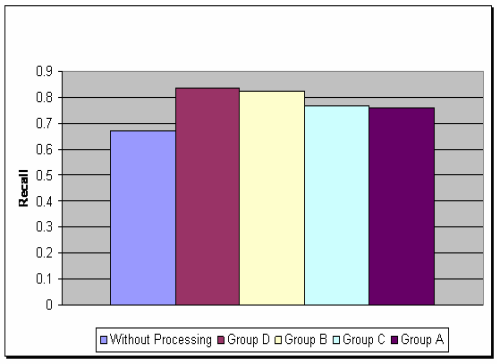
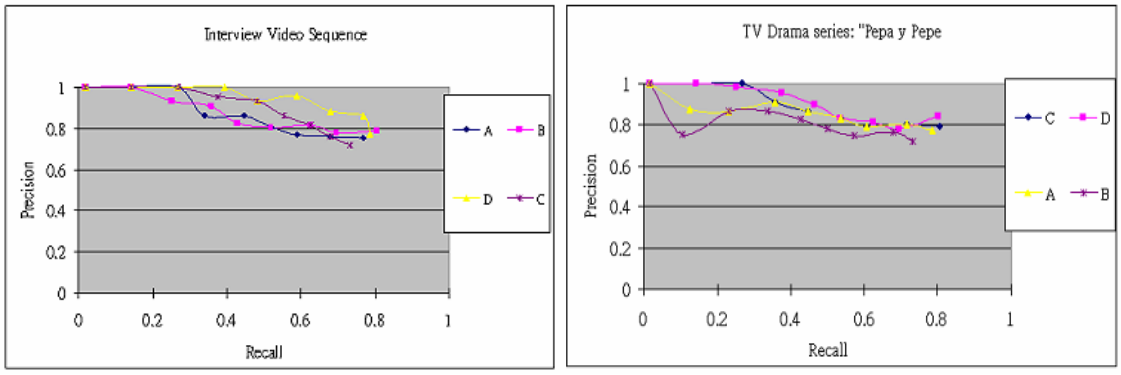
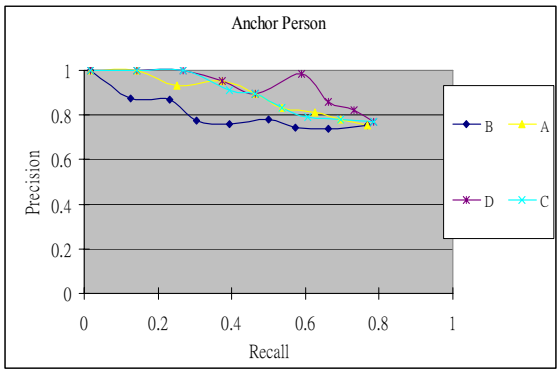



Fig. 8-8 Average Recall for 2nd sequence In walking person testing dataset

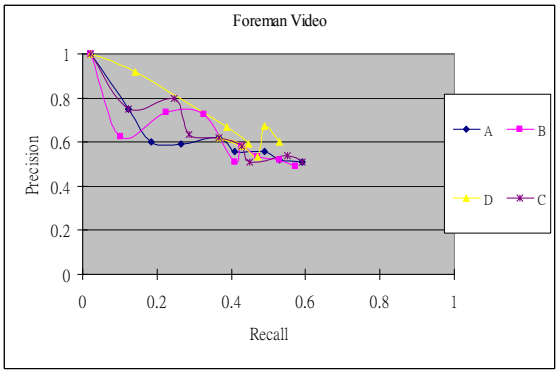
We then computed the average numbers of missing MBs, false MBs. Numbers of missing and false MBs are simply computed by comparing a segmented object mask with its related ground truth. Fig. 8-9 presents results of object detection approach on various video test sequences.



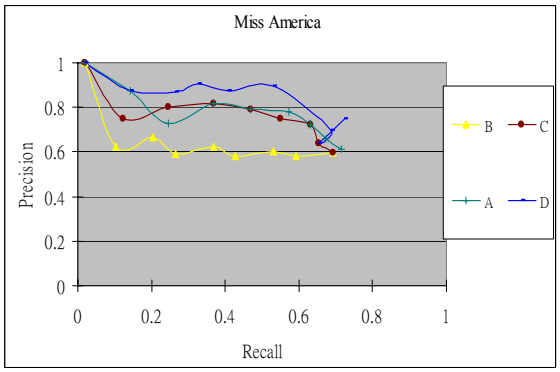

(a) (b)


Anchor Person
Precision
Recall
B
A
D
C

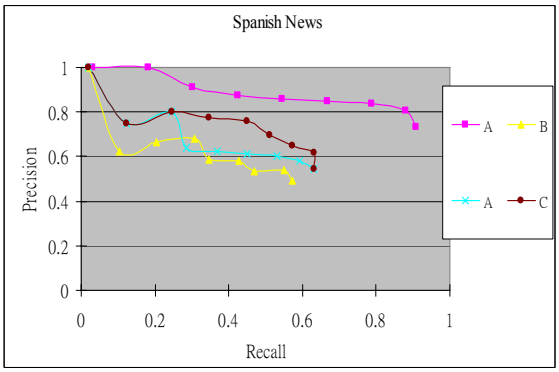
(c)


Foreman Video
Precision
Recall
A
B
D
C

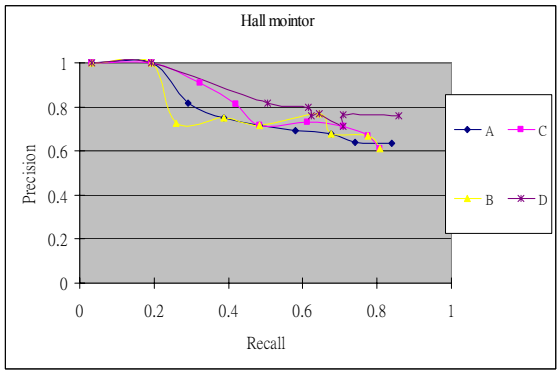
(d)


Miss America
Precision
Recall
B
C
A
D

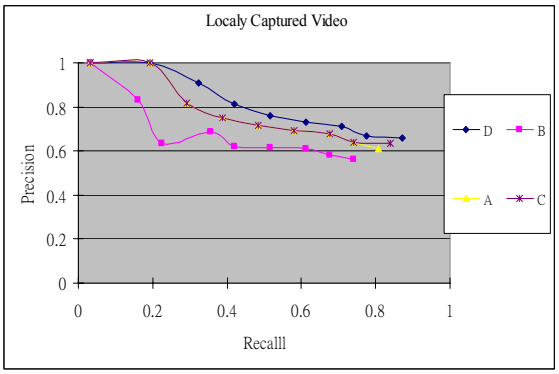
(e)


Spanish News
Precision
Recall
A
B
A
C

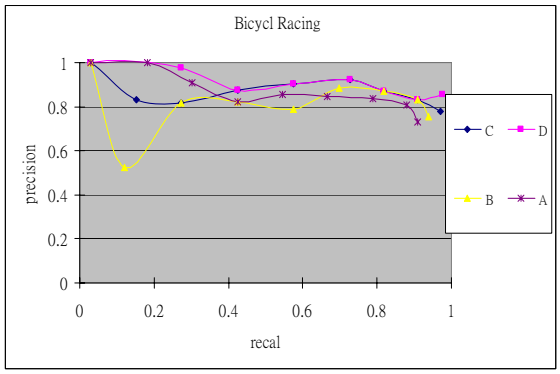
(f)


Hall mointor
Precision
Recall
A
C
B
D

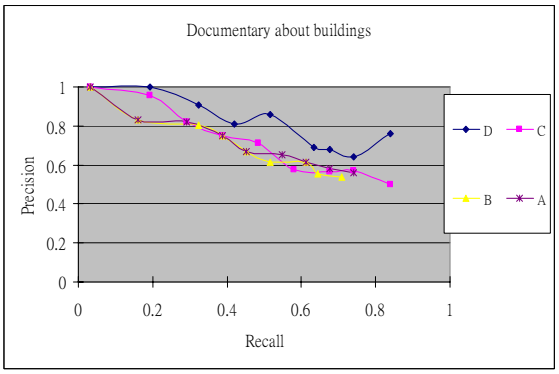
(g)


Localy Captured Video
Precision
Recalll
D
B
A
C

(h)


Bicycl Racing
precision
recal
C
D
B
A

(i)


Documentary about buildings
Precision
Recall
D
C
B
A

(j)

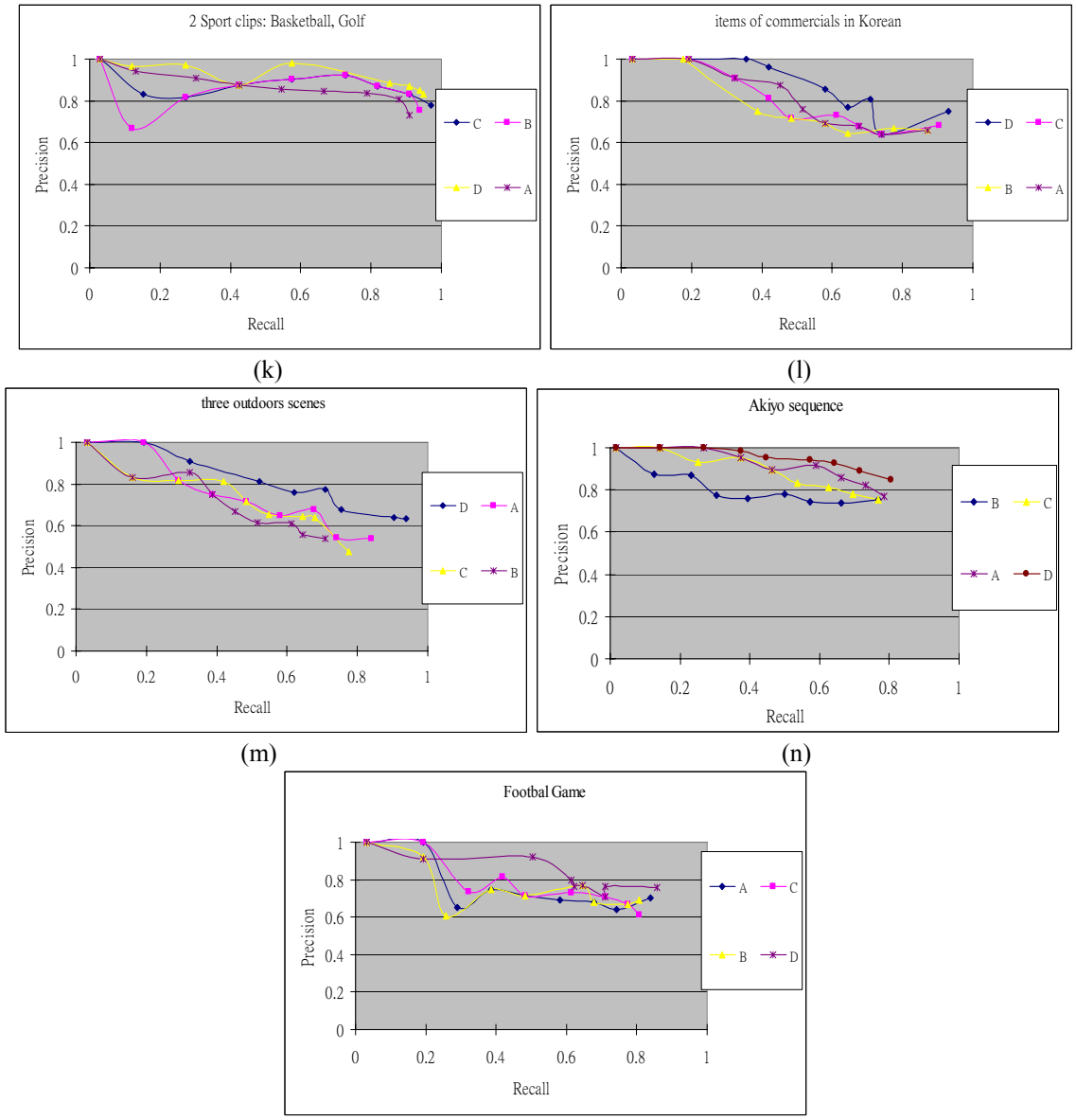



Fig. 8-9 Recall and precision for the following video clip (a) Interview (b) TV drama (c) Anchor person (d) Spanish news (e) Miss America (f) Spanish news (g) Hall monitor (h) Locally captured video (i) Documentary about buildings (j) Bicycle racing (k) Sport scenes for golf courts (l) commercial in Korean (m) three out door scenes (n)Akiyo (o) football game

## 8.1.2 Discussion

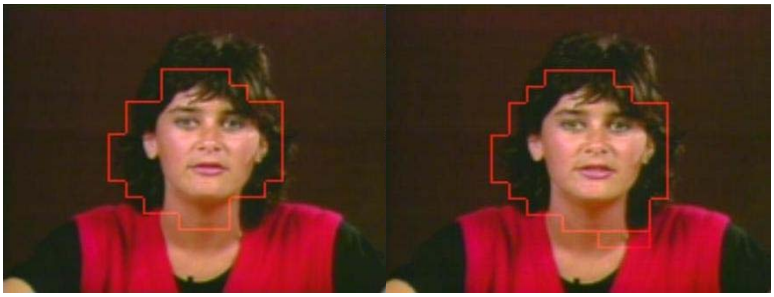
The recall rate drops in the speedways sequence because vehicles are very small when they are far away. "Hall Monitor," is surveillance type of video containing small moving objects and complex background. "Miss America" and "Akiyo" sequences are typical head-and shoulder type video in QCIF and CIF format, respectively. For

sequences which have temporarily still objects from the beginning, such as “Miss America” or “Akiyo,” our proposed approach still have sufficiently good detection results.

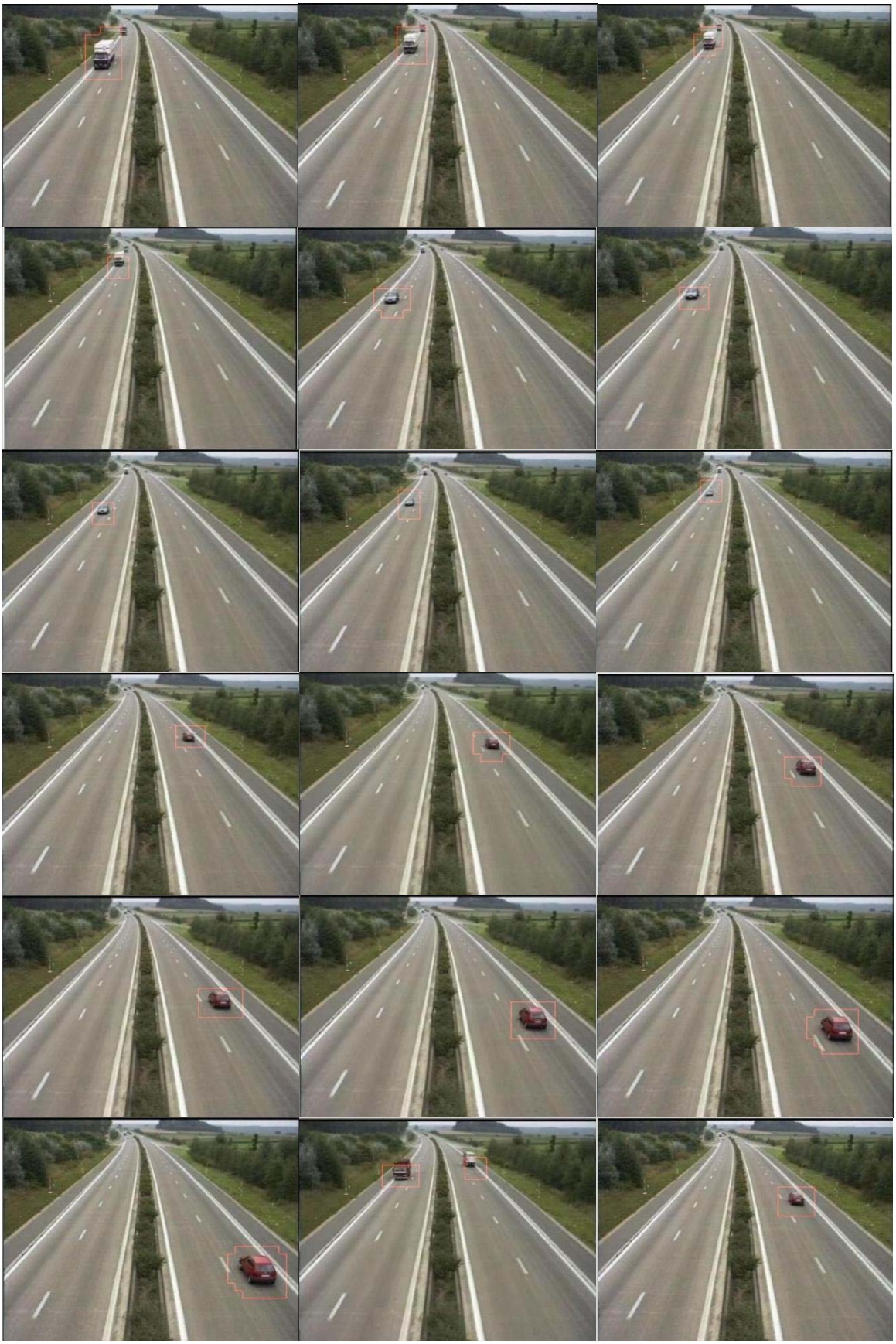
## 8.2 Subjective Evaluation

Fig.-8-10 shows the segmentation results for several benchmark sequences. The sequences are:

- Speedway video sequence
- Miss America video sequence
- Foreman video sequence
- Text in commercial video
- Actress in TV festival
- Actress in TV ceremony
- Interview “professor “ from Spanish news
- Interview “student” from Spanish news
- Oil industry report from Spanish news
- Sport Reporter
- Presidential talk
- Anchor person

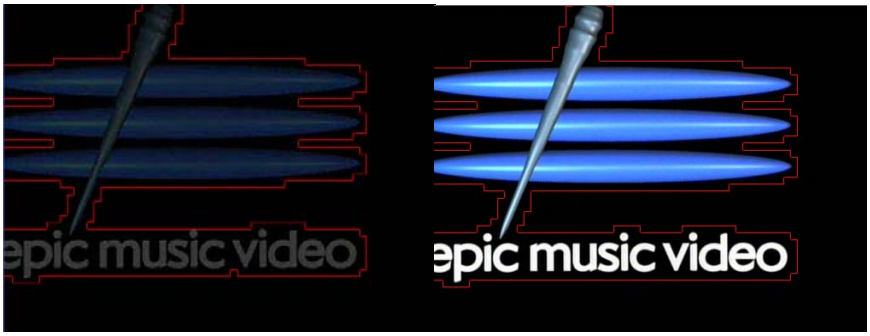


(a)

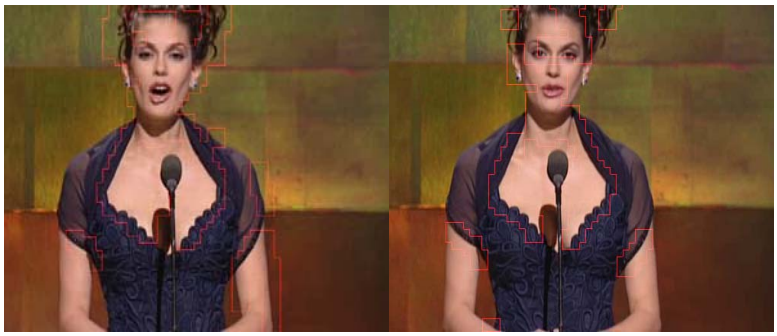


(b)

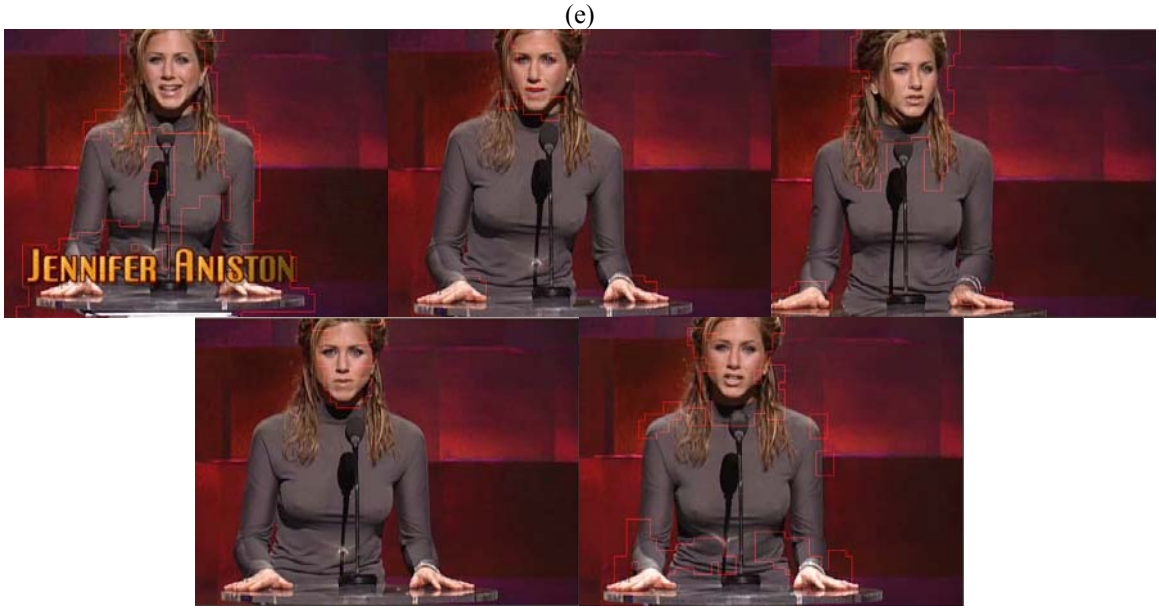
(c)



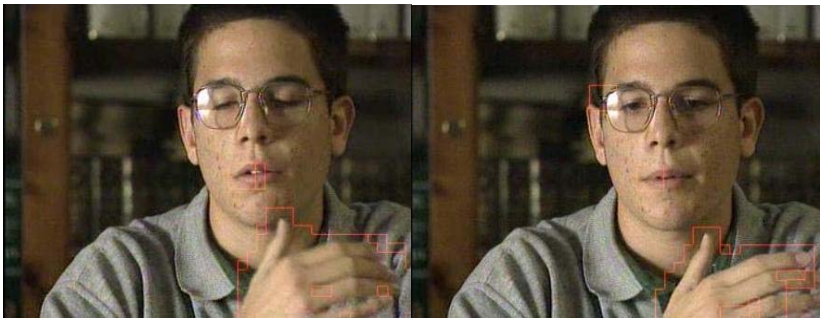

(d)



(e)



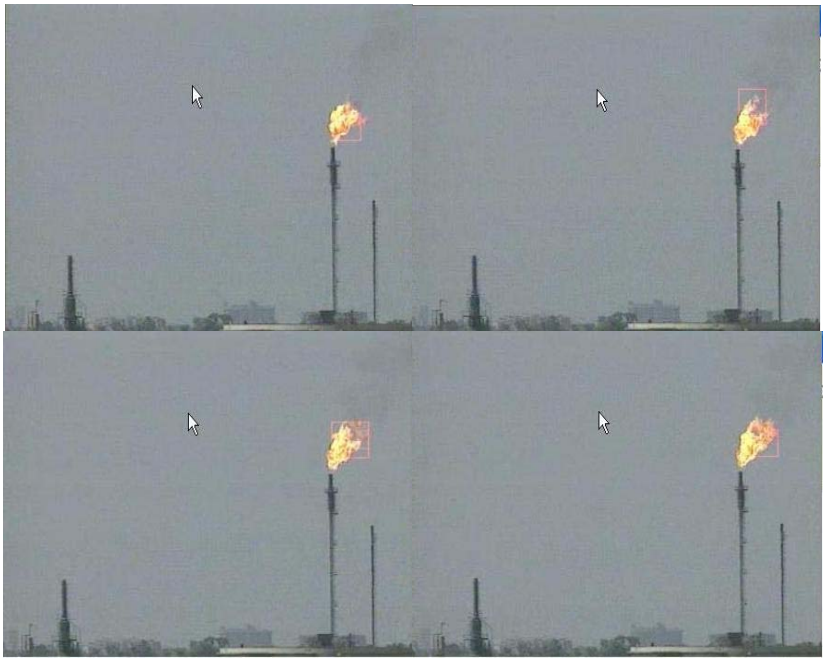

(f)



(g)

(h)

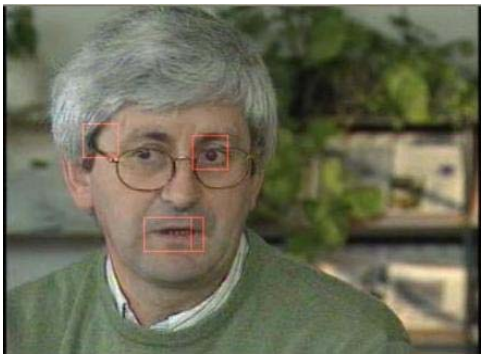

(i)



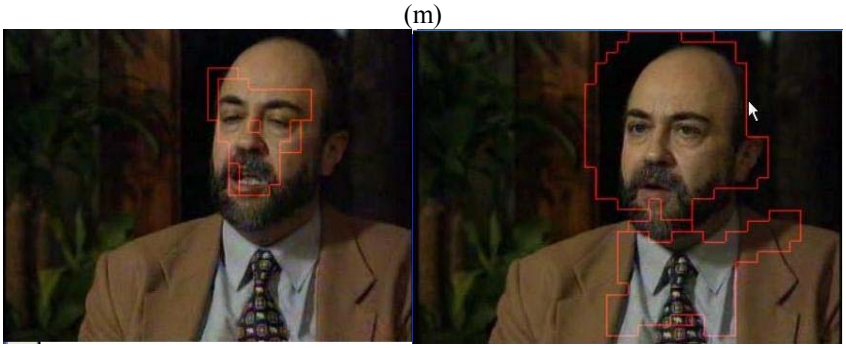
(j)

(k)


(l)



(m)



(n)

Fig.8-10: Subjective detection results for the following video sequences (a) Miss America video sequence (b) Speedway video sequence (c) Foreman video sequence (d) Text in commercial video (e) Actress in TV festival (f) Actress in TV ceremony (g) Interview "student" from Spanish news (h) Sport report from Spanish news (i) Interview from Spanish news (j) Oil industry report from Spanish news (k) Presidential talk (l) Anchor person (m) Interview "professor " from Spanish news (n) Interview from Spanish news

The Akiyo and Weather sequences do not have background noise so their segmentation results tend to be better than those of other sequences. Background noise does exist in the Hall Monitor sequence. Also, shadows cause by indoor illumination appear in the background region. Still our segmentation results track the object shape quite well and are subjectively better than previous results.

## 8.3 Run Time Evaluation

In order to verify the previous 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 anchor person video clip. The experiment result will be presented in the Table 8-2.

Table 8-2, shows the run time in milliseconds (ms) for the object extraction module using our system, other related work, and without any post preprocessing for the walking person clip. This video clip is among the MPEG7 testing dataset. The result is based on the first 500 frames. The groups A, B and C are the same related group used in comparing the objective result metrics. Also, we describe the I/O time needed to perform the object extraction task and 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 comes compatibly with our expectation and initial idea.

Table 8-2: Run time for object extraction in millisecond

| | Group A | Group B | Group C | Group D | Without Post Processing |
|---|---|---|---|---|---|
| Object Extraction Run time | 285 ms. | 269. | 258 ms. | 215 ms. | 300 ms. |
| I/O processing | 64 ms. | 60 ms. | 58 ms. | 48 ms. | 70 ms. |
| Total Time | 349 ms. | 329 ms. | 306 ms. | 263 ms. | 370 ms. |

Generally the segmentation speed depends on the object size and the complexity of the scene. For a typical object like Akiyo in CIF size images, it takes around 20 seconds per frame on a SUN UltraSparc-2 workstation. Note that this includes all the

computation processes described in the system architecture. Optimization may be performed to determine critical processes and non-critical processes may be ignored to reduce computations. Parallel processing can greatly improve the speed of our system.

As one can see, the computations of feature maps, which are the most computation intensive parts of the system, can easily be done in parallel. It can be seen that in the proposed approach, object detection is carried only on a part of the motion vectors, as many undesirable 'noisy' motion vectors have been eliminated. This drastically reduced the computation time as compared to existing algorithms for object detection.

# Chapter 9

# Conclusion and Future Work

## 9.1 Conclusion

It is a well-known fact that motion information is an important cue for humans to perceive video content. The proposed algorithms are appropriate for various kinds of video processing and communication applications in which motion vector is an important clue. The contributions and characteristics of the proposed scheme can be summarized as follows:

• Effectiveness: ability to refine motion vectors based on the texture, temporal and spatial features directly extracted from the compressed domain.

• Real-time Processing: ability to process in real time due to its low complexity and the property of processing in compressed domain.

• Open Framework: possibility to be incorporated to various kinds of applications.

Along with the increasing popularity of video over internet and versatility of video applications, such as video surveillance, vision-based control, human-computer interfaces, medical imaging, robotics and so on, the availability and efficiency of videos will heavily rely on object detection and related object tracking capabilities.

There are some special scenarios that automatic detection of video objects is a crucial issue. For instance, it is inevitable in developing object-based video application system, with on-line detection schemes. However, a desirable video objects extraction scheme for real-time object-based applications should meet the following criteria:

- Segmented objects should conform to human perception, i.e., semantically meaningful objects should be segmented.

- Segmentation algorithms should be efficient and achieve fast speed.
- Initialization should be simple and easy for users to operate (human intervention should be minimized).

This has been verified by examining the results of our experiments. We are achieving the additional advantages of efficiency and speed up by staying fully in the compressed domain, using only the P frames, and using a simple approach to filter implementation. Moreover, initialization and operation of our system is simple as well.

## 9.2 Future Work

In the future, we will use our proposed scheme in adopting a method to convert motion vectors in the MPEG coded domain to a uniform set, which is independent of the frame type and the direction of prediction. By utilizing these normalized motion vectors in our system, we expect to achieve better object detection. We propose a cost-efficient mechanism for improving the quality of service (QoS) delivered to the mobile users, by introducing a robust and efficient transcoding scheme as proven by extensive experiments. The proposed approach refines the motion vectors without the need to re-perform the motion estimation process.

Then the transcoding mechanism will be preformed using the refined motion vectors. Thus, great amounts of computing resources have been saved. Actually, extra investigation would result in good description for generic transcoding mechanism in any video processing domain. Currently, a good trend towards the mobile TV has been brought to the academia and industries. Therefore, video transcoding will be an interesting issue to be addressed for mobile TV field.

In the object based video streaming framework, video objects are extracted automatically and used to control video quality under various manipulations and network

resource requirements. Several future applications can be anticipated based on our proposed approach.

1. Mobile video communication is very promising and hot research topic in academia and industry. To allow 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.
2. Online video broadcasting, such as game broadcasting and news broadcasting, has requirements real-time processing and display. 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.
3. E-Learning and web distance education may take advantage of our scheme. But this time, lecturer face will be assumed as moving object.

On enhancing the proposed scheme issues on the quality of service in RTP stack level, need to be further addressed. We anticipate that further quality of service can be attained by co-operating the object detection result with the RTP stack definition.

# Reference:

[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, pp. 857-868, 2000.

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

[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, pp. 427-432, 2000.

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

[5] A. Pentland, R. Picard, and S. Sclaroff,Photobook "Tools for Content-Based Manipulation of Image Databases.Storage and Retrieval of Image and Video Databases II," Proc. Of The International Society for Optical Engineering, pp. 34-47, 1994.

[6] V. V. Vinod and H. Murase, "Video Shot Analysis using Efficient Multiple Object Tracking" Proc. Of the IEEE International Conference on Multimedia Computing and Systems, pp.501-508, 1997.

[7] P. Fieguth, "Color-Based Tracking of Heads and Other Mobile Objects at Video Frame Rates," Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 21-27, 1997.

[8] N. Brady and N. O'Connor, "Object detection and tracking using an EM-based motion estimation and seg-mentation framework," Proc. IEEE International Conference on

Image Processing , pp. 925-928, 1996.

[9] D. P. Elias, "The motion Based Segmentation of Image Sequences", Ph.D. thesis, Trinity College, Department of Engineering, University of Cambridge, Aug. 1998.

[10] N. Vasconcelos and A. Lippman, "Empirical Bayesian EM-based motion segmentation," Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 527-532, 1997.

[11] P. H. S. Torr, R. Szeliski, and P. Anandan, "An integrated bayesian approach to layer extraction from image sequences," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 23, No. 3, pp. 297-303, 2001.

[12] R. Wang, and T. Huang, "Fast Camera Motion Analysis in MPEG domain," Proc. ICIP, Vol. 3, pp. 691-694, 1999.

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

[14] 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, pp. 572-576, 2001.

[15] D.-Y. Chen, S.-J. Lin and S.-Y. Lee, "Motion Activity Based Shot Identification and Closed Caption Detection for Video Structuring," Proc. VISUAL, pp. 288-301, 2002.

[16] T. R. M. King, "Efficient and Effective Methods for Object Segmentation in Video Images," Final report, Fall Semester the Johns Hopkins University, http://www.apl.jhu.edu/Notes/Beser/525759/kingfinalreport.pdf, 2002.

[17] D.-Y. Chen, and S.-Y. Lee "Motion-Based Semantic Event Detection for Video Content Description in MPEG-7," Proc. IEEE Pacific-Rim Conference on

Multimedia , pp. 110-117, 2001.

[18] A. Eleftheriadis and D. Anastassiou, "Constrained and general dynamic rate shaping of compressed digital video," in Proc. IEEE Int. Conf. Image Processing, pp 396-399, 1995.

[19] G. Keesman et al., "Transcoding of MPEG bitstreams," Signal Process. Image Communication, Vol. 8, No.6, pp. 481-500, 1996.

[20] P. N. Tudor and O. H. Werner, "Real-time transcoding of MPEG-2 video bit streams," in Proc. Int. Broadcasting Con., pp. 286-301, 1997.

[21] P. Assuncao and M. Ghanbari, "Post-processing of MPEG2 coded video for transmission at lower bit rates," in proc. of IEEE International Conference On Acoustics, Speech And Signal Processing, pp.1998-2001, 1996,.

[22] D. G. Morrison, M. E. Nilsson, and M. Ghanbari, "Reduction of the bit-rate of compressed video while in its coded form," in Proc. Sixth Int. Workshop Packet Video, pp. 17-21, 1994.

[23] R. J. Safranek, C. Kalmanek, and R. Garg, "Methods for matching compressed video to ATM networks," in Proc. Int. Conf. Image, pp. 13-16, 1995.

[24] H. Sun, W. Kwok, and J. W. Zdepski, "Architecture for MPEG compressed bitstream scaling," IEEE Trans. Circuits Syst. Video Technology, Vol. 6, No. 2, pp. 191-199, 1996.

[25] Y.-P. Tan, D.D. Saur, S.R. Kulkami, P.J. Ramadge, "Rapid Estimation of Camera Motion from Compressed Video with Application to Video Annotation," Circuits and Systems for Video Technology, IEEE Transactions on Vol. 10, No. 1, pp. 133-146, Feb. 2000.

[26] S.-F. Chang, "Compressed-Domain Techniques for Image/ Video Indexing and

Manipulation" IEEE International Conference on Image Processing, pp. 325-331, 1995.

[27] S.-F. Chang, Compositing and Manipulation of Video Signals for Multimedia Network Video Services, Ph.D. Dissertation, U.C. Berkeley, Aug., 1993.

[28] S. Chien, S. Ma, and L. Chen, "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, 2002.

[29] G. Kühne, S. Richter, and M. Beier, "Motion-based segmentation and contour-based classification of video objects," Proc. ACM Multimedia Conference, pp.41-50, 2001.

[30] R. Curwen and A. Blake, "Dynamic Contours Real-Time Active Splines," In Active Vision MIT Press, pp. 39-58, 1992.

[31] R. Goldenberg, R. Kimmel, E. Rivlin, and M. Rudzsky, "Fast Geodesic Active Contours" .Proc. Int'l Conf. Scale-Space Theories in Computer Vision, pp. 34-45, 1999.

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

[33] Y. Ma, and H.-J. Zhang, "A New Perceived Motion based Shot Content Representation," IEEE International Conference on Image Processing, pp. 426-429, 2001.

[34] M. Pilu, "On Using Raw MPEG Motion Vectors To Determine Global Camera Motion," Digital Media Department HP Laboratories Bristol HPL-97-102,http://www.hpl.hp.com/techreports/97/HPL-97-

[35] N. W. Kim, T. Y. Kim, and J. S. Choi, "Motion analysis using the normalization of MVs on MPEG compressed domain", In Proc.The International Technical Conference On Circuits/System, Computers and Communications, pp.1408-1411, 2002.

[36] R. Ulichney, "Filter Design for void and cluster dither arrays", Proc. System Information Display Int. Symposium, pp. 809-812, 1994.

[37] R. V. Babu, and K. R. Ramakrishnan , "Compressed Domain Motion Segmentation for Video Object Extraction" Proc. of IEEE International Conference On Acoustics, Speech And Signal Processing, pp. 3788-3791, 2002.

[38] H.-L. Eng, and K.-K. Ma, "Motion trajectory extraction based on macroblock motion vectors for video indexing" Proc. international Image Processing conference, pp. 284-288, 1999.

[39] V. Kobla, and D. Doermann, "Compressed domain video indexing techniques using DCT and motion vector information in MPEG video," in Proc. SPIE Conf. Storage and Retrieval for Image and Video Databases, pp. 200-211, 1997.

[40] J. Meng, and S. Chang, "CVEPS - a compressed video editing and parsing system," in Proc. Of ACM Multimedia, pp. 56-66, 1996.

[41] D.-Y. Chen, S.-Y. Lee, and H.-T. Chen, "Motion Activity Based Semantic Video Similarity Retrieval". Proc. IEEE Pacific Pacific-Rim Conference, pp. 319-327, 2002.

[42] A. K. Jain, and S. Bhattacharjee, "Text Segmentation Using Gabor Filters for Automatic Document Processing", Journal of Machine Vision and Applications, Vol. 5, No. 3, pp. 169-184, 1992.

[43] A. K. Jain, and Y. Zhong, "Page Segmentation Using Texture Analysis", Journal Pattern Recognition, Vol. 29, No. 5, pp. 743-770, 1996.

[44] B. L. Yeo, and B. Liu, "Visual Content Highlighting via Automatic Extraction of Embedded Captions on MPEG Compressed Video", Proc. SPIE Digital Video Compression: Algorithms and Technologies, pp. 38-47,1995.

[45] D. LeGall, "MPEG: A Video Compression Standard for Multimedia Applications", Journal of Comm. ACM, Vol. 34, No. 4, pp. 46-58, 1991.

[46] D. Gatica, C. Gu , and M.-T. Sun, "Semantic video object extraction using four-band watershed and partition lattice operators," IEEE Trans, Circuits Syst. Video Technol., Vol.11, No. 3, pp.603-618, 2001.

[47] X. Haifeng, A. Younis, and R. Kabuka," Automatic Moving Object Extraction for Content-based Applications," IEEE Transactions on Circuits and Systems for Video Technology, Vol.14, No 6, pp.796-812, 2004.

[48] A. Ogale, C. Fermuller and Y. Aloimonos," Motion segmentation using occlusions," IEEE Transactions On Pattern Analysis and Machine Intelligence, Vol. 27, No 6, pp.988-992. 2005.

[49] J. Senders, “Distribution of Attention in Static and Dynamic Scenes,” In: proceedings The International Society for Optical Engineering, pp. 186-194, 1997.

[50] A. Yarbus, “Eye Movements and Vision” Plenum Press, NewYork NY, (1967)

[51] Y. Ma, H. Zhang, “A Model of Motion Attention for Video Skimming,” Proc of The International Conference on Image Processing, pp. 22-25, 2002.

[52] L-A. Zadeh, “A Note on Web intelligence, World Knowledge and Fuzzy Logic,” Journal of Data Knowledge Engineering, Vol.50, pp. 291-304, 2004.

[53] P. Sobey, and M. Srinvansan, "Measurement of Optical Flow by a Generalized Gradient Scheme," J. on Opt. Soc. Am, Vol.8, No.9, pp.1488-1498, 1991.

[54] D. Murray, and A. Basu, "Motion Tracking with an Active Camera," IEEE Trans.

PAMI, Vol.16, No.5, pp.449-459, 1994.

[55] J. Meng, and S.-F. Chang, "Tools for Compressed- domain Video Indexing and Editing", Proc. SPIE Storage and Retrieval for Still Image and Video Databases, pp.180-191, 1996

[56] Y. Nakajima, A. Yoneyama, H. Yanagihara, and M. Sugano, "Moving object detection from MPEG coded data", Proc. SPIE Visual Communications and Image Processing, pp.988-996, 1998.

[57] S.-F. Chang, D. Zhong, R. Kumar, "Real-Time Content-Based Adaptive Streaming of Sports Video." ADVENT Technical Report #121 Columbia University, 2001.

[58] M. Hsiao, H. Kuo, H. Wu, Y. Chen, and S.-Y. Lee, "Objected-Based Video Streaming Technique with Application to Intelligent Transportation Systems", in procceding of IEEE International Conference on Networking, Sensing and Control pp. 315 – 320, 2004.

[59] A. Ahmad, D.-Y. Chen and S.-Y. Lee, "Robust Object Detection Using Cascade Filter in MPEG Videos," IEEE Fifth International Symposium on Multimedia Software Engineering, pp.156-162, 2003.

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

[61] A. Ahmad; D.-Y. Chen, and S.-Y. Lee, "Robust Compressed Domain Object Detection In Mpeg Videos" Proc. of the 7th IASTED International Conference Internet and Multimedia System Applications, pp. 706-712, 2003,.

[62] Y. Hongjiang, and K. Jain, "Automatic Caption Localization in Compressed

Video", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, No. 4, pp. 385-392, 2000.

[63] A. Ortega, and M. Khansari, "Rate Control for Video Coding over Variable Bit Rate Channels with Applications to Wireless Transmission", Proceedings IEEE International Conference on Image Processing, pp. 3388-3393, 1995.

[64] A. Ortega, F. Carignano, S. Ayer, and M. Vetterli, "Soft caching: Web cache management techniques for images" Proceeding IEEE Signal Processing Society Workshop on Multimedia Signal Processing, pp. 45-52, 1997.

[65] A. Ahmad, B. Ahmad, S. Talat, and S.-Y. Lee, "Fast And Robust Object Detection Framework For Object Based Streaming System," Proceeding The Second International Conference on Advances in Mobile Multimedia, pp.77-86, 2004.

[66] A. Ahmad, and S.-Y. Lee, "Novel Object-Based Video Streaming Technique," Proc. of 2nd International Conference on Computing, Communications and Control Technologies, pp. 255-300, 2004.

[67] A M. A. Ahmad; B M. A. Ahmad, and S Talat, "A Novel Approach for Improving the Quality of Service for Mobile Video Transcoding," The Third International Conference on Advances in Mobile Multimedia, pp. 323-330, 2005.

[68] A. Tripathi, and M. Claypool, "Improving Multimedia Streaming with Content-Aware Video Scaling", In Proceedings of the Second International Workshop on Intelligent Multimedia Computing and Networking, pp. 99-106, 2002.

[69] D. Lin, and R. Morris, Dynamics of Random Early Detection. In Proceedings of ACM the Special Interest Group on Data Communication Conference, pp. 425-435, 1997.

[70] D. Reyes, A. Reibman, J. Chuang, and S.-F. Chang, "Video Transcoding for Resilience in Wireless Channels," IEEE International Conference on Image Processing, pp. 365-366, 1998.

[71] ISO/IEC, "Generic coding of moving pictures and associated audio information", 1995.

[72] ISO/IEC 11 172, "Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s", 1993.

[73] J. Chung, and M. Claypool, "Better-Behaved, Better-Performing Multimedia Networking" In Proceedings of Euromedia Conference, pp. 84-90, 2000.

[74] J. Mitchell and W. Pennebaker, "MPEG Video: Compression Standard," Chapman and Hall, pp. 90-101, 1996.

[75] J. Shin, J. Kim, and C. Kuo, "Content-Based Video Forwarding Mechanism in Differentiated Service Networks" In Proceedings of International Packet Video Workshop, pp. 77-83, 2000.

[76] J. Walpole, R. Koster, S. Cen, C. Cowan, D. Maier, D. McNamee, C. Pu, D. Steere, and L. Yu, "A Player for Adaptive MPEG Video Streaming over the Internet", In Proceedings of 26th Applied Imagery Pattern Recognition Workshop, pp. 99-105, 1997.

[77] M. Claypool, and J. Tanner, "The Effects of Jitter on the Perceptual Quality of Video", In Proceedings of ACM Multimedia Conference, pp. 1122-1131, 1999.

[78] M. Hemy, U. Hangartner, P. Steenkiste, and T. Gross, "MPEG System Streams in Best-Effort Networks", In Proceedings of Packet Video Workshop, pp. 90-102 1999.

[79] M. Miyabayashi, N. Wakamiya, M. Murata, and H. Miyahara, "Implementation of

Video Transfer with TCP-Friendly Rate Control Protocol", In Proceedings of International Technical Conference on Circuits/Systems, Computers and Communications, pp. 117—120, 2000.

[80] N. Yeadon, F. Garcia, and D. Hutchinson, "Filters: QoS Support Mechanisms for Multipeer Communications", IEEE Journal on Selected Areas in Communications, Vol 4, No 7, pp.1245 1262, 1996.

[81] P. Bocheck, A. Campbell, S.-F. Chang, and R. Lio, "Utility-based Network Adaptation for MPEG-4 Systems", In Proceedings of Ninth International Workshop on Network and Operating System Support for Digital Audio and Video, pp. 890-87, 1999.

[82] S. Chandra, and C. Ellis, "JPEG Compression Metric as a Quality Aware Image Transcoding", In Proceedings of Second Usenix Symposium on Internet Technologies and Systems, pp. 547-552, 1999.

[83] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-Based Congestion Control for Unicast Applications" In Proceedings of ACM the Special Interest Group on Data Communication Conference, 232-233, 2000.

[84] S. Floyd, and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking, pp. 1256-1262, 1993.

[85] S. Sakazawa, N. Ytakishima, K. Hashimoto,"Video Data Transmission Protocol "SVFTP" Using Multiple TCP Connections and Its Application", IEICE transaction on Information an systems, pp. 60-72, 2005.

[86] S. Jacobs, and A. Eleftheriadis, "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, 1998.

[87] V. McCanne, and M. Vetterli, "Receiver-driven Layered Multicast", In Proceedings of ACM the Special Interest Group on Data Communication Conference, pp. 90-99, 1996.

[88] S. McCanne, M. Vetterli, and V. Jacobson, "Low-complexity Video Coding for Receiver-driven Layered Multicast", IEEE Journal on Selected Areas in Communications, Vol. 16, No. 6, pp. 983-1001, 1997.

[89] R. Rejaie, M. Handley, and D. Estrin, "RAP: An End-to-End Rate-Based Congestion Control Mechanism," Proc Conference on Computer Communications, pp. 1337-1345, 1999.

[90] H. Leong, and A. Si, "Multi-resolution information transmission in mobile environments", Journal of Mobile Information Systems, IOS Press, Vol. 1, No. 1, pp. 25-40, 2005.

[91] T. Nguyen, P. Brezany, M. Tjoa, and E. Weippl, "Toward a Grid-Based Zero-Latency Data Warehousing Implementation for Continuous Data Streams Processing", International Journal of Data Warehousing and Mining, Vol. 1, No. 4, pp. 22-55, 2005.

[92] R. Han, P. Bhagwat, R. LaMaire, T. Mummert, V. Perret and J. Rubas, Dynamic Adaptation in an Image Transcoding Proxy for Mobile Web Browsing, IEEE Personal Communications, pp.8-17, 1998.

[93] T. Warabino, S. Ota, D. Morikawa, and M. Ohashi, Video Transcoding Proxy for 3Gwireless Mobile Internet Access, IEEE Communications Magazine, pp. 66-71, 2000.

[94] S. Buchinger, and H. Hlavacs, "Subjective Quality of Mobile MPEG-4 Videos

with Different Frame Rates", Journal of Mobile Multimedia, Rinton Press, Vol. 1, No. 4, pp. 327-341, 2005.

[95] T. Shanableh, and M. Ghanbari, “Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding formats, IEEE Transactions on Multimedia, Vol. 2, No 2, pp. 101-110, 2000.

[96] P. Correia, S. Faria, M. Assuncao, “Matching MPEG-1/2 Coded Video to Mobile Applications,” 4th International Symposium on Wireless Personal Multimedia Communications, pp. 699-704, 2001.

[97] M. El-Barachi, R. Glitho and R. Dssouli, "Developing Applications for Internet Telephony: A Case Study on the Use of Web Services for Conferencing in SIP Networks", International Journal of Web Information Systems, Vol. 1, No. 3, pp.58-68, 2005.

[98] J. Tang, A. Liew, and H. Yan, "Human Face Animation Based on Video Analysis, with Applications to Mobile Entertainment", Journal of Mobile Multimedia, Vol. 1, No. 2, pp. 133-148, 2005.

[99] P. Papadimitriou, and V. Tsaoussidis,"On Transport Layer Mechanisms for Real-Time QoS", Journal of Mobile Multimedia, Vol. 1 , No. 4, pp. 342-363, 2005.

[100] G. Kessman, R. Hellinghuizen, F. Hoeksma, and G. Heidman, "Transcoding of MPEG bitstreams," International Journal of Signal Processing: Image Communications, Vol. 8, No. 6, pp. 481-500, 1996.

[101] S. Chang, and D. Messerschmidt,"Manipulation and compositing of MC-DCT compressed video," IEEE J. Select. Areas Comm., Vol. 13, No. 1, pp.1-11, 1995.

[102] H. Sun, A. Vetro, J. Bao, and T. Poon, “A new approach for memory-efficient

ATV decoding," IEEE Trans. Consumer Electron., Vol. 43, No 16 pp.517-525, 1997.

[104] P. Assuncao, and M. Ghanbari,"Congestion control of video traffic with transcoders, proc. IEEE International Conference on Communications, pp. 523-527, 1997.

[105] J. Youn, and M. Sun,"Motion estimation for high performance transcoding," in IEEE Int. Conf. Consumer Electronics, pp. 89-96, 1998.

[106] N. Bjork, and C. Christopoulos,"Transcoder architecture for video coding," IEEE Trans. Consumer Electron., Vol. 44, pp. 88-98, 1998.

[107] J. Mantyjarvi, S. Kallio, P. Korpipaa, J. Kela, and J. Plomp, "Gesture Interaction for Small Handheld Devices to Support Multimedia Applications", Journal of Mobile Multimedia, Vol. 1, No.2, pp. 92-111, 2005.

[108] M. Barry, J. Gutknecht, I. Kulka, P. Lukowicz, and T. Stricker," From Motion to Emotion: A Wearable System for the Multimedia Enrichment of A Blutooh Dace Performance", Journal of Mobile Multimedia, Vol 1, No. 2, pp. 112-132, 2005.

[109] J. Youn, M. Sun, and C. Lin, "Motion Vector Refinement for High- Performance Transcoding," IEEE Transactions on Multimedia, Vol. 1, No. 1, pp. 30-41, 1999.

[110] N. Björk, and C. Christopoulos, "Transcoder architectures for video coding," in Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing, pp. 2813-2816, 1998.

[111] Y. Huang, L. Hui, "an adaptive spatial filter for additive Gaussian and impulse noise reduction in video signals," in the proceeding of International Conference on Information, Communications & Signal Processing, IEEE Pacific-Rim Conference On Multimedia, pp. 402-406, 2003.

[112] S. Chang, And D. Messerschmitt, “Manipulation and Compositing of MC-DCT Compressed Video,” IEEE Journal. On Selected Areas In Communications, Vol. 13, No. 1, pp. 54-66, 1995.