Journal of Mobile Multimedia, Vol. 2, No.2 (2006) 167-186

A NOVEL APPROACH FOR IMPROVING THE QUALITY OF SERVICE FOR WIRELESS VIDEO TRANSCODING

ASHRAF M. A. AHMAD

Department of Computer Science and Information Engineering

National Chiao-Tung University, 1001 Ta-Hsueh Rd, Hsinchu, Taiwan

ashraf@csie.nctu.edu.tw

SUH-YIN LEE

Department of Computer Science and Information Engineering

National Chiao-Tung University, 1001 Ta-Hsueh Rd, Hsinchu, Taiwan

Received February 22, 2006
Revised March 30, 2006

To deliver streaming video over wireless networks is an important component for most interactive multimedia applications running on personal wireless handset devices. Such personal 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 device. 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 have been saved. Exceptional performance is demonstrated in the experiment results, as these experiments were designed to verify and prove the robustness of the proposed approach.

Key words: Wireless and Mobile Computing, Video Transcoding, Quality of Service, Motion Estimation
Communicated by: I. Ibrahim

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 to communicate 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 [10]. 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 [19]. Figure 1 presents a general diagram for a typical video transmission system. In this diagram, video wireless transmission emerges as an essential component for all types of video transmission systems.

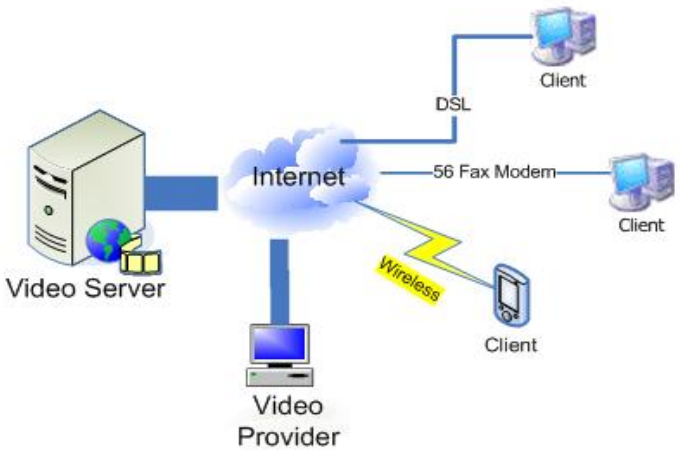

Figure 1. General Video Transmission Architecture.

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 [11, 27].

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 [3] 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 [22, 7, 16, 25]. 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. Although several compression techniques exist, MPEG 1/2/4 standards are among the most widely used compression algorithms for network video

applications. 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, [20] 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.

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 so called 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 MPEG-2 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. With the expansion and diversity of multimedia applications and the present communication infrastructure comprising of different underlying networks and protocols, 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.

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 [2, 9, 13, 18, 23, 24, 26]. 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 that is required by the network. 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 [2,14]. Bit-rate scaling can accommodate deficiency in available bandwidth. Resolution conversion [7,9,13] can also accommodate bandwidth limitations, but is primarily used to account for known limitations in the user device, such as processing power, display constraints or memory capability. To ensure adaptability across hybrid networks, syntax conversion [8,21] 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 [30]. 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 figure 2. 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.

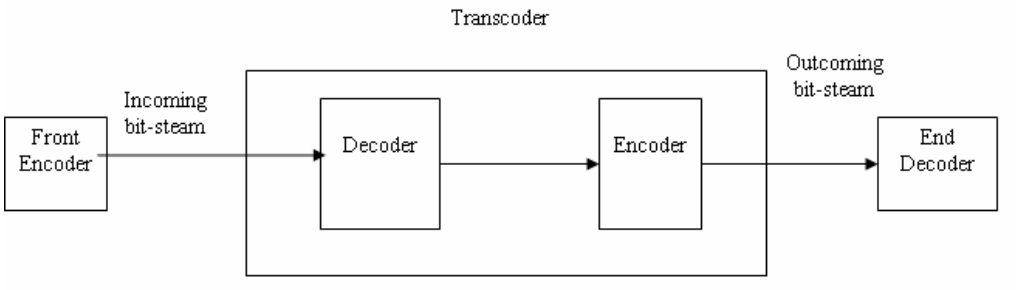

Figure 2. A typical transcoder architecture

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 [21], [9] 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. 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 [1] [28], [5] are not adequate. Figure 3 states the basic scheme of these approaches. They would produce an unacceptable texture distortion in the reconstructed signals, i.e., very low quality of service (QoS) delivered to the end user, which may not result in the best quality. Although an optimized motion vector can be obtained by full-scale motion estimation [17, 4], this is not desirable because of its high computational complexity.

Another related work, was suggested by [29], as they 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 draws in figure 4.

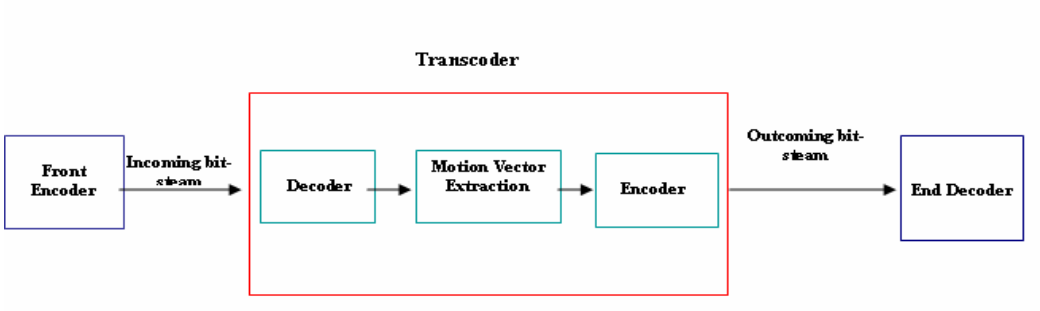

Figure 3. General scheme for motion vector reuse transcoding

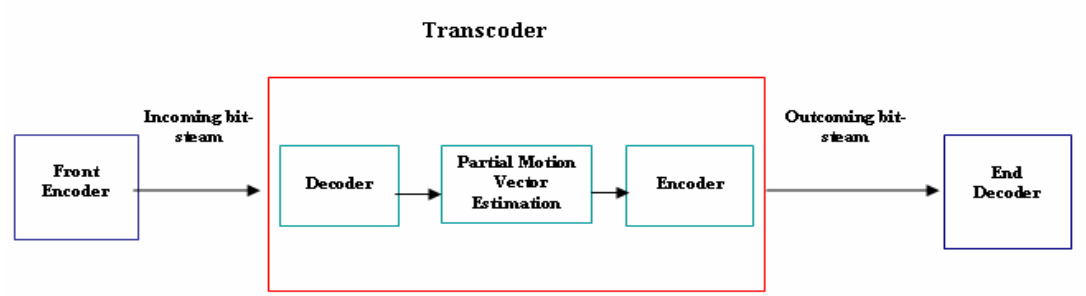

Figure 4. video transcoding scheme for partial motion vector estimation

In this paper, we consider the cascaded architecture as our 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 [5], [6][29]. 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 paper, we propose a new motion vector refinement scheme for a transcoder using the MPEG1/2 [3] 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 when compared to these approaches.

The reuse of 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 [2, 18, [23]. However, as proved in [5,6,29], 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. [29] 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.

The remainder of this paper is organized as the following. Section two draws a general overview of the proposed system. Section three analyzes the relationship between motion estimation and

transcoding. Section four presents the proposed scheme and verifies its theoretical validation. Section five expresses a general architecture for deploying the transcoded mechanism in mobile and wireless video streaming. Section six presents the results, analyzes the proposed scheme and results of related works. Section seven concludes this paper and describes future work.

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 Figure 5. 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. 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. These modules function is to produce fine motion vector from the motion vector which has been extracted. 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 user or client. End user is supposed to have end decoder to be able to render new video.

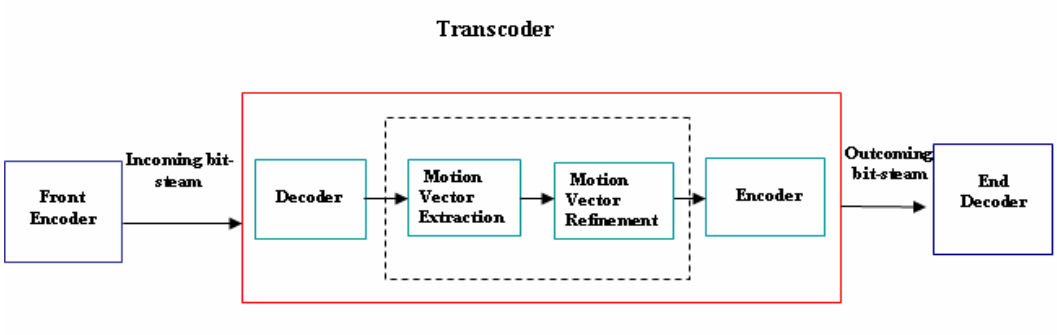

Figure 5. The system overview

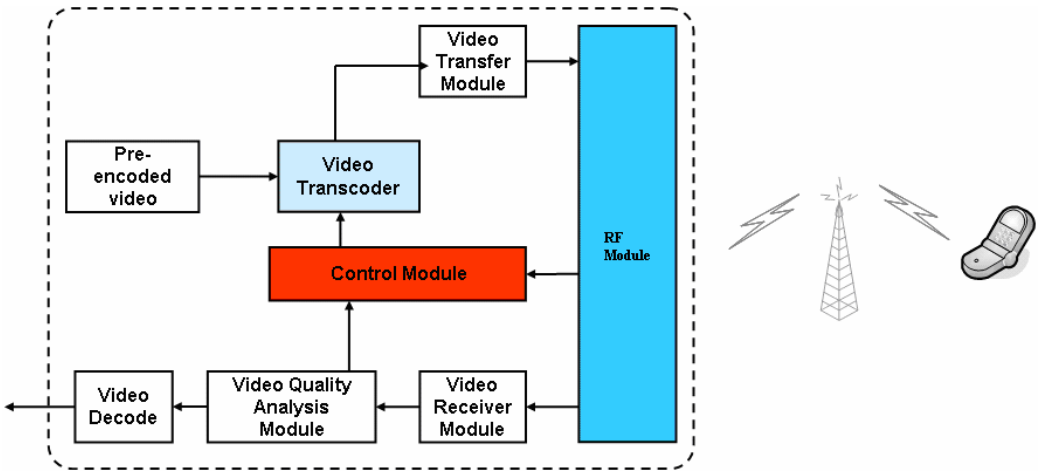

Figure 6. Overview of Mobile and Wireless Video Transcoding System Architecture

In order to state the mechanism to deploy our proposed method for high performance transcoding in mobile or wireless communication we propose the general system architecture for mobile and wireless video transcoding.

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 Figure 6, as the proposed system contains four important parts which are 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.

3.1 User Profile

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 behaviour. 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.

3.2 Video Transcoder

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 result we present in section 6, 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 in section 6. 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.

3.3 Control Module

The control module is responsible for creating a transcoding scheme according to the user profile and other information. The transcoding scheme will include some transcoding parameters. 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.

4 Motion Estimation in Transcoding

Current video compression techniques [3] 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 figure 7.

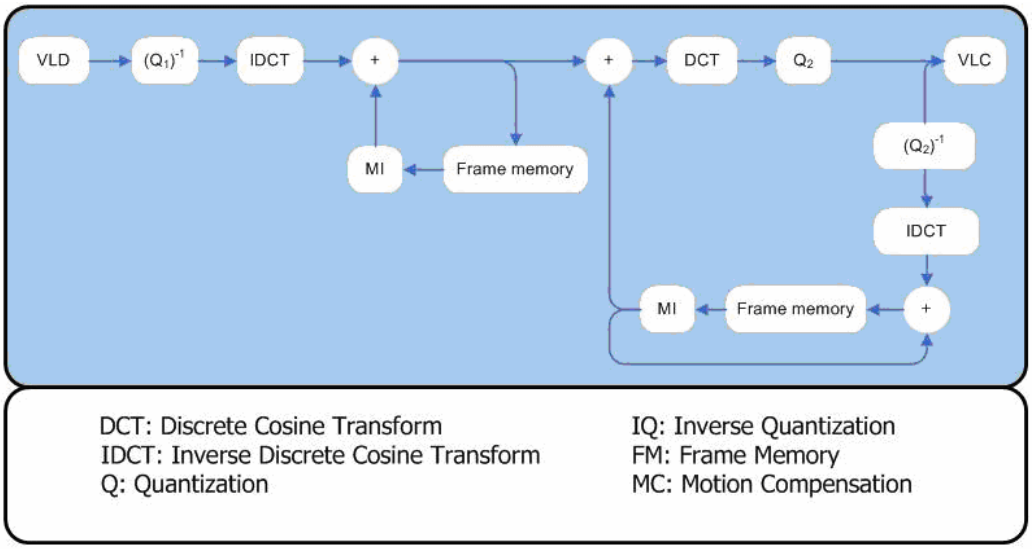

Figure 7. Cascaded pixel domain transcoder

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". Naturally, since the output bitrate is lower than the input bitrate, the quantization step size in Q2 in the transcoder is usually much more coarse than the quantizer step size in Q1 in the front encoder. According to [29], 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 we can be found.

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 [2, 18, 23] [extraction tool]. However, as discussed in the previous section, and proved by [29,5,6] simply reusing the incoming motion vectors is not optimal.

Simulation results (which will be presented in the later sections) show that its performance may be significantly worse than that which can be achieved with new motion estimation or our proposed motion vector refinement. In [29,5,6]'s analysis they showed that the differential reconstruction error causes incoming motion vectors to deviate from optimal values.

Therefore, a fine motion vector can be obtained by refining the incoming motion vector according to our proposed scheme as opposed to applying a full-scale motion estimation [5] or partial scale motion estimation [29].

Depending on the application, various types of noise can be distinguished. White 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 paper we will focus on the white Gaussian additive noise, and optionally try to defeat the impulsive noise effects.

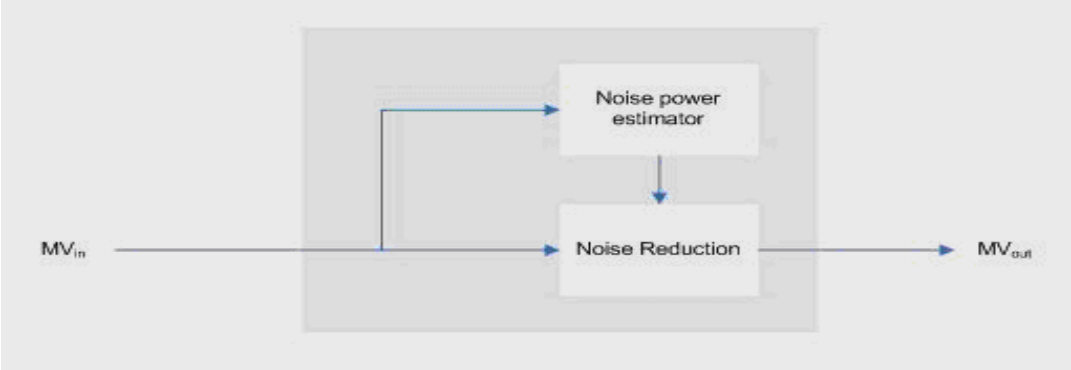

Figure 8. The overview architecture of the proposed scheme

In Figure 8, we introduce a general view of the proposed 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 Figure 9. It is a recursive spatial filter and includes a kernel and weighting block. The kernel consists of the motion vectors MV (i) (i=0, l...8) where MV(0) is the

central motion vector that will be filtered MV(1)-MV(8) are the neighbourhoods of MV(0). The kernel which is 2D window of motion vectors is shown in Figure 10.

Figure 9. Block diagram of the filter

MV_1	MV_2	MV_3
MV_4	MV_0	MV_5
MV_6	MV_7	MV_8

Figure 10. Working window

In the weighting block, the output motion vector can be calculated by:

$$MV_{out} = \frac{\sum_{i \in N_1} W_1(i) * MV'(i) + \sum_{i \in N_2} W_2(i) * MV(i)}{\sum_{i \in N_1} W_1(i) + \sum_{i \in N_2} W_2(i)} \quad (1)$$

Where N1 = (1, 2, 3, 4) and N2 = (5, 6, 7, 8) are sets of vectors defining a neighbourhood. And the weighs of each motion vector are defined as:

$$W_1(i) = z_1 K(i) \qquad i \in N_1 \quad (2)$$

$$W_2(i) = z_2 K(i) \qquad i \in N_2 \quad (3)$$

Where z1 and z2 are the parameters related to the position of the weighted Motion vector. To use more information of filter 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)). This function is defined by:

$$K(i) = e^{-\left(\frac{MV(i)-(MV(0)+\lambda)}{Noise\ Power}\right)^2} \tag{4}$$

Where λ is a parameter related to the local noise power of current input motion vector. It is ranged at $-Noise\ Power \le \lambda \le Noise\ Power$. This parameter can be estimated by:

$$\lambda = \arg\min_{\lambda_j}\left\{\sum_{i\in N_1}\left|MV'(i)-(MV(0)+\lambda_j)\right|+\sum_{i\in N_2}\left|MV(i)-(MV(0)+\lambda_j)\right|\right\} \tag{5}$$

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. In the case of central motion vector corrupted by impulse noise, the sum of the weights is taking over the Local neighbourhood, excluding the centre motion vector itself allows for good reduction of impulse noise.

5.1 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 block.

The Noise Power Estimator is used to estimate the noise power for every frame processed. Noise power values are to be used in the noise reduction block. The difference of every motion vector with its previous and next motion vector is calculated and the accumulated differences are delayed for 4 motion vectors and further accumulated into a variable entitled AC_DIFF. The Noise Power of each frame is updated as:

$$Noise\ Power(f) = \begin{cases} Noise\ Power(f)-1 & \text{NPI} > Thershold\ \Lambda Noise\ Power(f) \\ Noise\ Power(f)+1 & \text{NPI} \le \text{T}hershold\ \Lambda Noise\ Powe(f) \end{cases} \tag{6}$$

Where Noise Power (f) is the noise power of f frame and the threshold is an experimentally optimized constant defined as:

$$Threshold = (Number\ of\ M\text{otion}\ V\text{ector} * 60)/100 \tag{7}$$

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

The Noise Power Indicator (NPI) is defined as:

$$NPI = \sum_{MV} B(MV) \tag{8}$$

Where B(MV) is a binary variable assigned to every motion vector according to:

$$B(MV) = \begin{cases} 1, & Noise\ Power(f) < AC\_DIFF(MV) < 1.5 Noise\ Power(f)^{+} \\ 0, & else \end{cases} \tag{9}$$

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

$$AC\_DIFF(MV) = \sum_{n=1}^{4} \left| F(MV+n) + F(MV+n-1) \right| \tag{10}$$

So, count equals the number of times per field that the local noise estimate AC_DIFF(MV) lies within an interval, and this interval is shifted until the count equals a predefined Threshold value.

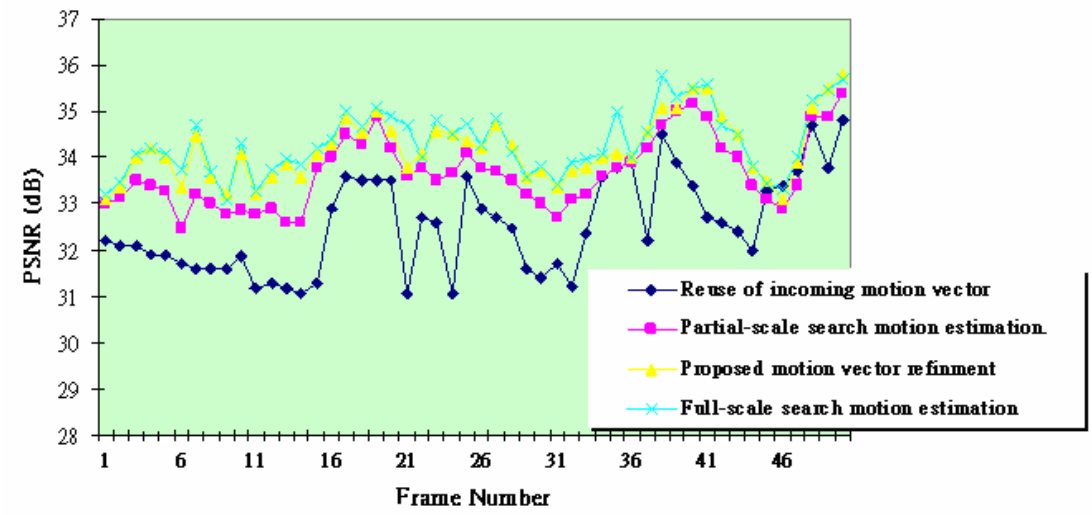

Figure 11. Performance of motion vector refinement ("Claire of QCIF format. Process 50 frames) Incoming but rate at 128 Kb/s was transcoded t0 32 Kb/s with 30 frames/s.

Applying the aforementioned scheme would result in fine motion vector values according to the performance result of implementing the proposed scheme. Figure 11 shows the performance of motion vector refinement. The quality degradation introduced by reusing incoming motion vectors is about 0.40 dB on average compared with the application of a full-scale search motion estimation and 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 has therefore been increased, by taking the noise weighting curve into account. Table 1 provided 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.

Table 1 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			

6 Result and Discussion

We have designed an experiment in order to verify the performance of the proposed scheme. The experiment has been designed to test the proposed scheme on several video clips. These video clips are in MPEG format and are part of the MPEG7 testing dataset and other video clips which have been widely used in such applications performance evaluation.

In all the simulations presented in this paper, test sequences of QCIF (176 144) 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. 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 hard 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 our experiment design. However, the idea of the proposed scheme can also be applied to B-frames.

Comparison is held among four types of work which are, transcoding scheme using full-scale motion estimation (FSME), transcoding scheme using proposed motion vector refinement (PMVR), transcoding scheme using re-use motion vector (RUMV) and finally transcoding using partial-scale motion estimation (PSME).The peak signal-to-noise ratio (PSNR) is taken to measure the video quality. We choose PSNR because is most commonly used to evaluate such system performance [23, 24, 29, 12].

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 follows:

$$MSE = \frac{\sum \left[f(x,y) - \tilde{f}(x,y) \right]^2}{N^2} \tag{11}$$

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

$$PSNR = 20 \log_{10} \left(\frac{255^2}{\sqrt{MSE}} \right) \tag{12}$$

In our result we present the PSNR values for Y components due to the paper space and readability issues. As we are emphasizing the processing performance and its importance to the handheld device, we measure the speed-up result according to the following equation.

$$Speed - up = \frac{Excution - Time(Y)}{Excution - Time(x)} \tag{13}$$

For comprehensive description sake, we provide simulation environment table 2. The CPU, memory and hard-disk capability have been listed under Hardware. The operating system, programming environment and profiler tool have been listed under Software.

Table 2. Simulation Environment

Hardware			**Software**		
CPU	**Memory**	**Hard Disk**	**Operating System**	**Programming Language**	**Profiler Tool**
Pentium-3 1GHz	256 SDRAM	10025 RPM	Windows XP Professional	ANSI C	NuMega TrueTime 2.1

Figures 12, 13 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.

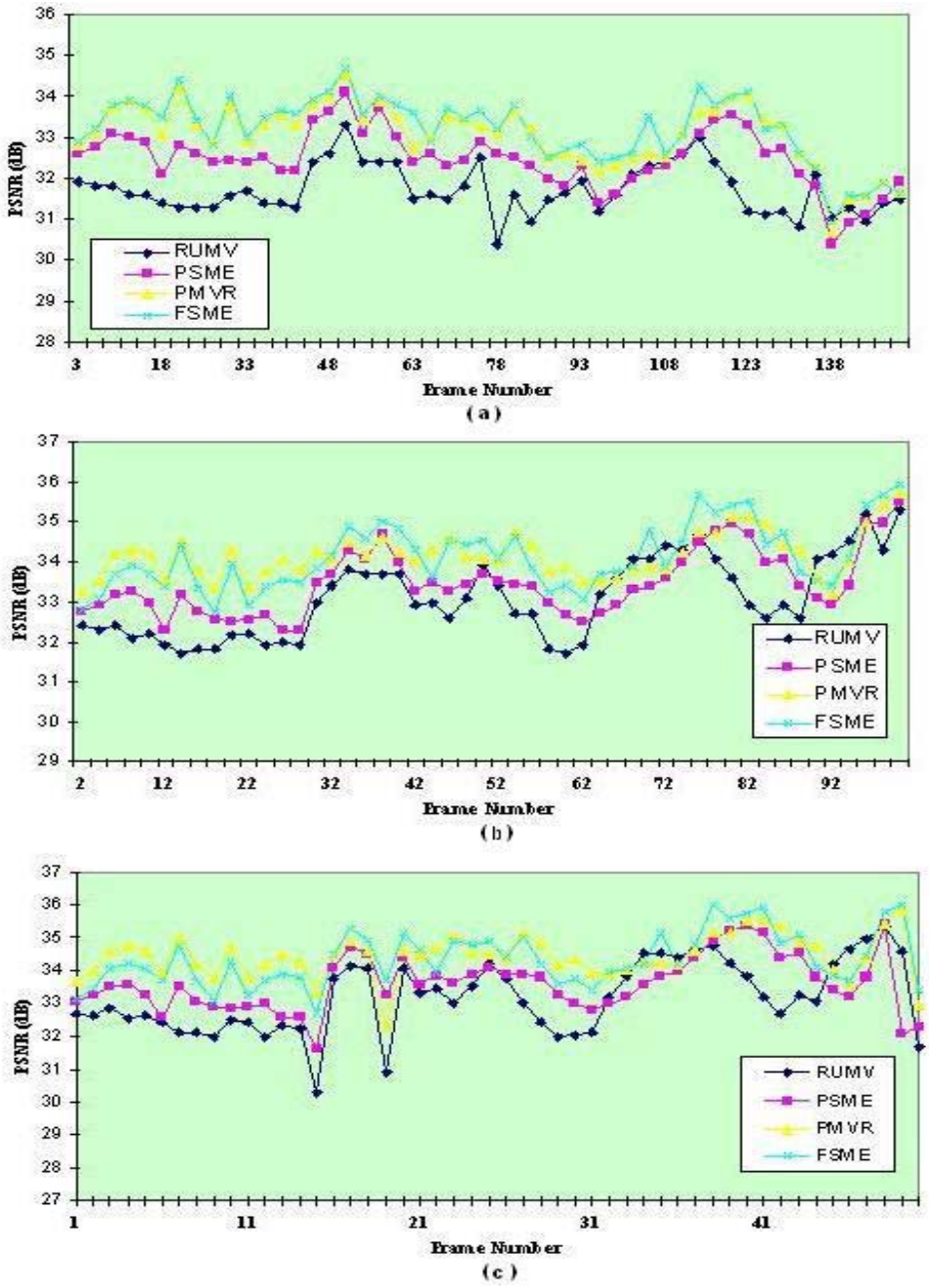

Figure 12. Performance of the proposed motion vector refinement against related works when frame rate was changed: (a) to 30 frames/s, (b) to 15 frames/s and (c) to 10 frames/s. Mother and daughter of 250 frames was encoded at 128 Kb/s with 30 frames/s, and then transcoded to 32 Kb/s. (a) outgoing frame-rate: 30 frames/s. (B) outgoing frame-rate: 15 frames/s and (C) outgoing frame-rate: 10 frames/s.

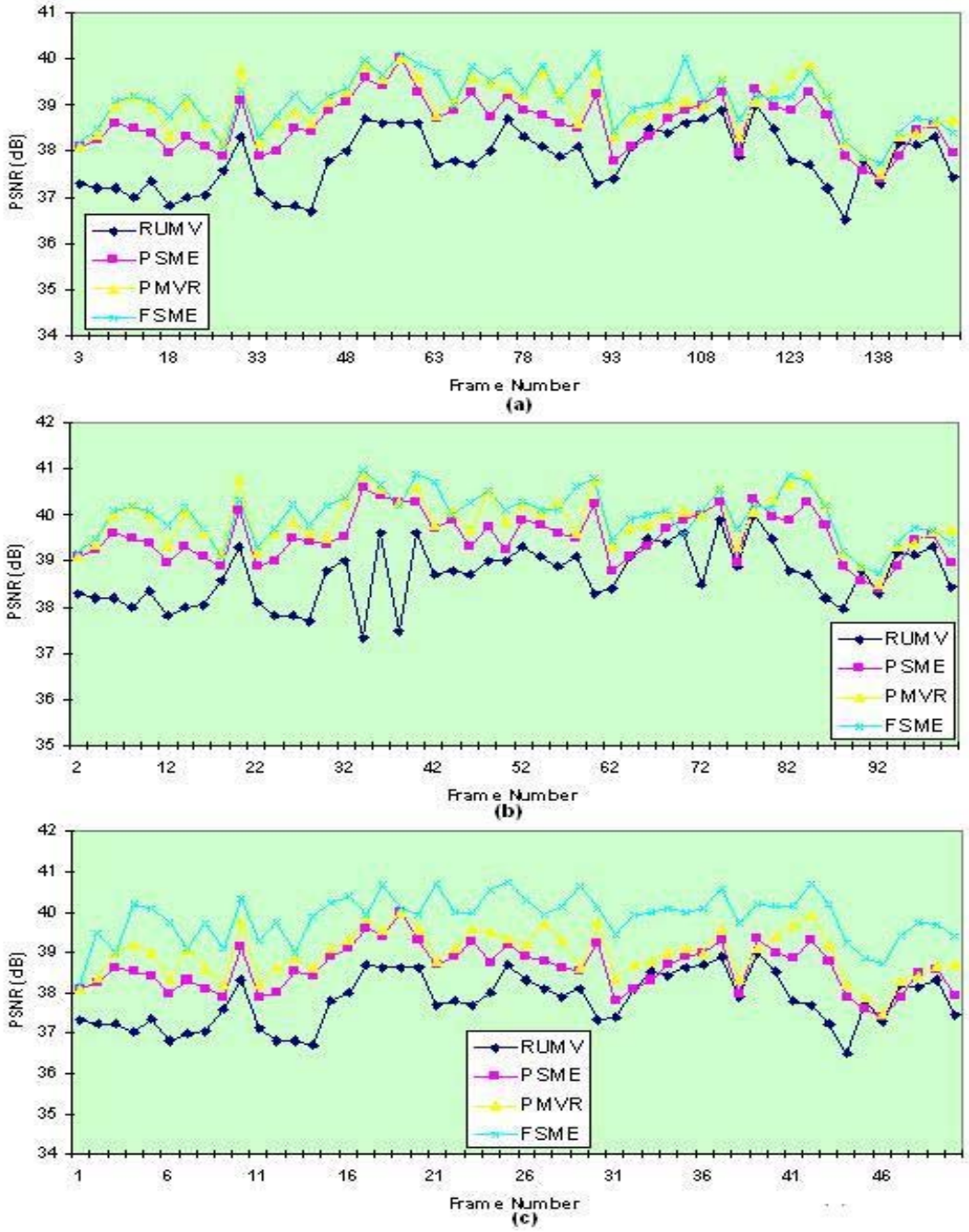

Figure 13 Performance of the proposed motion vector refinement against related works when frame rate was changed: (a) to 30 frames/s, (b) to 15 frames/s and (c) to 10 frames/s. Coast Guard sequence of 300 frames was encoded at 128 Kb/s with 30 frames/s, and then transcoded to 64 Kb/s. (a) outgoing frame-rate: 30 frames/s. (B) outgoing frame-rate: 15 frames/s and (C) outgoing frame-rate: 10 frames/s.

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 figure 12's results 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 rest of this section. Similarly, the presented results in figure 13 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 Tables 3, 4 as the performances of the FSME, PMVR, RUMV and PSME transcoding were compared. According to equation (x), speed up was computed. 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 in the opposed of full or partial scale motion estimation, as motion estimation is very time and power consuming. Even, PMVR has outperformed 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 3 Possessing performance 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 4 Possessing performance 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 Conclusion and Future work

In this paper, 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 which are 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 to find the optimal motion vectors which boosts the quality of the transcoding mechanism in favour of the high computation power consumed to realize these mechanisms. In this paper, 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 the future; we will use our proposed scheme in adopting the method in [15] 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. With the utilization of these normalized motion vectors in our system, the performance is anticipated to be higher. In addition, our transcoding mechanism have been applied in MPEG domain videos only, 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 very interesting issue to be addressed for mobile TV field.

Acknowledgements

The authors would like to express their appreciation for the editorial board of MoMM 2005 Journal special issue and Mr. Eric Pardede, regarding their helpful comments, reviews, and responsible prompt responses. In addition we wish to express our thanks for Mr. Ahmed (CoCo) Ma (Wen Hao, Ma, ahmed.coco.ma@gamil.com) for his wonderful proofreading our paper.

References

1. Assuncao P., and Ghanbari, M., Congestion control of video traffic with transcoders, IEEE International Conference on Communications, ICC'97, pp.523-527, Montreal - Canada, 1997.

2. Assuncao, P., and Ghanbari, M., Post-processing of MPEG2 coded video for transmission at lower bit rates, in ICASSP'96, vol. 4, pp.1998-2001, 1996.

3. Buchinger, S., and Hlavacs, H., 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.

4. Barry, M., Gutknecht, J., Kulka, I., Lukowicz, P., and Stricker, T., From Motion to Emotion: A Wearable System for the Multimedia Enrichment of A Butoh Dace Performance, Journal of Mobile Multimedia, Rinton Press, vol. 1, no. 2, pp.112-132, 2005.

5. Bjork N., and Christopoulos, C., Transcoder architecture for video coding, IEEE Trans. Consumer Electron., vol. 44, pp.88-98, Feb. 1998.

6. Björk, N., and Christopoulos, C., Transcoder architectures for video coding, in Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing, Seattle, WA, pp.2813-2816, 1998.

7. Correia, P., Faria, S. M., Assuncao, P. A., Matching MPEG-1/2 Coded Video to Mobile Applications, 4th International Symposium on Wireless Personal Multimedia Communications, Vol. 2, pp.699-704, Aalborg - Denmark, 2001.

8. Chang, S.F., and Messerschmidt, D.G., Manipulation and compositing of MC-DCT compressed video, IEEE J. Select. Areas Commun., vol. 13, pp.1-11, 1995.

9. Eleftheriadis, A., and Anastassiou, D., Constrained and general dynamic rate shaping of compressed digital video, in Proc. IEEE Int. Conf. Image Processing, Washington, DC, 1995.

10. Hong Va Leong and Antonio Si, Multi-resolution information transmission in mobile environments, Mobile Information Systems, IOS Press, vol. 1, no. 1, pp. 25 - 40, 2005.

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

12. Huang, Y., Hui, L. An adaptive spatial filter for additive Gaussian and impulse noise reduction in video signals, in the proceeding of ICICS PCM 2003, pp. 402-406.

13. Keesman, G., et al., Transcoding of MPEG bitstreams, Signal Process. Image Comm., vol. 8, pp. 481-500, 1996.

14. Kessman, G., Hellinghuizen, R., Hoeksma, F., and Heidman, G., Transcoding of MPEG bitstreams, Signal Processing: Image Communications, vol. 8, no. 6, pp. 481-500, 1996.

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

16. May El Barachi, Roch Glitho and Rachida 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, Troubador Publishing, vol. 1, no. 3, 2005.

17. Mantyjarvi, J., Kallio, S., Korpipaa, P., Kela, J., and Plomp, J., Gesture Interaction for Small Handheld Devices to Support Multimedia Applications, Journal of Mobile Multimedia, Rinton Press, vol. 1, no.2, pp. 92-111, 2005.

18. Morrison, D. G., Nilsson, M. E., and Ghanbari, M. Reduction of the bit-rate of compressed video while in its coded form, in Proc. Sixth Int. Workshop Packet Video, Portland, OR, 1994.

19. Nguyen, T. M., Brezany, P., Tjoa, A. M., and Weippl, E., Toward a Grid-Based Zero-Latency Data Warehousing Implementation for Continuous Data Streams Processing, International Journal of Data Warehousing and Mining, Idea Group Inc., vol. 1, no. 4, pp. 22 - 55, 2005.

20. Papadimitriou, P., and Tsaoussidis, V., On Transport Layer Mechanisms for Real-Time QoS, Journal of Mobile Multimedia, Rinton Press, vol. 1 , no. 4, pp. 342-363, 2005.

21. Sun, H., Vetro, A., Bao, J., and Poon, T., A new approach for memory-efficient ATV decoding, IEEE Trans. Consumer Electron., vol. 43, pp.517-525, 1997.

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

23. Safranek, R. J., Kalmanek, C., and Garg, R., Methods for matching compressed video to ATM networks, in Proc. Int. Conf. Image, Washington, DC, 1995.

24. Sun, H., Kwok, W., and Zdepski, J. W., Architecture for MPEG compressed bitstream scaling, IEEE Trans. Circuits Syst. Video Technol., vol. 6, pp.191-199, 1996.

25. Tang, J. S.-S., Liew, A. W.-C., and Yan, H., Human Face Animation Based on Video Analysis, with Applications to Mobile Entertainment, Journal of Mobile Multimedia, Rinton Press, vol. 1, no. 2, pp.133-148, 2005.

26. Tudor, P. N., and Werner, O. H., Real-time transcoding of MPEG-2 video bit streams, in Proc. Int. Broadcasting Conv., Amsterdam, The Netherlands, pp.286-301, 1997.

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

28. Youn, J., and Sun, M.-T., Motion estimation for high performance transcoding, in IEEE Int. Conf. Consumer Electronics, Los Angeles, CA, 1998.

28. Youn, J., Sun, M-T., and Lin, C-W., Motion Vector Refinement for High-Performance Transcoding, IEEE TRANSACTIONS ON MULTIMEDIA, vol.1, no.1, pp.30-41, 1999.

30. Youn, J., and Sun, M.-T., Video Transcoding with H.263 Bit-Streams, Journal of Visual Communication and Image Representation, 11, 2000.