The Hong Kong Polytechnic University

Department of Electronic and Information Engineering

Techniques and Algorithms for Video Transcoding

Kai-Tat Fung

A thesis submitted in partial fulfillment of the requirements for the Degree of Doctor of Philosophy

July 2005

Certificate of Originality

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____(Signed)

Fung Kai Tat (Name of student)

Abstract

Video transcoding becomes an important role for a video server to provide quality support services to heterogeneous clients or transmission channels. Video transcoding can be viewed as a process of converting a previously compressed video bitstream into a bit stream of different nature or lower bitrate bitstream. From this point of view, the focus introduced in this thesis is a ***strategy*** to convert the previously encoded video into other encoded format or reducing the bitrate of the encoded video according to the bandwidth requirement. To achieve this purpose, there are different ways to perform video transcoding. In this thesis, we will first look at the conventional video transcoder using three different techniques: frame skipping, video downscaling and requantization of DCT coefficients. The most straightforward method to implement these approaches is to cascade a decoder and an encoder, commonly known as pixel-domain transcoding. The input video bitstream is decoded in the pixel domain, and the decoded video signal is re-encoded at the desired output bit rate using frame skipping, requantization or video downscaling technique according to the capability of the clients' devices and the available bandwidth of the network. The major problem in this approach is that high processing complexity, large memory size, long delay and re-encoding errors will be introduced.

In the first half of the thesis, our studies focus on the homogeneous video transcoding. A DCT-based video frame skipping transcoder is proposed to provide low computational complexity. Using the proposed direct addition of DCT coefficients technique, the DCT coefficients can be re-estimated directly in the DCT domain without performing full decoding and re-encoding process. For the motion compensated

macroblock video transcoding, a DCT coefficient re-composition algorithm is proposed to reconstruct the re-estimated DCT coefficients from the pixel domain data and the transform domain data.

For video downscaling, a new architecture to obtain resampled DCT coefficients in the DCT domain by using the split and merge technique is proposed. In this approach, the macroblock is splitted into the dominant region and the boundary region. Then, the dominant region of the macroblock can be transcoded in the DCT domain using the proposed transcoding operators to achieve low computational complexity. By transcoding the boundary region adaptively, the computational complexity can be reduced and the re-encoding error introduced in the boundary region can be controlled more dynamically.

In the second half of the thesis, we address problems of heterogeneous video transcoding which is to transcode the video from one format to another format. A new compressed-domain heterogeneous video transcoder is proposed to convert a B-picture into a P-picture by making use of the techniques of motion compensation in the DCT domain and indirect addition of DCT coefficients. A set of equations, the formulation of the problem and solution are given in this new research area. Since the B-picture is transcoded to the P-picture, this new reconstructed P-frame will be used as a reference frame for the next incoming frame. Therefore, the next incoming P-frame needs to be transcoded in order to reference to the transcoded P-picture. A fast transcoding using the backward subtraction of the DCT coefficients is proposed to transcode the P frame in the DCT domain to achieve low computational complexity.

The H.264 video coding standard has been filed recently. This creates an important need for transcoding technologies that transcode the widely available H.263 compressed videos to H.264 compressed format and vice versa. Direct conversion of DCT coefficients to Integer Transform(IT) coefficients in the compressed domain is proposed for the H.263 to H.264 video transcoder. A set of operators is derived for converting the DCT coefficients to integer transform coefficients in the compressed domain directly. Also, a set of operators is proposed to obtain the motion-compensated integer transform coefficients(MC-I) form the incoming DCT coefficients directly for inter frame vide conversion. An approximation form of the DCT coefficients is proposed to speed up the transcoding process for low bitrate applications. In addition, the integer approximation of the operators is proposed to avoid the floating point implantation for the proposed transcoder.

It is exciting to report in this thesis that significant gains in terms of computation and high quality of video can be achieved by employing our proposed approaches. Undoubtedly, these advance techniques can enable the video transcoding to become more efficient and provide good quality video in practical situations.

Acknowledgements

I would like to express my greatest thank to my Supervisor, Professor W.C. Siu, for his continuous encouragement, guidance and care during the period that I worked on this thesis. He spent much of his invaluable time with me discussing my research, reviewing and correcting my articles and drafts of this thesis. His profound knowledge and experience in digital signal processing, his rigorous approach to research, many interesting and stimulating lectures that he has given, and his hard working style and devotion to science and reaserch, have inspired me during my work on the thesis. This will continually influence my future research and career.

I would like to thank all colleagues in the Mutlimedia Signal Processing Centre for their friendship, support and encouragement, especially, Dr. Chris Chan, Dr. Y.L. Chan, Dr. W.K Cheuk, Mr. W.L. Hui, Dr. T.C. Hung, Dr. Kenneth Lam, Dr. Bonnie Law, Dr. Daniel Lun, Mr. K.W. Wong and Mr. W.H. Wong and Dr. Michael Yiu.

It is my pleasure to acknowledge the Research Degrees Committee of the Hong Kong Polytechnic University for its generous support over the years.

Above all, I thank my family for their constant love, encouragement and support. Without their understanding and patience, it is impossible for me to complete this research study.

Table of Contents

List of Tables

List of Figures

Statement of Originality

The following contributions reported in this thesis are claimed to be original.

1. **New architecture for dynamic frame-skipping transcoder:** A direct addition of the DCT coefficients for macroblocks without motion compensation (non-MC macroblocks) is proposed. (Chapter 3, section 3.1)

   The proposed architecture is mainly performed on the discrete cosine transform (DCT) domain to achieve a low complexity transcoder. It is observed that the re-encoding error is avoided at the frame-skipping transcoder when the strategy of direct addition of DCT coefficients is employed. By using the proposed frame-skipping transcoder, the video qualities of the transcoded video sequences can be improved significantly.

2. **On Re-composition of Motion Compensated Macroblocks for DCT-based Video Transcoding:** Recomposition of MC macroblocks from the dominant region and the boundary region of MC macroblocks in the DCT domain using a shifted version of the dominant motion vector is proposed. (Chapter 3, section 3.2)

   The new architecture transcodes the dominant region of a motion compensated macroblock in the DCT domain by making use of the DCT coefficients of the incoming bistream and some pre-computed shift operators. By using a shifted version of the dominant vector, the re-encoding error introduced in the dominant region can be avoided. On the other hand, an adaptive transcoding architecture to transcode the boundary regions of MC marcoblocks and a way to perform error

compensation are proposed. This architecture can further speed up the transcoding process of the motion compensated macroblocks.

3. **DCT-based video downscaling transcoder using split and merge technique:** A new architecture to obtain resampled DCT coefficients in the DCT domain by using the split and merge technique is proposed. (Chapter 4, section 4.1)

   Using our proposed video transcoder architecture, a macroblock is splitted into two regions: dominant region and the boundary region. The dominant region of the macroblock can be transcoded in the DCT domain with low computational complexity and re-encoding error can be avoided. By transcoding the boundary region adaptively, low computational complexity can be achieved. More importantly, the re-encoding error introduced in the boundary region can be controlled more dynamically. A fast algorithm is also proposed to further speed up the transcoding process.

4. **Diversity and importance measures for video transcoding:** An adaptive motion vector re-composition algorithm using two new measures: the diversity and importance measures of motion vectors are proposed. (Chapter 4, section 4.2)

   Using the importance measure, our proposed scheme manages to differentiate the most representative motion vector as a consideration to recompose a new motion vector. In addition, the diversity measure provides information for the video transcoder controlling the size of the refinement window to achieve a significant reduction of computational complexity.

5. **A new Compressed-domain heterogeneous video transcoder:** A new DCT-based heterogeneous video transcoding architecture is proposed. (Chapter 6, section 6.2)

   The new transcoder architecture is to convert a B-picture into a P-picture by making use of the techniques of motion compensation in the DCT domain and

indirect addition of DCT coefficients is proposed. We derive a set of equations and formulate the problem of how to obtain the DCT coefficients in this research work. The major saving is due to the fact that the estimated DCT coefficients can be obtained in terms of the incoming DCT coefficients and the proposed operators. Since the operators to perform the motion compensation in the DCT domain can be pre-computed and the DCT coefficients can be reused, the coefficients for B frame to P frame can be achieved in the DCT domain directly without performing the full decoding and re-encoding process.

6. **Low Complexity H.263 to H.264 video transcoding using motion vector decomposition:** An effective motion vector decomposition algorithm is proposed for transcoding videos from the H.263 format to the H.264 format. (Chapter 7, section 7.1)

   The algorithm concentrates on macroblocks which consume more bits in the H.263 video sequence. In other words, a non-optimal motion vector is splitted into smaller sizes since the H.264 supports various block types. Using the proposed prediction error measure and diversity of the motion vector measure, the motion vector can be decomposed adaptively and the amount of the prediction error can be minimized. After re-estimating the new set of motion vectors, multiple reference frames estimation is applied to further reduce the bitrate and prediction error. Since the non-optimal motion vector can easily be identified inside the proposed transcoder, it is effective to improve the transcoding efficiency to avoid computational redundance.

7. **Compressed domain DCT to integer transform conversion:** A new compressed domain video transcoder architecture is proposed to transcode the DCT coefficients to the integer transform coefficients in the compressed domain. (Chapter 7, section 7.2)

Two sets of operators are derived to transcode the DCT coefficients into Integer transform coefficients for cases (i) without motion compensation and (ii) with motion compensation for intra frame and inter frame transcoding. To further speed up the transcoding process, a fast approximation of the DCT coefficients is used. Since the operators contain floating point implementation, an integer approximation form of the operators are derived for easy implementation.

List of Publications

(List of Publications of the Author on which this thesis is based)

International Journal Papers

Paper published or accepted:

1. *Kai-Tat Fung, Yui-Lam Chan and Wan-Chi Siu, "New architecture for dynamic frame-skipping transcoder," IEEE Transactions on Image Processing, Vol.11, pp.886-900, Aug 2002, USA.*
2. Kai-Tat Fung and Wan-Chi Siu, "DCT-based video downscaling transcoder using split and merge technique," paper accepted, to be appeared on IEEE Transactions on Image Processing.
3. Kai-Tat Fung and Wan-Chi Siu, "On Re-composition of Motion Compensated Macroblocks for DCT-based Video Transcoding," paper accepted, to be appeared on Signal Processing: Image communication, Elsevier Science, The Netherlands.

Paper submitted or in preparation:

4. "A new Compressed-domain heterogeneous video transcoder," Paper in preparation, to be submitted to IEEE Transactions on Circuit and System for Video Technology/Multimedia.
5. "Compressed domain DCT to integer transform conversion," Paper in preparation, to be submitted to IEEE Transactions on Circuit and System for Video Technology/Multimedia.
6. "Efficient video transcoding for H.263 to H.264 using motion vector decomposition," Paper in preparation, to be submitted to IEEE Transactions on Circuit and System for Video Technology/Image processing/Multimedia.
7. "Efficient Video Downscaling in DCT domain using Diversity and importance measures," Paper in preparation, to be submitted to IEEE Transactions on Circuit and System for Video Technology/Image processing/Multimedia.

International Conference Papers

8. Kai-Tat Fung and Wan-Chi Siu, “Conversion between DCT coefficients and integer transform coefficients in compressed domain for H.263 to H.264 video transcoding,” paper accepted, to be appeared on Proceedings, IEEE International Conference on Image Processing (ICIP’2005)
9. Kai-Tat Fung and Wan-Chi Siu, “Low Complexity H.263 to H.264 video transcoding using motion vector decomposition,” IEEE International Symposium on Circuits and Systems (ISCAS’2005), pp.908-911, May, 2005, Kobe, Japan.
10. Kai-Tat Fung and Wan-Chi Siu, “Diversity and importance measures for video transcoding,” IEEE International Conference on Acoustics Speech, and Signal Processing(ICASSP2005), pp.1061-1064.PA, U.S.A.
11. Wan-Chi Siu, Kai-Tat Fung and Yui-Lam Chan, “A compressed-domain heterogeneous video transcoder,” Proceedings, IEEE International Conference on Image processing(ICIP), pp.2761-2764, Oct, 2004, Singapore.
12. Kai-Tat Fung, Yui-Lam Chan and Wan-Chi Siu, “Dynamic frame skipping for high performance transcoding,” Proceedings, IEEE International Conference on Image processing(ICIP), pp.:425 – 428, Oct., 2001
13. Kai-Tat Fung and Wan-Chi Siu, “DCT-based Video Frame-Skipping Transcoder,” Proceedings, IEEE International Symposium on Circuits and Systems (ISCAS’2003), Vol. 2, pp.656-659, May 2003, Bangkok Thailand.
14. Kai-Tat Fung and Wan-Chi Siu, “Architecture of Wavelet-based Frame Skipping Transcoder,” Proceedings, IEEE International Conference on Neural Networks and Signal Processing, Vol. 2, pp.1229-1232, December 2003, Nanjing China.
15. Kai-Tat Fung and Wan-Chi Siu “New DCT-domain transcoding using split and merge technique,” Proceedings, IEEE international conference on Image Processing, vol.1, pp.197-200, September 2003, Barcelona Spain.
16. Kai-Tat Fung, Wan-Chi Siu and Yui-Lam Chan, “A Dynamic Frame-Skipping video Combiner for multipoint video Conferencing,” Proceedings, IEEE International

Symposium on Circuits and Systems (ISCAS'2002), pp.389-392, May 2002, Scottsdale Arizona, USA.

17. Kai-Tat Fung, Wan-Chi Siu and Ngai-Fong Law, "A Dynamic video Combiner for multipoint video Conferencing using wavelet transform," Proceedings, IEEE International Conference on Multimedia and Expo (ICME'2002), vol.2, pp.17-20, August 2002, Lausanne, Switzerland.

Book chapter of video transcoding-A book chapter is being written together with my supervisor which is going to give a more detailed description and discussion on the conventional and the advanced video transcoding techniques.

Chapter 1

1.1 Overview of video transcoding

With the advance of video compression and networking technologies, networked multimedia services, such as multipoint video conferencing, video on demand and digital TV, are emerging. Video transcoding becomes an important role for a video server to provide quality support services to heterogeneous clients or transmission channels[1-20]. It is in this scenario that the video server should have the capability of performing video transcoding. The function of video transcoding is to convert a previously compressed video bitstream into a lower bitrate bitstream without modifying its nature according to the clients devices in terms of its computational complexity and bandwidth constraint as shown in Figure 1.1.

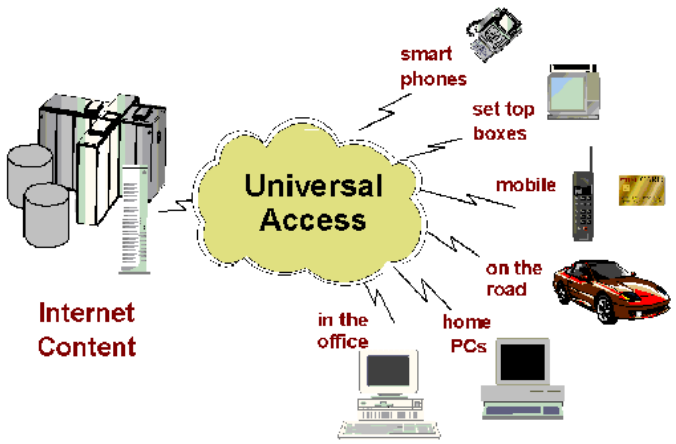


Figure 1.1 Video Transcoding application

Until now, there are three methods introduced in the literature for video bit-rate reduction: frame skipping[1,2,17,18,21-23], video downscaling[4,5,12,14-16,24.25] and requantization of DCT coefficients[3,6,9,13]. The most straightforward method to implement these approaches is to cascade a decoder and an encoder, commonly known as

pixel-domain transcoding[2]. The input video bitstream is decoded in the pixel domain, and the decoded video signal is re-encoded at the desired output bit rate using frame skipping, requantization or video downscaling technique according to the capability of the clients' devices and the available bandwidth of the network. The major problem in this approach is that high processing complexity, large memory size, long delay and re-encoding errors will be introduced.

In recent years, DCT-domain transcoding was introduced[1,8,9,13,18,26-29], under which the incoming video bitstream is partially decoded to the DCT coefficient level, and the transcoding technique is performed in the DCT domain. Since DCT-domain transcoding is carried out in the coded domain where complete decoding and re-encoding are not required, the processing complexity is significantly reduced. However, the problem with this approach is that the quantization errors will accumulate, and a prediction memory mismatch at the decoder will cause poor video quality. This phenomenon is called "drift" degradation, which often results in an unacceptable video quality[2,3,6,8]. Motivated by this problem, several advanced video transcoder are proposed to eliminate "drift" degradation[1,2,9,13,27,28]. In the next chapter, we will study some conventional approaches which are widely used in the video transcoding. After the discussion of these approaches, some advanced techniques will be given and its challenge will be presented.

1.2 Introduction of Video transcoding and Research Objective

As the number of networks, types of devices and video content representation formats increase, interoperability between different systems and different networks is

becoming more important. Therefore, gateways, multipoint control units and video servers have to be developed to provide a seamless interaction between video content creation and consumption. Transcoding of video content is a key technology to make this possible. In general, a transcoder relays video signals from a transmitter in one system to a receiver in another system. Generally speaking, transcoding can be defined as the conversion of one coded signal to another. While this definition can be interpreted quite broadly, it should be noted that research on video transcoding is usually very focused. In the earliest work on transcoding, the majority of interest focused on reducing the bitrate to meet an available channel capacity[3,6,9,13]. Additionally, researchers investigated techniques for the conversions between constant bitrate streams and variable bitrate streams to facilitate more efficient transport of videos. As time moved on and mobile devices with limited display and processing power became a reality, transcoding to achieve spatial resolution reduction[4,5,12,14-16,24-25], as well as temporal resolution reduction[1,2,17,18,21-23], has also been studied. Recently, with the emerging of H.264 standard, video transcoding has gained a significant amount of attention, where the aim is to support the conversion between the new standard and the existing video format.

In all of these cases, it is always possible to use a cascaded pixel-domain approach that decodes the original video signal, performs the appropriate intermediate processing, and fully re-encodes the processed video subject to any new constraints. This approach is the most straightforward way to perform video transcoding, however, high computational complexity is required and more efficient techniques are typically used. This quest for efficiency is the major driving force behind most of the transcoding

activity that we have seen so far. Of course, any gains in efficiency should only have a minimal impact on the quality of the transcoded video.

In this thesis, one of the major objectives is to reduce the bitrate while maintaining low complexity and achieving the highest quality possible. Applications requiring this type conversion include television broadcast and internet streaming. Ideally, the quality of the reduced rate bit stream should have the quality of a bit stream directly generated with the reduced rate. The most straightforward way to achieve this is to decode the video bitstream and fully re-encode the reconstructed signal at the new rate. This approach is illustrated in Figure 1.2. Figure 1.2 shows the details of a front video Encoder, a video transcoder and an end decoder in the multipoint video conferencing system[18]. The function of the video transcoder is used to convert a previously compressed bit stream into a lower bitrate bit stream and will be discuss in details in Chapter 2. First, the captured image in the client side is passed into the front encoder. In the front encoder, the image data undergo the 2D-DCT transform in order to achieve energy compaction. Since our human eyes are less sensitive to the high frequency, so the high frequency components will be discarded in the quantization process to achieve high compression ratio. Then the quantized DCT coefficients have to go through the variable length coding which encodes the DCT coefficients according to the statistical behaviours. The variable length coding tries to use less bits to represent the DCT coefficients which appear more frequently in order to improve coding efficiency. However, only use the spatial information is not enough to achieve high compression ratio. So, some motion estimation and compensation algorithms are used to reduce the temporal redundancy. These algorithms search for the best match for the current frame from a previous frame to

reconstruct the motion compensated frame. The motion estimation is performed on the luminance macroblocks based on the sum of absolute difference(SAD) or mean square error(MSE). Due to it simplicity, SAD is widely use as a measure criterion. In order to obtain a motion vector for the current macroblock, the best matching block that results in the minimal SAD is searched within a predefined search area S in the previous reconstructed reference frame such that

$$(u_t^S, v_t^S) = \arg \min_{(m,n)\in S} SAD_S(m,n) \tag{1.1}$$

where

$$SAD_S(m,n) = \sum\sum |P_S^C(i,j) - R_S^P(i+m, j+n)| \tag{1.2}$$

and m and n are the horizontal and vertical components of the motion vector. The $P_S^C(i,j)$ and $R_S^P(i+m, j+n)$ represent a pixel in the current frame and a displaced pixel by (m,n) in the previous reconstructed reference frame respectively. The superscript “c” or “p” denotes the “current” or “previous” frame respectively, and the subscript “s” indicates the “second” encoder. However, the motion compensated frame has a significant video quality degradation. In order to improve the video quality, the prediction errors (the difference between the motion compensated frame and the previous frame) are also transmitted. In this case, the difference frame, not the original image, is encoded using the DCT. In the end-decoder, the received bitstream is undergone the inverse variable length coding to get the quantized DCT coefficients. Then the quantized DCT coefficients are arranged for inverse quantization and inverse DCT to reconstruct the encoded image. If the motion estimation and motion compensation algorithm are used, the decoder needs a buffer to store the reference frame and perform motion compensation.

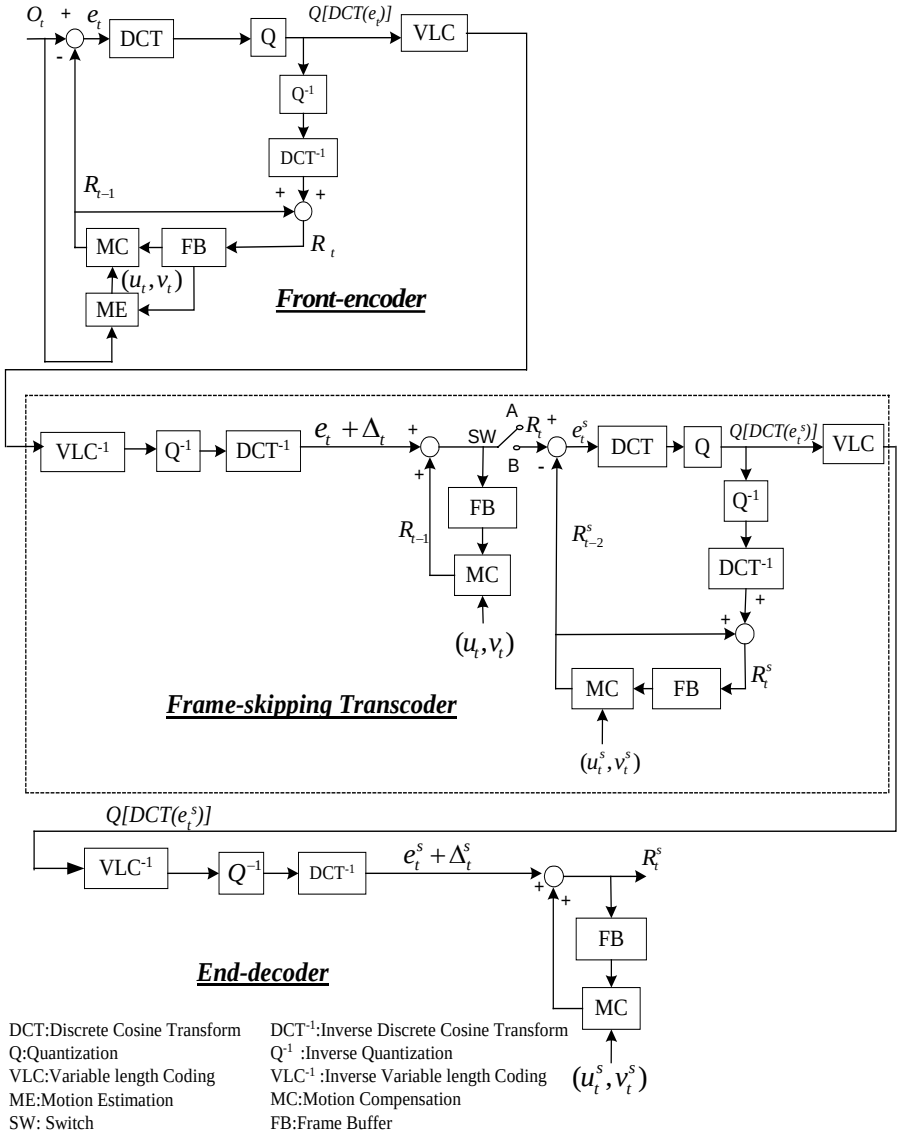


Figure 1.2 Details of the front video encoder, video transcoder and end decoder in the multipoint video conferencing system.

The best performance can be achieved by calculating the required new motion vectors and mode decisions for every macroblock at the new rate. However, significant complexity saving can be achieved while still maintaining acceptable quality, by reusing information contained in the original incoming bitstreams and also considering some simplified architectures.

In the next chapter, we review the progress made over past few years on bitrate reduction architectures and techniques. The focus of the study is to center around two specific aspects, complexity and drift problem. Drift can be explained as the blurring or smoothing of successively predicted frames. It is usually caused by the loss of high frequency data, which creates a mismatch between the actual reference frame used for prediction in the encoder and the degraded reference frame used for prediction in the transcoder and decoder.

1.3 Organization of the thesis

Before giving a description of the research work in this thesis, a review of video transcoding techniques is given in Chapter 2. This review gives focus on the practical aspects of conventional video transcoding techniques in frame skipping, video downscaling, requantization of DCT coefficients, motion vector re-composition and drift problem

For the convenience of discussion, we consider that the following chapters be divided into two parts and described as below.

Chapter 3, 4 and 5 form part I of the thesis. They mainly discuss techniques on homogeneous video transcoding. The emphasis is on DCT domain transcoding which performs the transcoding process in the compressed domain to achieve low computational complexity as well as to reduce re-encoding errors. In Chapter 3, a new frame skipping transcoder is proposed to greatly reduce the computational complexity and reduce the quality degradation. The proposed architecture is mainly performed on the discrete cosine transform (DCT) domain to achieve a low complexity transcoder. It is observed that the re-encoding error is significantly reduced at the frame-skipping

transcoder when the strategy of a direct addition of DCT coefficients is employed. By using the proposed frame-skipping transcoder, the video qualities of the transcoded video sequences can be improved significantly. In Chapter 4, a new DCT-based video downscaling architecture is proposed to achieve a good video quality with low complexity. The proposed video transcoder is based on the split and merge technique by which the video downscaling process can be performed for motion compensated video. In addition, an adaptive motion vector re-composition technique is proposed to re-estimated the new motion vector using diversity and importance measure. In Chapter 5, the proposed technique is applied in the DCT domain and wavelet domain for the application of the video combiner in a video conferencing system. The proposed architecture guarantees a high video quality in the region of interest while reducing the overall bit rate and the computation time even under low bit rates.

Chapter 6 and 7 forms part II of the thesis. These chapters mainly discuss the techniques on the heterogeneous video transcoding. Emphasis is on the compressed domain conversion which reduces the computational complexity and enhances the video quality. In Chapter 6, we address a conversion between B and P picture, which performs the transcoding process in the compressed domain. Using the proposed formulation, an heterogeneous video transcoder is developed. In Chapter 7, the problem of H.263 to H.264 video transcoding is addressed. In the first half of this chapter, a motion vector decomposition algorithm is proposed to speed up the transcoding process for motion re-estimation. Since the motion vector and activity information can be re-used, low computational complexity is achieved. In the second half of this chapter, a compressed domain conversion between the DCT coefficients and Integer Transform(IT) coefficients

is derived. Using the proposed formulation, the integer transform coefficients can be obtained directly from the DCT coefficients without performing full decoding and re-encoding process.

Chapter 8 is devoted to a summary of the work herein and the conclusions reached as a result. Suggestions are also included for further research in the general area of video transcoding.

Chapter 2

Review of Conventional Pixel domain video transcoding Techniques

The conventional approach for implementing transcoding is to cascade a decoder and an encoder, commonly known as pixel-domain transcoding. The incoming video bitstream is decoded in the pixel domain, and the decoded video frame is re-encoded at the desired output bitrate according to the capability of the clients' devices and the available bandwidth of the network. In the following, three conventional transcoding techniques will be described. In addition, some advance techniques such as motion vector re-composition, drift problem and motion compensation in the DCT domain will be discussed.

2.1 Frame skipping

Transcoding is a key technique for reducing the bitrate of a previously compressed video signal. A high transcoding ratio may result in an unacceptable picture quality when the full frame rate of the incoming video bitstream is used. Frame skipping is often used as an efficient scheme to allocate more bits to the representative frames, so that an acceptable quality for each frame can be maintained[1,2,17,18,21-23]. However, the skipped frame must be decompressed completely, which might act as a reference frame to non-skipped frames for reconstruction. The newly quantized DCT coefficients of the prediction errors need to be re-computed for the non-skipped frame with reference to the previous non-skipped frame. Also, motion vector re-estimation is required since the original reference frame is dropped.

Figure 2.1 shows the structure of a conventional frame-skipping transcoder in pixel-domain. In the front encoder, the motion vector, mv_t, for a macroblock with $N \times N$ pixels in frame O_t, the current frame, is computed by searching for the best matched macroblock within a search window S in the previous reconstructed frame, R_{t-1}, and it is obtained as follows:

$$mv_t = (u_t, v_t) = \arg \min_{(m,n) \in S} SAD(m,n) \tag{2.1}$$

$$SAD(m,n) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} | O_t(i,j) - R_{t-1}(i+m, j+n) | \tag{2.2}$$

where m and n are the horizontal and vertical components of the displacement of a matching macroblock, $O_t(i,j)$ and $R_{t-1}(i,j)$ represent a pixel in O_t and R_{t-1}, respectively. For the sake of convenience, we use the same convention for other symbols for the rest of this thesis; i.e. if Y_t represents a frame or error signal at time t, its corresponding value at spatial location (i,j) is denoted by $Y_t(i,j)$.

In transcoding the compressed video bitstream, the output bitrate is lower than the input bitrate. As a result, the outgoing frame rate in the transcoder is usually much lower than the incoming frame rate. Hence switch SW is used to control the desired frame rate of the transcoder. Table 2.1 summarises the operating modes of the frame-skipping transcoder.

Assume that R_{t-1} is dropped. However, R_{t-1} is required to act as the reference frame for the reconstruction of R_t such that

$$R_t(i,j) = R_{t-1}(i+u_t, j+v_t) + e_t(i,j) + \Delta_t(i,j) \tag{2.3}$$

where Δ_t represents the reconstruction errors of the current frame in the front-encoder due to quantization, and e_t is the residual signal between the current frame and the motion-compensated frame,

$$e_t(i,j) = O_t(i,j) - R_{t-1}(i+u_t, j+v_t) \tag{2.4}$$

Substituting (2.4) into (2.3), we obtain an expression for R_t,

$$R_t(i,j) = O_t(i,j) + \Delta_t(i,j) \tag{2.5}$$

In the transcoder, an optimized motion vector for the outgoing bitstream can be obtained by applying the motion estimation such that

$$mv_t^S = (u_t^S, v_t^S) = \arg \min_{(m,n)\in S} SAD^S(m,n) \tag{2.6}$$

$$SAD^S(m,n) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} | R_t(i,j) - R_{t-2}^S(i+m, j+n) | \tag{2.7}$$

where R_{t-2}^s denotes a reconstructed frame of the previous non-skipped reference frame. The superscript "*s*" is used to denote the symbol after performing the frame-skipping transcoder. Although the optimized motion vector can be obtained by making a new motion estimation[2], it is not desirable because of its high computational complexity. It is a common practice to reuse incoming motion vectors. The performance is found almost as good as the new full-scale motion estimation, and this approach is used in many transcoder architectures. The common approaches of motion re-estimation will be given in section 2.4[2]. Let us represent the new motion vector as (u_t^s, v_t^s). Hence, the reconstructed pixel in the current frame after the end-decoder is,

$$R_t^s(i,j) = R_{t-2}^s(i+u_t^s, j+v_t^s) + e_t^s(i,j) + \Delta_t^s(i,j) \tag{2.8}$$

where $e_t^s(i,j) = R_t(i,j) - R_{t-2}^s(i+u_t^s, j+v_t^s)$ and Δ_t^s represents the requantization errors due to the re-encoding in the transcoder, then,

$$R_t^s(i,j) = R_t(i,j) + \Delta_t^s(i,j) \tag{2.9}$$

This equation implies that the reconstructed quality of the non-skipped frame deviates from the input sequence to the transcoder. The effect of re-encoding errors is depicted in Figure 2.2 where the "Salesman" sequence was transcoded at half of the incoming frame rate. This Figure shows that re-encoding errors lead to a drop in the picture quality of about 3.5dB on average, which is a significant degradation. Table 2.2 also summarizes the performance of the conventional video transcoder. In the next chapter, we will study some advanced video transcoding technique to reduce the quality degradation and computational complexity during the transcoding process.

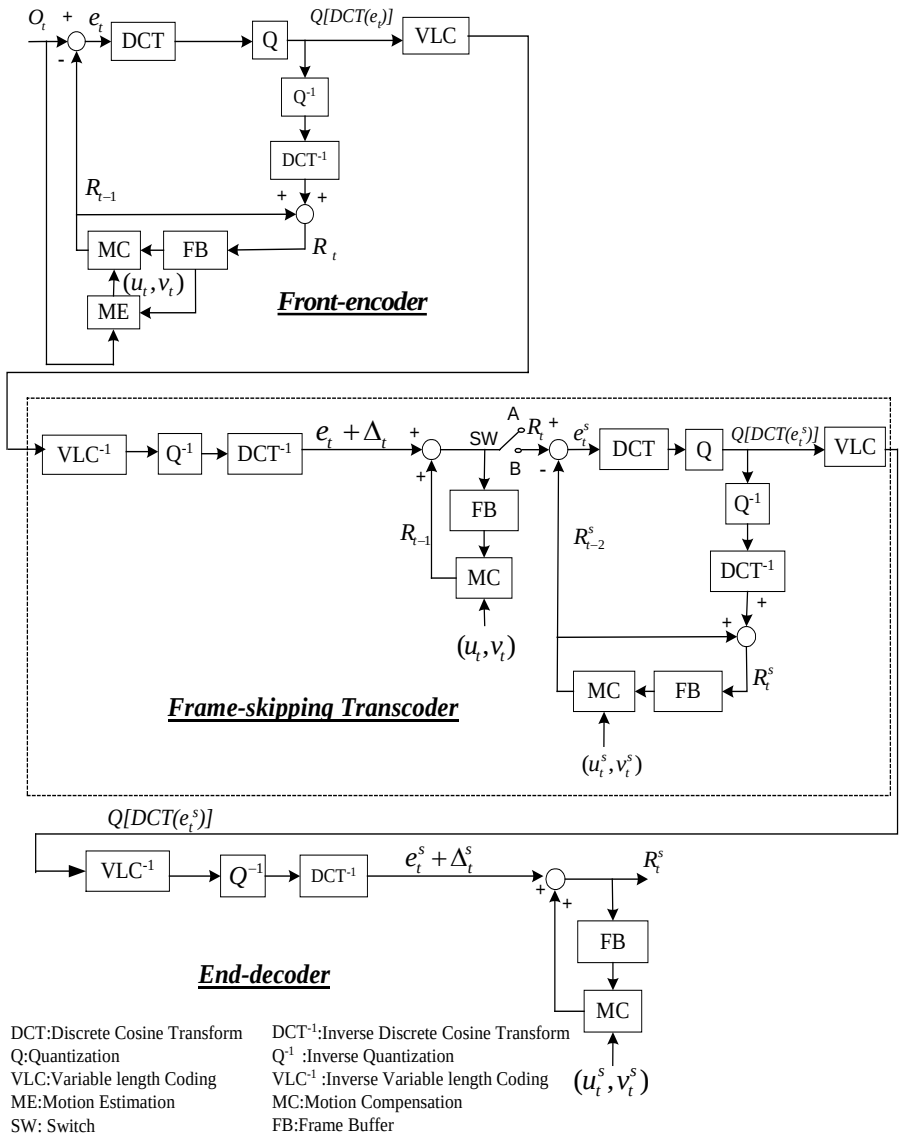


Figure 2.1. Frame-skipping transcoder in pixel-domain.

Table 2.1. Switch position for different modes of the pixel-domain transcoder.

Frame skipping mode	*SW* Position
Skipped frame	*A*
Non-skipped frame	*B*

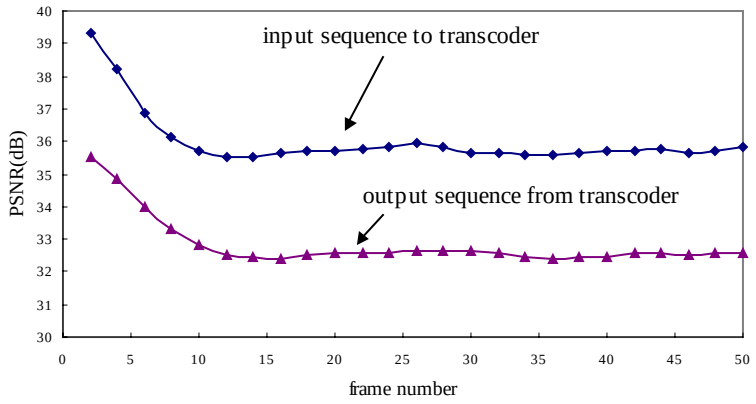


Figure 2.2. Quality degradation of conventional frame-skipping transcoder for the "Salesman" sequence. The peak signal-to-noise ratio (PSNR) of the frame-skipping pictures is plotted to compare with that of the same pictures which used directly a decoder without a transcoder.

Table 2.2 Average PSNR of the conventional transcoder without frame rate control, where the frame rate of the incoming bitstream was 30 frames/sec. The front encoder for encoding "salesman", "foreman" and "carphone" was H.263 TMN8; while MPEG2 TM5 was used to encode "table tennis" and "football".

Sequences	Input bitrate	PSNR performance of the transcoded video sequence		
		15 frames/sec	10 frames/sec	7.5 frames/sec
Salesman (176x144)	64k	33.20	33.28	33.20
	128k	36.71	36.77	36.69
Foreman (176x144)	64k	30.87	30.73	30.45
	128k	34.64	34.57	34.06
Carphone (176x144)	64k	32.28	32.25	32.21
	128k	34.92	34.90	34.87
Table Tennis (352x240)	1.5M	31.41	31.36	31.17
	3M	34.44	34.28	34.21
Football (352x240)	1.5M	30.03	29.96	29.68
	3M	33.87	33.80	33.63

2.2 Video downscaling

The most straightforward approach for implementing video downscaling transcoding is to cascade a decoder and an encoder, commonly known as pixel-domain transcoding as described in the previous section. That is, we have to downscale an encoded video produced by one of the current video compression standards such as

MPEG, H.261 or H.263 which employ motion compensated prediction to exploit the temporal redundancy to achieve low bitrates. The conventional approach is to decompress the video and perform downscaling of the video in the pixel domain. Then new motion vectors and DCT coefficients of this downscaled video have to be recomputed inside a transcoder. One simple approach to estimate the new motion vector is to take the average of four motion vectors of the associated four macroblocks and downscale it by two so that a resampled motion vector for the downscaled version of the video can be obtained[4,5,12,14-16,24,25]. Motion vectors obtained in this manner are not optimal. An adaptive motion vector resamling technique can be used which provides a possible solution to recompose new motion vectors. The resampling scheme suggested to align the weighting toward the worst prediction and to recompose an outgoing motion vector from the incoming motion vectors of the incoming frame which has a higher resolution. Using this technique, the conventional video downscaling transcoder can avoid full motion re-estimation to reduce the computational complexity[5].

Figure 2.3 shows the well-aligned case of the motion vectors during video downscaling. In this case, four motion vectors have the same direction and magnitude. Therefore, all the align-to-average weighting (AAW –which estimates the new motion vector by taking all incoming motion vectors with equal weighting as the consideration), align-to-best weighting (ABW –which estimates the new motion vector by taking more weighting for the motion vector with less prediction error as the consideration), align-to-worst weighting (AWW –which estimates the new motion vector by taking more weighting for the motion vector with more prediction error as the consideration) or adaptive motion vector resampling (AMVR –which estimates the new motion vector by

taking weighting adaptively) will provide the optimal motion vector as shown in Figure 2.4. However, when not all the motion vectors are well-aligned as shown in Figure 2.5, a good motion vector resampling or motion vector refinement(use the base motion vector as a center and refine the motion vector with a smaller size of the search window) is required in order to reduce the re-encoding error. In this case, new prediction errors have to be recomputed due to a mismatch of the DCT coefficients and the new motion vector which was estimated by using the adaptive motion vector resampling approach.

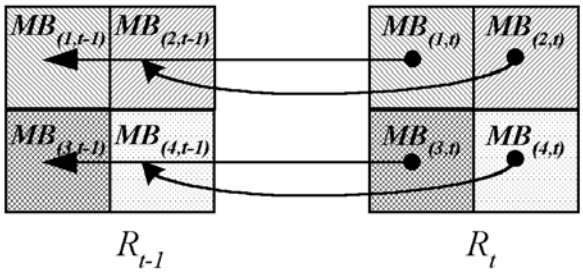


Figure 2.3. All motion vectors having the same direction and magnitude.

Figure 2.4. Optimal motion vector after video downscaling.

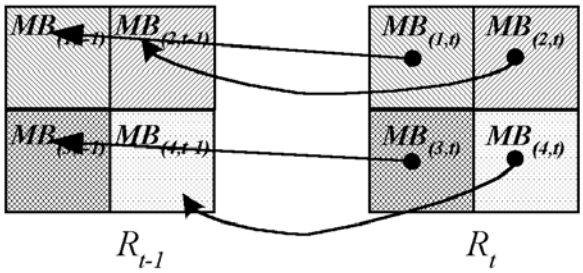


Figure 2.5. Motion vectors with different directions and magnitudes.

Figure 2.6 shows the architecture of the hybrid AMVR system. It is very similar to the cascaded pixel domain video transcoder except the AMVR block is included. In

this hybrid system, the spatial frames are reconstructed and downscaled in the pixel domain but the motion vectors are estimated directly from the existing motion vectors in the original sequence. Initially, the variable length decoding is performed and information of the motion vectors is extracted. Then inverse quantization and inverse DCT(IDCT) are performed for the incoming coefficients. After the motion compensation process and IDCT, the reconstructed frame is obtained and stored in the frame memory. This frame memory is used to reconstruct the next incoming frame. After the decoding process, a downscale process is applied to the decoded data in the pixel domain. In order to speed up the re-encoding process, motion re-estimation is not performed. In Figure 2.6, the AMVR block is responsible for resampling the motion vectors adaptively. The new motion vector, $\overline{mv}'$, is obtained by making use of the following equation:

$$\overline{mv}' = \frac{1}{2} \frac{\sum_{i=1}^{4} \overline{mv}_i A_i}{\sum_{i=1}^{4} A_i} \tag{2.10}$$

where $\overline{mv}_i$ denotes the motion vector of block *i* in the original *NxN* video and A_i denotes the activity measurement of the i^{th} residual block. The simplest way to calculate A_i is to count the number of non-zero AC coefficients. Figure 2.7 shows the new motion vector before downscaling. Due to the mismatch of this new motion vector and the incoming DCT coefficients, new prediction errors need to be recomputed. Therefore, the motion compensation process is applied using the new set of motion vectors for the lower resolution frame. New prediction errors have to be calculated. Then the DCT and quantization processes are applied to the new prediction errors. During the quantization process, re-encoding errors are introduced. In order to control the accumulation of errors,

inverse quantization and inverse DCT are performed. Therefore, re-encoding error can be fedback to the memory to avoid the accumulation of errors during the transcoding process. Although re-encoding errors can be controlled by using this architecture, however, high computational complexity is required and re-encoding errors still exist in the client decoding side. The effect of re-encoding errors is shown in Figure 2.8 where the “Table Tennis” sequence was transcoded at a quarter of the incoming frame size. This figure shows that re-encoding errors lead to a significant degradation of picture quality. Table 2.3 summarizes the performance of the conventional video transcoder.

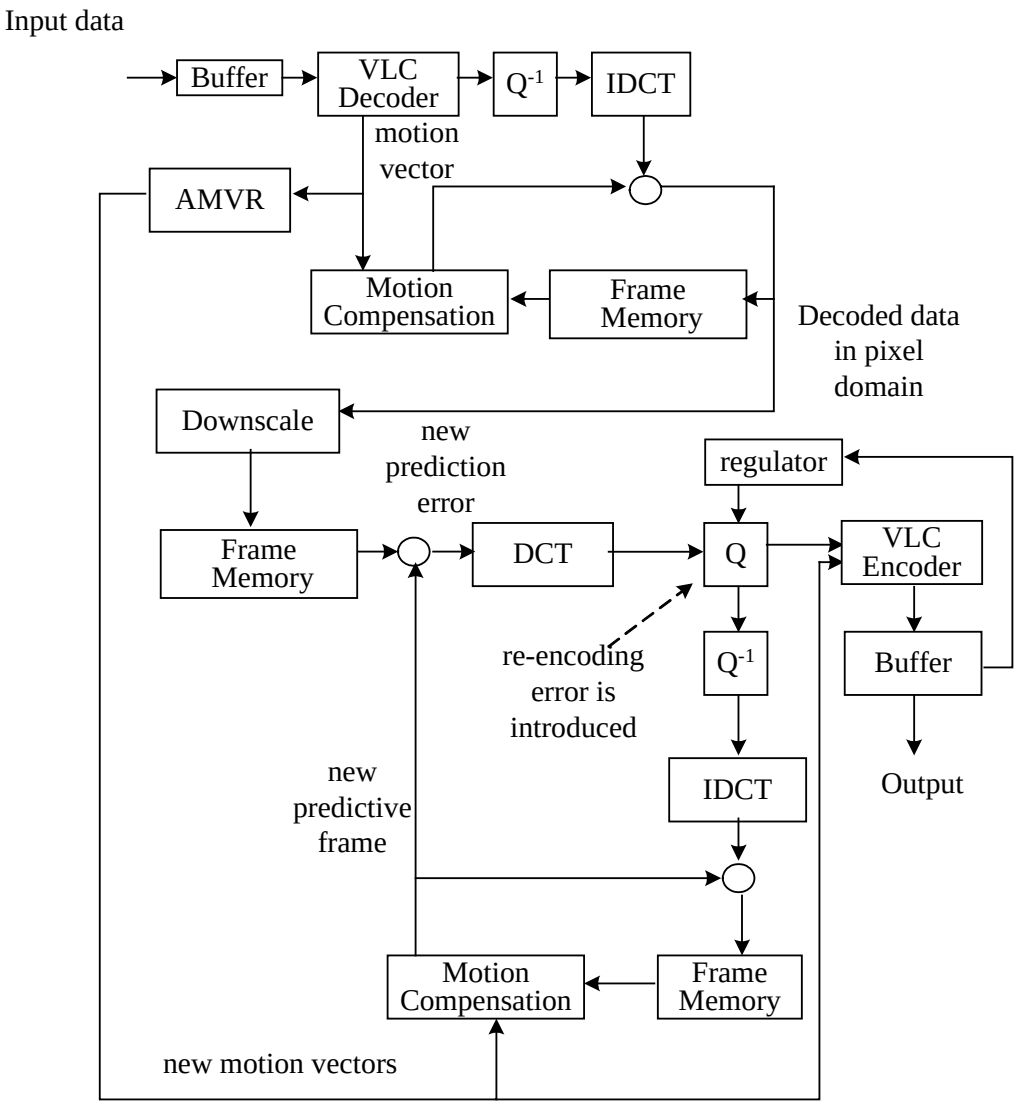


Figure 2.6. The architecture of the hybrid AMVR system.

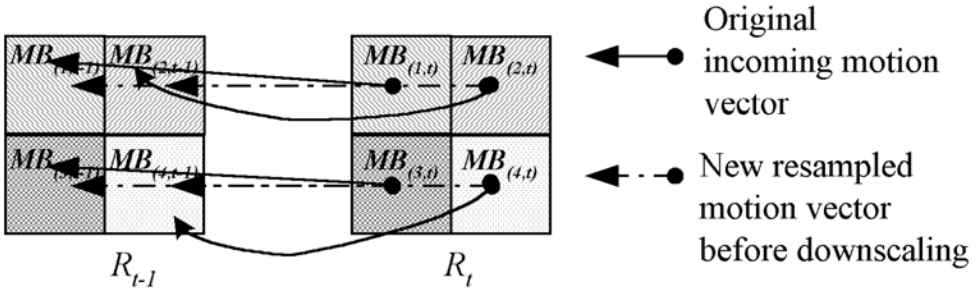


Figure 2.7. New motion vector estimated before downscaling.

Figure 2.8. Re-encoding error introduced by the video downscaling transcoder using AMVR system.

Table 2.3. Average PSNR of the conventional transcoder, where the frame rate of the incoming bitstream was 30 frames/s. MPEG2 TMN5 was used as the front encoder for encoding "Salesman", "Miss_America", "Hall", "Tennis", "Football" and "Flower".

Sequences	Input bitrate	Average PSNR difference as compared with CPDT+AAW for MC macroblock transcoding.	
		CPDT+ABW	AMVR
Salesman (352x288)	512k	0.06	0.41
	256k	0.05	0.39
Miss_America (352x288)	512k	0.09	0.39
	256k	0.07	0.36
Hall (352x288)	512k	0.11	0.42
	256k	0.08	0.38
Tennis (352x240)	3M	0.12	0.43
	1.5M	0.08	0.38
Flower (352x240)	3M	0.18	0.47
	1.5M	0.15	0.43
Football (352x240)	3M	0.21	0.50
	1.5M	0.27	0.56

2.3 Requantization of DCT coefficients

The simplest architectures for transcoding using the requantization of DCT coefficients is called open-loop transcoding. That is, the DCT coefficients can be

requantized, truncated or partially discarded for the high frequency components, to achieve the bitrate reduction in the coded domain. In other words, the transcoding process is carried out in the coded domain where complete decoding and re-encoding are not required. Hence, low computational complexity transcoding process can be achieved. However, the open-loop transcoding architecture produces "drift" degradations due to the mismatched of the reconstructed pictures in the front encoder and the end decoder, which leads to unacceptable video quality. In the following, we will introduce a conventional pixel domain video transcoder using the refinement scheme to reduce the computational complexity and avoid the drift problem (see Section 2.5).

Figure 2.9 shows the structure of the cascaded transcoder. The full motion estimation block is omitted for simplicity and motion vector refinement scheme is employed. Since the output bitrate is lower than the input bitrate, the quantization step size in quantizer Q2 inside the transcoder is usually much larger than the quantization step size in quantizer Q1 in the front encoder.

During the transcoding process, an optimized motion vector for the transcoded video can be obtained by applying the motion estimation such that

$$(O_x, O_y) = \arg \min_{(m,n)\in S} SAD_s(m,n) \tag{2.11}$$

$$SAD_s(m,n) = \sum_{i}^{M}\sum_{j}^{N} \left| P_s^c(i,\, j) - R_s^p(i+m,\, j+n) \right| \tag{2.12}$$

where (O_x, O_y) represents the optimal motion vector, the notation of P and R represent the input frame and reconstructed frame, respectively. The superscripts c and p represent the current and the previous frame, respectively. The subscript s represents the frame in the second encoder. SAD_s represents the sum of absolute different in the second encoder.

Although the optimized motion vector can be obtained by a full motion re-estimation, it is not desirable due to its high computational complexity. From the experimental results, it is found that the differential reconstruction error causes incoming motion vectors to deviate from optimal values. In practical situation, the deviation is within a small ranger and the position of the optimal vector will be closed to the incoming motion vectors. Therefore, we can obtain the optimal motion vector easily by refining the incoming motion vector within a small range instead of applying a full motion re-estimation.

For the refinement of incoming motion vectors, let us define a base motion vector (Bx, By) as the motion vector obtained from the incoming bitstream. The base motion vector can be obtained by using bilinear interpolation or forward dominant motion vector selection method (see Section 2.4). Then, a delta motion vector (Dx, Dy) can be estimated within a new search window S', around the point indicated by the base motion vector:

$$(Dx, Dy) = \arg \min_{(m,n) \in S'} SAD' \tag{2.13}$$

$$SAD' = \sum \sum | P_s^c(i, j) - R_s^p(i + Bx + m, j + By + n) | \tag{2.14}$$

The new search window S' can be set much smaller than the full scale window S which can still produce almost the same video quality as the full motion re-estimation. Table 2.4. shows the performance of motion vector refinement. The quality degradation introduced by reusing incoming motion vectors directly without refinement is about 0.47dB on average as compared with a new full search motion re-estimation for transcoding the "carphone" video sequence. However, the refinement of the incoming

motion vectors using a small search window increases the performance close to that of the full search motion re-estimation.

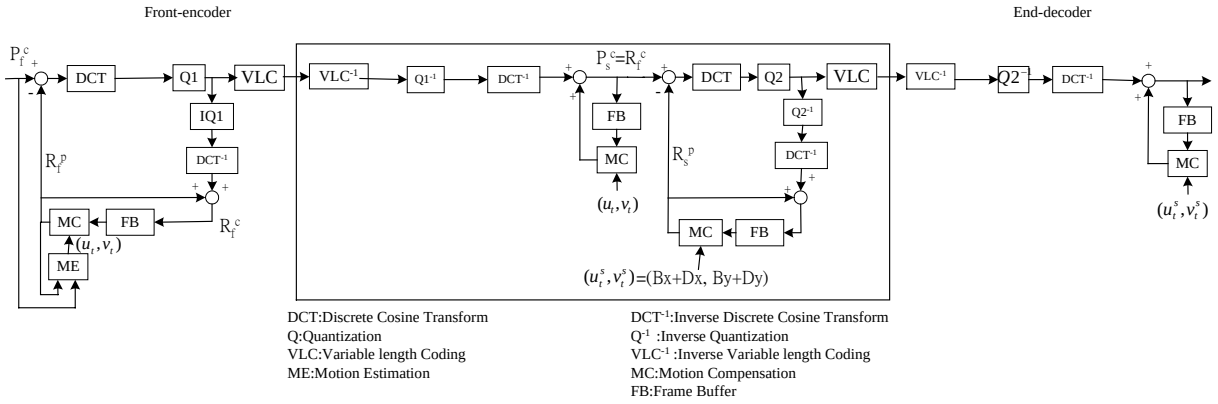


Figure 2.9. The cascaded pixel domain video transocder using requantization of DCT coefficients with refinement scheme .

Table 2.4 Performance of the conventional video transcoder using requantization of DCT coefficients. The incoming bitrate of the video is 128kb/s and the transcoded video bitrate is 64kb/s.

Video Sequence	Full motion re-estimation	Reuse the motion vector without refinement	Reuse the motion vector refinement
Carphone	33.6	33.13	33.58
Claire	39.58	39.47	39.54
Suzie	34.15	34.06	34.13

2.4 Methods of Motion vector recomposition

A video transcoder usually reuses the decoded motion vectors to re-encode the video sequences at a lower bit rate to speed up the transcoding process. When frame skipping is allowed in a transcoder, motion vectors cannot be reused directly since the motion vector of the current frame is no longer estimated from the immediate past frame. In other words, motion estimation of the non-skipped frames requires to follow the path of motion from the current frame to its previously non-skipped frame as shown in Figure 2.10. Hence, motion vector re-estimation is needed. To reduce the computational

complexity of motion vectors re-estimation, two major techniques are widely used in the motion vector re-composition: Forward Dominant motion Vector Selection(FDVS)[2] and Bilinear Interpolation motion Vector Selection(BiVS)[2].

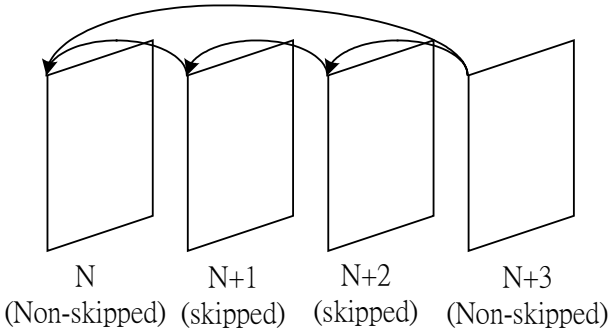


Figure 2.10 Motion tracing for the non-skipped.

The forward dominant motion vector selection method is to select the dominant motion vector from its four neighboring macroblocks. A dominant motion vector is defined as the motion vector carried by a dominant macroblcok. The dominant macroblock is the macroblock that has the largest overlapped segment with MB_{t-1} as shown in Figure 2.11. Hence, the recomposed motion vector becomes:

$$(u_t^s, v_t^s) = (u_t, v_t) + (u_{t-1}, v_{t-1}) \tag{2.15}$$

where (u_t^s, v_t^s) represents the recomposed motion vector, (u_t, v_t) and (u_{t-1}, v_{t-1}) represent the motion vectors in frame t and t-1, respectively.

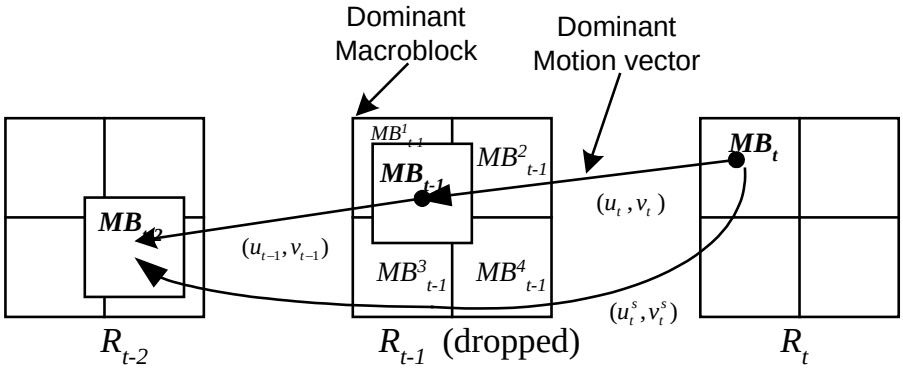


Figure 2.11 Forward dominant motion vector selection

The idea of the bilinear interpolation motion vector selection is to estimate the new motion vector according to the size of the overlapping area. Figure 2.12 shows the idea of how to recompose the new motion vector for frame N with reference to the previous non-skipped frame, frame N-2. The bilinear interpolation is defined as

$$\text{interpolated motion vector} = (1-a)(1-b)MV_1 + a(1-b)MV_2 + (1-a)bMV_3 + abMV_4 \quad (2.16)$$

where MV_1, MV_2, MV_3 and MV_4 are the motion vectors of the four macroblocks which have overlapping areas with the MB under question. a and b are constants determined by the pixel distance to MV_1. The weighting constant of each neighbor block is inversely proportional to the pixel distance. For example, one of the motion vector in frame N is (1,1) as shown in Figure 2.12. Therefore,

$$\text{the interpolated motion vector} = (1-1/16)(1-2/16)MV_1 + (1/16)(1-2/16)MV_2 + (1-1/16)(2/16)MV_3 + (1/16)(2/16)MV_4$$

Note that the size of the macroblock is 16x16. Repeatedly tracing the motion in this manner will create an extended motion vector for each macroblock in the current frame with reference to its previously non-skipped frame.

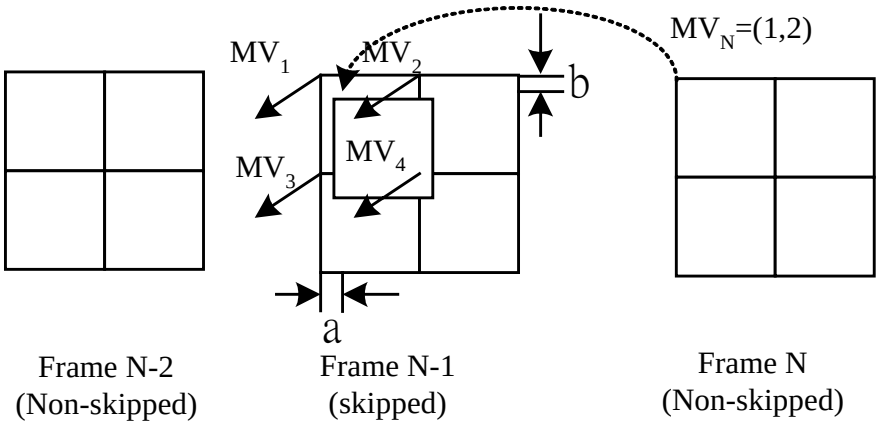


Figure 2.12 Motion vector recomposition using bilinear interpolation

2.5 What is Drift problem?

"Drift" problem is introduced due to a mismatch of the reconstructed pictures in the front encoder and the end decoder during the transcoding process. This phenomena can be explained in the following.

During the transcoding process of the DCT coefficients, requantization process is required for the transcoder using frame skipping, requantization of DCT coefficients or video downscaling technique (see Section 2.1, 2.2 and 2.3). Re-encoding errors are introduced during the requantization process. However, if the re-encoding errors are not feedback inside the transcoder (e.g.open-loop architecture, see Section 2.3), the reconstructed pictures in the front encoder and the end decoder will be mismatched. More importantly, the re-encoding errors will be accumulated and propagated to the latter frames since the transcoded frame need to act as a reference frame. This problem will lead to unacceptable video quality. However, the re-encoding errors need to decode in the pixel domain in order to avoid the accumulation of errors inside the transocder. Therefore, the conventional pixel domain transcoder is also called "drift-free" transcoder since the re-encoding errors are decoded in the pixel domain and feed back inside the

transcoder as shown in Figure 2.1, 2.6 and 2.9. This process introduces high computational complexity since full decoding and re-encoding process are required. In the next chapter, some advance techniques for the DCT-based transocder are proposed to control the drift problem. The major idea of these approaches tries to perform the motion compensation and feed back the re-encoding errors in the DCT domain.

2.6 Motion compensation in the DCT domain

In this section, we will introduce an advanced technique called Motion Compensation in the DCT domain(MCDCT) which is widely used in the DCT-based video transcoder[9,13,26-29].

The major idea to perform motion compensation in the DCT domain is to represent a shifted version of the DCT coefficients as the sum of all horizontally and/or vertically displaced anchor blocks. Then the motion compensated DCT values of B_0', B_1', B_2' and B_3' are constructed using the pre-computed DCT values of the shift operators.

Consider that P_0' is the block of interest in the pixel domain as shown in Figure 2.13. P_0, P_1, P_2 and P_3 are its four neighboring blocks in the pixel domain. The shaded regions in P_0, P_1, P_2 and P_3 are moved by an amount, (dx,dy). Then P_0' can be represented by the following equation.

$$P_0' = \sum_{i=0}^{3} S_{i1} P_i S_{i2} \qquad (2.17)$$

where S_{ij}'s are matrics tabulated in Table 2.5.

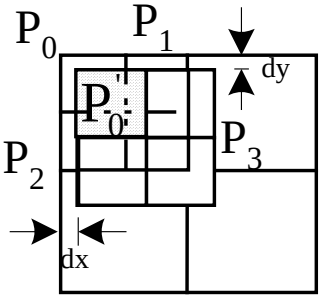


Figure 2.13 Dominant region and boundary region of MB_{t-1} in the pixel domain

Table 2.5. The shift operators.

Sub-block	S_{i1}	S_{i2}
P_0	$\begin{bmatrix} 0 & I_{hd} \\ 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 \\ I_{wd} & 0 \end{bmatrix}$
P_1	$\begin{bmatrix} 0 & I_{hd} \\ 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & I_{wb} \\ 0 & 0 \end{bmatrix}$
P_2	$\begin{bmatrix} 0 & 0 \\ I_{hb} & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 \\ I_{wd} & 0 \end{bmatrix}$
P_3	$\begin{bmatrix} 0 & 0 \\ I_{hb} & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & I_{wb} \\ 0 & 0 \end{bmatrix}$

Each I is an identity matrix of size hb=dy or hd=8-dy or wb=dx or wd=8-dx. As a pre-multiplication step, we have to shift the sub-block of interest horizontally while shift the sub-block vertically as a post-multiplication step. Four possible locations of the sub-block of interest: upper-left, upper-right, lower-left and lower-right are shown in Table 2.5.

Let us define the 2D-DCT of an 8x8 block, A, as

$$DCT(A) = \hat{A} = TAT^t \tag{2.18}$$

where T is an 8x8 DCT matrix with entries t(i,j) given by

$$t(i,\, j) = \frac{1}{2} k(i) \cos \frac{(2j+1)i\pi}{16} \tag{2.19}$$

where i represents the row index and j represents the column index and

$$k(i) = \begin{cases} \frac{1}{\sqrt{2}}, & i = 0 \\ 1, & otherwise. \end{cases} \tag{2.20}$$

Using the properties of DCT,

$$\text{DCT(CD)=DCT(C)DCT(D)} \tag{2.21}$$

where C and D are 8x8 matrices.

From eqn(2.17), we have,

$$DCT(P_0') = \sum_{i=0}^{3} DCT(S_{i1}) DCT(P_i) DCT(S_{i2}) \tag{2.22}$$

This implies, $B_0' = \sum_{i=0}^{3} DCT(S_{i1})(B_i) DCT(S_{i2})$

Note that the DCT of S_{i1} and S_{i2} can be pre-computed and B_i can be extracted from the incoming bitstream. From eqn(2.22), we can obtain the motion compensated DCT coefficients in the DCT domain by reusing the DCT coefficients from the incoming video bitstream and the pre-computed shift operators. In the next Chapter, we will make use of this technique to design the DCT-based video transcoder to speed up the transcoding process.

Chapter 3

Frame Skipping video transcoding

3.1 New architecture for dynamic frame-skipping transcoder

3.1.1 Introduction

In this chapter, a new algorithms and a novel architecture for frame-rate reduction to improve picture quality and to reduce complexity is introduced. The proposed architecture is mainly performed on the discrete cosine transform (DCT) domain to achieve a transcoder with low complexity. With the direct addition of DCT coefficients and an error compensation feedback loop, re-encoding errors are reduced significantly. Furthermore, a new frame-rate control scheme is proposed which can dynamically adjust the number of skipped frames according to the incoming motion vectors and re-encoding errors due to transcoding such that the decoded sequence can have a smooth motion as well as better transcoded pictures.

3.1.2 High-Quality Frame-Skipping Transcoder with Dynamic Control Scheme

In [23], we have proposed a direct addition of the DCT coefficients in frame-skipping transcoding to avoid re-encoding errors and to reduce the complexity for macroblocks coded without motion compensation. In this chapter, we present a new frame-skipping transcoding architecture which is an extension of the work of [23]. The new architecture has three new features:

(1) a direct addition of the DCT coefficients for macroblocks without motion compensation (non-MC macroblocks);

(2) a feedback loop for error compensation within motion-compensated macroblocks (MC macroblocks);

(3) an intelligent control scheme for a dynamic selection of the most representative and high-quality non-skipped frames.

The architecture of the proposed transcoder is shown in Figure 3.1. The input bitstream is first parsed with a variable-length decoder to extract the header information, coding mode, motion vectors and quantized DCT coefficients for each macroblock. Let us define the incoming residual signal with quantization errors due to the front encoder as $\hat{e}_t = e_t + \Delta_t$ and its quantized DCT coefficients as $Q[DCT(\hat{e}_t)]$. Each macroblock is then manipulated independently. Two switches, SW_1 and SW_2, are employed to update the DCT-domain buffer, FB_{DCT}, for the transformed and quantized residual signal depending on the coding mode originally used in the front encoder for the current macroblock being processed. The switch positions for different coding modes are shown in Table 3.1. For non-MC macroblocks, the previous residual signal in the DCT-domain is directly fed back from FB_{DCT} to the adder, and the sum of the input residual signal and the previous residual signal in the DCT-domain is updated in FB_{DCT}. Note that all operations are performed in the DCT-domain, thus the complexity of the frame-skipping transcoder is reduced. Also, the quality degradation of the transcoder introduced by Δ_t^s is avoided. When the motion compensation mode is used, motion compensation, DCT, inverse DCT, quantization and inverse quantization modules are activated to update FB_{DCT}. One problem with these MC macroblocks is that re-encoding errors will be generated, which introduces quality degradation in the transcoded sequence. Thus, the feedback loop in Figure 3.1 tries to compensate for the re-encoding errors introduced in the previous

frames. Note that switch SW_3 is used to adjust the frame rate and refresh FB_1 for the non-skipped frame. This switch is controlled by the proposed dynamic control scheme which makes use of both the motion vectors and re-encoding errors. Table 3.2 shows the frame-skipping modes of the proposed transcoder. The advantages of the DCT-domain buffer arrangement, together with the details of other methods, are described in the following subsections.

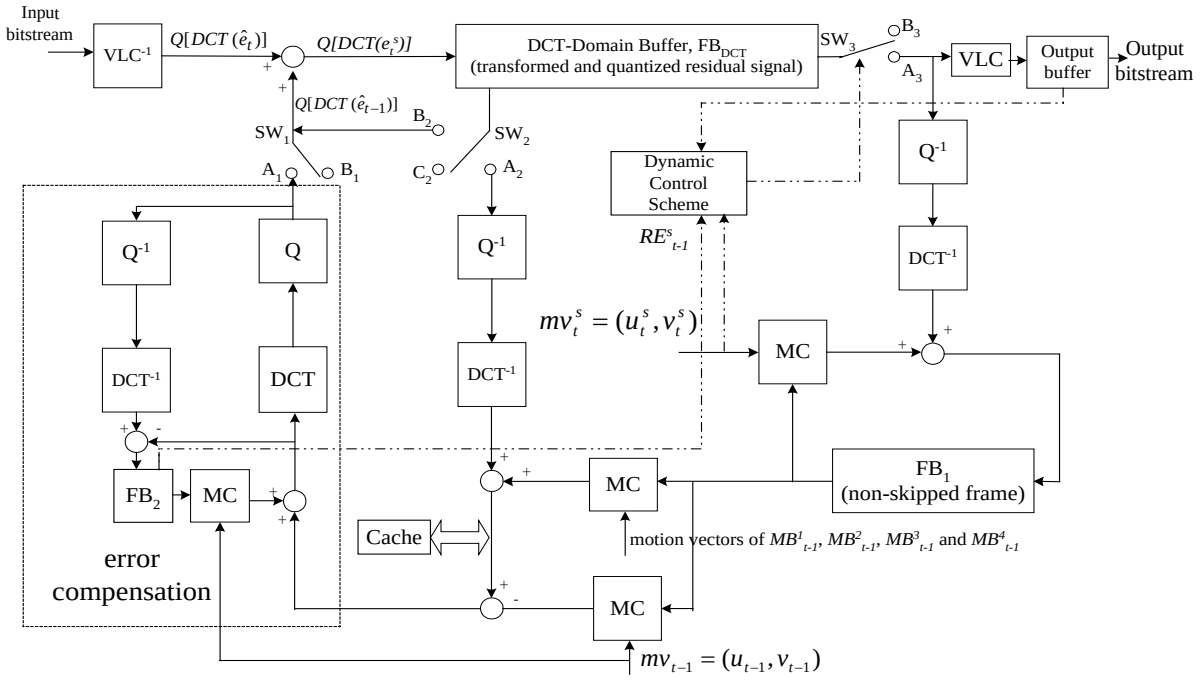


Figure 3.1 Architecture proposed for frame-skipping transcoder.

Table 3.1 Different coding modes of switches SW_1 and SW_2 of the proposed transcoder.

Coding mode	SW_1 Position	SW_2 Position
Non MC	B_1	B_2
MC	A_1	A_2

Table 3.2 Switch positions for different frame-skipping modes of the proposed transcoder.

Frame-skipping mode	SW_3 Position
Skipped frame	B_3
Non-skipped frame	A_3

A. Direct addition of DCT coefficients for non-MC macroblocks

In Figure 3.2, a situation in which one frame is dropped is illustrated. We assume that MB_t represents the current non-MC macroblock and MB_{t-1} represents the best matching macroblock with MB_t. Since MB_t is coded without motion compensation, the spatial position of MB_{t-1} is the same as that of MB_t, and MB_{t-2} represents the best matching macroblock with MB_{t-1}. Since R_{t-1} is dropped, for MB_t, we need to compute motion vector (u_t^s, v_t^s) and prediction errors in the DCT-domain, $Q[DCT(e_t^s)]$, by using R_{t-2} as a reference. Since the motion vector in MB_t is zero, then

$$(u_t^s, v_t^s) = (u_{t-1}, v_{t-1}) \tag{3.1}$$

Re-encoding can lead to additional errors, but this can be avoided if $Q[DCT(e_t^s)]$ is computed in the DCT-domain. In Figure 3.2, pixels in MB_t can be reconstructed by performing the inverse quantization and inverse DCT of $Q[DCT(\hat{e}_t)]$ and adding this residual signal to pixels in MB_{t-1} which can be similarly reconstructed by performing the inverse quantization and inverse DCT of $Q[DCT(\hat{e}_{t-1})]$ and adding this residual signal to pixels in the corresponding MB_{t-2}. The reconstructed macroblocks MB_t and MB_{t-1} are given by

$$MB_t = MB_{t-1} + \hat{e}_t \tag{3.2}$$

and

$$MB_{t-1} = MB_{t-2} + \hat{e}_{t-1} \tag{3.3}$$

Using (3.2) and (3.3), the prediction errors between the current non-MC macroblock and its corresponding reference macroblock in R_{t-2}, $e_t^s = MB_t - MB_{t-2}$, can be written as

$$e_t^s = \hat{e}_t + \hat{e}_{t-1} \tag{3.4}$$

By applying the DCT for e_t^s and taking into account the linearity of DCT, we obtain the expression of e_t^s in the DCT-domain.

$$DCT(e_t^s) = DCT(\hat{e}_t) + DCT(\hat{e}_{t-1}) \tag{3.5}$$

Then the newly quantized DCT coefficients of prediction errors are given by

$$Q[DCT(e_t^s)] = Q[DCT(\hat{e}_t) + DCT(\hat{e}_{t-1})] \tag{3.6}$$

Note that, in general, quantization is not a linear operation because of the integer truncation. However, $DCT(\hat{e}_t)$ and $DCT(\hat{e}_{t-1})$ are divisible by the quantizer step-size. Thus, we obtain the final expression of the prediction errors in the quantized DCT-domain by using R_{t-2} as a reference,

$$Q[DCT(e_t^s)] = Q[DCT(\hat{e}_t)] + Q[DCT(\hat{e}_{t-1})] \tag{3.7}$$

Equation (3.7) implies that the newly quantized DCT coefficient $Q[DCT(e_t^s)]$ can be computed in the DCT-domain by adding directly the quantized DCT coefficients between the data in the DCT-domain buffer, FB_{DCT}, and the incoming DCT coefficients, whilst the updated DCT coefficients are stored in FB_{DCT}, as depicted in Figure 3.1, when switches SW_1 and SW_2 are connected to B_1 and B_2 respectively. Since it is not necessary to perform motion compensation, DCT, quantization, inverse DCT and inverse

quantization, complexity is reduced. Furthermore, since requantization is not necessary for non-MC macroblocks, re-encoding errors Δ_t^s are also avoided.

For a real world image sequence, the block motion field is usually gentle, smooth, and varies slowly. As a consequence, the distribution of motion vector is center-biased [39-40], as demonstrated by the typical examples as shown in Table 3.3 which shows the distribution of the coding modes for various sequences including “Salesman”, “Foreman”, “Carphone”, “Table Tennis” and “Football”. These sequences have been selected to emphasize different amount of motion activities. It is clear that over 70% and 50% of the macroblocks are coded without motion compensation for sequences containing low and high amount of motion activities respectively. By using a direct addition of the DCT coefficients in the frame-skipping transcoder, the sequence containing more non-MC macroblocks can reduce the computational complexity and re-encoding errors more significantly.

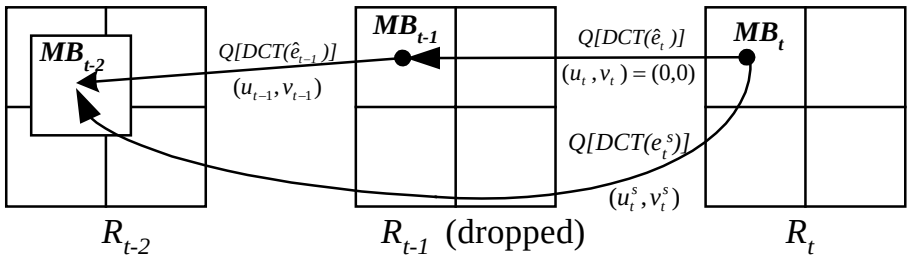


Figure 3.2 Residual signal re-computation of frame skipping for non-MC macroblocks.

Table 3.3 Percentage of non-MC macroblock for various sequences.

Salesman	Foreman	Carphone	Table Tennis	Football
95%	65%	69%	72%	53%

B. DCT-domain buffer updating for MC macroblocks with error compensation

For MC macroblocks, direct addition cannot be employed since MB_{t-1} is not on a macroblock boundary, as depicted in Figure 3.3. In other words, $Q[DCT(\hat{e}_{t-1})]$ is not available from the incoming bitstream. Figure 3.3 also shows that MB_{t-1} is formed by using parts of four segments which come from its four neighboring macroblocks. Let us define these four neighboring macroblocks as MB_{t-1}^1, MB_{t-1}^2, MB_{t-1}^3 and MB_{t-1}^4. It is possible to use the incoming quantized DCT coefficients of MB_{t-1}^1, MB_{t-1}^2, MB_{t-1}^3 and MB_{t-1}^4, to come up with $Q[DCT(\hat{e}_{t-1})]$. First, inverse quantization and inverse DCT of the quantized DCT coefficients of MB_{t-1}^1, MB_{t-1}^2, MB_{t-1}^3 and MB_{t-1}^4 are performed to obtain their corresponding prediction errors in the pixel-domain. Note that each macroblock is composed of four 8×8 blocks in video coding standards [10,30,31,34,44], and the DCT and quantization operations are performed on units of 8×8 blocks. When processing MB_{t-1}^1, MB_{t-1}^2, MB_{t-1}^3 and MB_{t-1}^4, only their corresponding 8×8 blocks, which have pixels overlapping with MB_{t-1}, are subject to the inverse DCT computation. In addition to only processing blocks that have pixels overlapping with MB_{t-1}, each 8×8 block is to be inverse transformed partially. The two-dimensional (2-D) inverse DCT's are implemented using one-dimensional (1-D) row transforms followed by 1-D column transforms. All eight rows for each required block receive a 1-D inverse DCT, but only the columns that have pixels overlapping to MB_{t-1} are subjected to a 1-D inverse DCT. In

most cases, the approach significantly reduces the required number of column inverse DCT.

Each segment of the reconstructed pixels in MB_{t-1} can be obtained by adding its prediction errors to its motion-compensated segment of the previous non-skipped frame stored in FB_1, as shown in Figure 3.1. After all pixels in MB_{t-1} are reconstructed, we need to find the prediction errors, $\hat{e}_{t-1}$. Actually, $\hat{e}_{t-1}$ is equal to the reconstructed pixel in MB_{t-1} subtracted from its corresponding MC macroblock from the previous non-skipped frame stored in FB_1, denoted as MB_{t-2} in Figure 3.3. In order to locate MB_{t-2}, we need to find a motion vector of MB_{t-1}. Again, MB_{t-1} is not on a macroblock boundary; it is possible to use the bilinear interpolation from the motion vectors mv_{t-1,MB_1}, mv_{t-1,MB_2}, mv_{t-1,MB_3} and mv_{t-1,MB_4} which are the four neighboring macroblocks, MB_{t-1}^1, MB_{t-1}^2, MB_{t-1}^3 and MB_{t-1}^4, of MB_{t-1} to come up with an approximation of mv_{t-1} [21]. However, the bilinear interpolation of motion vectors leads to inaccuracy in the resultant motion vector because the area covered by the four macroblocks may be too divergent and too large to be described by a single motion vector [2]. Thus, the forward dominant vector selection (FDVS) method is used [2] to select one dominant motion vector from four neighboring macroblocks. A dominant motion vector is defined as the motion vector carried by a dominant macroblock. The dominant macroblock is the macroblock that has the largest overlapped segment with MB_{t-1}.

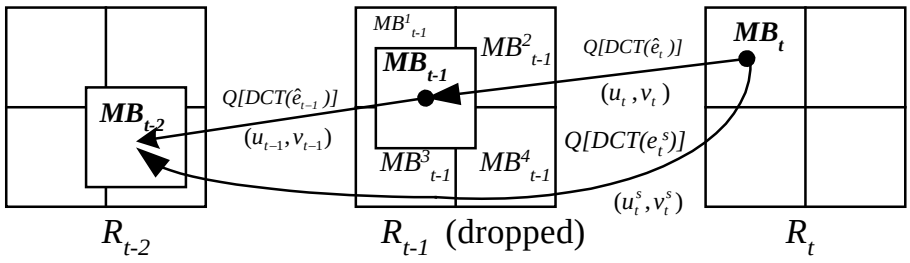


Figure 3.3. Residual signal re-computation of frame-skipping for MC macroblocks.

Hence, $\hat{e}_{t-1}$ can be computed and it is transformed and quantized to $Q[DCT(\hat{e}_{t-1})]$. In Figure 3.3, the newly quantized DCT coefficient $Q[DCT(e_t^s)]$ of a MC macroblock can then be computed by summing $Q[DCT(\hat{e}_{t-1})]$ and the incoming $Q[DCT(\hat{e}_t)]$ and it is quite similar to that of the non-MC macroblock as mentioned in (3.7) except the formation of $Q[DCT(\hat{e}_{t-1})]$. For non-MC macroblocks, $Q[DCT(\hat{e}_{t-1})]$ is available from the incoming bitstreams. Conversely, requantization is performed for the formation of $Q[DCT(\hat{e}_{t-1})]$ in MC macroblocks, which will introduce additional re-encoding errors Δ_{t-1}^s such that the reconstructed frame after the end-encoder is,

$$R_t^s = R_t + \Delta_{t-1}^s \tag{3.8}$$

Note that, as compared with Δ_t^s, Δ_{t-1}^s is the re-encoding due to frame *t-1* instead of frame *t*. Both of these errors will degrade the quality of the reconstructed frame. Since each non-skipped P-frame is used as a reference frame for the following non-skipped P-frame, quality degradation propagates to later frames in a cumulative manner. If the accumulated magnitude of re-encoding errors is large, it means that the quality of the transcoded sequence is degraded significantly. This is illustrated in Figure 6(a) which shows how re-encoding can lead to accumulated errors. According to the Figure, Δ_2^s is introduced for the formation of $Q[DCT(\hat{e}_2)]$ and such errors have the effect of degrading

the reconstructing quality of MB_3. When pixels in MB_3 are used as a reference for the subsequent non-skipped frame, for example, R_5^s in Figure 3.4(a), Δ_2^s will further affect the formation of $Q[DCT(\hat{e}_4)]$ and finally this error is accumulated in MB_5. These accumulated errors become significant in the sequence containing a large amount of MC macroblocks.

With the possibility of having re-encoding errors in MC macroblocks, it is obviously important to develop techniques to minimize the visual degradation caused by this phenomenon. Thus, a feedback loop is suggested as shown in Figure 3.1 to compensate for the re-encoding errors introduced in the previous frames. The forward and inverse DCT and quantization pairs in the feedback loop are mainly responsible for minimizing re-encoding errors. For these MC macroblocks, the quantized DCT coefficients are inversely quantized. An inverse DCT is then performed to form $\hat{e}_{t-1}$ with a re-encoding error, which subtracts the original $\hat{e}_{t-1}$ to generate the re-encoding error, Δ_{t-1}^s. The re-encoding error is stored in FB_2 can be written as

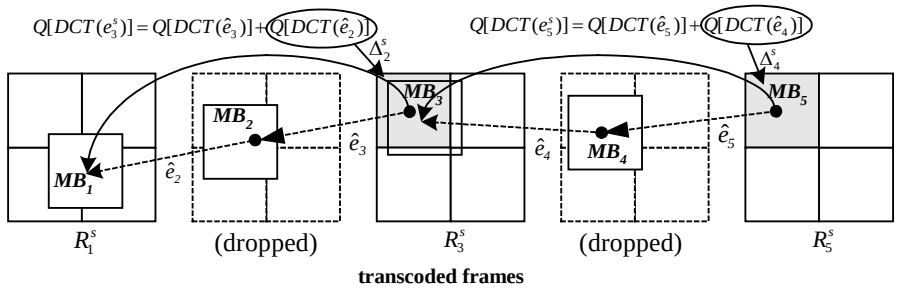
$$\Delta_{t-1}^s = DCT^{-1}\left(\left\lfloor \frac{DCT(\hat{e}_{t-1})}{q} \right\rfloor \times q\right) - \hat{e}_{t-1} \qquad (3.9)$$

where q is the quantization step-size and the floor function, $\lfloor a \rfloor$, extracts the integer part of the given argument a.

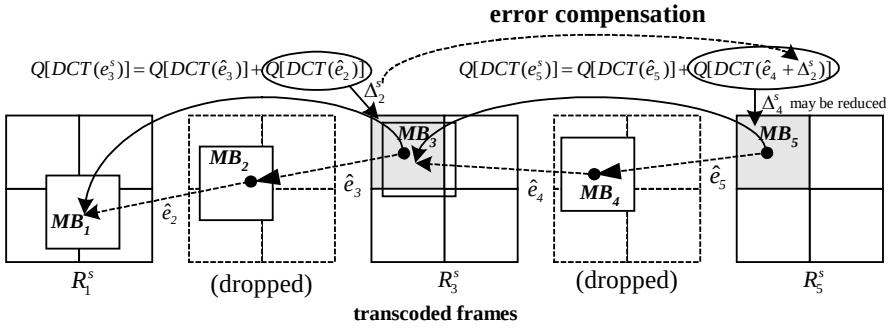
Since the motion vectors are highly correlated in the successive frames[41,45], it is observed that the spatial positions of MC macroblocks in certain frames are very close to the spatial positions of MC macroblocks in its subsequent frames. Thus, re-encoding errors stored in FB_2 are added to the prediction errors of MC macroblocks in the following P-frame to compensate for the re-encoding errors. For example, as shown in

Figure 3.4(b), Δ_2^s is added to $\hat{e}_4$ such that it is able to compensate for the re-encoding errors for the formation of $Q[DCT(\hat{e}_4)]$. Note that the feedback loop for error compensation cannot ensure the elimination of all the re-encoding errors generated by MC macroblocks and there still exists a certain amount of re-encoding errors after frame-skipping transcoding. However, these re-encoding errors are continuously accumulated in FB_2 such that most of them can be compensated for in the subsequent frames if the spatial positions of MC macroblocks between successive frames are highly correlated.

In order to reduce the implementation complexity of the MC macroblock, a cache subsystem is added to the proposed transcoder, as depicted in Figure 3.1. Since motion compensation of multiple macroblocks may require the same pixel data, a cache subsystem is implemented to reduce the redundant inverse quantization, inverse DCT and motion compensation computations. We have found that the arrangement is significant since the frequency of caching hits is high. This is due to the fact that the locality of motion often exists within each frame.



(a)



(b)

Figure 3.4 Effect of re-encoding error (a) without error compensation, and (b) with error compensation.

C. Buffer Arrangements for Multiple Frame Skipping

In Figure 3.1, it can be seen that three frame buffers were arranged in deriving the proposed architecture of frame-skipping transcoder. FB_1 is used to store the previous non-skipped frame. Since the main feature of the proposed transcoder is to operate the frame skipping in the DCT domain, the quantized DCT coefficients are updated in FB_{DCT}. To enhance the performance of the proposed transcoder, FB_2 is employed to store re-encoding errors to compensate for the accumulated errors. When multiple frames are dropped, the proposed frame-skipping transcoder can be processed in the forward order, thus eliminating the requirement for multiple buffers in FB_{DCT} and FB_2 which could be required to store the quantized DCT coefficients and re-encoding errors of all skipped frames respectively. Figure 3.5 shows a scenario when multiple frames are dropped. Assume that R_1 is the first non-skipped frame. At the beginning, the contents of FB_{DCT}

and FB_2 are initialized to zero. When R_2 is input to the transcoder, we directly store the incoming $Q[DCT(\hat{e}_2)]$ in FB_{DCT} since there is no skipped frame between R_1 and R_2. The stored coefficients of FB_{DCT} are used to update the DCT coefficients of the next skipped frame. This means that when R_2 is dropped, the proposed transcoder updates the DCT coefficients of prediction errors for each macroblock according to its coding mode. From (16), for non-MC macroblocks, $Q[DCT(e_3^s)]$ is obtained by a direct addition of $Q[DCT(\hat{e}_2)]$ in FB_{DCT} to the incoming $Q[DCT(\hat{e}_3)]$ of R_3. On the other hand, it is necessary to perform the re-encoding of non-MC macroblocks in order to compute $Q[DCT(\hat{e}_2)]$ which is added to the incoming $Q[DCT(\hat{e}_3)]$ for the formation of $Q[DCT(e_3^s)]$, FB_{DCT} is then updated with the new $Q[DCT(e_3^s)]$. Simultaneously, re-encoding errors Δ_2^s are stored in FB_2. For the next incoming frame R_4, $Q[DCT(e_4^s)]$ with error compensation can be iteratively computed as

$$Q[DCT(e_4^s)] = Q[DCT(\hat{e}_4)] + Q[DCT(e_3^s + \Delta_2^s)] \qquad (3.10)$$

Again, the re-encoding of $e_3^s + \Delta_2^s$ of non-MC macroblocks will generate accumulated errors Δ_3^s which are stored in FB_2. This iterative process has the advantage that only one pair of FB_{DCT} and FB_2 is needed for all skipped frames. The flexibility of multiple frame-skipping provides the fundamental framework for dynamic frame-skipping as described in the following.

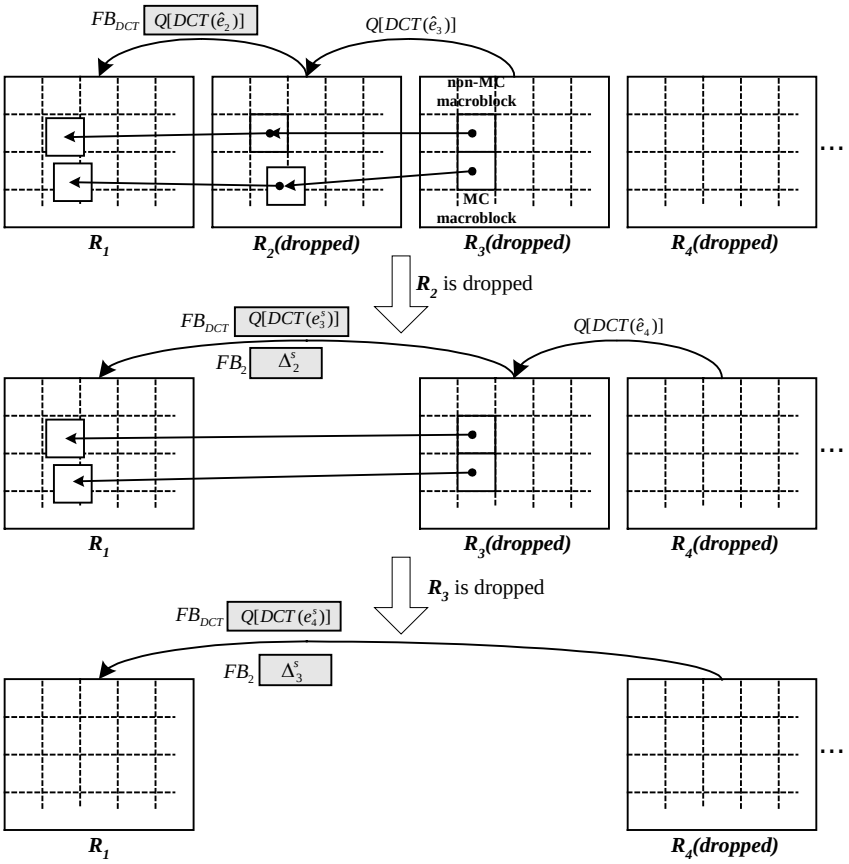


Figure 3.5 Multiple frame skipping of the proposed transcoder.

D. Dynamic Frame-Skipping Transcoder

Our proposed frame-skipping transcoder also develops a strategy for determining the length of the skipped frame such that it can reduce the quality degradation as well as minimize the motion jerkiness perceived by human beings. Traditionally, a motion vector is used to serve as a good indicator for dynamic frame skipping [21]. When multiple frames are dropped in the frame-skipping transcoder, re-encoding errors in MC macroblocks cannot be avoided entirely even though a feedback loop is applied to compensate for the accumulated errors, as mentioned in the previous section. It is observed that human eyes are sensitive to this type of quality degradation. Thus, it is necessary to regulate the frame rate of the transcoder by taking into account the effect of

re-encoding. The goal of the proposed dynamic frame-rate control scheme is to minimize the re-encoding errors as well as to preserve motion smoothness. To obtain a quantitative measure for frame-skipping, let us define a frame-skipping metric, $FSC_t^s(MA_t^s, RE_{t-1}^s)$, which is a function of the accumulated magnitude of the motion vectors and re-encoding errors due to transcoding for the macroblocks of the current frame. The metric is given by,

$$FSC_t^s(MA_t^s, RE_{t-1}^s) = \frac{\sum_{m=1}^{M}(MA_t^s)_m}{\sum_{m=1}^{M}(RE_{t-1}^s)_m} \tag{3.11}$$

where M is the total number of macroblocks in the current frame. Re-encoding errors after error compensation $(RE_{t-1}^s)_m$ are obtained by summing all requantization errors for the mth macroblock which can be written as,

$$(RE_t^s)_m = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\Delta_{t-1}^s(i,j) \tag{3.12}$$

where N is the size of macroblock and the corresponding motion activity $(MA_t^s)_m$ is given by

$$(MA_t^s)_m = \left|(u_t^s)_m\right| + \left|(v_t^s)_m\right| \tag{3.13}$$

where $(u_t^s)_m$ and $(v_t^s)_m$ are the horizontal and vertical components of the motion vector of the m^{th} macroblock which uses the previous non-skipped frame as its reference.

If the value of $FSC_t^s(MA_t^s, RE_{t-1}^s)$ following a non-skipped frame exceeds a predefined threshold, T_{FSC}, this incoming frame should be kept. $(MA_t^s)_m$ in (3.13) is used to detect the activity level of the m^{th} macroblock. If $\sum_{m=1}^{M}(MA_t^s)_m$ has a significant value, this implies that the incoming frame contains a certain amount of motion activities. It is

reasonable that the frame-rate control scheme be used to keep this frame since the previous non-skipped frame is not sufficient to represent the current frame. As a consequence, it is much better that the incoming frame be refreshed. However, we cannot guarantee the quality of the reconstructed frame due to re-encoding errors in the transcoder in cases where only the motion activity is used. Since the quality of the selected frame directly affects the motion smoothness of the transcoded sequence, it is usually beneficial to maintain the selected frame at a good reconstruction quality. The conventional algorithm fails to meet this objective. Thus, $\sum_{m=1}^{M}(RE_t^s)_m$ in (3.11) is used to measure re-encoding errors in the incoming frame. Also, a larger value of $\sum_{m=1}^{M}(RE_t^s)_m$ implies more re-encoding errors, and it will reduce the value of $FSC_t^s(MA_t^s, RE_{t-1}^s)$ such that the incoming frame is not kept even though it contains a certain amount of motion activities. Note that the threshold, T_{FSC}, is simply set according to the target frame rate of the transcoder, f_T. The feedback signal from the output buffer to the dynamic control scheme in Figure 3.1 is used to stabilize the outgoing frame rate of the transcoder, f_o, by adjusting the value of T_{FSC} dynamically. Initially, T_{FSC} is set to its initial value T_{init}. T_{FSC} can then be updated as follows:

- If $f_o > f_T$, increase T_{FSC} by T_{step}.
- If $f_o < f_T$, decrease T_{FSC} by T_{step}.
- Otherwise, keep the current value of T_{FSC}.

where T_{step} in above are the step size for adjusting T_{FSC}.

3.1.3 Simulation Results

Simulations have been performed to evaluate the overall efficiency of various frame-skipping transcoders. Two sets of experiments were carried out. First, in the front encoder, the first frame was encoded as intraframe (I-frame), and the remaining frames were encoded as interframes (P-frames). In the first simulation, bi-directional frames (B-frames) were not considered. Second, we demonstrate the performance of the proposed transcoder when the incoming bitstreams were encoded with B-frames. In both simulations, picture-coding modes were preserved during transcoding.

A. Performance of the proposed techniques on frame-skipping transcoder

The first set of experiments aims at evaluating the performances of the proposed techniques including the direct addition for non-MC macroblocks (DA), the error compensation for MC macroblocks (EC) and the dynamic selection of non-skipped frames by employing the incoming motion vectors and the re-encoding error ($FSC_t^s(MA_t^s, RE_{t-1}^s)$) when applied to the frame-skipping transcoder. Different front encoders were employed to encode two sets of video sequences with different spatial resolutions and motion characteristics. “Salesman”, “Forman” and “Carphone” are typical videophone sequences in QCIF (176×144) format, which are used to show the performance of the proposed frame-skipping transcoder in multipoint video conferencing [22]. H.263 is currently the best standard for practical video conferencing and its range of target bitrates is about 10-2048 Kb/s. Since H.263 could be superior to H.261 at any bitrate [42-43], an H.263 TMN8 front encoder [33] was employed to encode “Salesman”, “Forman” and “Carphone” at different bitrates (64 Kb/s and 128 Kb/s). On the other hand, “Tennis” and “Football” in SIF (352×240) containing high motion activities were

encoded by a MPEG2 TM5 front encoder [32] at different bitrates (1.5 Mb/s and 3 Mb/s), but only P-frames were generated. For all testing sequences, the frame-rate of the incoming bitstream was 30 frames/s.

To verify the performances of the proposed techniques, extensive simulations were carried out. Results of the simulations are used to compare the performance of a reference transcoder which is a conventional pixel-domain transcoder (CPDT) by employing FDVS to compose an outgoing motion vector from the incoming motion vectors of the skipped frames [2], named as CPDT+FDVS. Table 3.4 shows the simulation conditions for different transcoders examined. Note that BiVS represents the bilinear interpolation vector selection for composing an outgoing motion vector [21]. The detailed comparisons of the average PSNR between CPDT+FDVS and the proposed transcoders including DA+FDVS, DA+BiVS+EC, and DA+FDVS+EC are tabulated in Table 3.5 in which the frames are temporally dropped by a factor of 1, 2 and 3. We show that both DA+FDVS, DA+BiVS+EC and DA+FDVS+EC outperform CPDT+FDVS in all cases. The results are more significant for the non-MC macroblock because a direct addition of the DCT coefficients should not introduce any re-encoding errors. Also, Figure 3.6 shows that the proposed transcoders have a speed-up of about 4.5-10 times faster than that of CPDT+FDVS. This is because the probability of the non-MC macroblock happens more frequently in typical sequences, and we can achieve significant computational savings while maintaining good video quality on these non-MC macroblocks. In addition, the cache system in the proposed transcoders can reduce the computational burden of re-encoding the MC macroblocks.

Table 3.4. Simulation conditions.

	Conventional pixel-domain transcoder		Proposed transcoders			
	CPDT + FDVS	CPDT + FDVS + $FSC_t^s(MA_t^s)$	DA + FDVS	DA + FDVS + EC	DA + BiVS + EC	DA + FDVS + EC + $FSC_t^s(MA_t^s, RE_{t-1}^s)$
Motion vector composition method	FDVS	FDVS	FDVS	FDVS	BiVS	FDVS
Direction addition (DA)	—	—	ON	ON	ON	ON
Error compensation (EC)	—	—	OFF	ON	ON	ON
Dynamic control scheme	OFF	$FSC_t^s(MA_t^s)$	OFF	OFF	OFF	$FSC_t^s(MA_t^s, RE_{t-1}^s)$

Table 3.5 and Figure 3.6 also compare the average PSNR and complexities of two transcoders using different approaches for composing outgoing motion vectors: DA+FDVS+EC and DA+BiVS+EC. As shown in Table 3.5, DA+FDVS+EC is consistently better than DA+BiVS+EC at various outgoing frame rates. It is significant to note that the inaccuracy of the resultant motion vector of BiVS affects the average PSNR of the MC macroblock. Also, the complexity of DA+BiVS+EC is slightly higher than that of DA+FDVS+EC as shown in Figure 3.6. Therefore, FDVS is more suitable for frame-skipping transcoders.

In Table 3.5, it can be seen that DA+FDVS+EC has a PSNR improvement over DA+FDVS. This result is expected since the feedback loop of DA+FDVS+EC is enabled which can reduce the re-encoding errors of MC macroblocks. In other words, the average PSNR of the MC macroblock of DA+FDVS+EC is better than that of DA+FDVS. The advantage of error compensation is significant for sequences containing high motion activities. In the "Salesman" sequence, the average PSNR of DA+FDVS+EC is almost the same as that of DA+FDVS. However, DA+FDVS+EC significantly outperforms DA+FDVS for all "Foreman", "Carphone", "Table Tennis" and "Football" sequences which contain certain amount of motion activities. There is also about 0.3dB improvement of the non-MC macroblock in high moving sequences in which they are transcoded into half of the incoming frame rate. Let us use Figure 3.7(b) to give a clearer

account of this phenomenon. For DA+FDVS, pixels in region *A* have re-encoding errors since these pixels are located at the MC macroblocks of R_{t-2}. These errors can be accumulated to the non-MC macroblock in R_t. The PSNR improvement of the non-MC macroblock of DA+FDVS+EC is due to the contribution from the feedback loop of error compensation in which requantization errors are stored in FB_2 and is fed back to the quantizer to compensate for the requantization errors introduced in the previous frame. In Figure 3.7(a), we plot the average PSNR improvement of the non-MC macroblock for different skipping factors. For sequences containing high motion activities such as "Foreman" and "Carphone", significant improvement in average PSNR, of about 0.85dB and 1.25dB, have been achieved when the frames are temporally dropped by a factor of *2* and *3* respectively. The reason for this is illustrated in Figure 3.7(c) when the skipping factor is equal to *2*. Pixels in both regions *B* and *C* will contribute accumulated errors in the non-MC macroblock of R_t and the effect of accumulated errors is serious. However, the technique of error compensation can reduce these accumulated errors. The results of "Table Tennis" and "Football" sequences appear to be similar to that of the above, as shown in Table 3.5. On the other hand, the improvement for the "Salesman" sequence is not remarkable, because the sequence has only low motion activities.

Even though DA+FDVS+EC can greatly improve the overall performance as compared with CPDT+FDVS, the abrupt change in PSNR is significant, as shown in Figure 3.8 and it affects the motion smoothness of the transcoded sequence which probably introduces a flickering effect. Although the fluctuation of PSNR is not an exact measure of the flickering effect, it is fair to say that the flickering effect can be reduced by a smaller fluctuation of PSNR. DA+FDVS+EC is further enhanced by incorporating

the proposed frame rate control scheme $FSC_t^s(MA_t^s, RE_{t-1}^s)$, named as DA+FDVS+EC+ $\mathrm{FSC}_t^s(\mathrm{MA}_t^s, \mathrm{RE}_{t-1}^s)$. We have set the parameters T_{init} and T_{step} in $FSC_t^s(MA_t^s, RE_{t-1}^s)$ to *20* and *5*, respectively for the rest of the comparison. The PSNR performance of DA+FDVS+EC+ $\mathrm{FSC}_t^s(\mathrm{MA}_t^s, \mathrm{RE}_{t-1}^s)$ in transcoding various video sequences is shown in Table 3.6, Table 3.7 and Figure 3.8, for which the target frame rate f_T was set to 7.5 frames/s. In Table 3.6 and Table 3.7, DA+FDVS+EC+ $\mathrm{FSC}_t^s(\mathrm{MA}_t^s, \mathrm{RE}_{t-1}^s)$ increases the average PSNR of the proposed transcoder by about 0.2-0.3 dB while it reduces the fluctuation of PSNR, as depicted in Figure 3.8. The figure also shows that $FSC_t^s(MA_t^s, RE_{t-1}^s)$ outperforms the conventional frame-rate control scheme by using the incoming motion vectors only, $FSC_t^s(MA_t^s)$ [21]. This is because $FSC_t^s(MA_t^s, RE_{t-1}^s)$ can reduce re-encoding errors by preserving the high quality reference frame with high motion activities. These results show that $FSC_t^s(MA_t^s, RE_{t-1}^s)$ is able to strengthen our new architecture of frame-skipping transcoder and provides the decoded sequence with a smoother motion as well as better transcoded pictures.

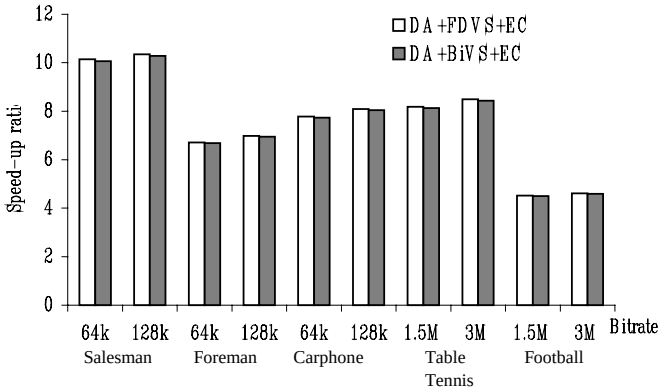
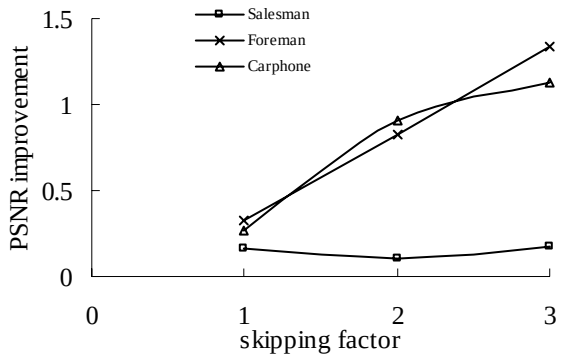
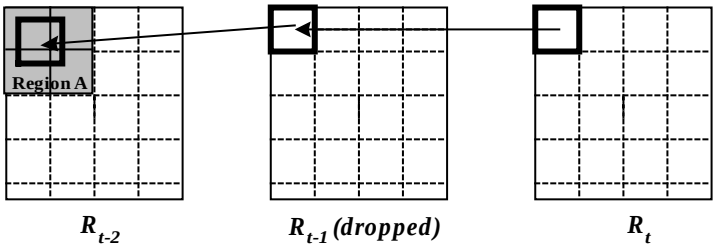


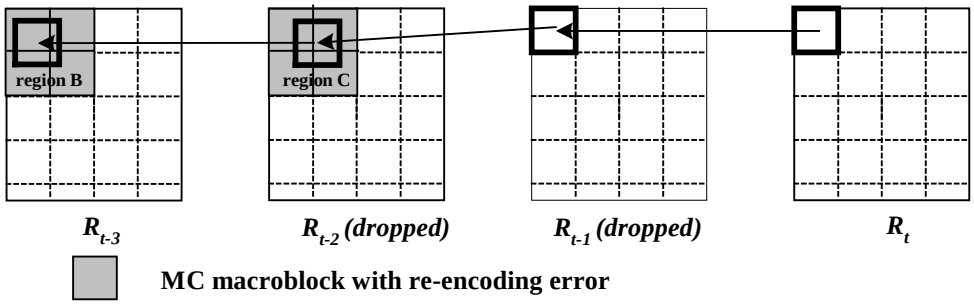
Figure 3.6 Speed-up ratio of various transcoders as compared with CPDT+FDVS, where the frame rate of the incoming bitstream was 30 frames/s and then transcoded to 10 frames/s. The front encoder for encoding "Salesman", "Foreman" and "Carphone" was H.263 TMN8 [33]; while MPEG2 TM5 [32] was used to encode "Table Tennis" and "Football".



(a)



(b)



(c)

Figure 3.7 Effect of error compensation in multiple frame skipping. (a) Average PSNR improvement of the non-MC macroblock against skipping factor for "Salesman", "Foreman" and "Carphone" sequences encoded at 128Kb/s. (a) Accumulated error in single frame skipping. (c) Accumulated error in multiple frame skipping.

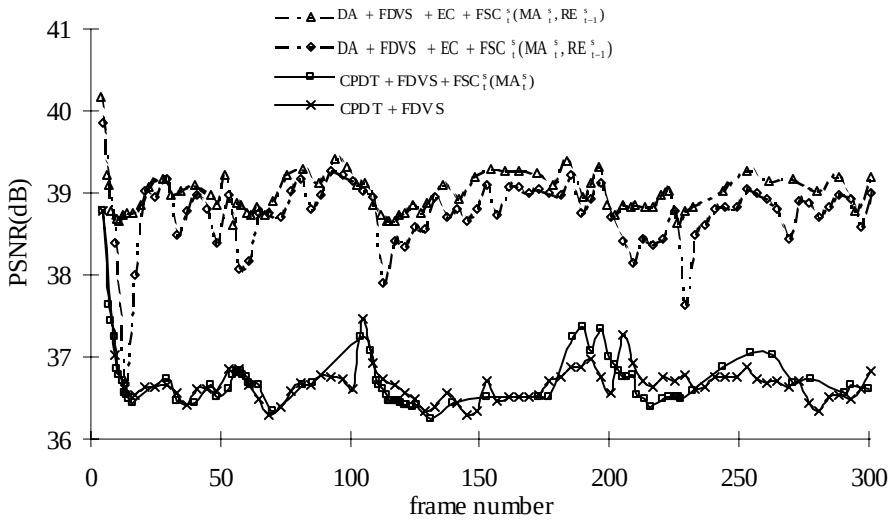


Figure 3.8 PSNR of the proposed dynamic frame-skipping transcoder of "Salesman" sequence encoded at 128 Kb/s with 30 frames/s, and then transcoded to 32 Kb/s with about 7.5 frames/s.

Table 3.5 Average PSNR of the proposed transcoders without frame-rate control, where the frame rate of the incoming bitstream was 30 frames/s. The front encoder for encoding "Salesman", "Foreman" and "Carphone" was H.263 TMN8 [33]; while MPEG2 TM5 [32] was used to encode "Table Tennis" and "Football".

Sequences	Input bitrate	CPDT+FDVS			DA+FDVS			DA+BiVS+EC			DA+FDVS+EC		
		MC region	Non-MC region	All region	MC Region	Non-MC region	All region	MC region	Non-MC region	All region	MC region	Non-MC region	All region
Output sequences are transcoded to 15 frames/s													
Salesman (176x144)	64k	30.76	33.54	33.20	31.69	35.77	35.33	31.56	35.82	35.37	31.99	35.89	35.42
	128k	34.41	36.91	36.71	34.86	39.14	38.62	34.60	39.22	38.69	35.03	39.30	38.78
Foreman (176x144)	64k	29.72	31.06	30.87	30.95	33.16	32.28	31.05	33.32	32.38	32.25	33.48	32.80
	128k	33.25	34.98	34.64	34.78	37.03	36.05	34.92	37.23	36.22	36.10	37.35	36.60
Carphone (176x144)	64k	30.86	32.59	32.28	31.29	34.47	33.71	31.38	34.62	33.86	32.36	34.77	34.14
	128k	33.40	35.25	34.92	34.89	37.12	36.25	35.14	37.28	36.37	36.01	37.39	36.64
Table Tennis (352x240)	1.5M	30.89	31.57	31.41	31.26	34.59	34.00	31.39	34.74	34.16	32.07	34.84	34.19
	3M	33.92	34.60	34.44	34.74	37.19	36.82	34.89	37.34	36.95	35.68	37.45	37.04
Football (352x240)	1.5M	29.69	30.73	30.03	30.89	32.76	32.04	31.05	32.94	32.18	32.16	33.08	32.64
	3M	33.60	34.34	33.87	35.19	36.41	36.02	35.46	36.62	36.25	36.57	36.78	36.68
Output sequences are transcoded to 10 frames/s													
Salesman (176x144)	64k	30.64	33.72	33.28	31.48	35.71	35.10	31.35	35.74	35.12	31.83	35.84	35.31
	128k	34.42	36.99	36.77	34.76	39.15	38.53	34.59	39.18	38.54	35.01	39.26	38.76
Foreman (176x144)	64k	29.58	30.92	30.73	30.21	32.58	31.69	30.26	33.09	32.24	31.61	33.29	32.70
	128k	33.16	34.92	34.57	33.68	36.08	35.15	33.77	36.75	35.81	35.03	36.90	36.42
Carphone (176x144)	64k	30.87	32.49	32.25	31.06	33.89	32.93	31.14	34.40	33.49	32.17	34.59	33.96
	128k	33.41	35.16	34.90	33.76	36.03	35.32	34.31	36.82	36.12	35.21	36.94	36.45
Table Tennis (352x240)	1.5M	30.68	31.47	31.36	31.05	34.24	33.35	31.17	34.51	33.57	31.80	34.67	33.87
	3M	33.76	34.48	34.28	34.19	36.29	35.70	33.97	36.89	36.07	34.83	37.06	36.44
Football (352x240)	1.5M	29.26	30.62	29.96	30.22	32.16	31.23	30.27	32.69	31.53	31.58	32.89	32.26
	3M	33.32	34.25	33.80	34.16	35.74	34.98	34.28	36.08	35.22	35.49	36.31	35.92
Output sequences are transcoded to 7.5 frames/s													
Salesman (176x144)	64k	29.90	33.71	33.20	31.33	35.56	34.94	31.19	35.60	35.07	31.72	35.73	35.22
	128k	34.36	36.92	36.69	34.57	38.98	38.36	34.42	39.05	38.49	34.95	39.16	38.74
Foreman (176x144)	64k	29.32	30.64	30.45	29.63	32.38	31.28	29.65	33.03	32.02	31.05	33.26	32.65
	128k	32.93	34.82	34.06	33.06	35.52	34.45	33.11	36.65	35.56	34.41	36.86	36.18
Carphone (176x144)	64k	30.89	32.40	32.21	31.98	33.49	32.55	31.09	34.20	33.28	32.04	34.37	33.74
	128k	33.40	35.09	34.87	33.64	35.76	35.05	33.98	36.75	35.87	35.08	36.89	36.21
Table Tennis (352x240)	1.5M	30.66	31.37	31.17	31.07	33.92	33.12	31.11	34.26	33.38	31.65	34.42	33.65
	3M	33.73	34.40	34.21	34.10	36.05	35.51	33.71	32.61	32.92	34.71	36.95	36.32
Football (352x240)	1.5M	28.97	30.33	29.68	29.62	31.88	30.81	29.70	32.60	31.21	31.05	32.85	31.99
	3M	33.09	34.12	33.63	33.58	35.24	34.44	33.56	35.95	34.81	34.86	36.26	35.59

Table 3.6 Average PSNR and speed-up ratio of various dynamic transcoders as compared with CPDT+FDVS using H.263 TMN8 [33] as a front encoder.

Sequence	Method	Average PSNR	Speed-up ratio	Average encoded frames per second
Input bitrate is 64kbps with 30 frames/s				
Salesman	DA + FDVS	33.20	-	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s)$	33.24	0.94	7.8
	DA + FDVS + EC	35.22	9.95	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s, RE_{t-1}^s)$	35.48	8.16	7.5
Foreman	DA + FDVS	30.45	-	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s)$	30.48	0.96	7.4
	DA + FDVS + EC	32.65	6.45	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s, RE_{t-1}^s)$	32.95	4.95	7.4
Carphone	DA + FDVS	32.21	-	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s)$	32.27	0.96	7.4
	DA + FDVS + EC	33.74	7.38	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s, RE_{t-1}^s)$	33.97	6.15	7.4
Input bitrate is 128kbps with 30 frames/s				
Salesman	DA + FDVS	36.69	-	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s)$	36.72	0.94	7.8
	DA + FDVS + EC	38.74	10.15	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s, RE_{t-1}^s)$	38.98	8.31	7.5
Foreman	DA + FDVS	34.06	-	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s)$	34.12	0.96	7.4
	DA + FDVS + EC	36.18	6.62	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s, RE_{t-1}^s)$	36.46	5.54	7.4
Carphone	DA + FDVS	34.87	-	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s)$	34.92	0.96	7.4
	DA + FDVS + EC	36.21	7.65	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s, RE_{t-1}^s)$	36.49	6.42	7.4

Table 3.7 Average PSNR and speed-up ratio of various dynamic transcoders as compared with CPDT+FDVS using MPEG2 TM5 [32] as a front encoder.

Sequence	Method	Average PSNR	Speed-up ratio	Average encoded frames per second
Input bitrate is 1.5Mbps with 30 frames/s				
Table Tennis	DA + FDVS	31.17	-	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s)$	31.24	0.96	7.4
	DA + FDVS + EC	33.65	7.86	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s, RE_{t-1}^s)$	33.88	6.65	7.4
Football	DA + FDVS	29.68	-	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s)$	29.77	0.97	7.4
	DA + FDVS + EC	31.99	4.39	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s, RE_{t-1}^s)$	32.30	3.85	7.4
Input bitrate is 3Mbps with 30 frames/s				
Table Tennis	DA + FDVS	34.21	-	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s)$	34.31	0.96	7.4
	DA + FDVS + EC	36.32	8.17	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s, RE_{t-1}^s)$	36.59	6.91	7.4
Football	DA + FDVS	33.63	-	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s)$	33.70	0.97	7.4
	DA + FDVS + EC	35.59	4.48	7.5
	DA + FDVS + EC + $FSC_t^s(MA_t^s, RE_{t-1}^s)$	35.91	3.96	7.4

B. Results with bi-directional frames (B-frames)

In this simulation, "Tennis", "Football" and "Stefan" sequences were encoded by a MPEG-2 TM5 front encoder with the structure "*IBBPBBPBBP…*" as the group of pictures (GOP). The bitstreams were then transcoded into lower frame rates. In Figure 3.9, frames are presented in the display order, but are numbered in the encoding order. Considering that the first input frame to the transcoder is I_o and its second one is P_1. The dynamic transcoding of P-frames involves the process of selecting the most representative frames according to the frame-skipping metric ($FSC_t^s(MA_t^s, RE_{t-1}^s)$) or the conventional scheme ($FSC_t^s(MA_t^s)$). The next incoming bi-directional frames, B_2 and B_3, are predicted from I_o and P_1. Due to this dependence, the selection of B-frames, B_2 and B_3, has to be postponed after their referenced frames, I_o and P_1, are transcoded. One straightforward approach to selecting a B-frame, which depends upon its referenced P-frames, is discussed as follows:

1. If one of its referenced frames is not selected, this B-frame is dropped. For example, as illustrated in Figure 3.9, when P_1 is not selected, all frames which reference to P_1 such as B_2, B_3, B_5 and B_6 are dropped.
2. If both of its referenced frames are selected, the B-frames between their reference frames may also be significant. In such cases, for each B-frame, we can approximate its value of frame-skipping metric, by assuming this metric between the frames is uniform in a short period of time, such that the frame-skipping metric of B-frame is a scaled version of its corresponding metric of P-frame. In Figure 3.9, P_4 and P_7 are selected. Then, the frame-skipping metrics of B_8 and B_9

become $\frac{1}{3}FSC_7^s$ and $\frac{2}{3}FSC_7^s$ respectively. If the scaled metric of B-frame is larger than T_{FSC}, this B-frame is selected. Otherwise, it is dropped.

In transcoding the B-frame, the quantized DCT coefficients of non-skipped B-frames can be directly obtained from the incoming bitstream because both of its reference frames are available. Besides, since a B-frame is not used as reference for further prediction, it is not necessary to update all buffers in the frame-skipping transcoder. This technique can be easily integrated into the DA+FDVS+EC+ $\mathrm{FSC}_t^s(\mathrm{MA}_t^s)$ and DA+FDVS+EC+ $\mathrm{FSC}_t^s(\mathrm{MA}_t^s, \mathrm{RE}_{t-1}^s)$, and the results are shown in Figure 3.10 and Table 3.8. It is seen that the proposed DA+FDVS+EC+ $\mathrm{FSC}_t^s(\mathrm{MA}_t^s, \mathrm{RE}_{t-1}^s)$ has about a 2.5dB PSNR improvement as compared to that of DA+FDVS+EC+ $\mathrm{FSC}_t^s(\mathrm{MA}_t^s)$. The complexity of DA+FDVS+EC+ $\mathrm{FSC}_t^s(\mathrm{MA}_t^s, \mathrm{RE}_{t-1}^s)$ is also less than that of DA+FDVS+EC+ $\mathrm{FSC}_t^s(\mathrm{MA}_t^s)$. These demonstrate the effect of the proposed frame-skipping transcoder when the incoming bitstream contains B-frames.

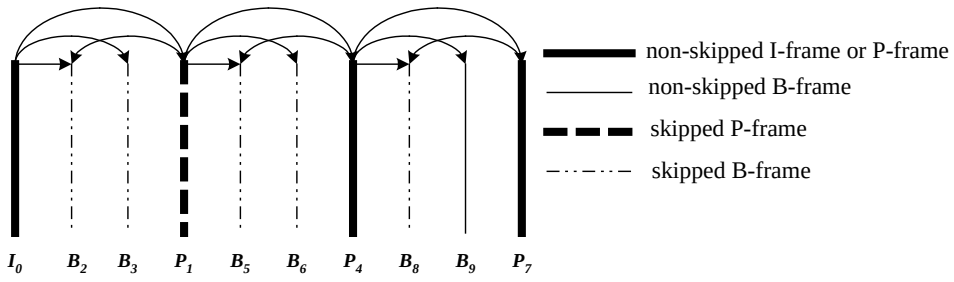


Figure 3.9 Input and output frames in the display order, but are numbered in the encoding order.

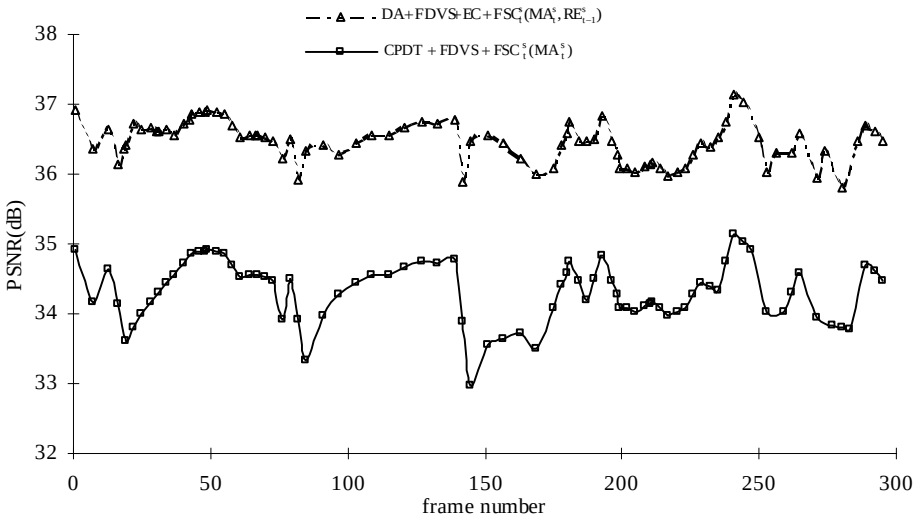


Figure 3.10 PSNR of the proposed dynamic frame-skipping transcoder. The "Table Tennis" sequence was encoded by MPEG2 TM5 [32] using a coding pattern "IBBPBBP…" with a bitrate of 3Mb/s. The incoming frame rate was 30 frames/s, and the frame rate after transcoding was 7.5 frames/s with 0.75 Mb/s.

Table 3.8 Average PSNR and speed-up ratio of the proposed dynamic transcoder as compared with $\mathrm{DA+FDVS+EC+FSC}_t^s(\mathrm{MA}_t^s)$. The front encoder was MPEG2 TM5 [32] with GOP structure of "IBBPBBP…".

Sequences	Input bitrate	Output bitrate	$\mathrm{DA+FDVS+EC+FSC}_t^s(\mathrm{MA}_t^s)$	$\mathrm{DA+FDVS+EC+FSC}_t^s(\mathrm{MA}_t^s, \mathrm{RE}_{t-1}^s)$	
			Average PSNR	Average PSNR	Speed-up ratio
Table Tennis (352x240)	1.5M	0.375M	31.26	33.95	7.05
	3M	0.75M	34.33	36.66	7.38
Football (352x240)	1.5M	0.375M	29.81	32.38	4.06
	3M	0.75M	33.73	35.98	4.17
Stefan (720x480)	7M	1.7M	30.61	33.11	4.32
	14M	3.5M	33.93	36.26	4.45

3.1.4 Conclusions

In this chapter, we have proposed a new architecture for a low-complexity and high quality frame-skipping transcoder. Its low complexity is achieved by: 1) a direct addition of the DCT coefficients for macroblocks coded without motion compensation to deactivate most of the complex modules of the transcoder, and 2) a cache subsystem for motion-compensated macroblocks to reduce the redundant IDCT and inverse quantization. Furthermore, we have also shown that a direct addition of the DCT coefficients on macroblocks without motion compensation and error compensation on

motion-compensated macroblocks can reduce significantly the re-encoding errors due to transcoding. The overall performance of the proposed architecture produces a better picture quality than the conventional frame-skipping transcoder at the same reduced bitrates.

Furthermore, our proposed frame-skipping transcoder can be processed in the forward order when multiple frames are dropped. Thus, only one DCT-domain buffer is needed to store the updated DCT coefficients of all skipped frames. By using such a mechanism, a new frame-rate control scheme for the proposed transcoder is also suggested in this chapter. Since the quality of the non-skipped frame impacts directly the motion smoothness of the transcoded sequence, it is beneficial to force the frame-rate control scheme to select frames which have good quality for reconstruction. The proposed scheme can dynamically adjust the number of skipped frames depending upon re-encoding errors as well as the accumulated magnitude of all of the motion vectors in the current frame. Experimental results show that our proposed dynamic frame-rate control scheme provides a decoded sequence with a smoother motion as well as better transcoded pictures.

3.2 On Re-composition of Motion Compensated Macroblocks for DCT-based Video Transcoding

3.2.1 Introduction

In this section, we focus on the transcoding process of the motion compensated macroblocks in the DCT domain. A new architecture of the frame-skipping transcoder to reduce the computational complexity of motion compensated macroblocks in the frame-skipping process is introduced. The new architecture transcodes the dominant region of a motion compensated macroblock in the DCT domain by making use of the DCT coefficients of the incoming bistream and some pre-computed shift operators. By using a shifted version of the dominant vector, the re-encoding error introduced in the dominant region can be avoided. On the other hand, an adaptive transcoding architecture to transcode the boundary regions of MC marcoblocks and a way to perform error compensation are proposed. This architecture can further speed up the transcoding process of the motion compensated macroblocks. Half pixel accuracy related to our proposed frame skipping transcoder is also addressed.

In the previous section, we proposed a new DCT-based video transocder to perform frame skipping in the DCT domain. Figure 3.11 shows a simplified architecture of video transcoder proposed in [1]. Two switches, SW_1 and SW_2, are employed to update the DCT-domain buffer for the transformed and quantized residual signal. The switching depends on the coding mode originally used in the front encoder for the current macroblock being processed. Note that switch SW_3 is used to adjust the frame rate and refresh the buffer for the reconstructed non-skipped frame. This switch is controlled by a

dynamic control scheme which makes use of both motion vectors and re-encoding errors. For transcoding non-MC macroblocks(see Figure 3.12), we have

$$Q[DCT(e_t^s)] = Q[DCT(\hat{e}_t)] + Q[DCT(\hat{e}_{t-1})] \quad (3.14)$$

where $Q[DCT(e_t^s)]$ represents the newly quantized DCT coefficients of prediction errors, $Q[DCT(\hat{e}_t)]$ represents the quantized DCT coefficients at time t and $Q[DCT(\hat{e}_{t-1})]$ represents the quantized DCT coefficients at time t-1[1]. Therefore, the previous residual signal in the DCT-domain is directly fed back from DCT domain buffer, FB_{DCT}, to the adder, which adds the input residual signal to update FB_{DCT} . Note that all operations are performed in the DCT-domain; thus the complexity of the frame-skipping transcoder is reduced. However for MC macroblock transcoding, full re-encoding process is required and the major problem with these MC macroblocks is that re-encoding errors will be generated, which introduces quality degradation in the transcoded sequence. Thus, the error compensation block as shown in Figure 3.11 tries to compensate the re-encoding errors introduced in the previous frames. This transcoding process of the motion compensated macroblocks introduces high computational complexity. In addition, for a real image sequence, the block motion field [36-38] is usually gentle and smooth, and varies slowly. As a consequence, the distribution of motion vectors is center-biased [39,40,52], as demonstrated by some typical examples as shown in Table 3.9, which gives the distribution of the coding modes for various sequences, including “Salesman”, “Foreman”, “Carphone”, “Table Tennis” and “Football”. These sequences have been selected to show different amount of motion activities. It is clear that, for encoding with half-pixel precision, about 76% and 27% of the macroblocks are coded without motion compensation for sequences containing small

and large amount of motion activities respectively. In order to support high transcoding ratio and to give high quality transcoded video for mobile or hand-held devices, a good MC macroblock transcoding technique is necessary. In this section, we propose a new scheme to the tackle problems of re-encoding error and high computational complexity introduced in the video transcoding process in the DCT domain. The major difficulty to transcode a MC macroblock is that the targeted DCT coefficients are not available from the incoming bitstream. In the following, we will give a detail analysis on how to re-estimate these DCT coefficients. Our approach gives a possible algorithm for the re-composition of the DCT coefficients. The proposed architecture can also be used to enhance an existing DCT-based spatial transocder to give wide applications.

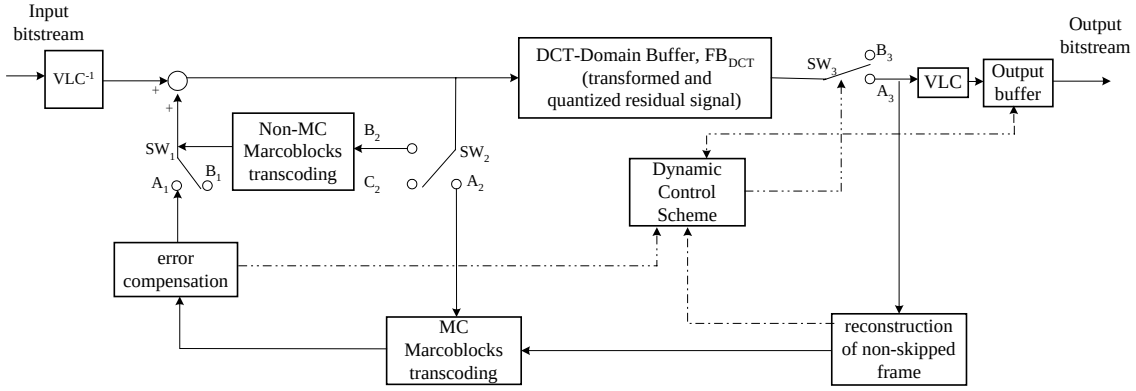


Figure 3.11 Simplified architecture of the video transcoder proposed in [1].

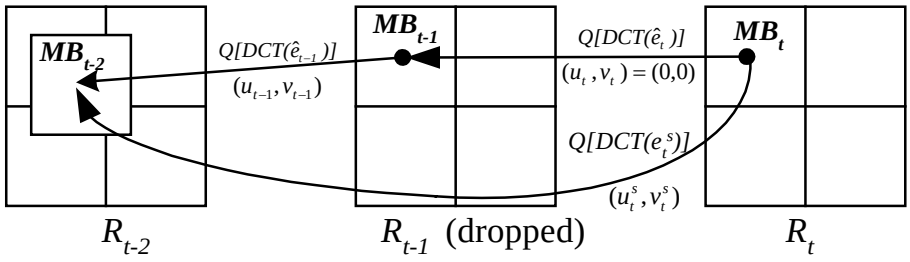


Figure 3.12. Residual signal re-computation of frame skipping for non-MC macroblocks.

Table 3.9. Percentage of non-MC macroblock for various sequences using half-pixel accuracy.

Salesman	Foreman	Carphone	Table Tennis	Football
76%	42%	48%	54%	27%

3.2.2 Low-complexity frame-skipping for high performance video transcoding on MC macroblocks

In [1], we have proposed a direct addition of the DCT coefficients in frame-skipping transcoding to avoid re-encoding errors and to reduce the computational complexity for macroblocks coded without motion compensation. In this section, we present a new frame-skipping transcoding architecture which is an extension of the work of [1]. The new architecture gives focus on the following areas.

1. Recomposition of MC macroblocks from the dominant region and the boundary region of MC macroblocks in the DCT domain using a shifted version of the dominant motion vector.
2. Suggesting an adaptive transcoding of MC macroblocks boundary.
3. Fast transcoding of the dominant region of MC macroblocks in DCT domain using significant DCT coefficients.

The architecture of the proposed transcoder is shown in Figure 3.13. The input bitstream is first parsed with a variable-length decoder to extract the header information, coding mode, motion vectors and quantized DCT coefficients for each macroblock. Let us define the incoming residual signal with quantization errors due to the front encoder as $\hat{e}_t = e_t + \Delta_t$ and its quantized DCT coefficients as $Q[DCT(\hat{e}_t)]$, where $\hat{e}_t$, e_t and Δ_t represent the incoming residual signal with quantization errors due to the front encoder, the quantized residual signal and the quantization errors due to front encoder respectively. Each macroblock is then manipulated independently. Two switches, SW_1 and SW_2, are

employed to update the DCT-domain buffer, FB_{DCT}, for the transformed and quantized residual signal. The selection depends on the coding mode originally used in the front encoder for the current macroblock being processed. The switch positions for different coding modes are shown in Table 3.10. For non-MC macroblocks, the previous residual signal in the DCT-domain is directly fed back from FB_{DCT} to the adder, which adds the input residual signal to update FB_{DCT}. Note that all operations are performed in the DCT-domain, thus the complexity of the frame-skipping transcoder is reduced. Also, the degradation in quality of the transcoder introduced by requantization error, Δ_t^s, is avoided.

When the motion compensation mode is used, appropriate shift operators in DCT domain, selective inverse quantization, selective inverse 1D-DCT, motion compensation, selective 1D-DCT, selective quantization modules are activated to update FB_{DCT}. The major difficulty to transcode these MC macroblocks is that re-encoding errors will be generated, which introduce quality degradation in the transcoded sequence. Also, high computational complexity is required. Motivated by this, we make use of the DCT properties to decompose the macroblock into two regions: dominant region and boundary region. Thus, some shift operators in the DCT domain and the shifted version of dominant motion vector are proposed to avoid the re-encoding errors introduced in the dominant region and to reduce the computational complexity by making use of the incoming DCT coefficients. Note that switch SW_3 is used to adjust the frame rate and refresh FB_1 for the non-skipped frame. This switch is controlled by a dynamic control scheme which makes use of the motion vectors. Table 3.11 shows the frame-skipping modes of the proposed transcoder. In order to reduce the computational complexity during the MC macroblock transcoding, switch SW_4 is used to further speed up the

transcoding process when fast dominant region transcoding is employed. Table 3.12 shows the switch modes of different dominant region transcoding approaches of the proposed transcoder. The following subsections will give the advantages of the DCT-domain buffer arrangement, together with the details of other methods.

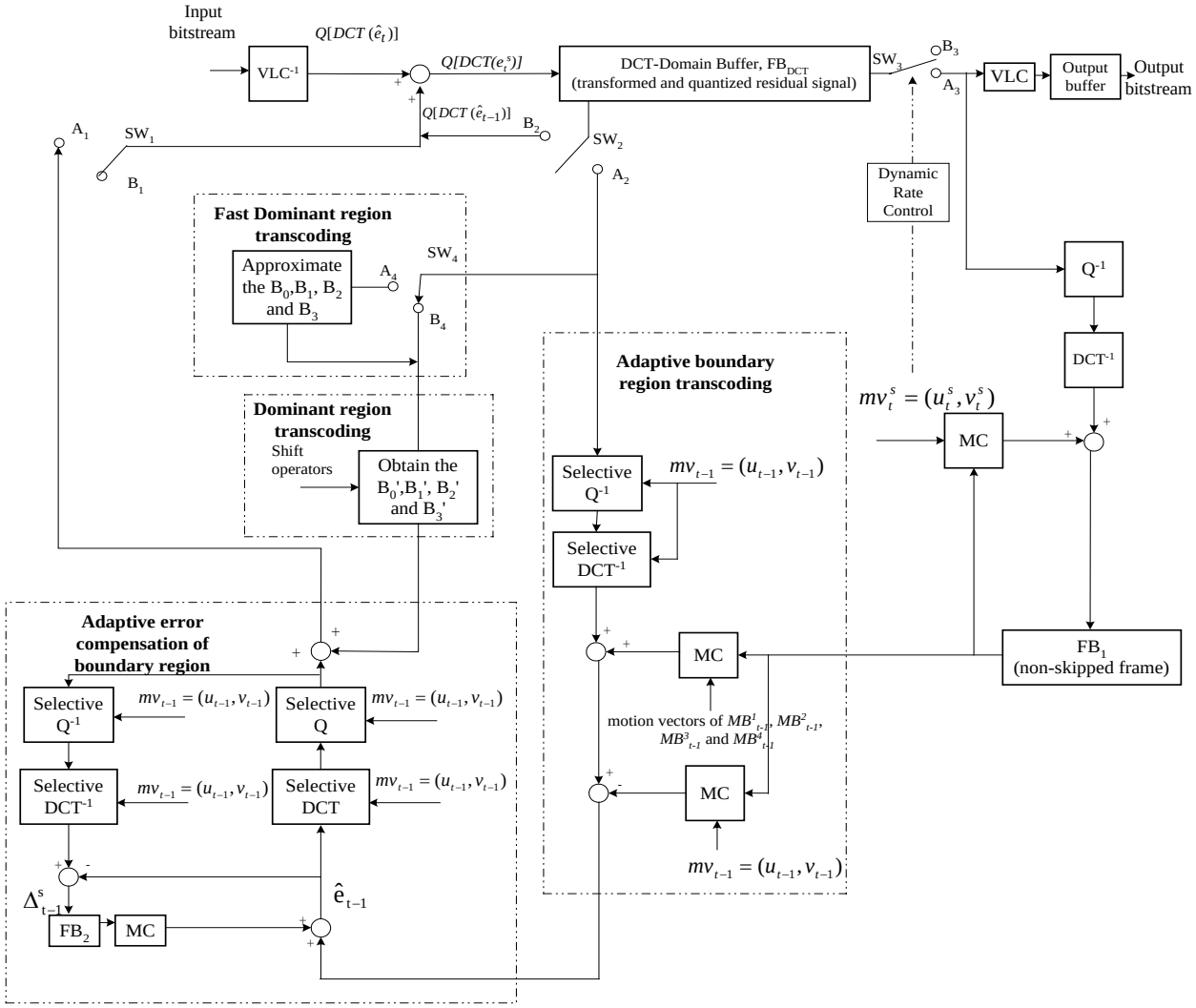


Figure 3.13 Architecture proposed for frame-skipping transcoder.

Table 3.10. Different coding modes of switches SW_1 and SW_2 of the proposed transcoder.

Coding mode	SW_1 Position	SW_2 Position
Non MC	B_1	B_2
MC	A_1	A_2

Table 3.11. Switch positions for different frame-skipping modes of the proposed transcoder.

Frame-skipping mode	SW_3 Position
Skipped frame	B_3
Non-skipped frame	A_3

Table 3.12. Switch positions for different dominant region transcoding approaches of the proposed transcoder.

Dominant region transcoding	SW_4 Position
Use all DCT coefficients	B_4
Fast Dominant region transcoding	A_4

A. Recomposition of MC macroblocks from the dominant region and the boundary region of MC macroblocks in DCT domain using a shifted version of the dominant motion vector

For MC macroblocks, direct addition cannot be employed since the macroblock in time t-1, MB_{t-1}, does not overlap completely with MB_{t-1}^1 as shown in Figure 3.14. In other words, $Q[DCT(\hat{e}_{t-1})]$ is not available from the incoming bitstream. Figure 3.14 also shows that MB_{t-1} is formed by using parts of four segments which come from its four neighboring macroblocks. Let us define these four neighboring macroblocks as MB_{t-1}^1, MB_{t-1}^2, MB_{t-1}^3 and MB_{t-1}^4. It is possible to use the incoming quantized DCT coefficients of MB_{t-1}^1, MB_{t-1}^2, MB_{t-1}^3 and MB_{t-1}^4, to come up with $Q[DCT(\hat{e}_{t-1})]$.

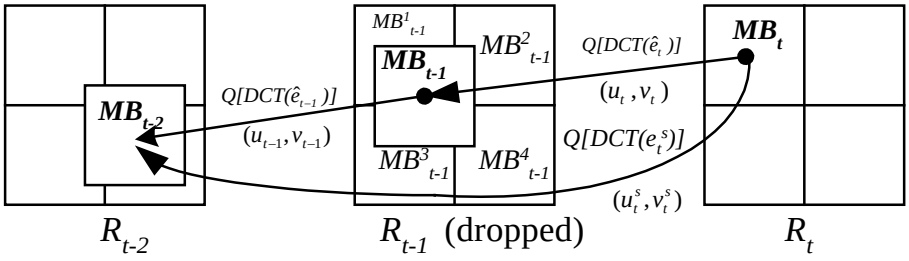


Figure 3.14. Residual signal re-computation of frame-skipping for MC macroblocks.

In[1], we obtained $Q[DCT(\hat{e}_{t-1})]$ by performing inverse DCT, inverse quantization, DCT and requantization. This process requires high computational complexity and introduces re-encoding errors. In this section, we split the MC macroblocks into two regions: dominant region and boundary region as shown in Figure 3.15. In the dominant region, we propose a shifted version of the dominant vector and a shift operator to compute the DCT coefficients in DCT domain. This is to achieve low computational complexity and avoid re-encoding errors.

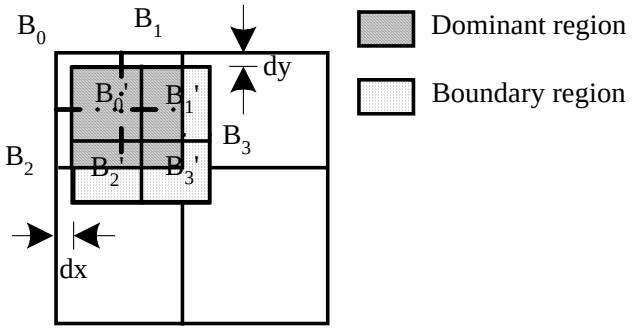


Figure 3.15. Dominant region and the boundary region of MB_{t-1} in the DCT domain.

Figure 3.16 shows the scenario that the dominant macroblock is pointing at the previous non-skipped frame. The prediction errors of R_{t-1} obtained from the incoming bitstream are B_0, B_1,...B_{15} as shown in Figure 3.17, where B_0, B_1, B_2, and B_3 have a motion vector which is considered as the dominant vector in this case, whilst all other blocks have three different motion vectors. Since this frame is skipped, the new

prediction errors B_0', B_1', B_2' and B_3' of the previous non-skipped frame as shown in Figure 3.15 have to be found in order to employ the technique of direct addition of DCT coefficients. Since B_0, B_1, B_2 and B_3 have the dominant vector (u_{t-1},v_{t-1})as shown in Figure 6, the new prediction errors B_0', B_1', B_2' and B_3' can be obtained by using a shifted version of the dominant vector, $(\hat{u}_{t-1},\hat{v}_{t-1})$, which is equal to the sum of the shift, (dx,dy) and (u_{t-1},v_{t-1}), where (dx,dy) is a shift determined by motion vectors from R_t. This is to avoid the re-encoding error in the dominant region if B_0', B_1', B_2' and B_3' are obtained directly from B_0, B_1, B_2, B_3, B_4, B_5, B_8, B_{10} and B_{12} in the DCT domain as shown in Figure 3.17.

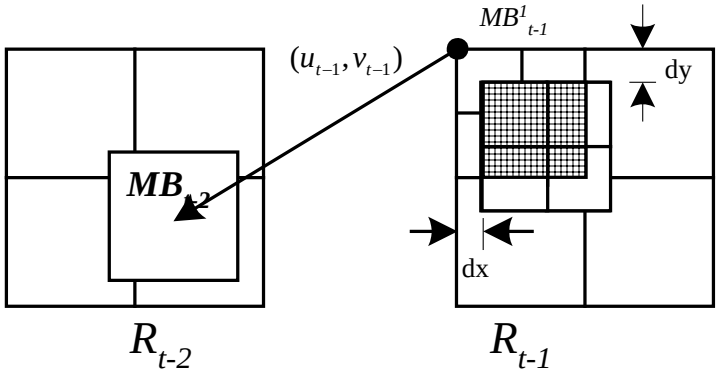


Figure 3.16. Dominant macroblock and the previous non-skipped frame.

B_0	B_1	B_8	B_9
B_2	B_3	B_{10}	B_{11}
B_4	B_5	B_{12}	B_{13}
B_6	B_7	B_{14}	B_{15}

Figure 3.17. Incoming DCT coefficients of four macroblocks, each of which consists of four blocks.

Since the shifted version of the dominant vector is used, the corresponding prediction error in the dominant region can be obtained without performing full re-

encoding process. If the other three motion vectors have the same direction as the dominant vector, the whole new prediction (B_0', B_1', B_2' and B_3') can be obtained in the DCT domain. In other words, no re-encoding error will be introduced. Otherwise, a decomposition of the block into a dominant region and three boundary regions is needed. In the following of this section, a way of transcoding the dominant region of an MC macroblock is introduced and an adaptive approach for transcoding the MC macroblock boundary regions is discussed in next section.

The major idea to obtain the DCT coefficients in the dominant region is introduced in Chapter 2.6. Note that the DCT of S_{i1} and S_{i2} can be pre-computed and B_i can be extracted from the incoming bitstream. In this case the re-encoding errors are avoided and hence the computational complexity can be reduced. If macroblocks MB_{t-1}^1, MB_{t-1}^2, MB_{t-1}^3 and MB_{t-1}^4 (see Figure 3.14) have the same motion vector as the dominant vector, B_1',B_2' and B_3' can be also obtained using this approach. Otherwise, a decomposition of the dominant region and boundary region is needed as shown in Figure 3.13. Using the following properties of the DCT,

$$DCT(A+B)=DCT(A)+DCT(B) \tag{3.15}$$

where A and B represent the pixels in the dominant region and three boundary regions with a size of 8x8 respectively. We can split each of blocks B_1',B_2' and B_3'in two regions: dominant region and boundary region as shown in Figure 3.15.

Then the dominant region of B_2' can be obtained by considering blocks B_4 and B_5 to have zero DCT coefficients as shown in Figure 3.17. Similarly, the dominant regions of B_1' and B_3' can be obtained by considering blocks B_8 and B_{10} to have zero DCT coefficients and blocks $B_{5,}$ B_{10} and B_{12} to have zero DCT coefficients. Hence the DCT

values of dominant region A can be obtained. Note that the DCT values of B_1', B_2' and B_3' can not be obtained since the DCT values in boundary region B have not been considered as described in equation (3.15). In the following, the DCT values of boundary region B have to be obtained separately by using 1-D inverse DCT, motion compensation, forward 1D-DCT and requantization as shown in Figure 3.13. In this process, re-encoding error is introduced due to requantization.

B. Adaptive Transcoding of MC macroblocks boundary

For transcoding an MC boundary, only selective inverse quantization and selective inverse 1D-DCT of the quantized DCT coefficients of MB_{t-1}^1, MB_{t-1}^2, MB_{t-1}^3 and MB_{t-1}^4 (excluding the dominant ones) need to be performed in the boundary regions as shown in Figure 3.15. Note that each macroblock is composed of four 8×8 blocks in video coding standards [10,30,31,34,44,], and the DCT and quantization operations are performed on units of 8×8 blocks. When processing MB_{t-1}^1, MB_{t-1}^2, MB_{t-1}^3 and MB_{t-1}^4 (excluding the dominant one), only their corresponding 8×8 blocks, which have pixels overlapping with the MB_{t-1} boundary region, are subject to partial inverse 1D-DCT computation. Hence, the partial inverse 1D-DCT is only performed in the boundary regions, and the motion vector, mv_{t-1} as shown in Figure 3.13, is needed as an input pointer for the selective inverse 1D-DCT module to control which rows or columns need the inverse 1D-DCT. In most cases, this approach significantly reduces the required number of column or row inverse DCTs as compared with the inverse 2D-DCT approach.

Each segment of the reconstructed pixels in the MB_{t-1} boundary region can be obtained by adding its prediction errors to its motion-compensated segment of the previous non-skipped frame stored in FB_1, as shown in Figure 3.13. After all pixels in

MB_{t-1} boundary region are reconstructed, we need to find the prediction error, $\hat{e}_{t-1}$. Actually, $\hat{e}_{t-1}$ is equal to the reconstructed pixel in MB_{t-1} boundary region subtracted from its corresponding MC macroblock from the previous non-skipped frame stored in FB_1, denoted as MB_{t-2} in Figure 3.14. In order to locate MB_{t-2}, we need to find a motion vector for MB_{t-1}. A shifted version of the dominant vector is used to select one dominant motion vector from four neighboring macroblocks with a shift of (dx,dy). A dominant motion vector is defined as the motion vector attached to a dominant macroblock which is the macroblock that has the largest overlapping segment with MB_{t-1}.

Hence, $\hat{e}_{t-1}$ can be computed and it is partially transformed by putting zero values in the dominant region and partially quantized to $Q[DCT(\hat{e}_{t-1})]$. After the DCT coefficients of region B are obtained, B_2', B_3'and B_1' can be reconstructed by adding DCT(A) and DCT(B) together using equation (3.15).

In Figure 3.14, the newly quantized DCT coefficient $Q[DCT(e_t^s)]$ of an MC macroblock can then be computed by adding $Q[DCT(\hat{e}_{t-1})]$ to the incoming $Q[DCT(\hat{e}_t)]$ and it is quite similar to that of the non-MC macroblock, except for the formation of $Q[DCT(\hat{e}_{t-1})]$. For non-MC macroblocks, $Q[DCT(\hat{e}_{t-1})]$ is available from the incoming bitstreams. Conversely, requantization has to be performed for the formation of $Q[DCT(\hat{e}_{t-1})]$ in MC macroblock boundaries, which will introduce additional re-encoding errors Δ_{t-1}^s such that the reconstructed frame after the end-encoder becomes,

$$R_t^s = R_t + \Delta_{t-1}^s \tag{3.16}$$

Note that, as compared with Δ_t^s in (3.16), Δ_{t-1}^s is the re-encoding error due to frame *t-1* instead of frame *t* and the re-encoding errors occur only at the boundary region.

However, these errors will degrade the quality of the reconstructed frame. Since each non-skipped P-frame is used as a reference frame for the following non-skipped P-frame, quality degradation propagates to later frames in a cumulative manner. If the accumulated magnitude of re-encoding errors is large, it means that the quality of the transcoded sequence degrades significantly. These accumulated errors become significant in the sequence containing a large amount of MC macroblocks and high motion activity.

With the possibility of having re-encoding errors in MC macroblocks, it is obviously important to develop techniques to minimize the visual degradation caused by this phenomenon. Thus, a feedback loop is suggested as shown in Figure 3.13 to compensate for the re-encoding errors introduced in the previous frames. The selective forward and inverse 1D-DCT and selective quantization pairs in the feedback loop are mainly responsible for the minimization of re-encoding errors. For these MC macroblocks, the quantized DCT coefficients are inversely quantized. An inverse 1D-DCT is then performed. The re-encoding error is obtained by the following equation and stored in FB_2:

$$\Delta^{s}_{t-1} = DCT^{-1}\left(\left\lfloor \frac{DCT(\hat{e}_{t-1})}{q} \right\rfloor \times q\right) - \hat{e}_{t-1} \tag{3.17}$$

where q is the quantization step-size and the floor function, $\lfloor a \rfloor$, extracts the integer part of the given argument a.

Since motion vectors are highly correlated in successive frames[41,45], it is observed that the spatial positions of MC macroblocks in certain frames are very close to the spatial positions of MC macroblocks in their subsequent frames. Thus, re-encoding

errors stored in FB_2 are added to the prediction errors of MC macroblocks in the following P-frame to compensate for the re-encoding errors. Note that the feedback loop for error compensation cannot ensure the elimination of all the re-encoding errors generated by MC macroblock boundary regions and there still exists a certain amount of re-encoding errors after frame-skipping transcoding. However, these re-encoding errors will be continuously accumulated in FB_2 such that most of them can be compensated for in the subsequent frames if the spatial positions of MC macroblocks between successive frames are highly correlated.

C. Fast DCT-based frame skipping transcoder on MC macroblocks using significant coefficients

Since the energy distributions of a DCT block obtained from the MC-DCT will mainly concentrate on the low frequency region. It is beneficial to approximate the DCT coefficients using significant DCT coefficients to speed up the transcoding process. Eqn. 3.18 gives the definition of the energy function, while eqn. 3.19 gives an approximation which can be used to find the number of significant DCT coefficients for a specific energy level.

$$energy = \sum_{l=0}^{N-1}\sum_{m=0}^{N-1} B^2(l,m) \tag{3.18}$$

$$\text{Approx. energy} = \sum_{l=0}^{j}\sum_{m=0}^{k} B^2(l,m) \tag{3.19}$$

where $B(l,m)$ represents row l and column m DCT coefficients. j and k represent the numbers of rows and columns to approximate the original DCT coefficients. Initially, j and k are set to zero. If the approximated energy is less than 0.95 of the original energy, j or k will be increased by 1 until the approximation of significant DCT coefficients are

obtained. From our experiment, this approach only introduces a drop of 0.04 to 0.19 dB video quality in the MC macroblock transcoding. However, it can further increase the speed of the MC-DCT transcoding between 1.6 and 5 times, especially the incoming video is encoded at a low bit rate. A detailed analysis will be discussed in the following section.

3.2.3 Experimental Results

A large number of experiments have been performed to evaluate the overall efficiency of various frame-skipping transcoders. Two situations were considered. In both cases, the first frame of a sample sequence was encoded as an intraframe (I-frame), and the remaining frames were encoded as interframes (P-frames). In the first situation, half-pixel accuracy was not considered, whilst in the second situation half-pixel precision was used. In both situations, picture-coding modes were preserved during transcoding.

3.1 Performance of the proposed techniques on frame-skipping transcoder

The first set of experiments aims at evaluating the performance of the proposed techniques including direct addition for MC macroblocks in the dominant region, error compensation for MC macroblocks boundary and fast MC-DCT transcoding using significant DCT coefficients when applied to frame-skipping transcoder. Different front encoders were employed to encode two sets of video sequences with different spatial resolutions and motion characteristics. “Salesman”, “Forman” and “Carphone” are typical videophone sequences in QCIF (176×144) format, which were used to show the performance of the proposed frame-skipping transcoder in multipoint video conferencing [7]. H.263 is currently the best standard for practical video conferencing and its range of target bitrates is about 10-2048 Kb/s. Since H.263 could be superior to H.261 at any

bitrate [21-22], an H.263 TMN8 front encoder [23] was employed to encode “Salesman”, “Forman” and “Carphone” at different bitrates (64 Kb/s and 128 Kb/s). On the other hand, “Tennis” and “Football” in SIF (352×240) containing high motion activities were encoded by an MPEG2 TM5 front encoder [24] at different bitrates (1.5 Mb/s and 3 Mb/s), but only P-frames were generated. For all testing sequences, the frame-rate of the incoming bitstream was set to 30 frames/s.

Results of the simulations are used to compare the performance of a reference transcoder which is a conventional pixel-domain transcoder (CPDT) employing forward dominant vector selection (FDVS) to compose an outgoing motion vector from the incoming motion vectors of the skipped frames [9-10], named as CPDT+FDVS. Table 3.13 shows the simulation conditions for different transcoders examined. Besides, a DCT-based frame-skipping transcoder which employs Direct Addition of DCT coefficients using FDVS[12], named as DA+FDVS, is also compared. A detailed comparison of the average PSNR between CPDT+FDVS, DA+FDVS and our proposed transcoder using MC macroblock transcoding in dominant and boundary regions, named as DA+FDVS+MCDCT, are given in Table 3.14 (see columns 1-11). In these evaluations, the frames were temporally dropped by a factor of 3. It is shown that DA+FDVS+MCDCT outperforms CPDT+FDVS and DA+FDVS in all cases. The results are more significant for MC macroblocks because our proposed MCDCT transcoding in the dominant region does not introduce any re-encoding errors and the transcoding of the dominant region is done in the DCT domain. Also, our proposed transcoders which have a speed-up of about 1.5-2.3 times faster than that of the DCT-based transcoder (see column 12). This is because we can achieve significant

computational savings while maintain good video quality on both non-MC macroblocks and dominant region of MC macroblocks in these sequences without performing full decoding and re-encoding process. For sequences with low motion activity, such as the salesman sequence, the motion vector is usually small. Therefore, the dominant region is large in MC macroblocks. Hence, a significant improvement of about 2.2-2.4dB in MC regions can be achieved. However, the average PSNR and the speed up ratio do not have significant improvement since less than 10% of the MC macroblocks are coded. For "Forman", "Carphone", "Table tennis" and "Football" sequences, the average PSNR and speed up ratio become significant since more MC macroblocks are coded. The average PSNR improvement is about 1.3-2dB and the speed up ratio is about 2.1-2.3 times. In the next section, our proposed video transocoding will be discussed when the front encoder takes half pixel precision.

Table 3.13. Simulation conditions.

	Conventional pixel-domain transcoder	DCT-domain transcoder		Proposed transcoders			
	CPDT + FDVS	DA + FDVS	DA + FDVS + EC	DA + FDVS + MCDCT	DA + FDVS + FMCDCT	DA + FDVS + MCDCT + EC	DA + FDVS + FMCDCT + EC
Direct addition on non-MC macroblock using dominant vector selection (DA+FDVS)	—	ON	ON	ON	ON	ON	ON
Error compensation (EC)	—	OFF	ON	OFF	OFF	ON	ON
Proposed MC-DCT transcoding using shifted version of dominant vector (MCDCT)	—	OFF	OFF	ON	OFF	ON	OFF
Proposed Fast MC-DCT transcoding using shifted version of dominant vector with significant coefficients approximation (FMCDCT)	—	OFF	OFF	OFF	ON	OFF	ON

Table 3.14. Performance of the proposed transcoders, where the frame rate of the incoming bitstream was 30 frames/s. The front encoder for encoding "Salesman", "Foreman" and "Carphone" was H.263 TMN8 [23]; while MPEG2 TM5 [24] was used to encode "Table Tennis" and "Football". All results are expressed in PSNR, except that column 12 and 14 indicate the speed up ratios.

Sequences	Input bitrate	CPDT+FDVS			DA+FDVS			DA+FDVS+MCDCT			Speed up ratio for DA+FDVS+MCDCT to DA+FDVS	DA+FDVS+FMCDCT(Fast approach)		DA+FDVS+EC	DA+FDVS+MCDCT+EC	DA+FDVS+FMCDCT+EC (Fast approach)
		MC region	Non-MC region	All regions	MC Region	Non-MC region	All regions	MC region	Non-MC region	All regions		All regions	Speed up ratio for DA+FDVS+FMCDCT to DA+FDVS+MCDCT			
Column 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Salesman (176x144)	64k	29.90	33.71	33.20	31.33	35.56	34.94	33.66	35.61	35.41	1.61	35.36	1.62	35.22	35.56	35.51
	128k	34.36	36.92	36.69	34.57	38.98	38.36	36.83	39.04	38.75	1.52	38.71	1.48	38.74	39.13	39.09
Foreman (176x144)	64k	29.32	30.64	30.45	29.63	32.38	31.28	31.79	32.99	32.63	2.2	32.47	4.03	32.65	33.39	33.23
	128k	32.93	34.82	34.06	33.06	35.52	34.45	35.58	36.8	36.47	2.18	36.30	3.83	36.18	37.19	37.02
Carphone (176x144)	64k	30.89	32.40	32.21	31.98	33.49	32.55	32.78	34.34	34.15	2.16	34.08	3.64	33.74	34.64	34.57
	128k	33.40	35.09	34.87	33.64	35.76	35.05	36.17	37.05	36.75	2.13	36.67	3.45	36.21	37.32	37.24
Table Tennis (352x240)	1.5M	30.66	31.37	31.17	31.07	33.92	33.12	32.79	34.68	34.42	2.12	34.27	3.36	33.65	34.89	34.74
	3M	33.73	34.40	34.21	34.10	36.05	35.51	36.08	37.19	37.06	2.10	36.90	3.18	36.32	37.60	37.44
Football (352x240)	1.5M	28.97	30.33	29.68	29.62	31.88	30.81	31.69	32.54	32.17	2.30	31.98	5.37	31.99	32.86	32.67
	3M	33.09	34.12	33.63	33.58	35.24	34.44	36.07	36.43	36.26	2.28	36.08	5.00	35.59	36.96	36.78

Columns 13 and 14 of table 3.14 also compare the average PSNR and complexities of our proposed transcoders: DA+FDVS+MCDCT and our proposed transcoder using significant DCT coefficients for MC macroblock transcoding in dominant region, named as DA+FDVS+FMCDCT. As shown in the table, DA+FDVS+FMCDCT has similar performance with the DA+FDVS+MCDCT with a quality degradation of about 0.04-0.19dB. Note that the speed up has been increased further from 1.4 to 5.3 times. Therefore, DA+FDVS+MCDCT is more suitable for low bitrate frame-skipping transcoders application. As expected the DA+FDVS+FDMCDCT has a slight PSNR degradation over DA+FDVS+MCDCT, since not all DCT coefficients are used in the MC transcoding process. However, the speed up of transcoding MC macroblocks has been increased significantly, especially in low bitrate applications. Since the DCT coefficients concentrate mainly on low frequency part and only a few

DCT coefficients are non-zero in low bitrate video coding. Therefore, the approximation is very near to the actual value and the number of significant DCT coefficients is much less than 64. In the "Salesman" sequence, the speed up performance is about 1.48, since most of the macroblocks were coded in non-MC format. For sequences "Foreman", "Carphone", "Table Tennis" and "Football", the average PSNR degradation of DA+FDVS+FMCDCT is about 0.07-0.19dB. However, the speed up of DA+FDVS+FMCDCT outperforms significantly the DA+MCDCT by about 3.1-5.3 times for all sequences examined.

The last three columns of table 3.14 also show the effect of error compensation. Note that our proposed transcoding approach outperforms the DCT-based approach since error compensation is used to control the accumulation of errors instead of eliminating all re-encoding errors in the MC transcoding process. In other words, the difference of PSNR performance between our proposed transcoder and the DCT trascoder is due to our design that re-encoding error is not introduced in the dominant region for using the proposed MCDCT transcoding approach. From columns 15-17, we can see that all DCT based approaches compensate a lot of the errors successfully. This error compensation scheme will introduce high computational complexity since the IDCT and inverse quantization need to be performed for MC macroblocks. However, the MC-DCT video transcoding algorithm involves re-encoding errors only in the boundary regions of MC macroblocks. Hence, error compensation needs to be performed only in these regions. In other words, the selective inverse 1D-DCT and selective inverse quantization only require to serve the boundary regions. Therefore, low computational complexity can be achieved in the error compensation process.

3.2 Results with half pixel precision

In this part of our experimental work, half-pixel precision for motion estimation has been used for the front encoder. Hence, the percentage of Non-MC macroblocks is lower than the front encoder in the previous experiments. The average PSNR performance of "Salesman", "Foreman", "Carphone", "Table Tennis" and "Football" sequences using the conventional approach increases about 0.21-0.7dB as compared with the previous experiment since the incoming video quality increases as shown in Table 3.14 and Table 3.15. However, the performance of the DCT-based transcoder has a slight quality degradation which is about 0.07-0.51dB. It is due to the fact that direct addition of DCT coefficients can only be applied to Non-MC macroblocks. Conversely, our proposed MCDCT transcoding approach can transcode the dominant region of MC macroblocks in the DCT domain to avoid quality degradation as shown in Table 3.15 and Figure 3.18. More importantly, the speed up ratio also becomes significant when our proposed fast algorithm is adopted. It is about 4-5 times faster than a direct realization of the algorithm since the percentage of MC macroblocks becomes significant. The major computational requirement comes from the process for transcoding the MC macroblocks since direct addition of DCT coefficients cannot be used.

Table 3.15. Average PSNR a of various dynamic transcoders as compared with CPDT+FDVS using H.263 TMN8 [23] as a front encoder for encoding "Salesman", "Foreman" and "Carphone"; while MPEG2 TM5 [24] was used to encode "Table Tennis" and "Football" with half pixel precision.

Sequences	Input bitrate	CPDT+ FDVS	DA+FDVS+ MCDCT+EC	DA+FDVS+ EC	DA+FDVS+ FMCDCT+EC
Salesman (176x144)	64k	33.41	35.74	34.72	35.68
	128k	36.91	39.33	38.23	39.28
Foreman (176x144)	64k	31.05	33.95	32.51	33.79
	128k	34.63	37.71	36.01	37.52
Carphone (176x144)	64k	32.74	35.16	33.67	35.08
	128k	35.38	37.80	36.02	37.71
Table Tennis (352x240)	1.5M	31.75	35.45	33.48	35.29
	3M	34.77	38.14	36.07	37.96
Football (352x240)	1.5M	30.38	33.41	31.74	33.24
	3M	34.28	37.58	35.32	37.38

Frame	196	200	204
(a) CPDT + FDVS			
(b) DA + FDVS +EC			
(c) DA + FDVS + MCDCT + EC (the present approach)			

Figure 3.18. Performance of the proposed video transcoder: (a) approach using CPDT+FDVS (b) approach using DA+FDVS+EC and (c) the present approach using DA+FDVS+MCDCT+EC.

3.2.4 Conclusion

In this section, we have proposed a new architecture for a low-complexity and high quality frame-skipping transcoder to resolve the problem of MC macroblcoks transcoding. Its low complexity is achieved by: 1) re-using the DCT coefficients for macroblocks coded with motion compensation to deactivate most of the complex modules of the transcoder, 2) proposing to use an adaptive MC macroblock boundary

transcoding and error compensation scheme using selective inverse 1D-DCT, 1D-DCT, quantization and inverse quantization to process motion-compensated macroblocks, and 3) employing a fast DCT-based frame skipping transcoder on MC macroblocks, which makes use of significant coefficients to speed up the transcoding process. Furthermore, we have also shown that a direct addition of DCT coefficients on macroblocks with motion compensation in dominant regions can reduce significantly the re-encoding errors due to transcoding. The overall structure of the proposed architecture produces a better picture quality than the conventional frame-skipping transcoder and DCT-based transcoder using the same reduced bitrates.

Furthermore, our proposed MCDCT frame-skipping transcoder can process frames in the forward order when multiple frames are dropped. Thus, only one DCT-domain buffer is needed to store the updated DCT coefficients of all skipped frames. Besides, the performance of the front encoder with half pixel accuracy is also evaluated. Since the percentage of MC-macroblocks becomes significant, our proposed transcoder outperform the DCT transcoder in terms of the video quality and computational complexity. It is due to the fact that our proposed transcoder decomposes MC macroblocks into a dominant region and three boundary regions. Therefore, low computational complexity can be achieved since only the boundary region is required to transcode in pixel-domain and perform error compensation. In other words, we can perform the transcoding process in most regions in the DCT domain and avoid quality degradation. Experimental results confirm that our proposed MCDCT transcoder has outstanding performance for transcoding MC macroblocks which are elementary items for hybrid video coding.

3.3 Architecture of Wavelet-based Frame Skipping Transcoder

3.3.1. Introduction

The MPEG video compression standard incorporated several scalable modes which include signal-to-noise ratio(SNR) scalability, spatial scalability and temporal scalability. The major concern of these modes are layered instead of being continuously scalable. Continuous rate scalability provides the capability of selecting arbitrarily the data rate within a scalable range. It is very flexible to allow the video server to fully use the available network bandwidth and adjust dynamically the data rate of the video.

One of the specific coding strategies known as embedded rate scalable coding is well suited for continuous rate scalable applications. In embedded coding, all compressed data are embedded in a single bitstream. This single bitstream can be decoded at different data rates. The decoder receives the compressed data from the beginning of the bitstream up to a point where a chosen data rate requirement is achieved. A decoded image at this data rate can then be reconstructed and visual quality corresponding to this data rate can be obtained. Thus, to achieve the best performance, the bits that convey the most important information need to be embedded at the beginning of the compressed bitstream.

With this advance of video compression[50], networking technologies and international standards, video conferencing is widely used in our daily life[19,20,22,23,30,31,44,51,53,54]. In fact, more and more video conferencing products are appearing on the market.

Figure 3.19 shows a typical scenario in which a video conferencing is held over a practical wide-area network such as the Public Switch Telephone Network (PSTN) or the

Integrated Service Digital Network (ISDN) in which the channel bandwidth is constant and symmetrical. Assume that encoded video bitstream of each conference participant is *R* kb/s in a quarter common intermediate format (QCIF: 176 × 144 pixels). The MCU receives and decodes the multiple video bitstreams from all the conference participants[54-55]. The decoded videos are combined in a common intermediate format (CIF: 352 × 288 pixels) through a video combiner. The combined video is re-encoded at *R* kb/s in order to fulfill the requirement of the channel bandwidth. Therefore video transcoding must be performed at the video combiner. Transcoding is regarded as a process of converting a previously compressed video bitstream into a lower bitrate compressed video bitstream.

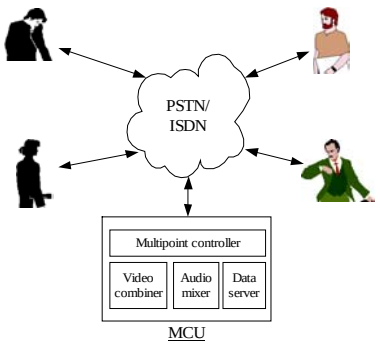


Figure 3.19. An example of multipoint video conferencing.

Transcoding is a practical tool for video combining in multipoint video conferencing over a symmetrical wide-area network. However, the computational complexity is inevitably increased since an individual video bitstream needs to be decoded and the combined video signal needs to be encoded. More importantly, additional degradation is introduced since the video quality of the transcoding approach suffers from this intrinsic double-encoding process. In order to provide a satisfactory visual quality of the combined and transcoded video, the process to re-distribute bits in the re-encoding process to different parts of the combined video is a crucial step. Some

of the widely used transcoders for video conferencing application adopt requantization of DCT coefficients or frame skipping [1-3,6,9,13,17,18,21-23] to reduce the bandwidth of inactive speakers. The incoming video bitstream is at first fully decoded in the pixel domain, and the decoded video frame is re-encoded at the desired output bitrate according to the capability of the clients' devices and the available bandwidth of the network for storage or transmission. This process involves high computational complexity, large memory size, and long delay. Recently, some fast algorithms have been proposed by making use of the information from the incoming bitstream [44,53] to reduce computational complexity. For example, motion vectors extracted from the incoming bitstream after decoding can be used to reduce significantly the complexity of the transcoding, since motion re-estimation [36-39,60] can be avoided. Note that the video quality of the pixel-domain transcoding approach suffers from its intrinsic double-encoding process, which introduces additional degradation. Motivated by this, we make use of the progressive properties of wavelet transform and the region of interest (ROI) features to build a video conferencing system in this section.

Using successive quantization of the wavelet coefficients, scalable video coding in both ROI and background can easily be obtained. Considering the unequal importance of various regions, higher decomposition levels of wavelet transform and finer quantization levels are not required in background region. Besides, difference kernels can be used to compromise the computational complexity and video quality. Then the video combiner calculates the motion activity of each conference participant and adjusts the video quality by discarding bitstream containing fine details information for the inactive conference participants or background information according to the bandwidth

requirement. Since no re-encoding process is required in the bit reallocation process, computational complexity can be greatly reduced in the video combiner. More importantly, video quality degradation can be avoided. In order to improve the quality of the transcoded video, a successive quantizations and a fast frame skipping of wavelet coefficients in the wavelet domain are proposed in this section. The novelty of this algorithm is that the designed wavelet coder can perform both SNR scalablity and temporal scalablity using direct addition of wavelet coefficients in the wavelet domain. These properties allow the video combiner to adjust the video quality more effectively. Due to the fact that full re-encoding process is not required, both computational complexity and video quality degradation can be reduced significantly as compared with the video conferencing system using the conventional transcoding approach.

3.3.2 The Proposed video Conferencing system

Figure 3.20 shows the system architecture of the wavelet-based video coder in multipoint video conferencing system. Since the video conferencing system is wavelet-based, blocking artifacts are avoided. The wavelet-based video conferencing system has four major features:

1. selecting region of interest and using adaptive bit allocation for the foreground (region of interest) and the background in the video coder,
2. progressive transmission updating,
3. fast frame skipping in wavelet domain, and
4. using adaptive bit reallocation algorithm in the video combiner.

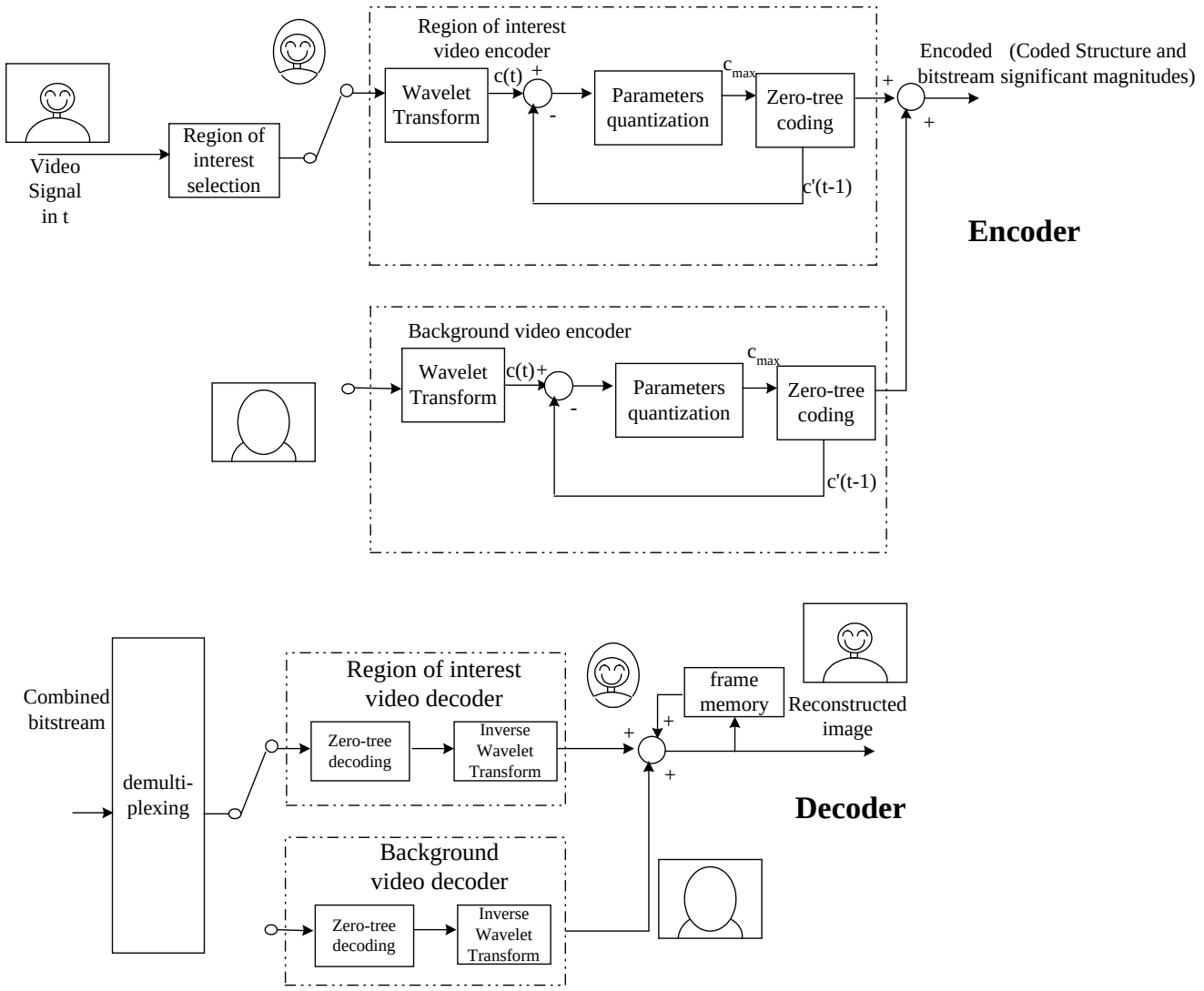


Figure 3.20. The system architecture for the proposed video coder in multipoint video conferencing.

In this section, we will focus on the first two major features while the transcoding techniques in the video combiner will be discussed in the following sections.

The purpose of region selection is to identify the region of interest in an image, e.g., the speaker's face in video conferencing. This region is updated automatically by tracking the object's motion using histogram information[51]. The wavelet-based coder is then applied separately to the foreground and the background. From Figure 3.20, a wavelet transform corresponds to two sets of analysis/synthesis digital filters $g/\hat{g}$ and

h/$\hat{h}$, where g is a high pass filter and h is a low pass filter. By using filters g and h, an image can be decomposed into four bands as shown in Figure 3.21. Subsampling is used to translate the subbands into a base band image. This is the first level of the wavelet transform. The procedure can be repeated on the low-low (LL) band. Thus, a typical 2-D discrete wavelet transform used in image processing will generate a hierarchical pyramidal structure as shown in Figure 3.22. The inverse wavelet transform is obtained by reversing the transform process, replacing the analysis filters with the synthesis filters and using upsampling as shown in Figure 3.23. The wavelet transform can decorrelate the image pixel values, and result in frequency and spatial orientation separation. The transform coefficients in each band exhibit unique statistical properties that can be used for encoding the image. Because the size of the region of interest is small, the computation time can be reduced significantly. Adaptive bit allocation is then performed in the structure coding stage. It makes sure that the video quality of the foreground is always better than that of the background. This is particularly important for unstable networks or low bit rate applications. In the process of structure coding, significant wavelet coefficients are calculated by

$$S^{x}_{k(i,j)}(t) = \text{rounding}\ \{\ c^{x}_{k(i,j)}(t) / \ c^{x}_{k\max}(t)\ \} \tag{3.20}$$

where $S^{x}_{k(i,j)}(t)$ =significant wavelets coefficients

$c^{x}_{k(i,j)}(t)$ =wavelet coefficients

$c^{x}_{k\max}(t)$ =maximum magnitude of the wavelet coefficients

i=horizontal direction

j=vertical direction

k=number of iterations at time t

and x=l, h, v or d represents the image after wavelet transform in low, horizontal, vertical or diagonal filter bank, respectively.

$S^{x}_{k(i,j)}(t)$ is then passed into the zero-tree coder. Note that $S^{x}_{k(i,j)}(t)$ becomes zero or one after the rounding process.

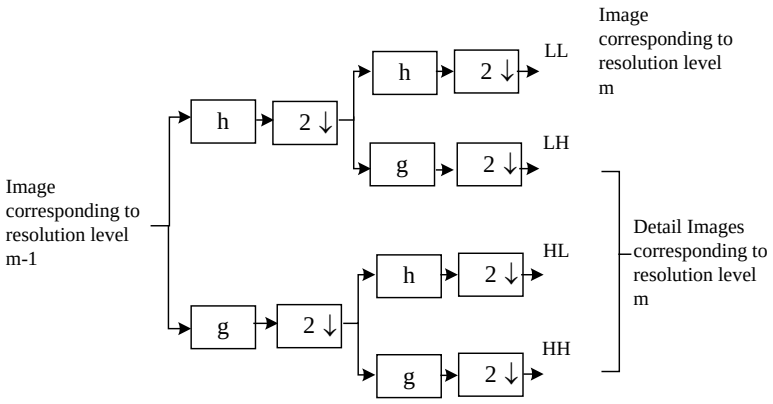


Figure 3.21. The first level of the wavelet transform.

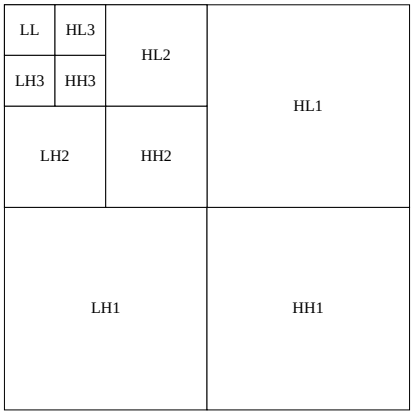


Figure 3.22. A hierarchical pyramidal structure

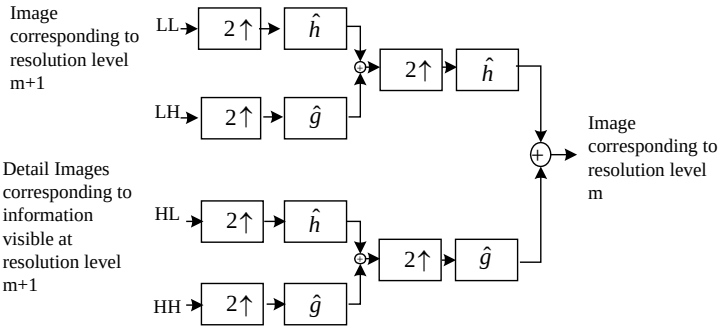


Figure 3.23. Inverse wavelet transform

Initially, a coarse quantization level is applied to the wavelet coefficients in both ROI and background. In embedded coding, a key issue is to embed more important information at the beginning of the bitstream. From the rate-distortion point of view, one wants to quantize the wavelet coefficients that cause a larger distortion in the decompressed image

initially. Let the wavelet transform be $c^{x}_{k(i,j)}(t)$=T(p), where p is the image pixel and $c^{x}_{k(i,j)}(t)$ is the wavelet transform coefficient at time t. The reconstructed image $\hat{p}$ is obtained by performing the inverse transform, $\hat{p}$ =T^{-1}($c^{x}_{k(i,j)}(t)$'), where $c^{x}_{k(i,j)}(t)$' is the quantized transform coefficient. The largest reduction in distortion can be achieved if transform coefficients with the largest magnitudes are quantized and encoded without distortion. Furthermore, this strategically distributes the bits in such a way that the decoded image will look "natural"; progressive refinement or bit-plane coding is used. Hence, in the coding procedure, multiple passes through the data are made. Since $c^{x}_{k\max}(t)$ is the largest magnitude at time t. In the first pass, those transform coefficients with a magnitudes greater than (1/2) $c^{x}_{1\max}(t)$ are considered significant and are quantized to a value of (3/4) $c^{x}_{1\max}(t)$. The rest of the data are quantized to 0. In the second pass, coefficients that have been quantized to 0 but have magnitudes in between (1/4) $c^{x}_{1\max}(t)$ and (1/2) $c^{x}_{1\max}(t)$ are considered to be significant and are quantized to (3/8) $c^{x}_{1\max}(t)$. Again the rest are quantized to zero. Also, those significant coefficients in the last pass are refined to one more level of precision, i.e. (5/8) $c^{x}_{1\max}(t)$ or (7/8) $c^{x}_{1\max}(t)$. This process can be repeated until the data rate meets the requirement or the quantization step size is sufficiently small. Thus, we can achieve the largest reduction in distortion with the smallest number of bits, while the coded information is distributed across the image.

It has been observed experimentally that coefficients which are qunatized to zero at a certain pass have structural similarity across the wavelet subbands with the same spatial orientation. Thus spatial orientation trees (SOT's) can be use to quantize large

areas of insignificant coefficients efficiently. The EZW algorithm and the SPHIT algorithm use slightly different SOT's as shown in Figure 3.24. The major difference between these two algorithms lies in the fact that they use different strategies to scan the transformed pixels.

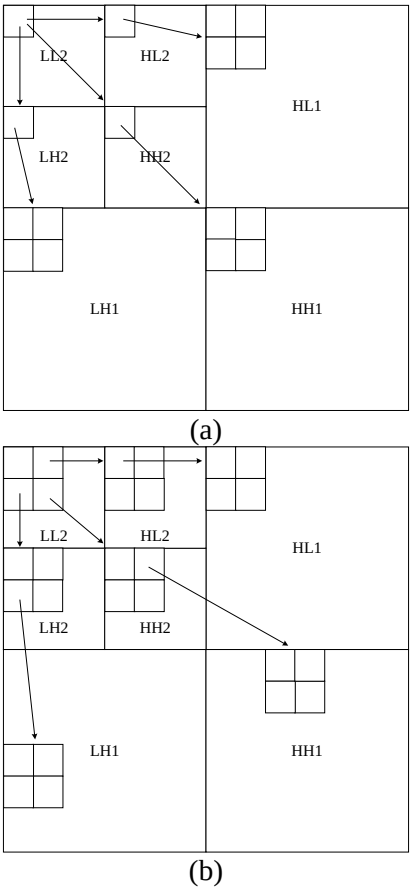


Figure 3.24. (a) The EZW algorithm and (b) the SPHIT algorithm.

In the first quantization, a base level video quality can be obtained as shown in Figure 3.25a. This coarse quantization level favours low bitrate video conferencing applications. This coarse quantization parameter is selected from equation(1). In order to enhance the video quality, a second level quantization level is applied to the residue of the quantized wavelet coefficients. Combining these two levels of quantized coefficients, an enhanced level video quality can be obtained as shown in Figure 3.25b. A final quantization level is applied to the residue of the quantized wavelet coefficients if the

bandwidth is available. Combining these three levels of quantized wavelet coefficients, a high video quality can be achieved as shown in Figure 3.25c. Due to the high correlation of the successive frames, these quantization parameters can be fixed for every group of picture to reduce the computational complexity in the encoding process and the transcoding process. Figure 3.20 gives the architecture of our proposed video coder using the wavelet transform. Since the image at time t and t-1 are usually highly correlated, it is beneficial to make use of this correlation. Initially, the input signal passes into the video encoder featured with the region of interest and the background encoder at time t. According to the region of interest selected by the user, a simple tracking will be applied using histogram or chrominance information. The difference between the encoded image at time t-1 in the wavelet domain and the input signal in the wavelet domain at time t is fed into the encoder as an input signal in order to speed up the encoding process as compared in [51]. After the encoding process, both video streams will be combined and transmitted to the client side for decoding. Note that the encoded image at time t-1 can be reconstructed in the front encoder in the wavelet domain. This arrangement is used to improve the efficiency of video coding and video quality and to facilitate progressive transmission process.

Since the difference between the current frame and the previous reconstructed frame is small in both background and slow motion regions, less bits are required to encode these regions in a practical situation. In practice, only 30% of the targeted bits are required for background video coding, the video quality still can be improved progressively. Besides, as compared with the traditional video transcoder, no re-encoding process is required. Hence, high computational complexity and quality

degradation can be avoided in the transcoding process. In the following discussion, we will focus on scalable video transcoder using successive quantization to perform frame skipping in the wavelet domain.

(3.25a)

(3.25b)

(3.25c)

Figure 3.25. Different levels of video qualities. (3.25a) first level, (3.25b) second level and (3.25c) third level).

3.3.3 Fast frame skipping algorithm in wavelet domain

Temporal scalability of the wavelet-based video can be achieved by our proposed fast frame skipping technique in the wavelet domain. Figure 3.26 shows the architecture of our proposed frame-skipping transcoder. This transcoder performs frame-skipping in the wavelet domain, so that low computational complexity can be achieved. Initially, the number of significant magnitudes is extracted for every incoming frame and will be accumulated until a non-skipped frame is transmitted, according to the following pseudo-code:

$$\hat{L} = L_i + \hat{L} \tag{3.21}$$

where $\hat{L}$ is the accumulated number of incoming significant magnitudes and L_i is the number of incoming significant magnitudes. This setting is used to control the frame-skipping rate. If $\hat{L}$ is larger than a frame-skipping threshold, it implies that the previous non-skipped frame does not have sufficient information to represent the current frame. So, it is necessary to update this frame in the client sides. Hence, switch S1 is closed and $\hat{L}$ is set to zero. Table 3.16 summaries the operating modes of the frame-skipping transcoder.

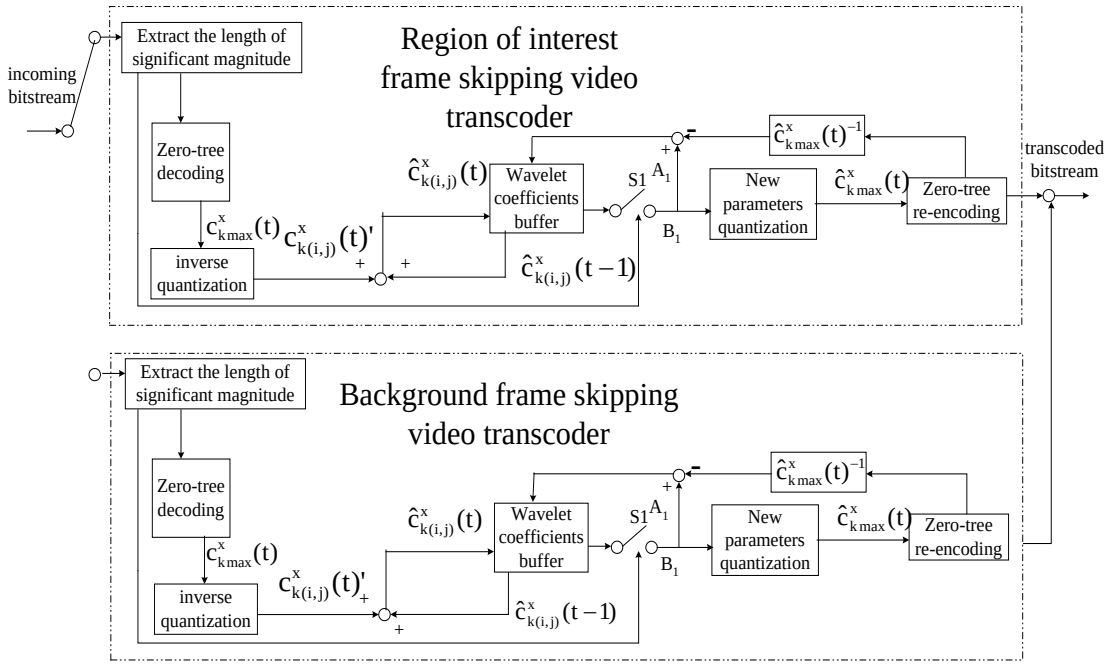


Figure 3.26. The architecture of our proposed frame-skipping transcoder.

Table 3.16. Switch position for different modes of frame skipping.

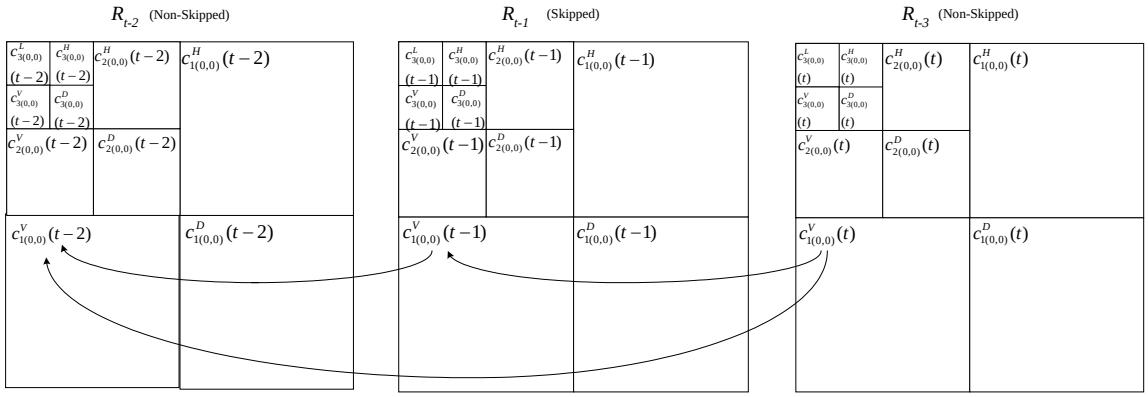
Frame skipping mode	*S1* Position
Skipped frame	A_1
Non-skipped frame	B_1

Then zero-tree decoding will be performed to reconstruct the wavelet coefficients and then $c^{x}_{k\max}(t)$ is obtained. The inverse quantization is applied to reconstruct the quantized wavelet coefficients, $c^{x}_{k(i,j)}(t)'$. The frame skipping process of wavelet coefficients in the wavelet domain can be achieved by employing the proposed direct addition of wavelet coefficients technique as shown in the following equation.

$$\begin{pmatrix} \hat{c}^{x}_{k0,0}(t) & \hat{c}^{x}_{k0,1}(t) & \hat{c}^{x}_{k0,2}(t) & \dots & \hat{c}^{x}_{k0,N-1}(t) \\ \hat{c}^{x}_{k1,0}(t) & \hat{c}^{x}_{k1,1}(t) & \hat{c}^{x}_{k1,2}(t) & \dots & \hat{c}^{x}_{k1,N-1}(t) \\ \hat{c}^{x}_{k2,0}(t) & \hat{c}^{x}_{k2,1}(t) & \hat{c}^{x}_{k2,2}(t) & \dots & \hat{c}^{x}_{k2,N-1}(t) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \hat{c}^{x}_{kN-1,0}(t) & \hat{c}^{x}_{kN-1,1}(t) & \hat{c}^{x}_{kN-1,2}(t) & \dots & \hat{c}^{x}_{kN-1,N-1}(t) \end{pmatrix} = \begin{pmatrix} c^{x}_{k0,0}(t)' & c^{x}_{k0,1}(t)' & c^{x}_{k0,2}(t)' & \dots & c^{x}_{k0,N-1}(t)' \\ c^{x}_{k1,0}(t)' & c^{x}_{k1,1}(t)' & c^{x}_{k1,2}(t)' & \dots & c^{x}_{k1,N-1}(t)' \\ c^{x}_{k2,0}(t)' & c^{x}_{k2,1}(t)' & c^{x}_{k2,2}(t)' & \dots & c^{x}_{k2,N-1}(t)' \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ c^{x}_{kN-1,0}(t)' & c^{x}_{kN-1,1}(t)' & c^{x}_{kN-1,2}(t)' & \dots & c^{x}_{kN-1,N-1}(t)' \end{pmatrix} +$$

$$\begin{pmatrix} c^x_{k0,0}(t-1)' & c^x_{k0,1}(t-1)' & c^x_{k0,2}(t-1)' & \dots & \dots & \dots & \dots & c^x_{k0,N-1}(t-1)' \\ c^x_{k1,0}(t-1)' & c^x_{k1,1}(t-1)' & c^x_{k1,2}(t-1)' & \dots & \dots & \dots & \dots & c^x_{k1,N-1}(t-1)' \\ c^x_{k2,0}(t-1)' & c^x_{k2,1}(t-1)' & c^x_{k2,2}(t-1)' & \dots & \dots & \dots & \dots & c^x_{k2,N-1}(t-1)' \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ c^x_{kN-1,0}(t-1)' & c^x_{kN-1,1}(t-1)' & c^x_{kN-1,2}(t-1)' & . & . & . & . & c^x_{kN-1,N-1}(t-1)' \end{pmatrix} \quad (3.22)$$

where $\hat{c}^x_{k(i,j)}(t)$ represents the newly reconstructed wavelet coefficient in the x-direction and y-direction at time t after frame-skipping process. This new wavelet coefficient can be obtained by direct addition of the wavelet coefficients for all skipped frame as shown in Figure 3.27. Then $\hat{c}^x_{k(i,j)}(t)$ will be passed into the wavelet coefficient buffer and accumulated with the previous skipped wavelet coefficients. The new quantization parameter, $\hat{c}^x_{k\,max}(t)$, will be found from $\hat{c}^x_{k(i,j)}(t)$. After $\hat{c}^x_{k\,max}(t)$ is obtained, the difference between the quantized wavelet coefficient $\hat{c}^x_{k(i,j)}(t)$' and the reconstructed wavelet coefficients $\hat{c}^x_{k(i,j)}(t)$ is fed back into the wavelet coefficient buffer to compensate for the re-encoding error due to requantization. Hence, the problem of accumulation of errors can be resolved. Then the reconstructed wavelet coefficients will undergo zero-tree coding and be transmitted to the decoder sides. Since all operations are performed in the wavelet domain, inverse wavelet transform and forward wavelet transform are not required. Hence, low computational complexity can be achieved.



$$\hat{c}^{x}_{k(i,j)}(t) = c^{x}_{k(i,j)}(t)' + c^{x}_{k(i,j)}(t-1)'$$

Figure 3.27. Direct addition of the wavelet coefficients.

Another advantage of our proposed frame-skipping transcoder is that when multiple frames are dropped, it can be processed in the forward order, thus eliminating multiple wavelet-domain buffers that are needed to store the incoming quantized wavelet coefficients of all dropped frames. Figure 3.28 shows a scenario in which two frames are dropped. When R_{t-2} is dropped, we store the wavelet coefficients into the wavelet-domain buffer. The stored wavelet coefficients will be used to update the new wavelet coefficients for the next dropped frame. This means that when R_{t-1} is dropped, our proposed scheme updates the wavelet coefficients. From Figure 3.28, the wavelet coefficients in the wavelet-domain buffer are added to the corresponding incoming wavelet coefficients of the frame in R_{t-1}. The buffer is then updated with the new wavelet coefficients. By using the proposed scheme, only one wavelet-domain buffer is needed for all dropped frames. The flexibility of multiple frame-skipping provides the fundamental framework for dynamic frame-skipping, which is used in multipoint video conferencing.

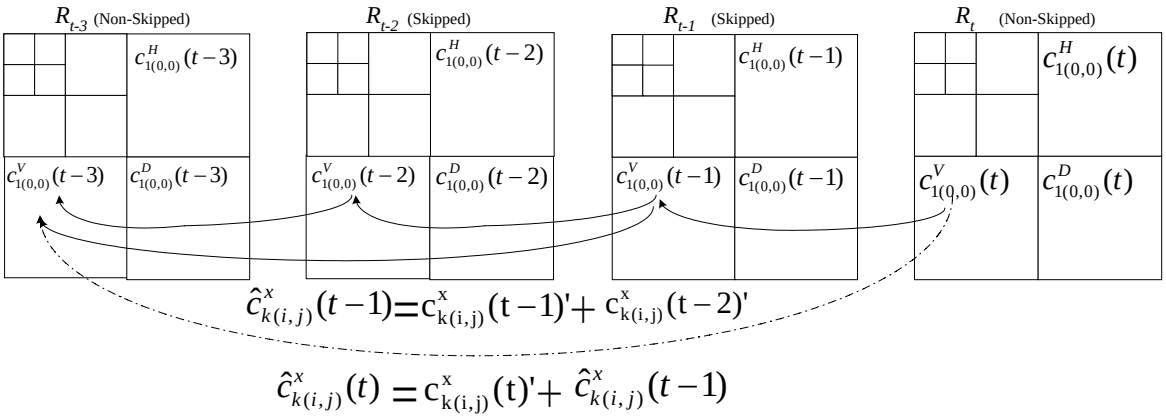


Figure 3.28 Multiple frame skipping of our proposed transcoder.

3.3.4. Adaptive bit reallocation in the video combiner

Based on our proposed frame skipping transcoder, the transcoded video can achieve temporal scalability. In this section, we will discuss our arrangement to achieve SNR scalability. Since the video is scalable as produced by the proposed video coder, no transcoding process is required. In the adaptive bit reallocation process, the video combiner extracts the motion activities of the incoming video bitstreams. The video combiner only needs to calculate the motion activities and the targeted number of bits for each conference participant. According to the targeted number of bits for each conference participant, the video combiner adjusts the level of video quality by discarding the high quality level video bitstream or performing the frame skipping in wavelet domain if necessary.

According to the motion activities and the bandwidth constraints of each conference participant, the video combiner adjusts the scalable video quality levels[31-32] for each conference participant. We propose a video combiner which extracts the motion activities information from the incoming bitstreams as shown in Figure 3.29.

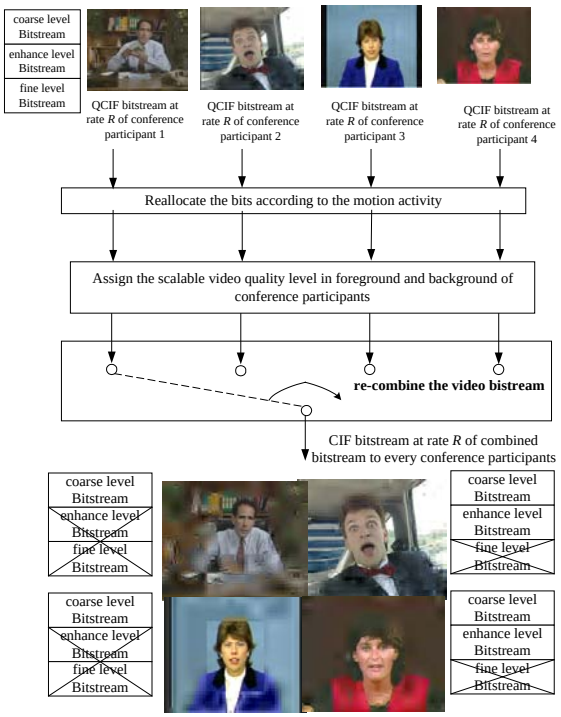


Figure 3.29. Our proposed video combiner.

In most multipoint video conferencing applications, usually only one or two conference participants are active at a given time, while other participants have no or little motion activities. To make the best use of the available bit rates, a rate control scheme is used to calculate the targeted number of bits based on the motion inside the region of interest. The motion activity is defined in terms of the number of significant magnitudes and the energy of the audio signal as shown in following formulations:

$$\cos t_{video} = \hat{L} \tag{3.23}$$

and

$$If \mid Audioenergylevel \mid > audio\_background\_energy\_threshold$$

$$cost_{audio} = \lambda \tag{3.24}$$

$$cost_i = cost_{audio} + cost_{video} \tag{3.25}$$

where $\hat{L}$ is the accumulated number of incoming significant magnitudes, $cost_{video}$ and $cost_{audio}$ represent the importance of the video and audio of user i, respectively. And

$cost_i$ represents the importance of user i and λ is a constant. The video combiner evaluates the percentage of bits for every conference participant by normalizing the speaker activities as described in equation (3.26).

$$\%\_of\_Bits\_assigned\_to\_CP_i = \frac{cost_i}{\sum_{i=1}^{N} cost_i} x100\% \tag{3.26}$$

After the percentage of bits for every speaker is calculated, the video combiner adjusts the video quality level or performs frame skipping according to the bandwidth constraint for each conference participant. Consequently, more bits will be allocated to those sub-sequences with higher motion activities. Since the incoming video is scalable, our proposed video combiner can achieve a low complexity by discarding the fine level or the second level quantized coefficients or performing frame skipping according to the bandwidth requirement and the activities of users. A detailed discussion of the performance of our techniques is given in section 3.3.6.

3.3.5 Analysis of computational complexity

The wavelet based video coder is composed of two major components. For the wavelet transform part, a sophisticated analysis is required in order to select the most suitable wavelet kernal for video conferencing application. In this section, three common kernels including the Harr, B93 and B97 are considered. The major advantage of using the Harr kernal is its lowest computational complexity as compared with other kernals. However, the quality performance is another important factor since this kernal introduces some obvious visual artifacts. For image compression, B93 and B97 are widely used because of its good reconstruction quality. However, the computational complexity is much higher than the Harr Kernal. Hence, in this section, we have made an analysis on

combination kernels such as Harr and B93. For the decomposition, the Harr kernal will initially be used and, for higher decomposition level, B93 filter will be used. Figure 3.30 shows the average PSNR using different numbers of iterations for three wavelet kernals to encode a typical video conferencing sequence- the salesman. Note that in order to achieve higher video quality, three iterations have been used. Usually three iterations are not sufficient to achieve a good image quality. However, the video quality can be upgrade due to the recursive properties of our designed video coder. Besides the video quality, programming speed is also important for real time applications. The computational complexities for the three wavelet kernels using different numbers of iterations are shown in Figure 3.31. From these two figures, we can conclude that by using filters of different kernels, both high PSNR and high frame rate(25/sec) can be achieved for typical video conferencing applications. Hence in the following discussion, we give our experimental results on practical video conferencing applications.

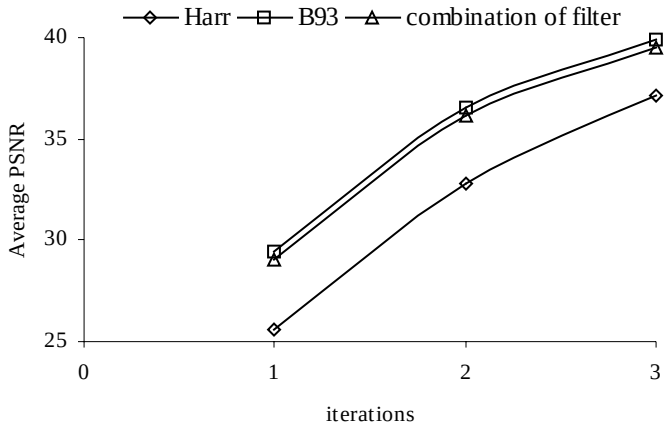


Figure 3.30 Average PSNR using different numbers of iterations for wavelet kernals

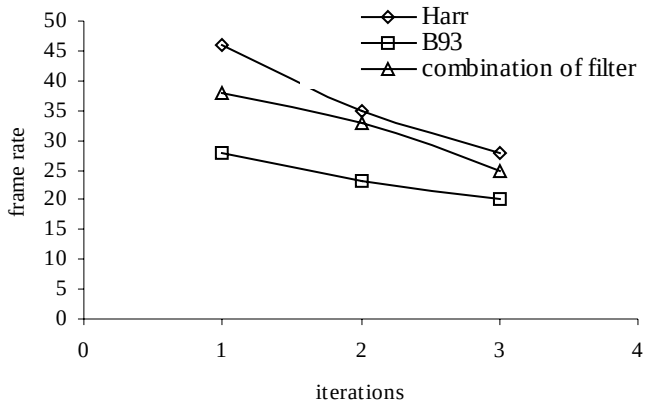


Figure 3.31 Computational complexities of the three wavelet kernals using different iterations

3.3.6 Experimental Results

A series of computer simulations were conducted to evaluate the overall efficiency of the proposed frame-skipping transcoder. The performance of the proposed video combiner for multipoint continuous presence video conferencing is also presented below.

A. Performance of the Frame-Skipping Transcoder

To evaluate the overall efficiency of the proposed frame-skipping transcoding approach, all test sequences, in QCIF (176×144) format, were encoded at high bitrates (64kb/s and 128kb/s) using a fixed quantization parameter. For the front encoder, the first frame was coded and the region of interest is defined in the center of the image, and the remaining frames were encoded using the difference frame which is the difference between the current frame and the previous frame as the input signal. These picture-coding modes were preserved during the transcoding. In the following, the peak signal-to-noise ratio (PSNR) of the transcoded sequence was measured against the original sequence.

The first experiment demonstrates the performance of direct addition of wavelet coefficients. In other words, only the temporal scalability is under examined. The PSNR performance of the proposed frame-skipping transcoder for the "Salesman" sequence is shown in Figure 3.32. At the front encoder, the original test sequence "Salesman" was encoded at 128kb/s as shown in Figure 3.32, and then transcoded into 64kb/s at half of the incoming frame rate. As shown in Figure 3.32, the proposed transcoder outperforms the pixel-domain transcoder. The PSNR performance is also close to the incoming video since the re-encoding error is significantly reduced. Note that in the pixel-domain approach, the re-encoding error introduced could be a problem since all macroblocks suffer quantization errors. In Figure 3.32, the performance of our proposed transcoder is very close to the incoming video since the transcoding process is performed in the wavelet domain. Since the motion activity becomes significant after the 7^{th} frame, this results in the quality degradation. Note that re-encoding error is introduced in all macroblocks and is accumulative in the pixel-domain approach. As shown in Table 3.17, the average PSNR performance in the proposed transcoder is significantly better than that of the pixel-domain transcoder. Thus, we can achieve significant computational savings while maintain good video quality. On the other hand, our proposed transcoder also allows SNR scalability.

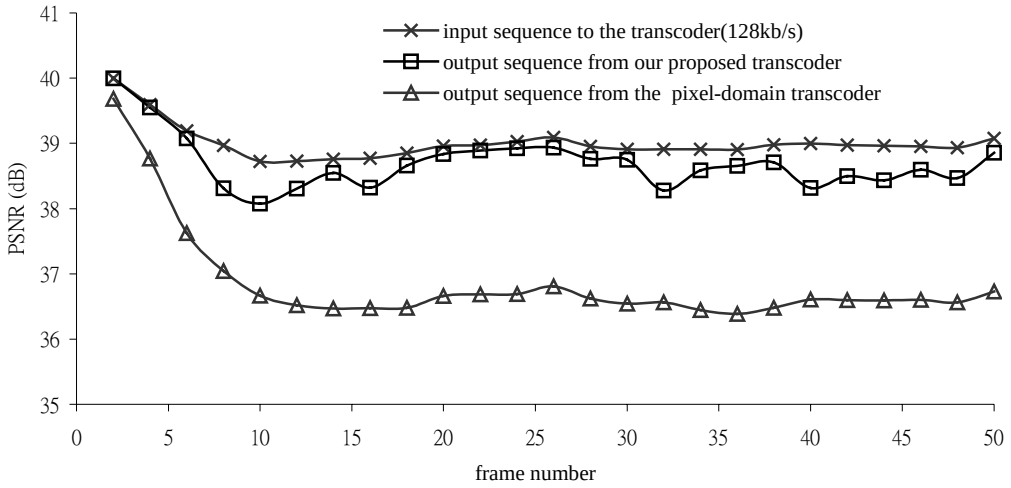


Figure 3.32. Performance of the proposed transcoder. The frame-rate of incoming bitstream is 30 frames/s and it is then transcoded to 15 frames/s.

Table 3.17. Performance of the proposed transcoder. The frame-rate of incoming bitstream is 30 frames/s and it is then transcoded to 15 frames/s.

Sequences	Input bitrate	Pixel-domain transcoder			Our proposed transcoder		
		MC region	Non-MC region	All region	Region of interest	background	All region
Salesman	64k	30.25	34.31	33.77	32.25	36.02	35.34
	128k	33.58	37.14	36.85	34.84	38.92	38.42
News	64k	30.44	34.66	34.03	32.13	36.14	35.36
	128k	34.07	37.47	37.13	35.45	39.29	38.57
Hall	64k	36.14	37.07	36.93	37.06	37.59	37.36
	128k	38.3	39.43	38.94	38.76	39.93	39.47

All these advantages combined give rise to significant computational saving as well as quality improvement. These demonstrate the effectiveness of the proposed frame-skipping transcoder. The simulation results of three video test sequences are summarized in Table 3.17.

In order to illustrate the effects of the proposed frame-skipping transcoder with multiple frame dropping, Table 3.18 set forth the results of the frame-skipping transcoding for which the frames are temporally dropped by a factor of *2*. The results are

similar to the above three video test sequences. It is clear that the pixel-domain transcoder gives the worst performance, and our proposed transcoder provides a significant improvement. Also the computational complexity is reduced remarkably.

Table 3.18 Performance of the proposed transcoder. The frame-rate of incoming bitstreams is 30 frames/s and the bitstreams are subsequently transcoded to 10 frames/s.

Sequences	Input bitrate	pixel-domain transcoder			Our proposed transcoder		
		MC region	Non-MC region	All region	Region of interest	background	All region
Salesman	64k	29.85	34.12	33.45	32.09	35.86	35.10
	128k	33.59	37.03	36.70	34.66	38.85	38.34
News	64k	30.04	34.49	33.69	31.94	36.03	35.23
	128k	34.11	37.34	36.93	35.22	39.16	38.42
Hall	64k	34.2	36.37	35.96	36.84	37.45	37.21
	128k	38.2	38.81	38.52	38.51	39.82	39.31

B. Performance of Continuous Presence Video Conferencing System

Let us report the results of a four-point video conferencing session. The video of each conferee was encoded into the QCIF format at 128kb/s, as shown in Figure 3.33. Four video sequences were then transcoded and combined into a CIF format. We then selected segments of the combined sequence to form a 400-frame video sequence in which the first person was most active in the first 100 frames, the second person was most active in the second 100 frames, and so on. For the front encoder, the first frame was coded and the region of interest is defined in the center of the image, and the remaining frames were encoded using the difference frame which is the difference between the current frame and the previous frame as the input signal.

Figure 3.33. Encoded frame 194 of the four conferee's videos, which are received by the MCU.

The motion activities for the four conference participants and the combined video sequence are shown in Figure 3.34. In this figure, the top four curves correspond to four participants in the upper left, upper right, lower left and lower right corners, respectively. The bottom curve represents the motion activity of the combined video sequence. Figure 3.34 shows that although there were short periods of time when multiple participants were active, only one participant was active during most of the time, while other participants were relatively inactive. The overall motion activity of the combined video sequence is relatively random. This indicates that dynamic allocation of the encoding frames to each participant is suitable for the multipoint video conferencing environment.

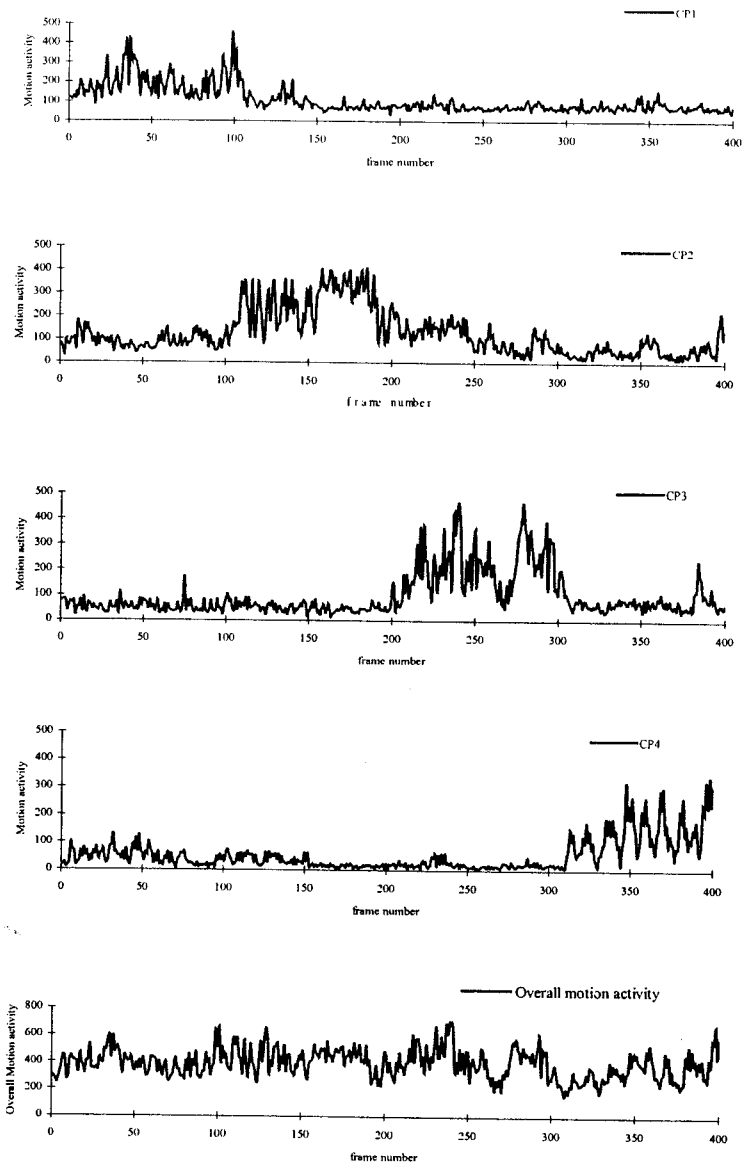


Figure 3.34. Motion activity of a multipoint videoconference.

In the following discussion, we will analyze the performance of the proposed video combiner for continuous presence multipoint video conferencing system as compared to the pixel-domain combiner with dynamic frame-skipping (PDCOMB-DFS) [21]. Figure 3.35 shows the PSNR performance of the conference participants for different video combining approaches. For example, the participant is most active from

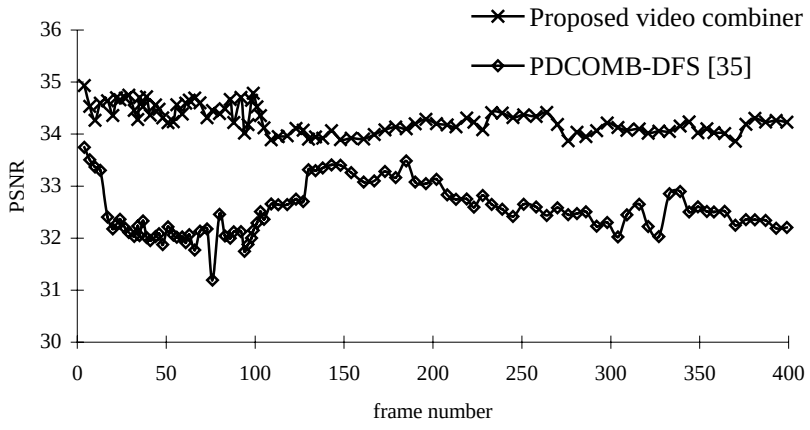
frame 0 to frame 100 in Figure 3.35(a) (as shown in the motion activity plot in Figure 3.34. This active period is transcoded more frequently following the motion activities of the sub-sequence, therefore the videos displayed on the receiver are smoother. It can be seen from Figure 3.35 that the PDCOMB-DFS approach is inferior due to the re-encoding process while the proposed video combiner offers a much better quality as compared with the PDCOMB-DFS. The gain can be as high as 2.6 dB for the active periods due to the efficiency of our proposed frame-skipping transcoder. A frame (the 194^{th} frame) in the combined sequence is shown in Figure 3.36. It can be seen that the video quality of the active participant (at the upper right corner) with the proposed video combiner is much better than that of the PDCOMB-DFS. In Figure 3.36(d), we observe from frames 300 to frame 400 that, the PSNR of the PDCOMB-DFS drops significantly. This is because each non-skipped frame is used as a reference frame of the following non-skipped frame; quality degradation propagates to later frames in a cumulative manner. However, our proposed video combiner suffers less error accumulation as compared to the PDCOMB-DFS since the proposed direct addition of wavelet coefficients can be applied in the wavelet domain and requantization errors is feed back to the wavelet buffer to avoid an accumulation of the re-encoding errors. Furthermore, due to the SNR scalability of the video, the video quality of the participants can be adjusted easily. Comparing with the PDCOMB-DFS, quality degradation is very visible and this argument is also supported by Figures 3.33 and 3.36. Table 3.19 shows the PSNR of each participant of all 400 frames of the video sequence at 128 kb/s using different video combiners. The diagonal values indicate more active motion activities in different time slots of individual conference participants. The table shows that, by using the proposed video combiner, the

PSNR's of all conference participants are greatly improved as compared to the PDCOMB-DFS during both the active and non-active periods. In a practical multipoint video conferencing system, active participants should be given most attention, and the video quality of these active participants is particularly important. The proposed video combiner can achieve this goal significantly.

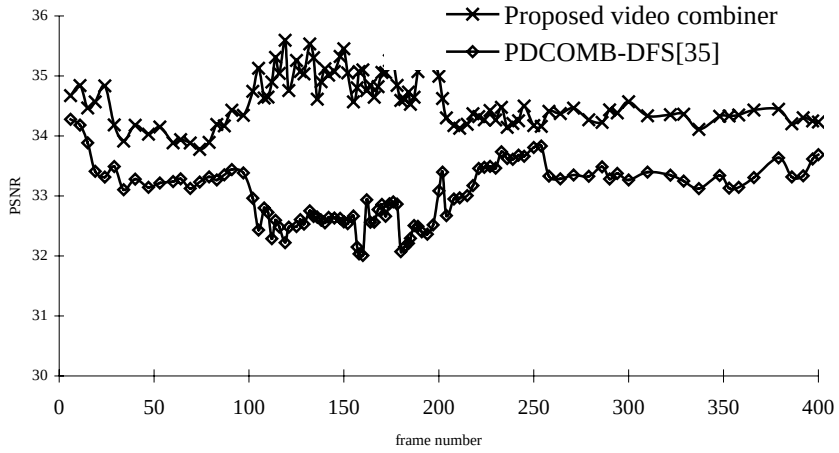
Table 3.19. Average PSNR's of the combined video sequence.

	Frame 1-100		Frame 101-200		Frame 201-300		Frame 301-400	
	A	B	A	B	A	B	A	B
1st conference participant (most active during frame 1 –100)	**32.21**	**34.51**	32.99	34.06	32.60	34.21	32.41	34.11
2nd conference participant (most active during frame 101 –200)	33.42	34.23	**32.55**	**34.99**	33.39	34.32	33.36	34.31
3rd conference participant (most active during frame 201 –300)	34.11	34.84	34.31	34.85	**33.32**	**35.71**	33.85	34.73
4th conference participant (most active during frame 301 –400)	33.08	34.35	32.84	34.24	32.91	33.72	**31.66**	**34.30**

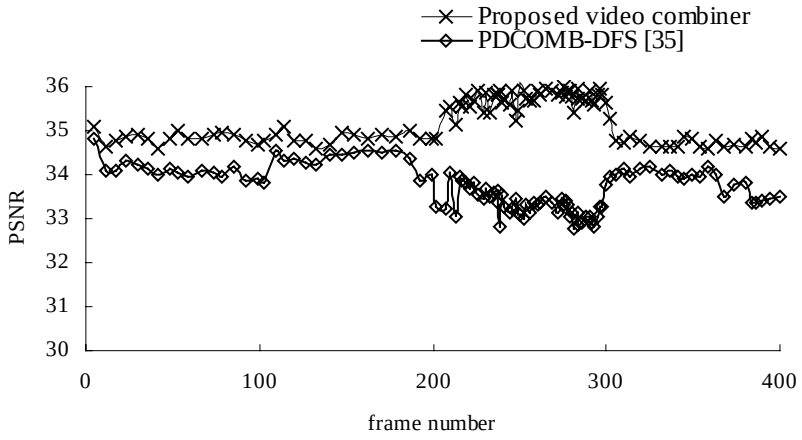
A - PDCOMB-DFS [35].
B - Proposed video combiner.



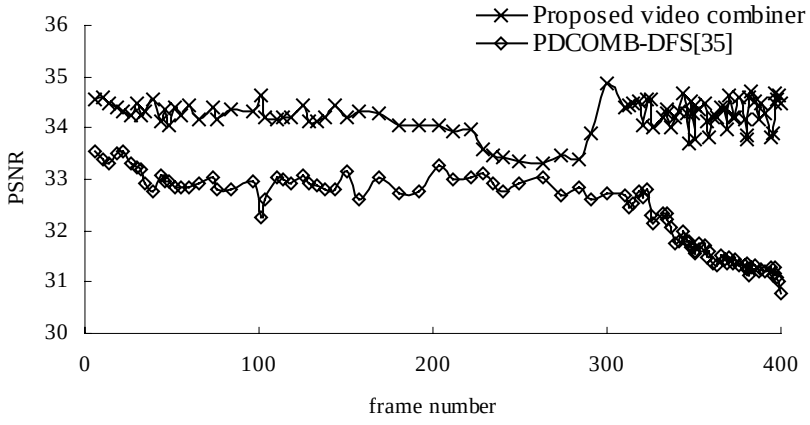
(a)



(b)



(c)



(d)

Figure 3.35. PSNR performance of a conference participant who is most active (a) between frame 0 and frame 100, (b) between frame 101 and frame 200, (c) between frame 201 and frame 300, (d) between frame 301 and frame 400.

(a)

(b)

Figure 3.36. Frame 194 of the combined video sequence using (a) PDCOMB-DFS [35] (b) our video combiner using the proposed frame-skipping transcoder. The active conference participant is at the upper right corner.

3.3.7 Conclusions

This section proposes a low-complexity and high quality frame-skipping transcoder. Its low complexity is achieved by using: 1) a direct addition of the wavelet coefficients in wavelet domain, and 2) an adaptive bit re-allocation strategy in the video

combiner to adjust the quality of the scalable video. We have also shown that a direct addition of the wavelet coefficients can reduce re-encoding errors significantly. Furthermore, our proposed frame-skipping transcoder can be processed in an novel arrangement when multiple frames are dropped. Thus, only one wavelet-domain buffer is needed to store the updated wavelet coefficients of all dropped frames. Overall, the proposed frame-skipping transcoder produces a better picture quality than the pixel-domain approach at the same reduced bitrates.

We have also integrated our proposed frame-skipping transcoder into a new video combining architecture for continuous presence multipoint video conferencing. In multipoint video conferencing, usually only one or two participants are active at any given time. When the frame skipping transcoding approach is used, the frame rate of coding a sub-sequence needed to achieve a certain quality level depends very much on its motion activity using the frame-skipping transcoding approach. We can achieve a better video quality by combining the frame skipping technique and bit reallocation strategy in scalable videos. Since re-encoding is minimized in our frame-skipping transcoder, the proposed architecture provides a better performance than a conventional video combiner in terms of quality and complexity.

Chapter 4

Video Downscaling transcoding

4.1 DCT-based video downscaling transcoder using split and merge technique

4.1.1 Introduction

In this chapter, a new architecture to obtain resampled DCT coefficients in the DCT domain by using the split and merge technique is introduced. Using our proposed video transcoder architecture, a macroblock is splitted into two regions: dominant region and the boundary region. The dominant region of the macroblock can be transcoded in the DCT domain with low computational complexity and re-encoding error can be avoided. By transcoding the boundary region adaptively, low computational complexity can also be achieved. More importantly, the re-encoding error introduced in the boundary region can be controlled more dynamically. A fast algorithm is also proposed to further speed up the transcoding process.

4.1.2 Low Complexity and High quality Video Downscaling for Transcoding in the DCT Domain using Split and Merge technique

In this section, we present a new DCT-based video downscaling transcoding architecture. The new architecture has the following main features:

1 transcoding the overlapping regions of MC macroblocks in the DCT domain using the split and merge technique,

2 adaptively transcoding the non-overlapping regions of MC boundary macroblocks,

3 reconstructing the new prediction errors with the architecture using the adaptive re-encoding error control, and

4 Fast DCT-based transcoding of MC macroblocks using significant coefficients.

Architecture of the proposed transcoder is shown in Figure 4.1. The input bitstream is firstly parsed with a variable-length decoder to extract the header information, coding mode, motion vectors and quantized DCT coefficients for each macroblock. Note that each macroblock is manipulated independently. Switch, SW_1 is employed to pass the reconstructed and quantized DCT coefficients to the DCT-domain downsampling operator for the transformed and quantized residual signal. The selection depends on the coding mode originally used in the front encoder for the current macroblock being processed. The switch positions for different coding modes are shown in Table 4.1. For non-MC macroblocks or the well-align case (e.g. all motion vectors have the same magnitudes and directions), the incoming prediction error in the DCT-domain is directly downsampled in DCT domain. Hence, low computational complexity can be achieved and the quality degradation introduced using the pixel domain approach can be avoided.

When the motion vectors are not well aligned, direct downsampling in DCT domain cannot be achieved since the incoming prediction errors mismatch with the reconstructed new motion vector as shown in Figure 4.2. The major difficulty to transcode these MC macroblocks is that re-encoding errors will be generated due to the re-encoding process of the new DCT coefficients, which introduces quality degradation in the transcoded sequence. Also, high computational complexity is required. Motivated by this, our proposed architecture transcodes the new prediction errors mainly in the DCT

domain by splitting the macroblock into overlapping region and boundary region. By calculating the new motion vector with the minimum distance(MVMD) among the four incoming motion vectors as shown in Figure 4.3, the overlapping region between the incoming DCT coefficients and the target new DCT coefficients can be reused. In other words, full inverse DCT, forward DCT, quantization and requantization are not required in the overlapping region. Therefore, the video quality degradation in the overlapping region can be avoided and low computational complexity can be achieved. For the boundary regions, adaptive DCT, adaptive IDCT, adaptive quantization and adaptive requantization are used to calculate the DCT coefficients. Frame buffer FB2 is proposed to feedback the re-encoding errors introduced in the boundary regions of the macroblocks. Hence, the re-encoding error introduced in the boundary regions of the macroblocks can be controlled more dynamically without introducing any redundant operations. In order to reduce the computational complexity during the MC macroblock transcoding, switch SW_2 is used to further speed up the transcoding process when fast overlapping region transcoding is employed. Table 4.2 shows different overlapping region transcoding approaches of the proposed transcoder. The advantages of the DCT-domain downscaling arrangement, together with the details of other methods, are described in the following subsections.

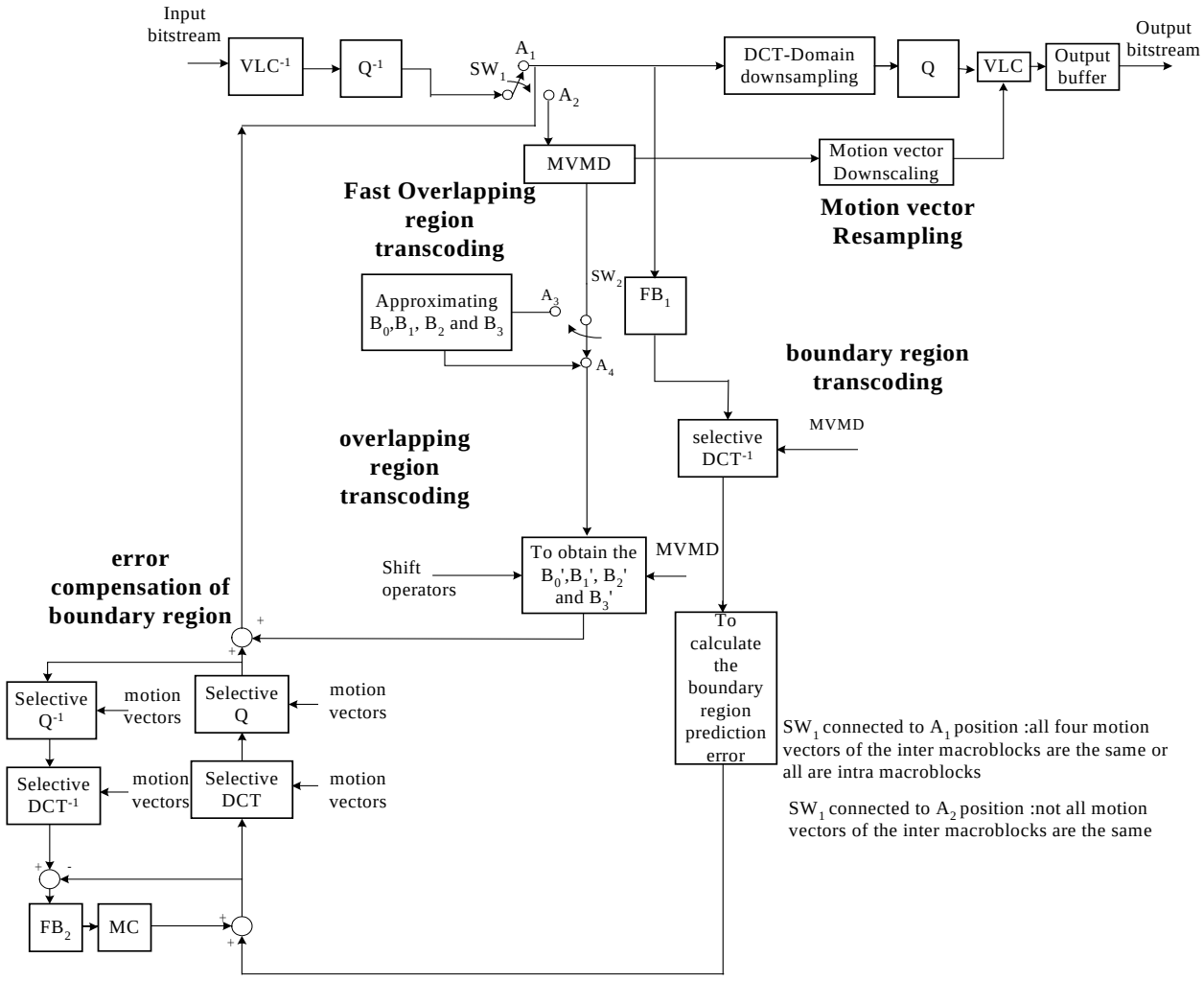


Figure 4.1. Architecture proposed for DCT-based video downscaling video transcoder.

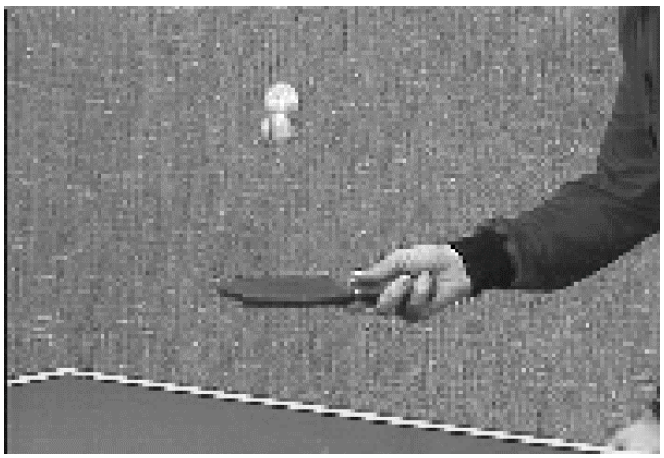


Figure 4.2. Result of mismatching between the incoming DCT coefficients and the new reconstructed motion vector.

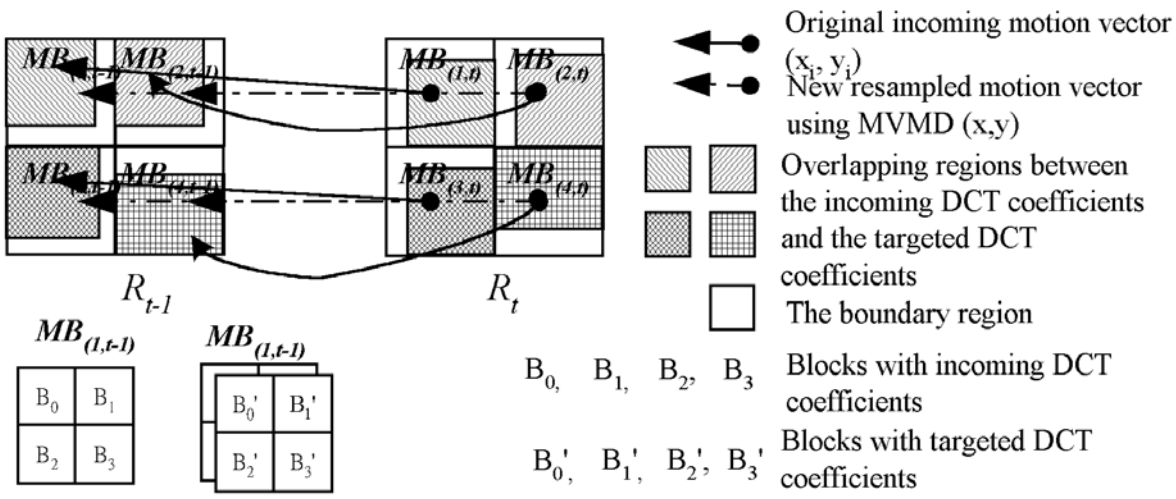


Figure 4.3. Diagram showing the way to avoid video quality degradation in the overlapping region.

Table 4.1. Different coding modes of switches SW_1 of the proposed transcoder.

Coding mode	SW_1 Position
Non MC/ well aligned	A_1
Not well aligned	A_2

Table 4.2. Switch positions for different overlapping region transcoding approaches of the proposed transcoder.

Coding mode	SW_2 Position
Fast Overlapping region transcoding	A_3
use all DCT coefficients	A_4

A Transcoding the overlapping region of MC macroblocks in the DCT domain using the Split and Merge Technique

For MC macroblocks, direct downscaling of the DCT coefficients cannot be employed since there is a mismatch between new resultant motion vector and the incoming DCT coefficients. In other words, the DCT coefficients corresponding to the new resampled motion vector are not available from the incoming bitstream. Figure 4.4 shows the overlapping area of the incoming DCT coefficients. Our objective is to obtain new DCT coefficients in the $MB_{(1,t-1)}$ by using parts of the four segments which come from its four neighboring macroblocks.

In this chapter, we split an MC macroblock in two types of regions: overlapping region and boundary region as shown in Figure 4.4. In the overlapping region, we propose a new minimum distance motion vector and a shift operator to compute the new DCT coefficients. This is to achieve low computational complexity and avoid re-encoding errors.

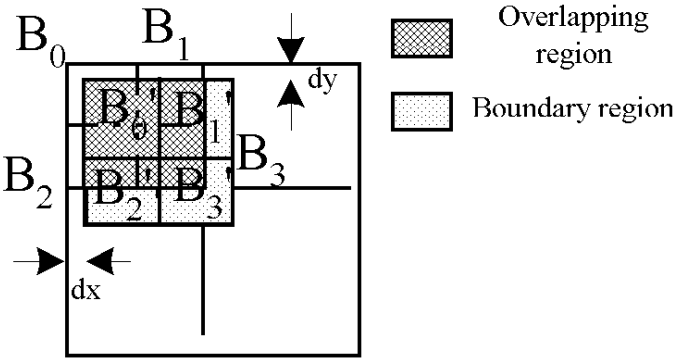


Figure 4.4. The overlapping region and the boundary region of MB_{t-1}

Figure 4.3 shows the scenario that the current macroblock is referring to the macroblock in the previous frame. The prediction errors obtained from the incoming bitstream are B_0, B_1,...B_{15} as shown in Figure 4.5, where B_0, B_1, B_2, and B_3 have the same motion vector whilst other blocks have three different motion vectors. Since our proposed new resampled motion vector before downscaling is different from the incoming motion vectors as shown in Figure 4.3, the new prediction errors B_0', B_1', B_2' and B_3' referencing to the previous frame as shown in Figure 4.5 have to be obtained in order to employ the downscaling of DCT coefficients in the DCT domain. The resampled motion vector can be obtained by minimizing the following cost functions:

$$Cost\_f(x) = (x - x_1)^2 + (x - x_2)^2 + (x - x_3)^2 + (x - x_4)^2 \quad (4.1)$$

$$Cost\_f(y) = (y - y_1)^2 + (y - y_2)^2 + (y - y_3)^2 + (y - y_4)^2 \quad (4.2)$$

where x and y are resampled motion vectors in the horizontal and vertical directions respectively and $(x_1,y_1),\ldots(x_4,y_4)$ are the motion vectors from the incoming bitstream as shown in Figure 4.3. Note that if all incoming motion vectors are the same, i.e. $x_1=x_2=x_3=x_4$ and $y_1=y_2=y_3=y_4$, the new motion vectors become (x_1,y_1). Otherwise, we need to minimize the cost function in both horizontal and vertical directions as shown below:

$$\frac{d}{dx}Cost\_f(x) = -2(x - x_1) - 2(x - x_2) - 2(x - x_3) - 2(x - x_4) \tag{4.3}$$

$$\frac{d}{dy}Cost\_f(y) = -2(y - y_1) - 2(y - y_2) - 2(y - y_3) - 2(y - y_4) \tag{4.4}$$

Let us set the derivatives to zero. We have

$$0 = -2(x - x_1) - 2(x - x_2) - 2(x - x_3) - 2(x - x_4) \tag{4.5}$$

$$0 = -2(y - y_1) - 2(y - y_2) - 2(y - y_3) - 2(y - y_4) \tag{4.6}$$

Hence, we can obtain

$$x = \frac{x_1 + x_2 + x_3 + x_4}{4}, y = \frac{y_1 + y_2 + y_3 + y_4}{4} \tag{4.7}$$

Then new prediction errors, B_0', B_1', B_2' and B_3', can be obtained by using this resampled motion vector *(x,y)* as well as B_0, B_1, B_2, B_3, B_4, B_5, B_8, B_{10} and B_{12} (see also Figure 4.5) to avoid the re-encoding errors in the overlapping region if they can be obtained in the DCT domain directly.

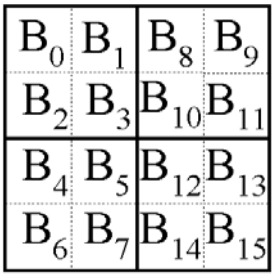


Figure 4.5. Incoming DCT coefficients of four macroblocks.

Since a shifted version of the original motion vector is used, the corresponding prediction errors in the overlapping region can be obtained without performing the full re-encoding process. If the other three motion vectors have the same direction and magnitude (i.e. well-aligned), all new prediction errors (B_0', B_1', B_2' and B_3') can be obtained in the DCT domain. In other words, no re-encoding error will be introduced. Otherwise, a decomposition of the overlapping region and boundary regions are needed.

Using the MC-DCT technique introduced in Chapter 2.6, the DCT of S_{i1} and S_{i2} can be pre-computed and B_i can be extracted from the incoming bitstream, so the amount of re-encoding can be reduced and the computational complexity is simplified. If macroblocks $MB_{(1,t)}$, $MB_{(2,t)}$, $MB_{(3,t)}$ and $MB_{(4,t)}$ have the same motion vector, a direct downsampling of the DCT coefficients can be applied in the DCT domain. Otherwise, a decomposition of the overlapping and boundary regions are needed as shown in Figure 4.4.

Recall the following property of the DCT,

$$DCT(A+B)=DCT(A)+DCT(B) \quad (4.8)$$

where A and B represent pixels in the overlapping and boundary regions with the size equal to 8x8 respectively. We can split blocks B_1', B_2' and B_3' in two regions: overlapping region and boundary region as shown in Figure 4.4.

Using eqn(4.8), we have

$$DCT(A+B)= \sum_{i=0}^{3} DCT(S_{i1})(B_i)DCT(S_{i2}) + DCT(B) \quad (4.9)$$

Due to the difference between the incoming motion vectors and the resampled motion vector, B_4 and B_5 are not required in order to obtain the overlapping region of B_2' (see also Figure 4.5). Similarly, (B_8, B_{10}) and (B_5, B_{10} and B_{12}) are not required for finding the overlapping regions of B_1'and B_3' respectively. Note that the DCT coefficients of B_1', B_2' and B_3' can not be obtained completely since the DCT coefficients as described in eqn.4.9 of the boundary region, B, have not been considered yet. The DCT coefficients of boundary region B have to be obtained separately by using 1-D inverse DCT, motion compensation, forward 1D-DCT and requantization as shown in Figure 4.1. In this process, re-encoding error cannot be avoided due to requantization.

B. Adaptive transcoding the non-overlapping regions of MC boundary macroblocks

In order to transcode the MC boundary, the boundary region is extracted. Selective quantization and selective 1D-DCT of the quantized DCT coefficients of $MB_{(1,t)}$, $MB_{(2,t)}$, $MB_{(3,t)}$ and $MB_{(4,t)}$ have to be performed in the boundary as shown in Figure 4.4. Note that each macroblock composes of four *8×8* blocks in common video coding standards [30,31,34,44], and the DCT and quantization operations are performed on units of *8×8* blocks. When processing $MB_{(1,t)}$, $MB_{(2,t)}$, $MB_{(3,t)}$ and $MB_{(4,t)}$, only their corresponding *8×8* blocks which have pixels overlapping with MB_t boundary are subject to the selective inverse 1D-DCT computation. Hence, part of the inverse 1D-DCT is performed in the boundary, while the motion vector, mv_{t-1}, is needed as an input to the adaptive 1D-DCT module to control which rows or columns the 1D-DCT has to be performed as shown in Figure 4.1. In most cases, this approach is able to reduce

significantly the required number of column or row DCTs as compared with that of the 2D-DCT approach.

Therefore, the new DCT coefficients of the boundary region can be obtained by performing DCT operations on part of the data and putting zero values in the overlapping region. Adaptive quantization is then used to achieve low computational complexity.

C. Reconstruction of the new prediction errors and adaptive re-encoding error control architecture

After obtaining the DCT coefficients of region B, B_1', B_2'and B_3' can be reconstructed by adding DCT(A) and DCT(B) together as shown in eqn.4.9. In Figure 4.4, the newly quantized DCT coefficients of an MC macroblock can then be further processed by downscaling these coefficients in the DCT domain as described in Figure 7. For the well-aligned case, downscaling the incoming DCT coefficients can be performed directly in the DCT domain. Conversely, requantization is required for the formation of new DCT coefficients in the macroblock boundary if not all the motion vectors are the same. This will introduce additional re-encoding errors.

Note that re-encoding errors are introduced in the boundary region only as shown in Figure 4.4. However, these errors will degrade the quality of the reconstructed frame. Since each P-frame is used as a reference frame for the following P-frame, quality degradation will propagate to later frames in a cumulative manner. If the accumulated sum of the re-encoding errors is large, it means that the quality of the transcoded sequence is degraded significantly. These accumulated errors become significant in the sequence containing a large amount of MC macroblocks and high motion activity.

With the possibility of having re-encoding errors in MC macroblocks, it is obviously important to develop techniques to minimize the visual degradation caused by this phenomenon. Thus, a feedback loop is suggested as shown in Figure 4.1 to compensate for the re-encoding errors introduced in the boundary region. The adaptive forward and inverse 1D-DCT and adaptive quantization pairs in the feedback loop are mainly responsible for minimizing re-encoding errors. For these MC macroblocks, the quantized DCT coefficients are inversely quantized and the inverse 1D-DCT is performed adaptively. The re-encoding errors introduced in the boundary region can be obtained by subtracting the original signal from the recovered signal after quantization. This re-encoding error is then stored in FB_2 and fed back to latter frames to avoid the accumulation of re-encoding errors.

Since motion vectors are highly correlated in the successive frames[36-41,45,52], it is observed that the spatial positions of MC macroblocks in certain frames are very close to the spatial positions of MC macroblocks in its subsequent frames. Thus, re-encoding errors stored in FB_2 are added to the prediction errors of MC macroblocks in the following P-frame to compensate for the re-encoding errors. Note that the feedback loop for error compensation cannot ensure the elimination of all re-encoding errors generated by MC macroblock boundary. However, these re-encoding errors are continuously accumulated in FB_2 such that most of them can be compensated for in the subsequent frames if the spatial positions of the MC macroblocks between successive frames are highly correlated.

After the reconstruction of new prediction errors, domain downsampling will be performed in the DCT domain, and the resampled motion vector will also be downscaled

(e.g. half of the original incoming one). After the downscaling process, variable length encoding is applied. Then the output data are stored inside the output buffer for transmission.

D. Fast DCT-based transcoding on MC macroblocks using significant coefficients

Since the energy distributions of DCT blocks obtained from an incoming bitstream mainly concentrate on the low frequency region, it is beneficial to approximate the DCT coefficients using significant DCT coefficients to speed up the transcoding process. The number of significant DCT coefficients can be obtained by using the following equation which defines the energy of DCT coefficients of a block with size NxN,

$$energy = \sum_{l=0}^{N-1}\sum_{m=0}^{N-1} B^2(l,m) \tag{4.10}$$

$$Approx.\,energy = \sum_{l=0}^{j}\sum_{m=0}^{k} B^2(l,m) \tag{4.11}$$

where $B(l,m)$ represents the l^{th} row and m^{th} column of the DCT coefficients. j and k represent the numbers of rows and columns to approximate the original DCT coefficients. Initially, j and k are set to zero. If the approximated energy is less than 0.9 times of the original energy, j or k will be increased by 1 until an approximation of the significant DCT coefficients are obtained. Our experimental work shows that this approach only introduces a video quality drop of about 0.06 to 0.22 dB in the MC macroblock transcoding. However, it can further increase the speed of the DCT-based transcoding about 2.82 to 4.51 times, especially when the incoming video is encoded at a low bit rate.

4.1.3 Experimental Results

Extensive experiments have been performed to evaluate the overall efficiency of various video downscaling transcoders. In the front encoder, the first frame was encoded as an intraframe (I-frame), and the remaining frames were encoded as interframes (P-frames). Picture-coding modes were preserved during transcoding.

These experiments aim at evaluating the performances of the proposed techniques including (i) transcoding the overlapping region of MC macroblocks in the DCT domain using the minimum distance motion vector with unchange video resolution, (ii) adaptive transcoding the non-overlapping region MC macroblocks boundary and (iii) adaptive re-encoding error control architecture when applied to the video downscaling transcoder. The front encoder was employed to encode video sequences with different spatial resolutions and motion characteristics. "Salesman", "Miss_america" and "Hall" in CIF (352×288) containing low motion activities and "Tennis", "Football" and "Flower" in (352x240) containing high motion activities were encoded by an MPEG2 TM5 front encoder [32], but only P-frames were generated. For all testing sequences, the frame-rate of the incoming bitstream was 30 frames/s.

A large number of experimental works have been done to compare the performance of our architecture with conventional pixel-domain transcoders (CPDT) employing align-to-average weighting (AAW)[5], align-to-best weighting (ABW)[5] or adaptive motion vector resampling (AMVR) [5] to resample a downscaled motion vector from the incoming motion vectors of the four macroblocks. Table 4.3 shows the simulation conditions for different transcoders examined. Detailed comparisons of the average PSNR between CPDT+AAW, CPDT+ABW, CPDT+AMVR and our proposed

DCT-based transcoder using motion vector with the minimum distance(DCT+MVMD) are given in Table 4.4. It shows that our proposed DCT-based transcoders outperform CPDT+AAW and CPDT+ABW and CPDT+AMVR in all cases. These results are more significant for sequences with low motion activity because our proposed DCT-based transcoder does not introduce any re-encoding error in the overlapping region since the transcoding is performed in DCT domain. Also, Table 4.5 shows that our proposed transcoders have a speed-up of about 3.52-4.58 times faster than that of the conventional transcoder. This is done without performing full decoding and re-encoding process. Our transcoder transcodes all MC macroblocks mainly in DCT domain. For sequences with low motion activity, such as salesman, miss_America and Hall sequences, the motion vectors are small due to slow motion activity. Therefore, the overlapping region is large in the MC macroblock. Hence, significant improvement can be achieved, which is about 1.56-1.83dB as shown in Table 4.4. For “Table tennis”, “Football” and “Flower” sequences, the average PSNR and speed-up also have significant improvement as compared with the conventional approaches. It is due to the fact that re-encoding process is performed in the pixel domain[5]. Full decoding and re-encoding processes are required for the conventional pixel domain transcoder to transcode the MC macroblocks. Hence, the computational complexity as well as re-encoding errors become significant for transcoding these video sequences. Only a small amount of region of the MC macroblock requires full re-encoding for using our proposed video transcoder, and pixel domain transcoding occurs only in boundary regions. In other words, the transcoding process is performed mostly in the DCT domain hence quality degradation can be

avoided. On the average, about 1.19-1.45dB PSNR improvement and a speed-up of 3.52-3.97 times have been achieved as shown in Table 4.5.

Table 4.3. Simulation conditions.

Approaches / Condition	Proposed DCT-domain transcoder		Conventional pixel-domain transcoders			
	DCT+MVMD	FDCT+MVMD	CPDT+AAW	CPDT+ABW	CPDT+AWW	CPDT+AMVR
Proposed DCT-based transcoding using motion vector with minimum distance (DCT+MVMD)	ON	ON	OFF	OFF	OFF	OFF
Proposed Fast DCT –based transcoding using motion vector with minimum distance with significant coefficients approximation (FDCT+MVMD)	OFF	ON	OFF	OFF	OFF	OFF

Table 4.4. Average PSNR of the proposed transcoder, where the frame rate of the incoming bitstream was 30 frames/s. MPEG2 TMN5 [32] was used as the front encoder for encoding “Salesman”, “Miss_America”, “Hall”, “Tennis”, “Football” and “Flower”.

Sequences	Input bitrate	Average PSNR difference as compared with CPDT+AAW for MC macroblock transcoding.		
		CPDT+ABW	AMVR[5]	DCT+MVMD
Salesman (352x288)	512k	0.06	0.41	1.83
	256k	0.05	0.39	1.78
Miss_America (352x288)	512k	0.09	0.39	1.74
	256k	0.07	0.36	1.71
Hall (352x288)	512k	0.11	0.42	1.62
	256k	0.08	0.38	1.56
Tennis (352x240)	3M	0.12	0.43	1.45
	1.5M	0.08	0.38	1.40
Flower (352x240)	3M	0.18	0.47	1.31
	1.5M	0.15	0.43	1.25
Football (352x240)	3M	0.21	0.50	1.25
	1.5M	0.27	0.56	1.19

Table 4.5. Average PSNR and speed-up ratio of our proposed transcoder as compared with CPDT+AAW using MPEG2 TMN5 [32] as a front encoder.

Sequences	Input bitrate	DCT+MVMD	
		Average PSNR difference as compared with CPDT+AAW	Speed-up ratio as compared with CPDT+AAW
Salesman (352x288)	512k	1.83	4.58
	256k	1.78	4.52
Miss_America (352x288)	512k	1.74	4.47
	256k	1.71	4.42
Hall (352x288)	512k	1.62	4.21
	256k	1.56	4.17
Tennis (352x240)	3M	1.45	3.97
	1.5M	1.40	3.95
Flower (352x240)	3M	1.31	3.63
	1.5M	1.25	3.61
Football (352x240)	3M	1.25	3.54
	1.5M	1.19	3.52

Table 4.6 compares the average PSNR and complexities of our proposed transcoders, DCT+MVMD, and our proposed transcoder using significant DCT coefficients for MC macroblock transcoding in the overlapping region, FDCT+MVMD. As shown in Table 4.6, FDCT+MVMD gives similar performance with the DCT+MVMD and it has a slight quality degradation of about 0.06-0.22dB. Note that the speed-up can further be increased from 2.82 to 4.51 times. Therefore, FDCT+MVMD is more suitable for video downscaling transcoders with low bitrate applications.

We have pointed out, in Table 4.6, that FDCT+MVMD has a slight PSNR degradation over DCT+MVMD. This result is expected since not all DCT coefficients are used in the MC transcoding process. However, the speed-up of transcoding MC macroblocks can be further increased significantly, especially in low bitrate applications.

Since the major contents of the DCT coefficients concentrate mainly on the low frequency part and only a few DCT coefficients are non-zero in low bitrate video coding, the approximation is very close to the actual value and the number of significant DCT coefficients is much less than 64. In the "Salesman", the speed-up performance is about 2.82, since most of the macroblocks are coded in the non-MC mode. For the "Table Tennis", "Football" and "Flower" sequences, the average PSNR degradation using the FDCT+MVMD is about 0.12-0.22dB. However, the speed-up of FDCT+MVMD significantly outperforms DCT+MVMD for all these sequences, which is about 3.89-4.51 times.

Table 4.6. Average PSNR and speed-up of the proposed transcoder using significant DCT coefficients (FDCT+MVMD) as compared with the proposed transcoder DCT+MVMD. MPEG2 TMN5 [32] was used as the front encoder for encoding "Salesman", "Miss_America", "Hall", "Tennis", "Football" and "Flower".

Sequences	Input bitrate	FDCT+MVMD	
		Average PSNR difference as compared with DCT+MVMD	Speed-up ratio as compared with DCT+MVMD
Salesman (352x288)	512k	-0.09	2.82
	256k	-0.06	2.88
Miss_America (352x288)	512k	-0.13	2.97
	256k	-0.10	3.06
Hall (352x288)	512k	-0.16	3.34
	256k	-0.12	3.45
Tennis (352x240)	3M	-0.18	3.89
	1.5M	-0.12	4.03
Flower (352x240)	3M	-0.20	4.21
	1.5M	-0.14	4.36
Football (352x240)	3M	-0.22	4.37
	1.5M	-0.16	4.51

4.1.4 Conclusion

In this section, we have proposed a new architecture for low-complexity and high quality video downscaling transcoder to solve the problem of MC macroblcoks transcoding. Its low complexity is achieved by: 1) re-using the DCT coefficients for macroblocks coded with motion compensation to deactivate most of the complex modules of the transcoder, 2) using an adaptive MC macroblock boundary to help the transcoding, 3) using selective inverse 1D-DCT, 1D-DCT, quantization and inverse quantization for motion-compensated macroblocks in error compensation, and 4) using a fast DCT-based video downscaling transcoding approach for MC macroblocks with significant coefficients to speed up the transcoding process. Furthermore, it is shown that re-encoding errors can be reduced significantly by 1) transcoding the overlapping region of MC macroblocks in the DCT domain using the minimum distance motion vector without changing its video resolution, 2) adaptively transcoding the non-overlapping regions of MC boundary macroblocks and 3) making use of an adaptively re-encoding error control architecture. On the whole, the proposed architecture produces a picture with the quality better than that of the conventional video downscaling transcoder at the same reduced bitrates. Furthermore, low computational complexity can be achieved since only the boundary region is required to make transcoding in the pixel-domain and perform error compensation. In other words, the transcoding process can be performed in the DCT domain for most of the regions, which avoids quality degradation. Experimental results show that our proposed DCT-based video downscaling transcoder has outstanding performance to transcode MC macroblocks of various video sequences.

4.2 Diversity and importance measures for video transcoding

4.2.1 Introduction

In video downscaling, simply reusing the motion vectors extracted from an incoming video bitstream may not result in good quality pictures. Several refinement schemes have been proposed recently to correct their recomposed new motion vectors in order to optimize the coding efficiency during the transcoding process. However, the major concern is that an optimal resampled motion vector may not exist during video downscaling process. In other words, it is difficult to use one motion vector to represent four motion vectors when the diversity of the incoming motion vectors is high. Besides, redundant computation for refinement has also been carried out even if the resampled motion vector is already optimal. In this section, an adaptive motion vector re-composition algorithm using two new measures: the diversity and importance measures of motion vectors are introduced. Using the importance measure, our proposed scheme manages to differentiate the most representative motion vector as a consideration to recompose a new motion vector. In addition, the diversity measure provides information for the video transcoder controlling the size of the refinement window to achieve a significant reduction of computational complexity.

4.2.2 An Adaptive Motion Vector Re-composition for high performance spatial video transcoder using Diversity and Important Measure (AMVR-DIM)

In this section, we present a new motion vector re-composition video downscaling transcoding architecture. The architecture of the transcoder is shown in Figure 4.6. The input bitstream is firstly parsed with a variable-length decoder to extract the header information, coding mode, motion vectors and quantized DCT coefficients for each

macroblock. Each macroblock is then manipulated independently. Switch *SW* is used to select appropriate tools to re-estimate the new motion vector. The selection depends on the motion vector and the amount of non-zero DCT coefficients. The switch positions for different coding modes are shown in Table 4.7. For non-MC macroblocks or the well-align case (e.g. all motion vectors having the same magnitudes and directions), the motion vector refinement process can be avoided. Hence, low computational complexity can be achieved. When the motion vectors are not well aligned, motion vector refinement process is necessary since the incoming prediction errors mismatch with the reconstructed new motion vector. However, high computational complexity is required for a large refinement window. Motivated by this, our proposed architecture estimates the new motion vector by considering its diversity and importance. The proposed diversity and importance measures are defined as follow:

The diversity measure of a resampled macroblock j is defined as

$$Diversity_j = | Mv_{hi} - \hat{M}v_h | + | Mv_{vi} - \hat{M}v_v | \tag{4.12}$$

where Mv_{hi} and Mv_{vi} represent the horizontal and vertical components of the motion vector of the original macroblock i(i=0 to 3), respectively. $\hat{M}v_h$ and $\hat{M}v_v$ represent the average horizontal components and vertical components of all motion vectors, respectively.

The importance measure of a resampled macroblock j is defined as

$$Importance_j = \sum_{i=0}^{3} DCT_i \tag{4.13}$$

where DCT_i represents the number of non-zero quantized DCT coefficients of the original macroblock i.

Then, high diversity of a resampled macroblock j can be defined as

$$\frac{Diversity_j}{\frac{1}{N}\sum_{j=0}^{N-1} Diversity_j} > 1 \tag{4.14}$$

and high importance of resampled macroblock j can be defined as

$$\frac{Im\,por\,tan\,ce_j}{\frac{1}{N}\sum_{j=0}^{N-1} Im\,por\,tan\,ce_j} > 1 \tag{4.15}$$

There are four types of conditions in this motion vector re-composition. The classification is made based upon their diversity and importance measures as shown in Table 4.7.

Type 1 According to the diversity and importance measures, if both measures are low, the transcoder uses one of the motion vectors with the highest importance and performs motion vector refinement within ± 1 pixel.

Type 2 If the diversity measure is low but the importance measure is high, the transcoder uses the motion vector with the highest importance and performs motion vector refinement within ± 3 pixels.

Type 3 If the diversity measure is high but the importance measure is low, four-motion vector mode is employed in the transcoder.

Type 4 According to the diversity and importance measures, if both measures are high, Intra-refresh mode is employed in the transcoder.

If the diversity and importance measures are high, it is difficult to obtain a new motion vector to represent the original four motion vectors. Since the average energy is high in these macroblocks, intra-refresh is a good technique to apply in this case. On the other hand, if the diversity is high but the importance measure is low, it is beneficial to

use four-motion vector mode since the incoming motion vectors have already minimized the prediction error successfully. It is advantageous to maintain these motion vectors instead of finding other solutions. If the diversity is low but the importance measure is high, a large refinement window is required since every macroblock contains a lot of prediction errors. A large refinement window can guarantee to find an optimal motion vector in this case. When the diversity and importance measures are low, only a small refinement window is necessary since the incoming motion vectors have already minimized the prediction error with good performance and it is easy to obtain a new motion vector when the diversity is low. By using this adaptive motion vector re-composition algorithm, the proposed transcoder can optimize the resultant motion vector in terms of good quality and low computational complexity. Besides, the transcoder can detect whether the new motion vector is good enough or has to choose an alternative solution to tackle the motion vector re-composition problem.

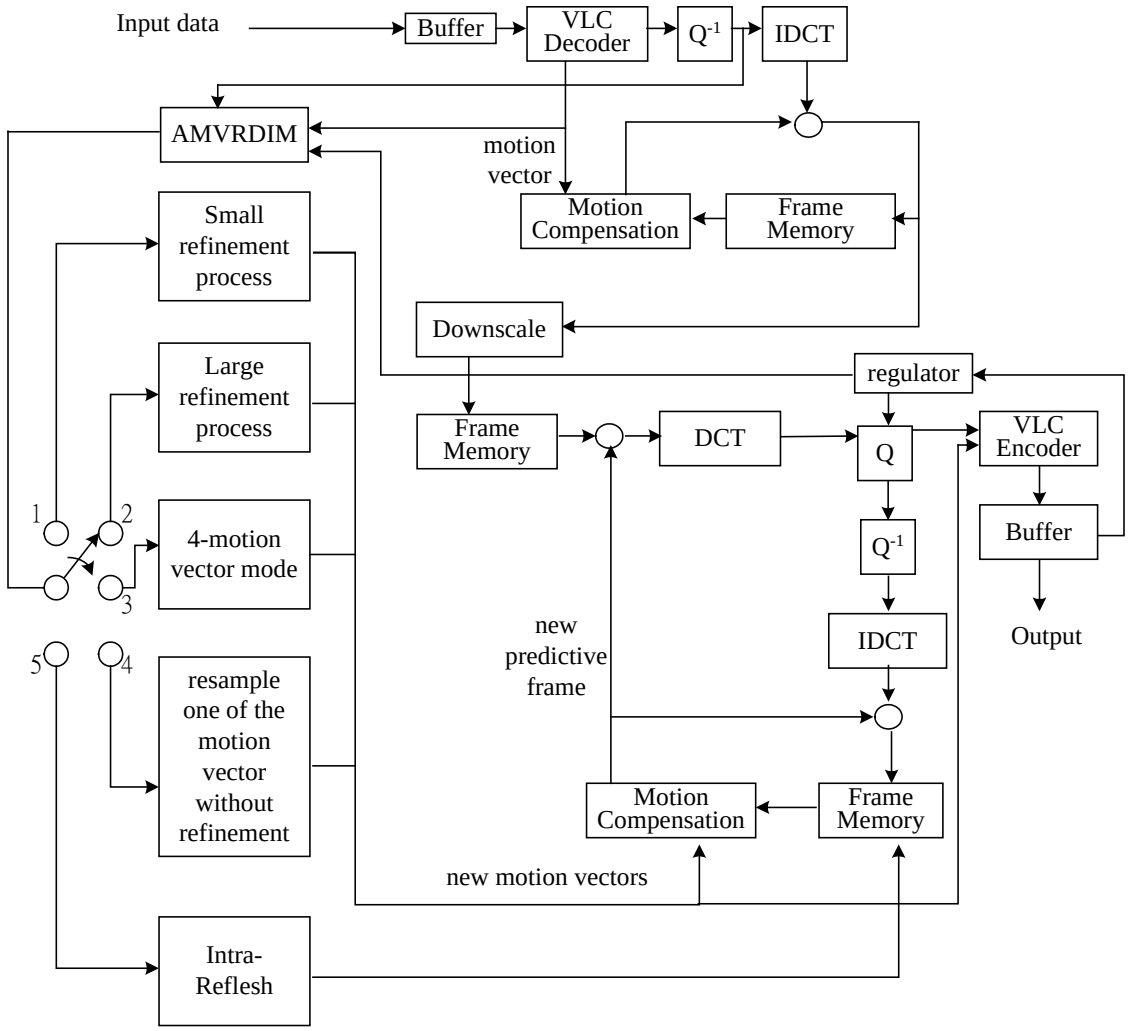


Figure 4.6. Architecture proposed for the video downscaling transcoder.

Table 4.7. Different coding modes of switches *SW* of the proposed transcoder.

Position	Diversity	Importance	Motion vector re-composition
5	High	High	Intra-refresh
3	High	Low	Four-motion vector mode
2	Low	High	Large refinement window is used
1	Low	Low	Small refinement window is used
4	Zero(well -align)	High/Low	No refinement process

4.2.3. Experimental Results

Extensive simulations and performance comparison have been done with reference to a conventional pixel-domain transcoder (CPDT) for employing align-to-

average weighting (AAW), align-to-best weighting (ABW) or adaptive motion vector resampling (AMVR) which is to resample a downscaled motion vector from the incoming motion vectors of the four macroblocks. In the front encoder, the first frame was encoded as intraframe (I-frame), and the remaining frames were encoded as interframes (P-frames). Picture-coding modes were preserved during transcoding. The pre-defined threshold can be determined adaptively by considering jointly the size of the buffer and the bitrate requirement. Larger diversity threshold and larger importance threshold will be set for the low bitrate application. Under this condition, the transcoder tends not to use the 4-motion vector and Intra Refresh mode since more bits are required in these two modes. This architecture is able to make the video downscaling transcoder adjust the video quality more dynamically according to the bandwidth requirement. Results of our experimental work show that the proposed video transcoders outperform CPDT+AAW, CPDT+ABW and CPDT+AMVR in all cases as shown in Table 4.8. The results are more significant for sequences with high motion activities because the proposed transcoder provides an optimal solution to resample the motion vectors according to the diversity and importance measures. Significant improvement in motion compensated regions can be achieved, which is about 0.7-1.9dB as compared with the conventional video downscaling transcoder. As compared with the refinement scheme with +/-3, 40 to 60% of the computational time can be saved for the proposed transcoder with the similar video quality of the transocded video.

Table 4.8. Average PSNR of the proposed transcoder, where the frame rate of the incoming bitstream was 30 frames/s. H.263 was used as the front encoder for encoding "Salesman", "Miss_America", "Hall", "Tennis", "Football" and "Flower".

Sequences	Input bitrate	Average PSNR difference as compared with CPDT+AAW for MC macroblock transcoding.		
		CPDT+ABW	AMVR[5]	AMVR-DIM
Salesman (352x288)	512k	0.06	0.41	0.76
	256k	0.05	0.39	0.70
Miss_America (352x288)	512k	0.09	0.39	0.74
	256k	0.07	0.36	0.71
Hall (352x288)	512k	0.11	0.42	1.24
	256k	0.08	0.38	1.16
Tennis (352x240)	3M	0.12	0.43	1.45
	1.5M	0.08	0.38	1.41
Flower (352x240)	3M	0.18	0.47	1.71
	1.5M	0.15	0.43	1.55
Football (352x240)	3M	0.21	0.50	1.91
	1.5M	0.27	0.56	1.87

4.2.4. Conclusion

In this section, we have proposed a simple architecture to form a low-complexity and high quality video downscaling transcoder to resolve the problem of Motion vector resampling. Using the diversity and importance measures, the proposed video transcoder is able to detect whether the optimal motion vector can be obtained during the resampling process. By jointly considering the diversity and importance measures, the proposed video transcoder can control the refinement window dynamically to achieve low computational complexity. Besides, the four-motion vector and Intra-Refresh modes provide alternative solutions for the video transcoder to tackle the problem of motion vectors with high diversity. Results of our experimental work show that the proposed architecture produces pictures quality with better as compared with the conventional video downscaling transcoder at the same reduced bitrates.

Chapter 5

Video Combiner

5.1 A Dynamic Frame-Skipping video Combiner for multipoint video Conferencing

5.1.1 Introduction

With the advance of video compression, networking technologies and international standards, video conferencing is widely used in our daily life[34]. In recent years, more and more video conferencing products are appearing on the market. The rapid growth of video conferencing has driven the development of mutlipoint video conferencing which can be integrated into personal computers thus providing an efficient way for more than two people to exchange information at multiple locations.

For multipoint video conferencing over a wide-area network, the conference participants are connected to a multipoint control unit (MCU) [54,55] which coordinates and distributes audio, video and data streams among multiple participants in multipoint video conferencing according to the requirement of the channel bandwidth. Figure 5.1 shows the scenario of four persons participating in a multipoint video conferencing with a MCU. An audio mixer in the MCU accepts audio data in a variety of formats, with different data rates. The audio mixer must decode and mix these different audio bitstreams from all the conference participants and the mixed audio signal is encoded again for distribution to the conference participants. Similarly, a video combiner is also included in the MCU to combine the multiple coded video bitstreams from the conference participants into a coded video bitstream which conforms to the video coding

standard such as H.263[34], and send it back to the conference participants for decoding and presentation.

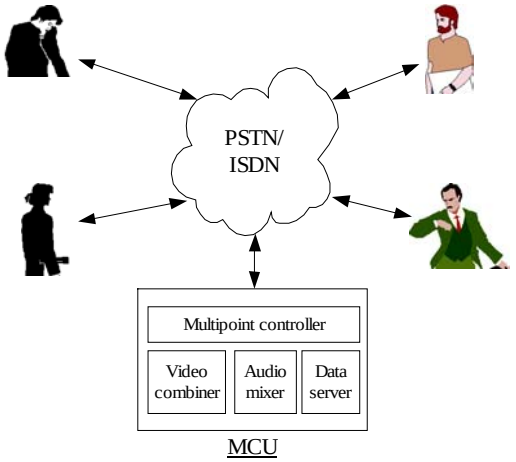


Figure 5.1. An example of multipoint video conferencing.

The four-point video conferencing as shown in Figure 5.1 is held over a practical wide-area network such as the Public Switch Telephone Network (PSTN) or the Integrated Service Digital Network (ISDN) in which the channel bandwidth is constant and symmetrical. Assume that encoded video bitstream of each conference participant is *R* kb/s in a quarter common intermediate format (QCIF: 176 × 144 pixels). The MCU receives and decodes the multiple video bitstreams from all the conference participants. The decoded videos are combined in a common intermediate format (CIF: 352 × 288 pixels) through the video combiner. An example of the video combining of four QCIF frames into a single CIF frame is illustrated in Figure 5.2. The combined video is re-encoded at *R* kb/s in order to fulfill the requirement of the channel bandwidth for purpose of sending back the encoded video to all conference participants. Therefore video transcoding must be performed at the video combiner. Transcoding is regarded as a process of converting a previously compressed video bitstream into a lower bit-rate

bitstream. Figure 5.3 is a block diagram of video combining for multipoint video conferencing using the transcoding approach[3,6,23].

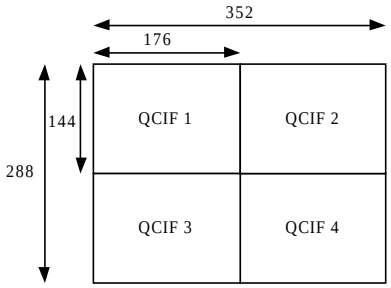


Figure 5.2. Combining four QCIF frames into a single CIF frame.

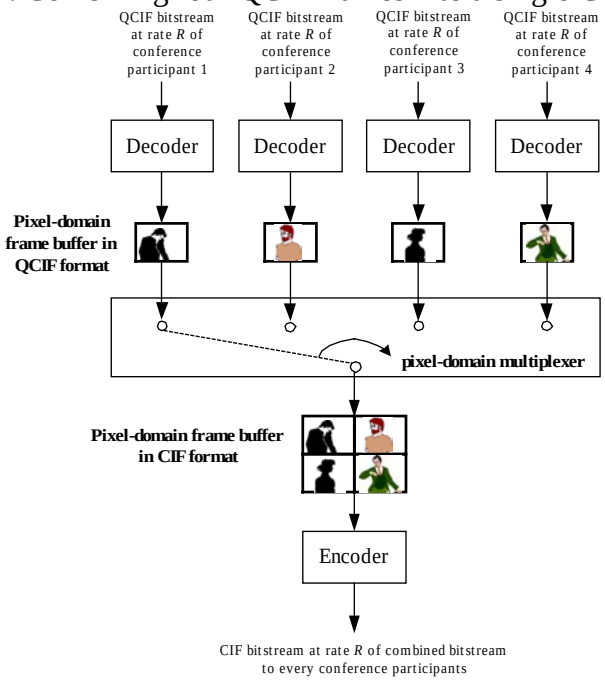


Figure 5.3. Pixel-domain video combiner using the transcoding approach.

Transcoding is a very practical approach for video combining in multipoint video conferencing over a symmetrical wide-area network. However, the computational complexity is inevitably increased since the individual video bitstream needs to be decoded and the combined video signal needs to be encoded. In addition, the video quality of the transcoding approach suffers from its intrinsic double-encoding process which introduces additional degradation. The visual quality and the computational

complexity need to be considered in video transcoding of multipoint video conferencing. In order to provide a satisfactory visual quality of combined and transcoded video, the re-distribution of the limited bits in the reencoding process to different parts of the combined video is a critical step. In most multipoint video conferencing, usually only one or two conference participants are active and talking at any given time, while the other participants are listening with little motion. To make best use of the available bit rates, a rate control scheme is proposed in [20] to measure the motion activity of each sub-sequence by computing the sum of the magnitudes of its corresponding motion vectors, and to allocate the bit rates to each sub-sequence according to its activity. Consequently, more bits will be allocated to those sub-sequences with higher motion activities, and this control scheme will produce a much more uniform visual quality.

In this section, a new architecture of video combiner which allows the sub-sequences to be transcoded in different frame rates according to their motion activities and audio levels is introduced. By dynamically allocating more frames (bits) to the active and talking conference participants in video combining, we are able to make a better viewing experience of the sub-sequences and the visual qualities of the active sub-sequences can be improved significantly. The proposed video combiner has two new main features:

1. a dynamic sub-frame skipping (DSFS) scheme for a dynamic selection of the most representative sub-frames according to the motion activities and the audio levels of the conference participants;
2. a high-quality and low-complexity frame-skipping transcoder which provides a satisfactory visual quality and reduces the computational burden of the MCU by

employing a direct summation of the DCT coefficients and fast motion re-estimation in the transcoder.

5.1.2 The proposed video combiner

Figure 5.4 shows the proposed system architecture for a video combiner in multipoint video conferencing. In contrast to [20], our approach to video combining is based on frame-skipping transcoding which is mainly performed on the discrete cosine transform (DCT) domain to achieve a transcoder with low complexity. For simplicity and without loss of generality, four QCIF video bitstreams are received by the video combiner from the conference participants. Each QCIF bitstream is first parsed with a variable-length decoder (VLD) to extract the header information, coding mode, motion vector and quantized DCT coefficients for each macroblock. The frame-skipping transcoder processes and updates the quantized DCT coefficients of the current frame in the DCT-domain buffer for each QCIF sub-sequence and the detail will be described in section 5.1.3. The transcoded and quantized DCT coefficients in the DCT-domain buffers in QCIF format are subsequently combined into a single buffer in CIF format through a multiplexer. At the beginning of the formation of a combining sequence, a decision is made as to which DCT-domain buffers (QCIF format) should be included in the new buffer (CIF format) by the dynamic sub-frame skipping (DSFS) controller.

As mentioned above, in most multipoint video conferencing, usually only one or two participants are active and talking at any given time, while the other participants are listening with little motion. The talkers usually attract most of the attention and they have larger motion than others. Therefore, allocating sub-sequences with higher frame rates (bit rates) can provide a much better visual experience to all conference participants.

On the other hand, more sub-frames of the inactive sub-sequences are dropped and the saved bits are reallocated to the active sub-sequences. Since the motion activities of the inactive sub-sequences are relatively slow and they are not the focus of the conferencing, the effect of frame-rate reduction by skipping the inactive sub-frames can often be masked by the active and talking sub-sequences. This loss is not noticeable to the conference participants.

To make the best use of this property, a dynamic sub-frame skipping (DSFS) is proposed which can dynamically distribute the encoded sub-frames to each sub-sequence by considering the motion activities and audio levels. As a consequence, the visual experience to the conference participants of the combined video can be enhanced. Thus, it is necessary to regulate the frame rate of each sub-sequence according to the motion activity in the current frame (MA_t) of the sub-sequence and its corresponding audio level. To obtain a quantitative measure for MA_t, we use the accumulated magnitudes of all of the motion vectors estimated for the macroblocks in the current frame, i.e.,

$$MA_t = \sum_{i=1}^{M} \left| (u_t^n)_i \right| + \left| (v_t^n)_i \right| \tag{5.1}$$

where M is the total number of macroblocks in the current sub-frame, and $(u_t^n)_i$ and $(v_t^n)_i$ are the horizontal and vertical components of the motion vector of the ith macroblock which uses the previous non-skipped sub-frame as a reference.

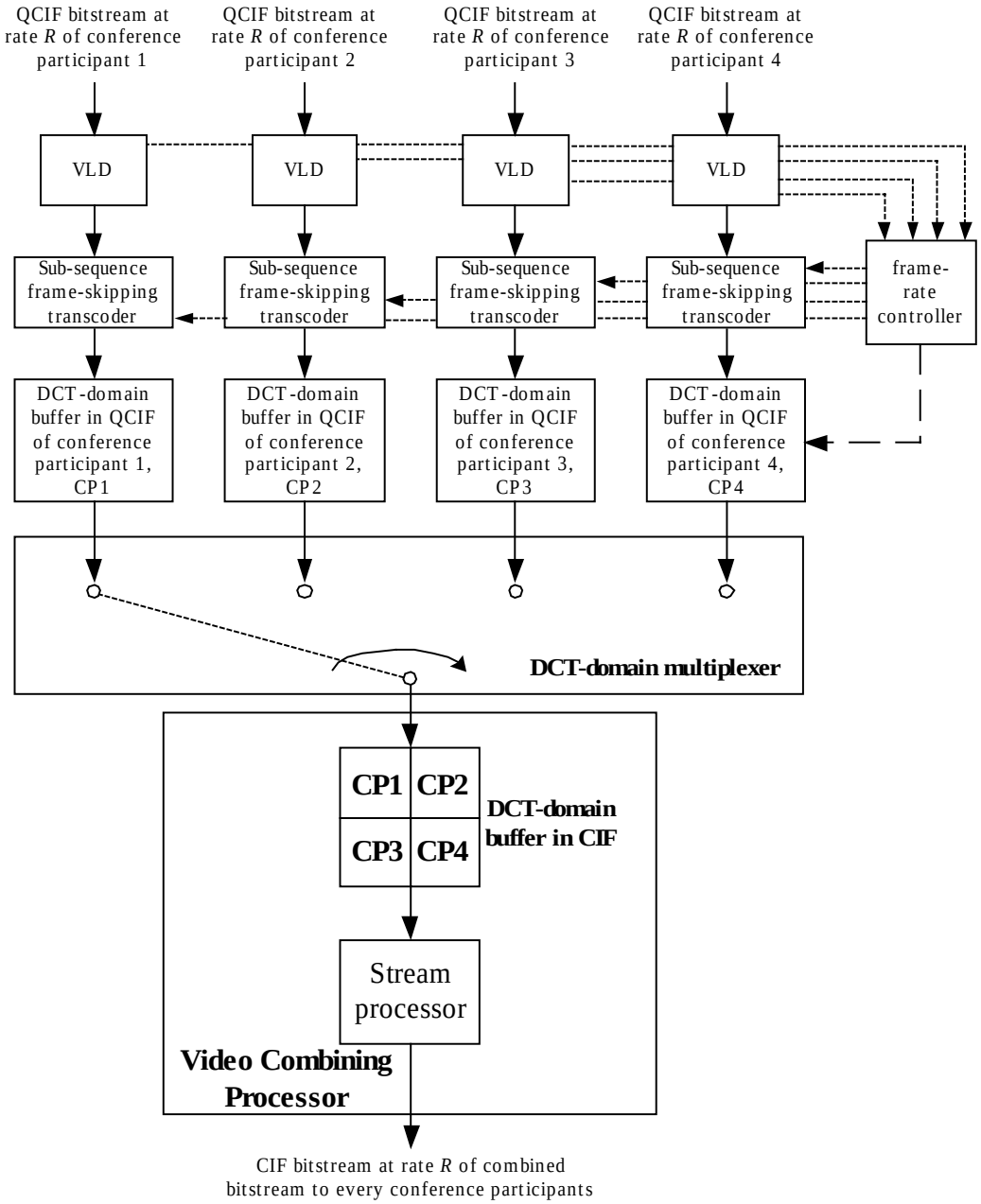


Figure 5.4. Architecture of the proposed video combiner.

If the value of MA_t after a non-skipped sub-frame exceeds the predefined threshold, T_{MA}, the incoming sub-frame should be kept. By adaptively adjusting the frame rate of each sub-sequence according to the MA_t, the proposed architecture can allocate more sub-frames for a sub-sequence with high motion activity and less sub-frames for a sub-sequence with low motion activity.

It is interesting to note that the different sub-sequences have different values of T_{MA} which are set according to the audio level of the sub-sequence. The larger the T_{MA} is set in the sub-sequence, the more the sub-frames will be skipped. Consequently, more saved bits will be used in other active and talking sub-sequences. The threshold setting process consists of testing the linear samples of the audio signal of a sub-sequence to detect whether this channel is active or silent. In the threshold setting process, the active talker can be identified and a lower T_{MA} will be used. The DSFS scheme is summarized in Figure 5.5.

If the current sub-frame of a sub-sequence is selected, it is picked out by the DCT-domain multiplexer and the DCT coefficients in the corresponding DCT-domain buffer are copied in the video combining processor's buffer with the size of CIF. The quantized DCT coefficients of each conference participant only need to be mapped according to Figure 5.4. Hence, the DCT-domain buffer of conference participant 1, *CP1*, is mapped to the first quadrant of the combined picture, the DCT-domain buffer of conference participant 2, *CP2,* is mapped to the second quadrant, etc. The data in the DCT-domain buffer will be encoded in the compressed bitstream by the stream processor. If the current sub-frame of a sub-sequence is skipped, it simply sets all the COD (coded macroblock indication) bits in the H263 syntax[34] of the macroblocks which belong to the skipped sub-frames to "1" to avoid sending the associated DCT coefficients, motion vectors, and macroblock overhead bits. Only 99 COD bits are required to represent a skipped QCIF sub-frame, thus the overhead is relatively negligible.

T_{MA} setting process
if (energy level of sub-sequence's audio < T_{audio})
then
$T_{MA} = T_1$;
else
$T_{MA} = 0.5 \times T_1$;
where T_{audio} and T_1 are the predefined threshold.

Dynamic sub-frame skipping
if ($MA_t < T_{MA}$)
then
skip the current sub-frame;
else
keep the current sub-frame;

Figure 5.5. Pseudocode of the DSFS scheme.

5.1.3 The High performance frame-skipping transcoder

According to the skipping decision provided by the DSFS controller, the frame-skipping transcoders in Figure 5.4 must perform the sub-frame skipping on different sub-sequences. Each transcoder also has the responsibility for updating its corresponding DCT-domain buffer (QCIF format) in the proper manner such that these buffers can be multiplexed by the video combining processor as mentioned in section 5.1.2. However, the skipped sub-frame must be decompressed completely, and should act as the reference sub-frame to the non-skipped sub-frame for reconstruction. The newly quantized DCT coefficients of prediction error and the motion vectors need to be re-computed for the non-skipped sub-frame with reference to the previous non-skipped sub-frame, which introduces a re-encoding process; this can create an undesirable complexity in real time application as well as introduce re-encoding error. In this section, we propose a new sub-sequence frame-skipping transcoder for improving picture quality and reducing complexity. The proposed transcoder is mainly performed on the discrete cosine transform (DCT) domain to achieve a low complexity transcoder. In [23], we have proposed a direct summation of the DCT coefficients in the frame-skipping transcoding

to avoid the re-encoding errors and to reduce the complexity for macroblocks coded without motion compensation. In this section, we present a new frame-skipping transcoder which is an extension of the work of [23]. The new transcoder has the following features:

1. a direct summation of the DCT coefficients for re-computing the newly quantized DCT coefficients of prediction error in the macroblocks without motion compensation;
2. an adaptive motion vectors composition for the new motion vectors.

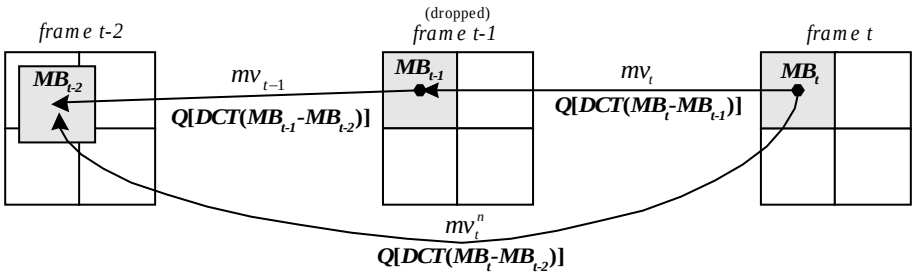
A. Macroblocks without motion compensation

For those macroblocks coded without motion compensation, a direct summation of the DCT coefficients is employed such that the DCT transform and motion compensation operations are not needed for reconstructing the newly quantized DCT coefficients of prediction error. For typical video conferencing, a majority of the video signal is coded without motion compensation, and hence the complexity reduction realised by using the direct summation is significant. In Figure 5.6, a situation in which one frame is dropped is illustrated. We assume that MB_t represents the current reconstructed macroblock in frame t and MB_{t-1} represents the best matching reconstructed macroblock to MB_t in frame $t-1$. Since MB_t is coded without motion compensation, the spatial position of MB_{t-1} is the same as that of MB_t, and MB_{t-2} represents the best matching reconstructed macroblock to MB_{t-1} in frame $t-2$. Since frame $t-1$ is dropped, for MB_t, we need to compute new motion vector mv_t^n and the prediction error in the DCT-domain, $Q[DCT(MB_t - MB_{t-2})]$, by using frame $t-2$ as a reference. Since the motion vector in MB_t is zero, then

$$mv_t^n = mv_{t-1} \tag{5.2}$$

Re-encoding can lead to an additional error, but it can be avoided if $Q[DCT(MB_t - MB_{t-2})]$ is computed in the DCT-domain. The newly quantized DCT coefficients of prediction error, $Q[DCT(MB_t - MB_{t-2})]$, can then be computed as,

$$\begin{aligned} &Q[DCT(MB_t - MB_{t-2})] \\ &= Q[DCT[(MB_t - MB_{t-1}) + (MB_{t-1} - MB_{t-2})]] \end{aligned} \tag{5.3}$$



Q(.) - Quantization operator
DCT(.) - DCT operator

Figure 5.6. Macroblocks without motion compensation.

Taking into account the linearity of DCT, equation (5.3) can be expressed as,

$$\begin{aligned} &Q[DCT(MB_t - MB_{t-2})] \\ &= Q[DCT(MB_t - MB_{t-1}) + DCT(MB_{t-1} - MB_{t-2})] \end{aligned} \tag{5.4}$$

In Figure 5.6, $Q[DCT(MB_t - MB_{t-1})]$ and $Q[DCT(MB_{t-1} - MB_{t-2})]$ are the inputs to the transcoder. An inverse quantization can be performed to get back the $DCT(MB_t - MB_{t-1})$ and $DCT(MB_{t-1} - MB_{t-2})$. The new $Q[DCT(MB_t - MB_{t-2})]$ can be obtained according to equation (5.4).

However, if the same quantizer step-size is used in both the input and output of the transcoder, the quantization becomes a linear operation since integer truncation is not involved in that case. Equation (5.4) can be written as,

$$\begin{aligned} & Q[DCT(MB_t - MB_{t-2})] \\ &= Q[DCT(MB_t - MB_{t-1})] + Q[DCT(MB_{t-1} - MB_{t-2})] \end{aligned} \quad (5.5)$$

Equation (5.5) implies that the newly quantized DCT coefficients $Q[DCT(MB_t - MB_{t-2})]$ can be computed in the DCT-domain by summing directly the incoming quantized DCT coefficients, $Q[DCT(MB_t - MB_{t-1})]$ and $Q[DCT(MB_{t-1} - MB_{t-2})]$. Since it is not necessary to perform the motion compensation, DCT, quantization, inverse DCT and inverse quantization, the complexity is reduced. Furthermore, since requantization is not necessary for this type of macroblocks, the quality degradation of the transcoder introduced by re-encoding is also avoided. By using a direct summation of the DCT coefficients for the non-moving macroblocks, the computational complexity involved in processing these macroblocks can be reduced significantly and the additional re-encoding error can be avoided.

B. Motion compensated macroblocks

For motion-compensated macroblocks, the motion vectors in the dropped sub-frames are usually not available in the incoming bitstreams, as depicted in Figure 5.7, thus new motion vectors for the transcoded bitstream need to be re-computed. Motion vector re-estimation is undesirable due to the intensive computation. Instead, motion vector re-estimation using the available motion vectors is a better approach [2]. It is possible to use bilinear interpolation from the motion vectors mv_{t-1,MB_1}, mv_{t-1,MB_2}, mv_{t-1,MB_3} and mv_{t-1,MB_4} which are the four neighboring macroblocks, MB_{t-1}^1, MB_{t-1}^2, MB_{t-1}^3 and MB_{t-1}^4, of MB_{t-1} to come up with an approximation of mv_{t-1} [21]. However, bilinear interpolation of motion vectors leads to inaccuracy of the resultant motion vector because the area

covered by the four macroblocks may be too divergent and too large to be described by a single motion vector [2]. Thus, the dominant vector selection approach is used [2] to select one dominant motion vector from four neighboring macroblocks. A dominant motion vector is defined as the motion vector carried by a dominant macroblock. The dominant macroblock is the macroblock that has the largest overlapped segment with MB_{t-1}. For example, the new motion vectors in Figure 5.7 can be composed by

$$mv_t^n = mv_t + mv_{t,MB_1} \tag{5.6}$$

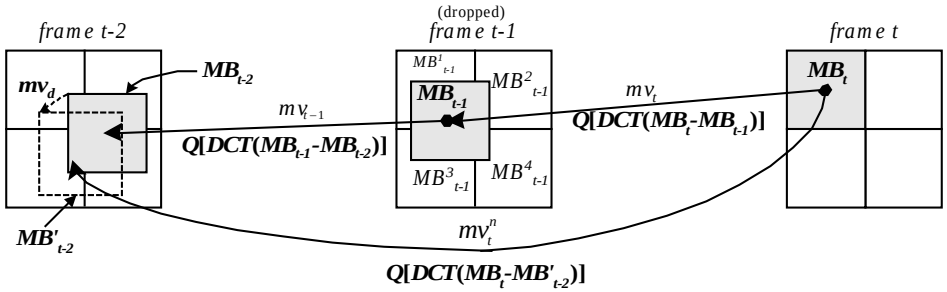


Figure 5.7. Motion-compensated macroblocks.

In [2], the motion vector refinement scheme uses the motion vector computed in equation (5.6) as the base motion vector and then performs a motion estimation in a very small search range around the base motion vector. The performance of this scheme is further improved in this chapter. If there is a strong dominant macroblock which overlaps the reference macroblock in a significantly large area, this motion vector is reliable for composing the new motion vector. In other words, refinement of the motion vector is not required. The adaptive motion vector refinement scheme is summarized as follows:

1. The base motion vector, mv_t^n, is composed according to equation (5.6).

2. If the largest overlapping area is greater than a T_{oa}, which is a predefined threshold (e.g. say 80% of the macroblock area), then select the base motion vector as the final motion vector, otherwise go to step (3).
3. The delta motion vector, mv_d, is estimated within a small search area (±*3* pixels) around the base motion vector (motion vector refinement scheme), as depicted in Figure 5.7, and the new motion vector is computed by

$$mv_t^n = mv_t + mv_{t,MB_1} + mv_d \tag{5.7}$$

After obtaining the new motion vector, the newly quantized DCT coefficients, $Q[DCT(MB_t - MB'_{t-2})]$, are required to be computed. Again, since MB_{t-1} is not on a macroblock boundary, direct summation cannot be employed. In other words, re-encoding of the motion-compensated macroblocks is inevitable.

C. DCT-domain buffer updating for multiple sub-frame skipping

Another advantage of the proposed transcoder is that when multiple frames are dropped, it can be processed in the forward order, thus eliminating the multiple DCT-domain buffers that are needed to store the incoming quantized DCT coefficients of all dropped frames.

Figure 5.8 shows a scenario when two frames are dropped. When frame *t-2* is dropped, we store the DCT coefficients of its prediction errors in the DCT-domain buffer. The stored DCT coefficients of prediction errors will be used to update the DCT coefficients of prediction errors of the next dropped frame. This means that when frame *t-1* is dropped, our proposed scheme updates the DCT coefficients of prediction errors for each macroblock according to its coding mode. For example, macroblocks, MB_{t-1}^2, MB_{t-1}^3,

and MB^4_{t-1}, in frame *t-1* are coded without motion compensation. From eqn (5.5), the DCT coefficients of prediction errors in the DCT-domain buffer are added to the corresponding incoming prediction errors of the macroblock in frame *t-1*. The buffer is then updated with the new DCT coefficients. In Figure 5.8, MB^1_{t-1} is a motion-compensated macroblock. It is necessary to perform the re-encoding of this macroblock and then update the corresponding incoming DCT-coefficients to form the updated data in the DCT-domain buffer. By using our proposed scheme, only one DCT-domain buffer is needed for all dropped frames. The flexibility of multiple frame-skipping provides the fundamental framework for dynamic sub-sequence frame-skipping.

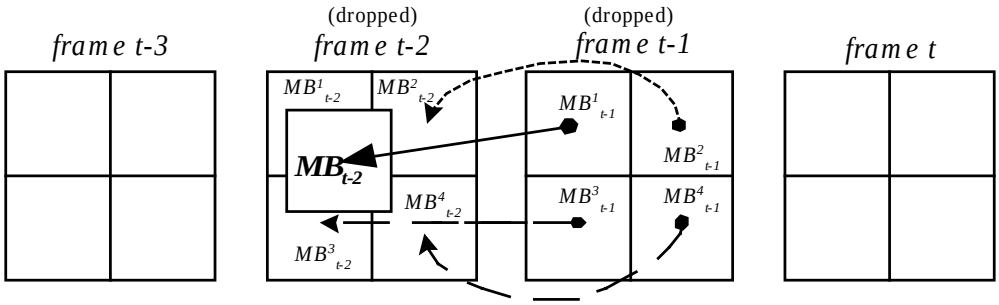


Figure 5.8. Multiple frame skipping of our proposed transcoder.

5.1.4. Experimental Results

A series of experiments were conducted to evaluate the overall efficiency of the proposed video combiner for multipoint video conferencing. For our experiments, we recorded a four-point video conferencing session. Each conferee's sequence was encoded into a QCIF format at 128kb/s according to the H.263 standard [34], as depicted in Figure 5.9. At the front encoder, the first frame was coded as intraframe (I-frame), and the remaining frames were encoded as interframes (P-frames). These picture-coding modes were preserved during the transcoding. The four sequences were then transcoded

and combined into a CIF format. We then selected segments of the combined sequence to form a 400-frame video sequence in which the first person was most active in the first 100 frames, the second person was most active in the second 100 frames, and so on. The motion activities for the four conference participants and the combined video sequence are shown in Figure 5.10. In the figure, the four curves correspond to the four conference participants in the upper left, upper right, lower left and lower right corners, respectively. The figure shows that although there were short periods of time when multiple participants were active at the same time, only one participant was active during most of the time, while other participants were relatively inactive. This indicates that multipoint video conferencing is a suitable environment for dynamic allocation of the encoding sub-frames to each sub-sequence.

In the following discussion, we will analyse the performance of the proposed video combiner as compared to the conventional pixel-domain combiner proposed by Sun *et al* [20]. Since the active participants need higher frame rates to produce an acceptable video quality while the inactive participants only need lower frame rates to produce an acceptable quality, the dynamic sub-frame skipping (DSFS) scheme in the proposed video combiner is used to distribute the sub-frames to each sub-sequence according to the motion activities and the energy of the audio signals. For example, the participant is most active from frame 0 to frame 100 in Figure 5.11(a) (as shown in the motion activity plot in Figure 5.10). This active period is transcoded more frequently following the motion activities of the sub-sequence, therefore the sub-sequence displayed on the receiver is smoother. Inevitably, this improvement is made by sacrificing a certain amount of quality of the motion inactive periods. Figure 5.11 shows the PSNR

performances of all the frames for different video combining approaches. The PSNR of a skipped sub-frame is calculated from the incoming sub-frames and the latest previously non-skipped sub-frame, since the sub-frame repetitions will occur for the skipped sub-frames at the video decoders. It can be seen from Figure 5.11 that the Sun *et al's* video combiner loses due to the double-encoding aspect and the proposed video combiner offers a significant gain for both the active and non-active periods. This is because the probability of the macroblock coded without motion compensation happens more frequently in typical video conferencing sequences, and this type of macroblock should not introduce any re-encoding error due to the direct summation of the DCT coefficients. Also, the direct summation and adaptive motion refinement in the proposed transcoder can reduce the computational burden of re-encoding, which gives rise to a significant speed-up of about 9.2 times as compared with the Sun *et al's* video combiner.

A frame (184^{th} frame) in the combined sequence is shown in Figure 5.9. It can be seen that the video quality of the active participant (at the upper right corner) produced by the proposed video combiner is much better than that of the Sun *et al's* video combiner. Due to the DSFS, the degradation of the video quality of the inactive participants is not very visible and this is also supported by Figure 5.9 and Figure 5.. The detailed PSNR of all sub-sequences using different video combiners are summarized in Table 5.1 and Table 5.2, which represents the average PSNR's of the sub-sequence of all frames and non-skipped frames, respectively. The diagonal PSNR's indicate more active motion in different time slots of individual conference participants. These tables show that by using the proposed video combiner the PSNR's among all sub-sequences are much improved as compared to the Sun *et al's* video combiner during both the active and

non-active periods. In practical multipoint video conferencing, active participants are given most attention. Improvement of the video quality of the active participants is particularly important and we have shown that the proposed video combiner can achieve a significant improvement.

Table 5.1. Average PSNR's of the sub-sequence of all frames.

	Frame 1 – 100 (most active: CP1)		Frame 101 – 200 (most active: CP2)		Frame 201 – 300 (most active: CP3)		Frame 301 – 400 (most active: CP4)	
	A	B	A	B	A	B	A	B
CP1	**32.62**	**35.58**	34.68	36.54	34.48	36.57	34.39	36.54
CP2	33.95	36.57	**32.80**	**35.89**	34.45	36.18	34.38	36.32
CP3	33.81	36.32	33.99	36.49	**31.95**	**35.13**	34.32	36.38
CP4	33.86	36.42	33.91	36.69	33.77	36.66	**32.21**	**35.68**

A – Sun *et al.*'s video combiner [20].
B – Proposed video combiner.

Table 5.2. Average PSNR's of the sub-sequence of non-skipped frames.

	Frame 1 – 100 (most active: CP1)		Frame 101 – 200 (most active: CP2)		Frame 201 – 300 (most active: CP3)		Frame 301 – 400 (most active: CP4)	
	A	B	A	B	A	B	A	B
CP1	**32.62**	**35.77**	34.68	36.66	34.48	36.72	34.39	36.68
CP2	33.95	36.71	**32.80**	**36.06**	34.45	36.30	34.38	36.48
CP3	33.81	36.48	33.99	36.65	**31.95**	**35.34**	34.32	36.55
CP4	33.86	36.59	33.91	36.83	33.77	36.81	**32.21**	**35.89**

A – Sun *et al.*'s video combiner [20].
B – Proposed video combiner.

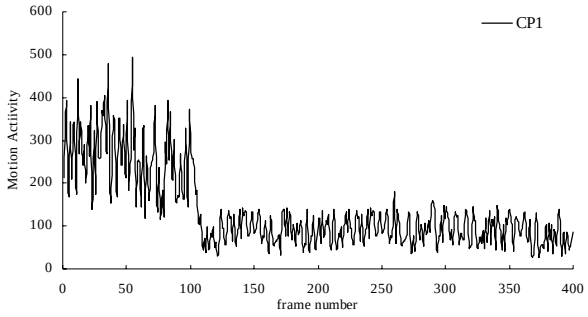
(a)

(b)

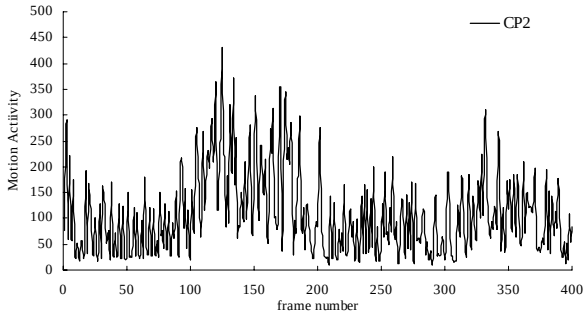
(c)

(d)

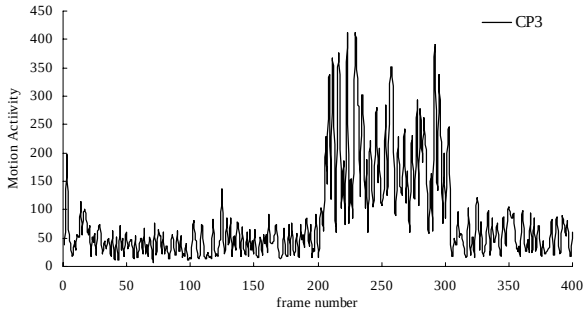
Figure 5.9. Encoded frame 184 of the four conferees' videos which are received by the video combiner. (a) CP1, (b) CP2, (c) CP3 and (d) CP4.



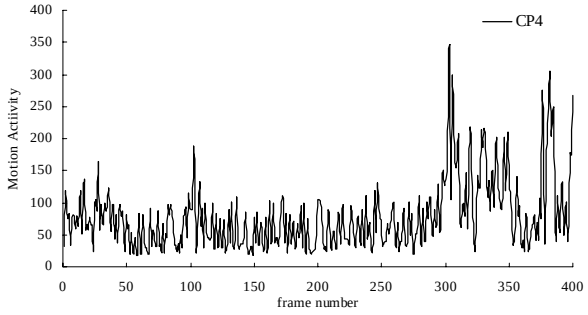
(a)



(b)

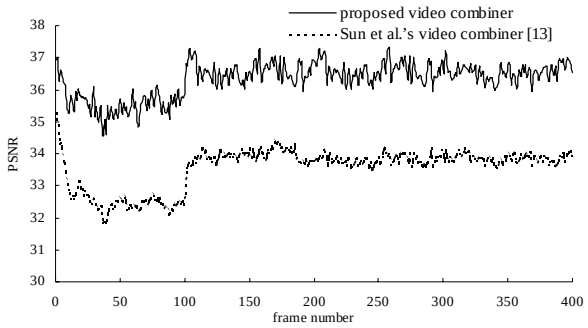


(c)

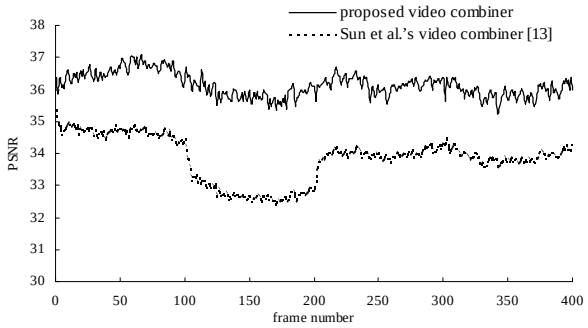


(d)

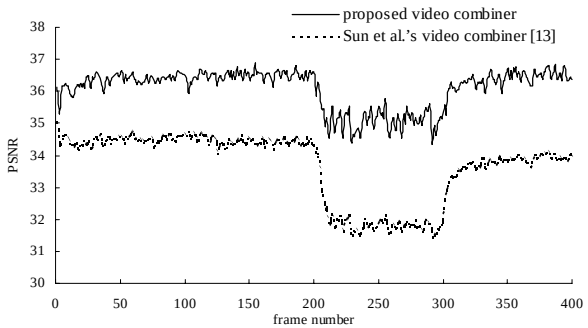
Figure 5.10. Motion activity of multipoint video conferencing.



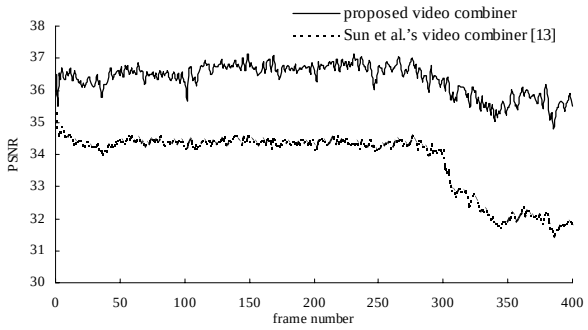
(a)



(b)



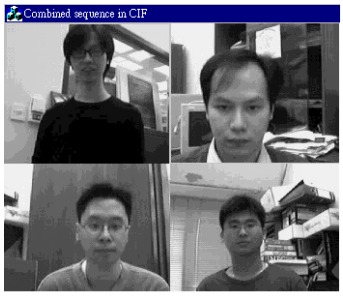
(c)



(d)

Figure 5.11. PSNR performance of a conference participant who is most active (a) between frame 0 and frame 100, (b) between frame 101 and frame 200, (c) between frame 201 and frame 300, (d) between frame 301 and frame 400.

(a)



(b)

Figure 5.12. Frame 184 of the combined video sequence using (a) Sun *et al.*'s video combiner [20] (b) our video combiner. The active conference participant is at the upper right corner

5.1.5 Conclusions

In this section, we have implemented a new video combiner for multipoint video conferencing by using the dynamic sub-frame skipping (DSFS) scheme and the high-performance frame-skipping transcoder. The DSFS scheme can improve the visual quality of the active and talking sub-sequences by re-allocating the saved bits from the inactive sub-sequences. The frame rate of coding a sub-sequence depends very much on its motion activity and audio level. We also propose a frame-skipping transcoder for minimizing re-encoding error and reducing computational burden by employing the direct summation of the DCT coefficients and the adaptive motion refinement. This transcoder can be processed in the forward order when multiple frames are dropped. Thus, only one DCT-domain buffer is needed to store the updated DCT coefficients of all

dropped frames. The proposed video combiner can provide a better performance than the conventional video combiner in terms of better quality and reduced complexity. In summary, it can be seen that a video combiner using the frame-skipping transcoding approach is able to provide a new and viable multipoint video conferencing service for the near future.

5.2 A Dynamic video Combiner for multipoint video Conferencing using wavelet transform

5.2.1 Introduction

This section introduces a new architecture of video combiner for multipoint video conferencing. The proposed video combiner is wavelet-based which extract the motion activities information from the video bitstreams produced by a wavelet-based video coder. By making use of the progressive properties of wavelet transform, the video quality level of inactive sub-sequences can be easily adjust in the video combiner, so that more bits can be reallocated to the active sub-sequences such that better visual perception as well as a smoother motion can be achieved. Since the video coder is region based, the computational complexity can be reduced significantly by using different wavelet kernels for the foreground and background. Due to progressive properties of the wavelet transform, the video combiner can easily adjust the video quality, so that the video quality in foreground always can be guaranteed and the background quality can be maintain in a acceptable level even under low bitrate environments. The new transcoder is then used to realize the multipoint video conferencing and some results are presented to show the improvement in performance due to our proposed architecture.

5.2.2 The Proposed video coder and video combiner

Figure 5.13 shows the system architecture of the wavelet-based video coder in multipoint video conferencing system[51]. Since the video conferencing system is wavelet-based, blocking artifacts are avoided. The wavelet-based video conferencing system has four major features:

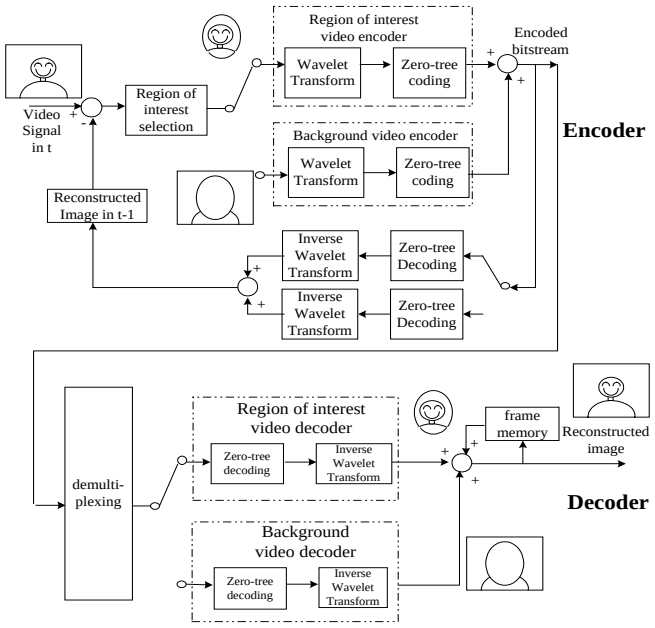


Figure 5.13. The system architecture for the proposed video coder in multipoint video conferencing.

1. selection of the region of interest;
2. adaptive bit allocation for the foreground (region of interest) and the background,
3. progressive transmission updating, and
4. adaptive bit reallocation algorithm in the video combiner

The purpose of region selection is to identify the region of interest in an image, e.g., the speaker's face in video conferencing. This region is updated automatically by tracking the object's motion using histogram information[51]. The wavelet-based coder is then applied separately to the foreground and the background. Because the size of the

region of interest is small, the computation time can be reduced significantly. Adaptive bit allocation is then performed. It makes sure that the video quality of the foreground is always better than that of the background. This is particularly important for unstable networks or low bit rate applications. The progressive transmission updating is employed so that the video quality and the coding efficiency can be improved even under low bitrate environment. The video combiner extracts motion activities from the incoming video bitstreams and performs bit reallocation such that the output bitrate is the same as the incoming bitrate. In the bit reallocation process, low complexity can be achieved since the video is scalable and no re-encoding process is required. The video combiner only needs to calculate the motion activities and the targeted number of bits for each conference participants. According to the targeted number of bits for each conference participants, the video combiner adjusts the video quality level by discarding part of the high quality level video bitstreams, if necessary. In the following discussion, we will focus on scalable video combiner using successive quantization techniques.

5.2.3 Progressive transmission in video conferencing system

In the zero-tree coding, a successive quantization scheme is proposed. Initially, a coarse quantization level is applied to the wavelet coefficients in both ROI and background. In this quantization process, a base level video quality can be obtained as shown in Figure 5.14a. This coarse quantization level favours for low bitrate video conferencing applications. In order to enhance the video quality, a second level quantization level is applied to the residue of the quantized wavelet coefficients. Combining these two levels of quantized coefficients, an enhanced level video quality can be obtained as shown in Figure 5.14b. A final quantization level is applied to the

residue of the quantized wavelet coefficients if the bandwidth is available. Combining these three levels of quantized wavelet coefficients, a high video quality can be achieved as shown in Figure 5.14c. The encoded wavelet bitstreams show no redundant information. Figure 5.13 presents the architecture of our proposed video coder using wavelet transform. Since the image at time t and t-1 are highly correlated, it is beneficial to make use of this correlation. Firstly, the difference between the video signal at time t and the encoded image at time t-1 are fed into the encoder as an input signal. Then the input signal passes into the region of interest video encoder and the background encoder. According to the region of interest selected by the user, a simple tracking will be applied using histogram or chrominance information. After the encoding process, both video streams will be combined and transmitted to the client side for decoding. Note that the encoded image at time t-1 can be reconstructed in the front encoder. This arrangement is used to improve the video coding efficiency, video quality and facilitate the progressive transmission process.

Since the difference between the current frame and the previous reconstructed frame is small in both background and slow motion regions, less bits are required to encode these regions in a practical situation. In practice, even 30% of the targeted bits are spent in the background video coding, the video quality still can be improved progressively. Besides, as compared with the traditional video transcoder, no re-encoding process is required. Hence, high computational complexity and quality degradation can be avoided in the transcoding process.

(5.14a)

(5.14b)

(5.14c)

Figure 5.14. Different levels of video qualities. (5.14a) first level, (5.14b) second level and (5.14c) third level).

5.2.4. Adaptive bit reallocation algorithm in the video combiner

According to the motion activities and the bandwidth constraints of each conference participants, the video combiner adjusts the scalable video quality level for each conference participants. Firstly, our proposed video combiner extracts the motion activities information from the incoming bitstreams as shown in Figure 5.15.

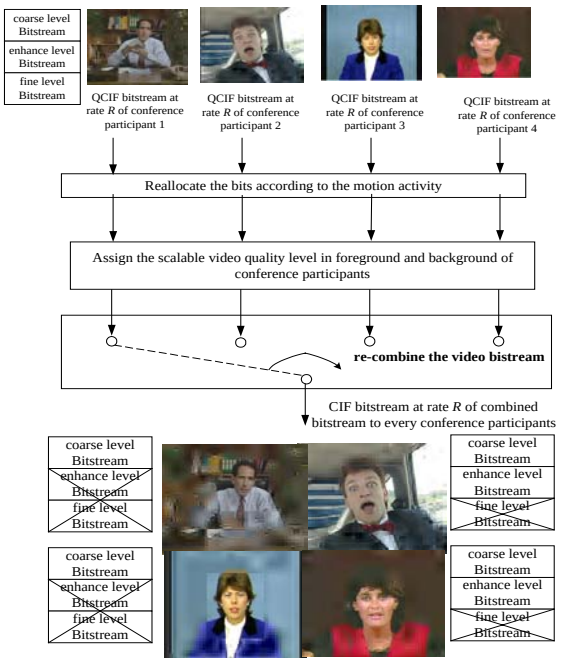


Figure 5.15. Our proposed video combiner.

In most multipoint video conferencing application, usually only one or two conference participants are active at a given time, while other participants have no or little motion. To make best use of the available bit rates, a rate control scheme is proposed to calculate the targeted bits based on the motion inside the region of interest. Since the speaker could have little motion when active, the motion activities is defined in terms of the magnitude of the motion of the region of interest and the energy of the audio signal as shown in following equations:

$$cost_{video} = |motion\_of\_the\_region\_of\_interest| \tag{5.8}$$

and

$$If\ |Audioenergylevel| > audio\_background\_energy\_threshold$$

$$cost_{audio} = \lambda \tag{5.9}$$

$$cost_t = cost_{audio} + cost_{video} \tag{5.10}$$

where $cost_i$ represents the importance of user i and λ is a constants. The video combiner evaluates the percentages of bits for every conference participants by normalizing the speaker activities as described in equation (5.11).

$$\%\_of\_Bits\_assigned\_to\_CP_i = \frac{cost_i}{\sum_{i=1}^{N} cost_i} x100\% \tag{5.11}$$

After the percentages of bits for every speaker are calculated, the video combiner adjusts the video quality level according to the bandwidth constraints for each conference participants. Consequently, more bits will be allocated to those sub-sequences with higher motion activities. Since the incoming video is scalable, our proposed video combiner can achieve a low complexity by discarding the fine level or the second level quantized coefficients according to the bandwidth requirement and the users' activities.

5.2.5 Experimental Results

The proposed video conferencing system is tested for the 64kbit/sec case. Figure 5.16 and Figure 5.17 show a comparison between our proposed system and the conventional DCT-based system. The overall performance is shown in Table 5.3. Although the background has a lower PSNR as compared to the conventional approach[21], the foreground has a much higher PSNR. Moreover, the subjective performance is much better than the conventional approach and the blocking artifacts are avoided. In fact, the subjective superiority is even more profound at low bit rates. In this case, the blocking artifacts associated with the DCT-based encoders are severe. However, by using our proposed video conferencing system, the quality in the foreground can still be maintained.

Table 5.3. Comparison of average PSNR's performances with the conventional video transcoder.

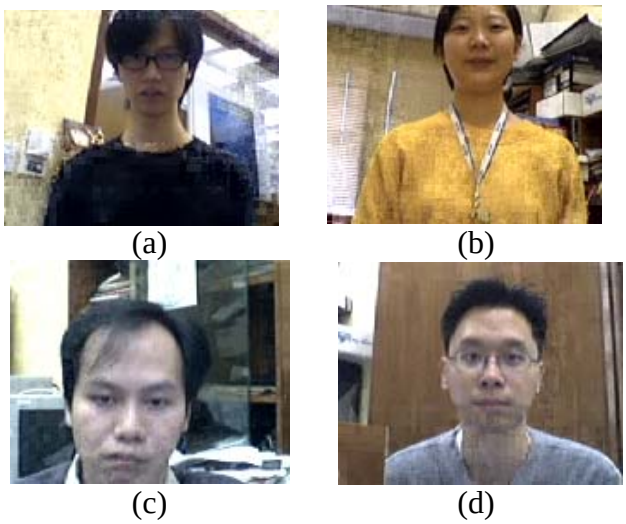
	Our proposed video conferencing system			Conventional video conferencing system[21]		
	Foreground	Background	Overall	Foreground	Background	Overall
CP1	36.4	30.4	32.3	30.6	32.6	32.1
CP2	36.8	30.8	32.9	29.7	33.6	32.8
CP3	35.4	34.7	34.9	29.9	36.6	33.4
CP4	35.6	34.3	34.8	30.3	35.2	34.3

5.2.6 Conclusions

A region-based video combiner is proposed. The proposed architecture consists of region of interest selection, wavelet-based encoders for the foreground and the background, motion-based region updating steps and adaptive bit re-allocation strategy in the video combiner. Based on the motion activities of the users, our proposed video combiner adjusts the speaker video quality without performing re-encoding process. Experimental results confirm that our proposed system produces a good video quality even under low bit rates for real-time video conferencing applications.

(a) (b) (c) (d)

Figure 5.16. Four conferees' output videos using conventional video transcoder (a) CP1, (b) CP2, (c) CP3 and (d) CP4.



(a) (b)

(c) (d)

Figure 5.17. Four conferees' output videos using our proposed video conferencing system (a) CP1, (b) CP2, (c) CP3 and (d) CP4.

Chapter 6

Heterogeneous Video Transcoding

6.1 A new Compressed-domain heterogeneous video transcoder

6.1.1 Introduction

Homogeneous transcoding techniques of MPEG-2 to MPEG-2, H.261 to H.261, H.263 to H.263 have been investigated[30-34]. However, there are requirements for heterogeneous transcoding, such that decoders of one form (e.g., H.263) have to receive videos which were previously coded by another form (e.g., MPEG-2). This is particularly important for transmitting videos over low bandwidth channels or hostile environments such as mobile networks and internet. The emergence of the forthcoming Universal Mobile Telecommunication System (UMTS) carrying video, voice, and data is a good example. In this case, in contrast to homogeneous transcoding, picture type, directionality of motion vectors, and picture rate might all change. To resolve this problem, one straightforward approach for implementing heterogeneous transcoding is to cascade a decoder and an encoder, commonly known as pixel-domain transcoding as shown in Figure 6.1. The incoming MPEG-2 video bitstream is decoded in the pixel domain, and the decoded video frame is re-encoded by an H.263 encoder at the desired output bitrate according to the capability of the clients' devices and the available bandwidth of the network. This involves high processing complexity, large memory and long delay. Recently, some fast algorithms have been proposed by making use of the information from the incoming bitstream [1-5,6,8,35] to reduce the computational complexity. For example, motion vectors extracted from the incoming bitstream after decoding can be used to reduce significantly the complexity of transcoding, since motion

re-estimation can be avoided[36-43]. However, the video needs to be decoded fully in the pixel domain and re-encoded according to the H.263 in this example. This pixel-domain transcoding approach suffers from its intrinsic double-encoding process, which introduces additional degradation and high computational complexity.

Recently, some discrete cosine transform(DCT) domain transcoding processes were introduced[9,26-28]. In these approaches, the incoming video bitstream is partially decoded in the DCT domain. Then the operations such as frame skipping, requantization, video downscaling can be performed in this domain. The approach can provide an efficient way for bitrate reduction with low computational complexity. The major concern is that this is an homogeneous transcoding algorithm. However, the type of the incoming video is out of our control. In order to support different kinds of video devices and video formats, heterogeneous transcoding becomes important for mobile video applications.

Motivated by this, the main objective of this paper is to investigate these bottlenecks and propose possible solutions to alleviate the problems. The major problem is that when a B-picture is transcoded to a P-picture, the DCT coefficients are not available from the incoming bitstream, and the new P-picture, it is used immediately as a reference by the incoming frame. In other words, the next incoming P-picture needs to point to the previous frame instead of the original reference frame. In the next section, we will have a detailed analysis of this situation. As a consequence, the compressed-domain heterogeneous transcoding will be derived, under which the transcoding process is carried out in the DCT domain where complete decoding and re-encoding are not required; hence the processing complexity is significantly reduced as well as the quality

of the transcoded video can be preserved. In addition, a fast DCT-domain video transcoding is proposed by using the correlations between pictures which reflect the motion activities.

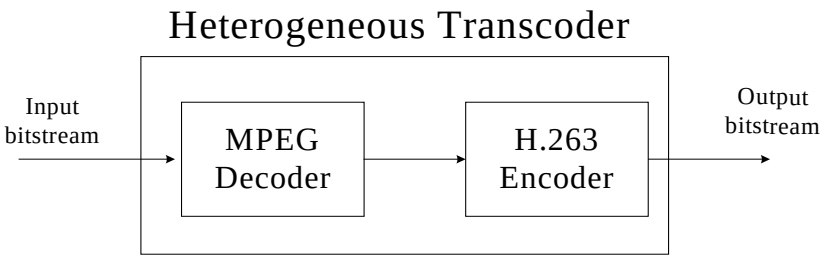


Figure 6.1. Conventional Heterogeneous transcoder

6.2. DCT-Based heterogeneous video transcoder

In [4], a heterogeneous transcoder is proposed to convert a B picture to a P picture. However, the major concern is that the transcoded DCT coefficients need to be recomputed in the pixel domain. In other words, re-encoding errors and high computational complexity are introduced during the transcoding process. In this section, we present a new DCT-based heterogeneous video transcoding architecture. The new architecture gives focus on the following areas.

6.2.1 Transcoding the B frame to a P frame by re-estimating the DCT coefficients and motion vectors in the DCT domain using direct addition of DCT coefficients and motion compensation in the DCT domain.

6.2.2 Fast algorithm for transcoding the incoming P frame with reference to the previous transcoded B frame using the approach of backward subtraction of DCT coefficients.

6.2.3 Fast algorithm for transcoding the B frame using uniform motion assumption and checking.

The architecture of the proposed transcoder is shown in Figure 6.2. The input bitstream is firstly parsed with a variable-length decoder to extract the header information, coding mode, motion vectors and quantized DCT coefficients for each macroblock. In the following, we will discuss the operations of the switches and buffers in details. The proposed DCT-based heterogeneous video transcder is used to transcode a video sequence from MPEG-2 to H.261 or H.263. Thus, we have to design efficient algorithms for transcoding among I-pictures, P-pictures and B-pictures. For standard codecs, a variety of transcoding formats exist. Even though our approach is useful to most of these formats; for the sake of simplicity, let us confine our discussion with a sample case. In the following, let us assume a group of pictures (GOP) in the incoming bitstream, which has a length of 8 pictures ($N=8$), the distance between the anchor I/P pictures is set to 4 pictures ($M=4$) and the format of the output picture sequence is either H.261 or H.263, with the sequence structures ($N=\infty$, $M=1$), as depicted in Figure 6.3. The superscript, $'$, is used to denote the picture after performing the transcoding. In Figure 6.3, pictures are presented in the display order, but are numbered in the encoding order. We consider these scenarios as two likely cases of the conversion, and the method can be generalized easily to other picture formats of the incoming and outgoing bitstreams. The following subsections will give our way to transcode the incoming bitstream in the DCT domain, the functions of the switches, the advantages of the DCT-domain buffer arrangement, together with the details of other methods.

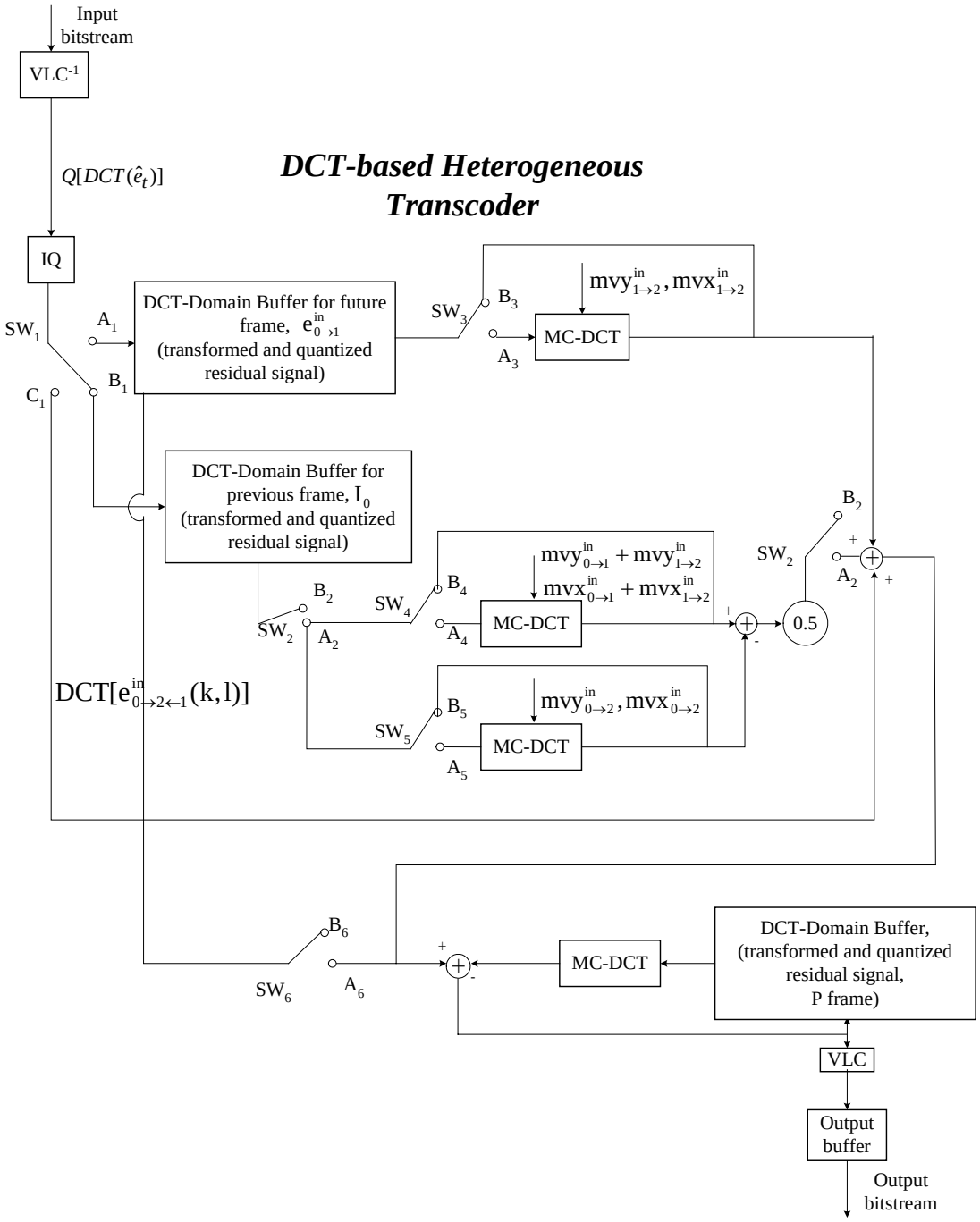


Figure 6.2. Architecture of the proposed heterogeneous video transcoder.

6.2.1 Re-estimate the DCT coefficients and motion vectors in the DCT domain

Let us consider to convert the first B-picture in a sub-group of an input bitstream, B_2, into a P-picture, $P_1^{'}$ which is predicted from frame I_0, as shown in Figure 6.3. The

new output motion vector, $mv_{0\to1}^{out}$, can be obtained directly from the forward motion vector of B_2, $mv_{0\to2}^{in}$, as depicted in Figure 6.4. However, the output prediction error in the DCT-domain, $DCT[e_{0\to1}^{out}]$, is not available from the incoming bitstream since the prediction error of incoming B_2 is derived from the previous I_0 and following P_1, such that:

$$\mathrm{e}_{0\to2\leftarrow1}^{\mathrm{in}}(\mathrm{k+j,l+i}) = \frac{\mathrm{B}_2(\mathrm{k+j,l+i}) - \mathrm{I}_0(\mathrm{k+j+mvy}_{0\to2}^{\mathrm{in}}, \mathrm{l+i+mvx}_{0\to2}^{\mathrm{in}})}{2} + \frac{\mathrm{B}_2(\mathrm{k+j,l+i}) - \mathrm{P}_1(\mathrm{k+j+mvy}_{1\to2}^{\mathrm{in}}, \mathrm{l+i+mvx}_{1\to2}^{\mathrm{in}})}{2} \quad (6.1)$$

where *(k,l)* represents the location of the upper left corner of a macroblock, *(i,j)* represents the spatial location within the macroblock, *mvx* represents the x-displacement of motion vector *mv* and *mvy* represents the y-displacement of motion vector *mv*.

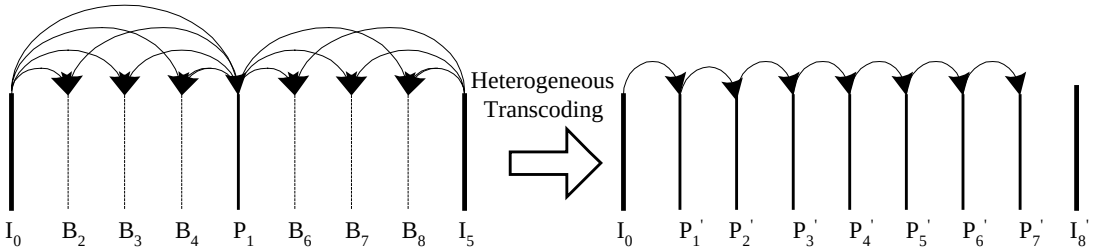


Figure 6.3: Typical heterogeneous transcoding.

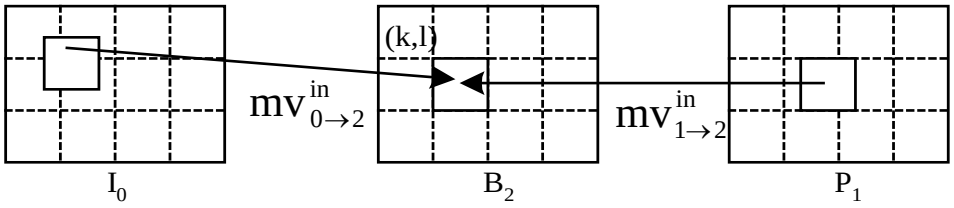


Figure 6.4: Forward and Backward motion vectors of B_2.

By applying the DCT to $e_{0\to2\leftarrow1}^{in}$ and taking into account of the linearity of the DCT, we obtain an expression for $e_{0\to2\leftarrow1}^{in}$ in the DCT-domain.

$$2DCT[e_{0\to2\leftarrow1}^{in}(k,l)] = DCT[B_2(k,l) - I_0(k+mvy_{0\to2}^{in}, l+mvx_{0\to2}^{in})] + DCT[B_2(k,l) - P_1(k+mvy_{1\to2}^{in}, l+mvx_{1\to2}^{in})] \quad (6.2)$$

From eqn (6.2), as it might be expected that $DCT[e_{0\to1}^{out}]$ cannot be obtained directly from the incoming bitstream. Let us make an approximation. Let the quantized DCT coefficients of the output prediction errors $DCT[e_{0\to1}^{out}]$ of frame 1 be given by the following equation,

$$DCT[e_{0\to1}^{out}(k,l)] = DCT[B_2(k,l) - I_0(k + mvy_{0\to1}^{out}, l + mvx_{0\to1}^{out})] \tag{6.3}$$

Using the incoming forward motion vector $mv_{0\to2}^{in}$ of the B frame as the estimated new motion vector $mv_{0\to1}^{out}$ of the transcoded P frame, we have

$mv_{0\to1}^{out} = mv_{0\to2}^{in}$, from eqn (6.2) and eqn (6.3), we obtain

$$\begin{aligned} 2DCT[e_{0\to1}^{out}(k,l)] &= 2DCT[B_2(k,l)] - 2DCT[I_0(k + mvy_{0\to1}^{out}, l + mvx_{0\to1}^{out})] \\ &= 2DCT[e_{0\to2\leftarrow1}^{in}(k,l)] + DCT[I_0(k + mvy_{0\to2}^{in}, l + mvx_{0\to2}^{in})] \\ &\quad + DCT[P_1(k + mvy_{1\to2}^{in}, l + mvx_{1\to2}^{in})] \\ &\quad - 2DCT[I_0(k + mvy_{0\to1}^{out}, l + mvx_{0\to1}^{out})] \end{aligned}$$

Therefore,

$$\begin{aligned} 2\mathrm{DCT}[\mathrm{e}_{0\to1}^{\mathrm{out}}(\mathrm{k},\mathrm{l})] &= 2\mathrm{DCT}[\mathrm{e}_{0\to2\leftarrow1}^{\mathrm{in}}(\mathrm{k},\mathrm{l})] \\ &\quad + \mathrm{DCT}[\mathrm{P}_1(\mathrm{k} + \mathrm{mvy}_{1\to2}^{\mathrm{in}}, \mathrm{l} + \mathrm{mvx}_{1\to2}^{\mathrm{in}})] \\ &\quad - \mathrm{DCT}[\mathrm{I}_0(\mathrm{k} + \mathrm{mvy}_{0\to2}^{\mathrm{in}}, \mathrm{l} + \mathrm{mvx}_{0\to2}^{\mathrm{in}})] \end{aligned} \tag{6.4}$$

By observing the prediction error of P_1 and making further manipulation of these equations, we can obtain an expression for the output prediction errors in the DCT-domain as shown below,

$$\begin{aligned} \mathrm{DCT}[\mathrm{e}_{0\to1}^{\mathrm{out}}(\mathrm{k},\mathrm{l})] &= \mathrm{DCT}[\mathrm{e}_{0\to2\leftarrow1}^{\mathrm{in}}(\mathrm{k},\mathrm{l})] \\ &\quad + \tfrac{1}{2}\mathrm{DCT}[\mathrm{e}_{0\to1}^{\mathrm{in}}(\mathrm{k} + \mathrm{mvy}_{1\to2}^{\mathrm{in}}, \mathrm{l} + \mathrm{mvx}_{1\to2}^{\mathrm{in}})] \\ &\quad + \tfrac{1}{2}\mathrm{DCT}[\mathrm{I}_0(\mathrm{k} + \mathrm{mvy}_{0\to1}^{\mathrm{in}} + \mathrm{mvy}_{1\to2}^{\mathrm{in}}, \mathrm{l} + \mathrm{mvx}_{0\to1}^{\mathrm{in}} + \mathrm{mvx}_{1\to2}^{\mathrm{in}})] \\ &\quad - \tfrac{1}{2}\mathrm{DCT}[\mathrm{I}_0(\mathrm{k} + \mathrm{mvy}_{0\to2}^{\mathrm{in}}, \mathrm{l} + \mathrm{mvx}_{0\to2}^{\mathrm{in}})] \end{aligned} \tag{6.5}$$

From eqn(6.5), we can see that the first term can be obtained directly from the incoming bitstream. The second term is the motion-compensated prediction error of the incoming bitstream and it can be computed mainly on the DCT-domain. By using the DCT-domain inverse motion-compensation proposed in [28], we can also obtain the third and fourth terms in the DCT-domain. The major idea to obtain the motion compensated DCT coefficients is to represent it as the sum of all horizontally and/or vertically displaced anchor blocks. Then the motion compensated DCT values of B_0', B_1', B_2' and B_3' (see Figure 6.5) are constructed using the pre-computed DCT values of the shift operators as mentioned in section 2.6.

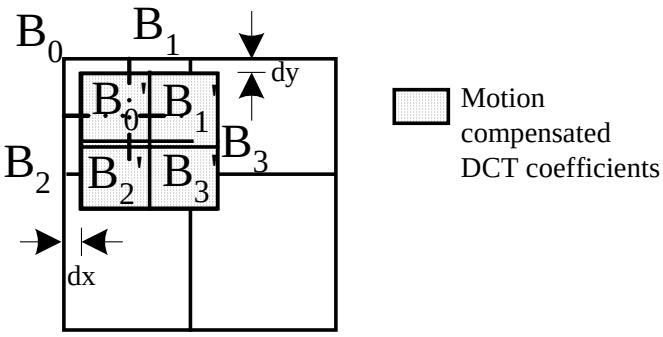


Figure 6.5. Motion compensated DCT coefficients and DCT coefficients from the incoming bitstream.

6.2.2 Fast transcoding

Since the P frame from the incoming bitstream refers to the previous I/P frame, it is necessary to transcode this P frame due to the fact that its previous B frame has been converted to the new P frame(see Figure 6.3). In this situation, we can employ the dominant motion vector selection scheme and MC-DCT to transcode this P frame.

For the non-motion compensated macroblock as shown in Figure 6.6, we propose to use the technique of direct subtraction of the DCT coefficients. Since the motion vectors are equal to zero in this case, we have

$mv^{in}_{1\to 2} = 0$, $mv^{in}_{0\to 2} = 0$ and $mv^{in}_{1\to 0} = 0$

Using eqn(6.2), we obtain

$$2DCT[e^{in}_{0\to 2\leftarrow 1}(k,l)] = DCT[B_2(k,l) - P_1(k,l)] + DCT[B_2(k,l) - I_0(k,l)] \quad (6.6)$$

Therefore, we have

$$DCT[B_2(k,l) - P_1(k,l)] = 2DCT[e^{in}_{0\to 2\leftarrow 1}(k,l)] - DCT[B_2(k,l) - I_0(k,l)] \quad (6.7)$$

In eqn(6.7), the first term can be obtained directly from the incoming bitstream. The second term is derived in the previous subsection and it can be computed mainly on the DCT-domain. For the motion compensated macroblock, we can employ the direct subtraction of DCT coefficients, DCT-domain inverse motion-compensation proposed in [28] and dominant motion vector selection techniques to transcode the DCT coefficients.

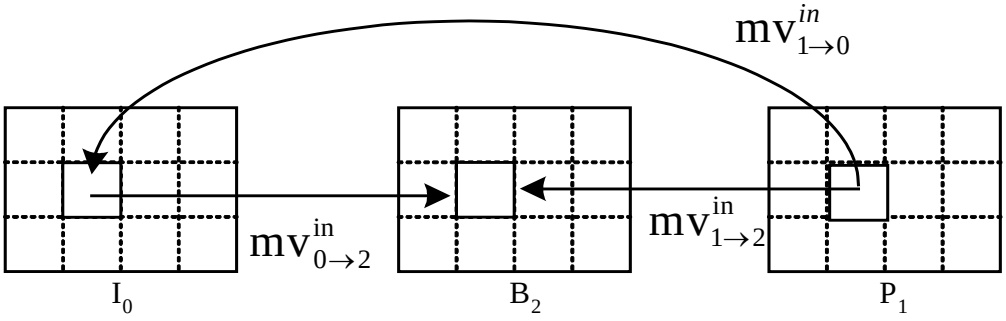


Figure 6.6. Forward and Backward motion vectors of $B_2$for non-motion compensated macroblocks.

6.2.3 Fast transcoding of the B frame using uniform motion assumption and checking.

In the previous section, we have successfully obtained the DCT coefficients in DCT domain. A further simplification can be done by an approximating the $DCT[e^{out}_{0\to 1}]$. This is done by assuming that motion activities between pictures are uniform(UMA), such that,

$$mv^{in}_{0\to 2} = mv^{in}_{0\to 1} + mv^{in}_{1\to 2} \quad (6.8)$$

Hence eqn.(6.5) can be further simplified as

$$DCT[e^{out}_{0\to 1}(k,l)] = DCT[e^{in}_{0\to 2\leftarrow 1}(k,l)] + \tfrac{1}{2} DCT[e^{in}_{0\to 1}(k + mvy^{in}_{1\to 2}, l + mvx^{in}_{1\to 2})] \quad (6.9)$$

In order to achieve high video quality for transcoding and low computational complexity, this assumption has to be verified. A process, designated as uniform motion checking(UMC), is applied to check how far the eqn (6.8) be valid. If the motion activities between pictures cannot fulfill this requirement, SW_2 will be switched to position A_2 to estimate the new DCT coefficients using eqn (6.5) instead of eqn (6.9) (see also Figure 6.2).

In the next section, the performance of using uniform motion assumption(UMA) and uniform motion checking(UMC) will be compared. Note that both approaches can obtain the targeted DCT coefficients in the DCT domain by reusing the incoming DCT coefficients in the above formulation. Using the above formulation, the architecture of the proposed transcoder is as shown in Figure 6.2.

In Figure 6.2, the video transcoder performs a variable-length decoding to extract the header information, coding mode, motion vectors and quantized DCT coefficients from the incoming bitstream. Switches SW_1, SW_2 and SW_6 are used to update the DCT-domain buffer for the transformed and quantized residual signal depending upon the coding mode originally used in the front encoder for the current macroblock being processed. SW_1 is used to update the DCT-domain buffer based on the frame type from the incoming bitstream. SW_2 is used to check the condition on uniform motion assumption(UMC) and SW_6 is used to convert the incoming P frame to a new reconstructed P frame since the reference frame is changed. Also, SW_3, SW_4 and SW_5 are used to control the MC-DCT modules. Functions of the switches are shown in Tables

6.1 to 6.4. Note that all operations are performed in the DCT-domain, full re-encoding process is not required. Also, MC-DCT modules will be deactivated for the Non-MC macroblock to speed up the transcoding process. Therefore, the computational complexity of the transcoder can be reduced significantly. In addition, quality degradation during the transcoding process can be avoided.

Table 6.1. Switch positions for different frame types.

Frame Type	SW_1 Position
Previous frame	B_1
Future frame	A_1
B	C_1

Table 6.2. Switch positions for using uniform motion assumption(UMA).

Uniform motion assumption(UMA)	SW_2 Position
Yes	B_2
No	A_2

Table 6.3. Switch positions for converting different frame type.

Incoming B frame	SW_6 Position
Yes	B_6
No	A_6

Table 6.4. Switch positions for different coding modes.

Coding mode	SW_3 Position	SW_4 Position	SW_5 Position
Non MC	B_3	B_4	B_5
MC	A_3	A_4	A_5

6.3. Experimental results

Extensive experiments have been performed to evaluate the overall efficiency of the proposed heterogeneous video transcoder. A fixed quantization parameter was used for all tested sequences, with I-B-P as the sequence structure(for M=3 and Intra

period=15). All B frames have to be converted into P frames inside the transcoder. These experiments aimed at evaluating the performances of the proposed techniques including (i) the DCT-domain transcoder with uniform motion assumption (DA+MCDCT+UMA), and (ii) the DCT-domain transcoder with motion checking(DA+MCDCT+UMC). The front encoder was employed to encode video sequences with different spatial resolutions and motion characteristics. "Salesman", "Foreman", "Carphone" in QCIF (176×144) containing low motion activities and "Table Tennis" and "Football" with size 352x240 containing high motion activities were encoded by an MPEG2 TM5 front encoder. For all testing sequences, the frame-rate of the incoming bitstream was 30 frames/s.

A large number of experimental exercises have also been done to compare the performance of our architecture with a conventional cascaded pixel-domain transcoder (CPDT) as shown in Figure 6.1. Detailed comparisons of the average PSNR between the CPDT and our proposed DCT-based transcoder using uniform motion assumption are given in Table 6.5. It shows that our proposed DCT-based transcoders outperform the CPDT in all cases.

In terms of video quality, the results are more significant for sequences with high motion activities because our proposed architecture transcodes the DCT coefficients in the DCT domain to achieve low computational complexity and reduce re-encoding errors. Significant improvement can be achieved which is about 2-2.3dB as compared with the conventional video transcoder. In terms of computational complexity, the speed up factor is about 4.5 to 7.3 times as compared with the conventional video transcoder[2]. Significant improvement can be achieved especially for the video sequences with low

motion activities since MC-DCT is not required. In other words, only addition / subtraction of DCT coefficients in the DCT domain are performed to achieve low computational complexity and reduce the re-encoding error during the transcoding process.

For sequences with low motion activities such as salesman and carphone sequences, the motion vectors are small. Therefore, the MC-DCT module can be deactivated in these cases. Hence, significant improvement can then be achieved, which is about 5.6 to7.3 times as shown in Table 6.5 For "Football" and "Foreman" sequences, the average PSNR and speed up also have significant improvement as compared with the conventional approaches. It is due to the fact that re-encoding process is performed in the pixel domain[35]. Full decoding and re-encoding processes are required for the conventional pixel domain transcoder. Hence, the re-encoding errors become significant to transcode these video sequences. Since the transcoding process performs mainly in the DCT domain, quality degradation can be avoided. The average PSNR improvement has been found to be about 2.1-2.3dB as shown in Table 6.5

Table 6.5 Performance comparison for all transcoded B-frames using the proposed DCT-based heterogeneous transcoder with uniform motion assumption (UMA) and conventional video transcoder

Sequences	Input bitrate	CPDT	DA+MCDCT+UMA	
		Average PSNR	Average PSNR	Speed-up ratio as compared with CPDT
Salesman (176x144)	64k	33.34	35.61	6.89
	128k	36.78	39	7.33
Foreman (176x144)	64k	30.53	32.69	4.65
	128k	34.28	36.56	4.85
Carphone (176x144)	64k	32.11	34.15	5.55
	128k	34.74	36.93	5.72
Table Tennis (352x240)	1.5M	32.41	34.54	5.06
	3M	35.04	37.33	5.30

Football (352x240)	1.5M	30.18	32.32	4.47
	3M	33.89	36.20	4.73

Table 6.6 also compares the average PSNR and complexities of our proposed transcoders using uniform motion assumption: DA+MCDCT+UMA and our proposed transcoder using uniform motion checking named as DA+MCDCT+UMC. As shown in Table 6.6, DA+MCDCT+UMA has similar performance with the DA+MCDCT+UMC in terms of the computational complexity. Note that the quality can be further increased. Therefore, DA+MCDCT+UMC is more suitable for high quality video transcoding application.

In Table 6.6, it can be seen that DA+MCDCT+UMA has a slight PSNR degradation over DA+MCDCT+UMC. This result is expected since not all the motion vectors obey eqn(6.6). For video sequences with low motion activities, eqn(6.6) is satisfied for most macroblocks. Therefore, DA+MCDCT+UMA works well. However, it is beneficial to check this assumption to avoid unnecessary quality degradation especially for transcoding the video sequences containing high motion activities. In the "Salesman" and Carphone" sequences, the speed up performance is about 3.3 to 6.3, since most of the macroblocks are coded in non-MC mode. And the average PSNR performance is very close as compared with the case with DA+MCDCT+UMA. For the "Football" and "Foreman" sequences, the average PSNR performance can be further improved which is about 0.53-0.6dB. The speed up performance is about 2.2-2.5 times.

Table 6.6. Performance comparison for all transcoded B-frames using the proposed DCT-based heterogeneous transcoder with uniform motion assumption (UMA) and uniform motion checking(UMC)

Sequences	Input bitrate	DA+MCDCT+UMA	DA+MCDCT+UMC	
		Average PSNR	Average PSNR	Speed-up ratio as compared with CPDT
Salesman (176x144)	64k	35.61	35.62	5.30
	128k	39	39.01	6.33
Foreman (176x144)	64k	32.69	33.22	2.32
	128k	36.56	37.1	2.49
Carphone (176x144)	64k	34.15	34.34	3.22
	128k	36.93	37.08	3.43
Table Tennis (352x240)	1.5M	34.54	34.78	2.69
	3M	37.33	37.54	2.94
Football (352x240)	1.5M	32.32	32.91	2.16
	3M	36.20	36.80	2.38

6.4. Conclusion

In this chapter, we have proposed a new architecture for a low complexity and high quality heterogeneous video transcoder to convert a B-picture into a P-picture. We have derived a set of equations and formulated the problem of how to obtain the DCT coefficients. By using the motion vector and the DCT coefficients from the incoming bitstream, the proposed video transcoder can transcode the B-picture into a P-picture in the DCT domain. In addition, a fast algorithm to transcode a B frame into a P frame using backward subtraction of DCT coefficients is proposed. The low computational complexity is achieved by performing video transcoding in the DCT-domain 1) using the techniques relating to motion compensation in the DCT domain and 2) an indirect addition of DCT coefficients to re-estimate DCT coefficients in the DCT domain. Besides, we have proposed a fast algorithm to further speed up the transcoding process in

the DCT domain using correlations which can reflect motion activities between pictures. Since the transcoding process is performed in the DCT domain, it is shown that re-encoding errors can be reduced significantly. The overall performance of the proposed architecture produces a picture with a quality better than that of the conventional video transcoder. Experimental results confirm that the proposed video transcoder achieves better performance as compared to the conventional video transcoder in terms of both quality and complexity.

Chapter 7

H.264 video transcoding

7.1 Low Complexity H.263 to H.264 video transcoding using motion vector decomposition

7.1.1 Introduction

The H.264 adopts various block types and multiple reference frames for motion compensation. For transcoding a video sequence from the H.263 format to the H.264 format, it is beneficial to reuse as much information as possible in the original sequence. It is always a good idea to design a video transcoder which aims at achieving a transcoded video sequence with good quality and lower bitrate, and without introducing high computational complexity during the transcoding process. In this chapter, an effective motion vector decomposition algorithm is introduced for transcoding videos from the H.263 format to the H.264 format. The algorithm concentrates on macroblocks which consume more bits in the H.263 video sequence. In other words, a non-optimal motion vector is splitted into smaller sizes since the H.264 support various block types. Using the proposed prediction error measure and diversity of the motion vector measure, the motion vector can be decomposed adaptively and the amount of the prediction error can be minimized. After re-estimating the new set of motion vectors, multiple reference frames estimation is applied to further reduce the bitrate and prediction error. Since the non-optimal motion vector can easily be identified inside the proposed transcoder, it is effective to improve the transcoding efficiency to avoid computational redundance. The proposed video transcoder can reduce substantially the amount of computation and

provide transcoded videos with better quality as compared with those obtained from the conventional cascaded video transcoder.

7.1.2 Motion Vector Decomposition for H.263 to H.264 Video transcoder

The proposed video transcoding architecture consists of an H.263 decoder and an H.264 encoder. The primary objective of the proposed video transcoder is to achieve a low computational complexity to produce a similar video quality of the transcoded video sequence with reduced bitrate requirement. A motion vector decomposition scheme is used instead of using motion estimation inside the transcoder as shown in Figure 7.1. In other words, no full motion estimation will be performed during the transcoding process.

The input bitstream is firstly parsed with a variable-length decoder to extract the header information, coding mode, motion vectors and quantized DCT coefficients for each macroblock. After decoding the incoming H.263 bitstream, a motion vector decomposition scheme is used to speed up the re-encoding process. The motion vector decomposition algorithm as shown in Figure 7.1 is used to re-estimate a new set of motion vectors. This is done by checking if the incoming motion vector be optimally estimated. In an ideal case, the optimal motion vector can minimize the amount of the prediction errors. However, it is difficult to obtain an optimal motion vector when the block consists of two objects as shown in Figure 7.2. In this case, the prediction error is large.

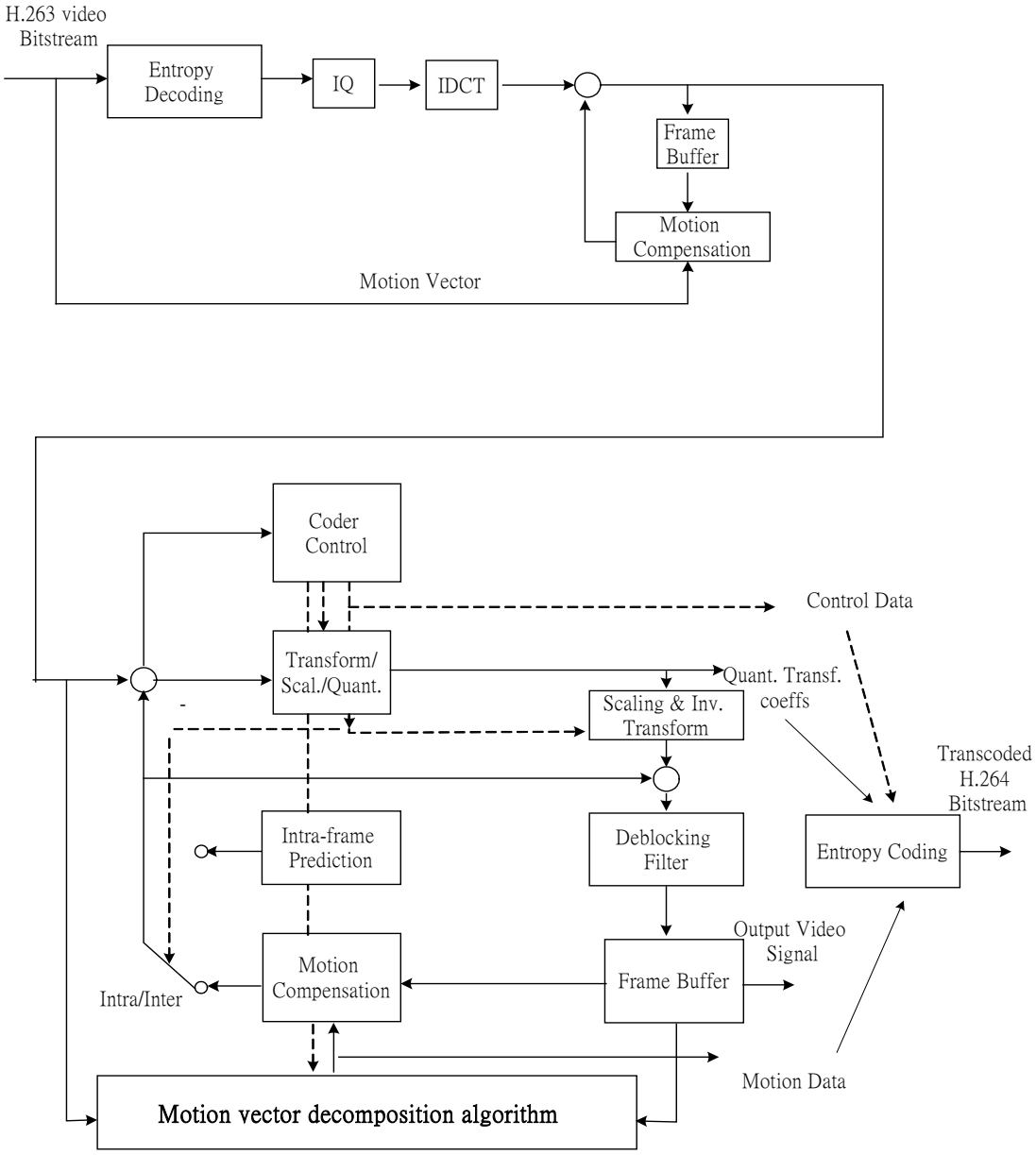


Figure 7.1. The proposed video transcoder for H.263 to H.264

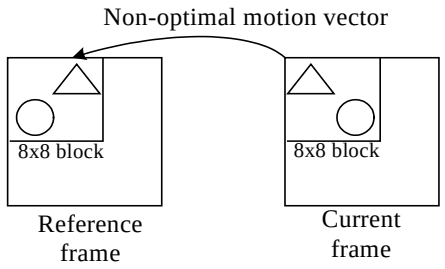


Figure 7.2. The non-optimal motion vector in H.263 video

Motivated by this, the architecture of our video transcoder is to aim at reducing the bitrate of the encoded video by using the motion vector decomposition which can minimize the amount of the prediction error. Let us define the average approximation of the prediction error, AAPerror, in the incoming frame as

$$AAPerror = \frac{1}{N}\sum_{i=0}^{N-1} DCT_i \tag{7.1}$$

where DCT_i represents the sum of the non-zero DCT coefficients in block i and N represents the total number of blocks in the incoming frame. The motion vector will be decomposed for a block which contains a large amount of prediction errors as defined in the following equation:

$$\frac{DCT_i}{AAPerror} > 1 \tag{7.2}$$

From eqn.7.2, if the approximation of the prediction error of the required block is larger than the average prediction error of the incoming frame, motion vector decomposition will be applied since the original incoming motion vector may not be a good motion vector to represent the block in the H.264 standard. In other words, the original motion vector from the incoming bitstream will be decomposed as shown in Figure 7.3. Then, a motion vector refinement will be applied for the decomposed motion vectors with a 5x5 refinement window size as shown in Figure 7.3. Let us define the diversity measure of a block i as

$$Diversity_i = | Mv_{hi} - \hat{Mv}_h | + | Mv_{vi} - \hat{Mv}_v | \tag{7.3}$$

where Mv_{hi} and Mv_{vi} represent the horizontal and vertical components of the motion vector of block i(i=0 to 3), respectively. $\hat{M}v_h$ and $\hat{M}v_v$ represent the averages of the horizontal and vertical components of all motion vectors, respectively.

Then, a block, block i, with high diversity is defined as

$$\frac{Diversity_i}{\frac{1}{N}\sum_{i=0}^{N-1} Diversity_i} > 1 \tag{7.4}$$

After the refinement of motion vectors, we measure the diversity of the new set of motion vectors using eqn.7.4. If the diversity is high, a further motion vector decomposition is applied. In other words, the motion vector will be splitted into four motion vectors with a smaller size again(e.g size of 4x4). Otherwise, the motion vector re-estimation with various block sizes is completed. After obtaining the new set of motion vectors with various block sizes, multiple reference frame estimation will be performed.

In the transcoding process from H.263 to H.264, if the diversity is low for the new set of the motion vectors, it implies that the obtained motion vectors are close the optimal solution. Therefore, further decomposition of the motion vectors will not lead to a significant saving in terms of bitrate reduction. In other words, high diversity implies the block may contain more than one objects. Further decomposition allows the encoder to choose a more suitable block size to represent the objects. After determining the various block sizes, multiple reference frames estimation allows the transcoder to have one more dimension to optimize the bitrate. This feature also allows the transcoder to find a more suitable reference frame in order to minimize the prediction error as much as possible. Note that the new set of the motion vectors consumes more bits and has higher prediction

error as compared with other regions. Motivated by this, the proposed transcoder performs the multiple reference frames estimation only on the new set of the motion vectors to further speed up the transcoding process.

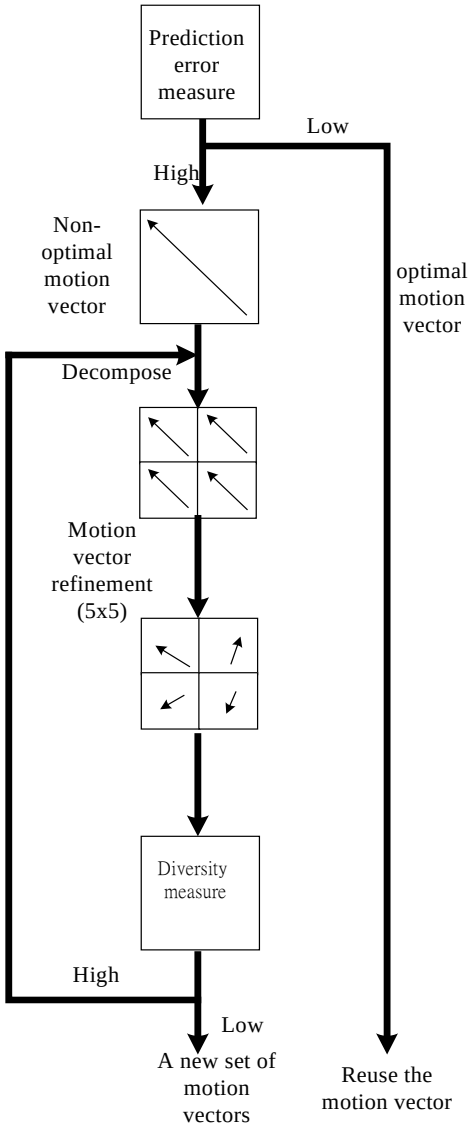


Figure 7.3. The flow diagram of motion vector decomposition

7.1.3 Experimental results

Extensive simulations and performance comparison have been done with reference to a cascaded pixel-domain transcoder using the full search motion re-

estimation. The cascaded transcoder was implemented using an H.263 video codec, an H.264 video codec, a variable block-based ME/MC module, with motion vector search range of $\pm$16, 4x4 integer DCT and 3 reference frames. In the front encoder, the first frame was encoded as an intraframe (I-frame), and the remaining frames were encoded as interframes (P-frames). Results of our experimental work show that the proposed video transcoders outperform the conventional cascaded video transcoder in terms of computational complexity as shown in Table 7.1. The proposed video transcoding architecture can achieves about 3 to 8 times speed up in the motion re-estimation process with a similar transcoded video quality. The results are more significant for sequences with low motion activities because the proposed transcoder provides a significant simplification in computational complexity during the motion re-estimation process. For a sequence with low motion activities such as salesman, many of the motion vectors are near optimal. The proposed motion vector decomposition scheme can achieve similar performance in terms of the transcoded video quality as compared with the conventional video transcoder using full search but with speed up ratio about 8 times. This is because the proposed motion vector decomposition scheme allows to identify the possible locations of a new set of motion vectors during the H.264 transcoding process to avoid the redundant motion re-estimation. For a sequence with high motion activities, only less than 0.3dB of quality degradation is introduced as compared with the conventional video transcoder using the full search motion re-estimation. However, in this case the speed up of the transcoder process is about 3 times as compared with the conventional video transcoder.

Table 7.1. Average PSNR of the proposed transcoder, where the frame rate of the incoming bitstream was 30 frames/s. H.263 was used as the front encoder for encoding "Salesman", "Miss_America", "Hall", "Tennis", "Football" and "Flower".

Sequences	Input bitrate	Average PSNR difference as compared with the cascaded video transcoder using full search motion re-estimation.	Speed up ratio as compared with the conventional cascaded video transcoder using full search motion re-estimation
Salesman (352x288)	512k	0.08	8.1
	256k	0.07	8.2
Miss_ America (352x288)	512k	0.119	7.7
	256k	0.09	7.8
Hall (352x288)	512k	0.13	6.6
	256k	0.10	6.7
Tennis (352x240)	3M	0.15	5.9
	1.5M	0.11	5.9
Flower (352x240)	3M	0.22	4.2
	1.5M	0.19	4.2
Football (352x240)	3M	0.25	3.3
	1.5M	0.30	3.4

7.1.4 Conclusion

In this section, we have proposed a simple architecture to form a low-complexity and high quality video transcoder to speed up the motion re-estimation for H.263 to H.264 video transcoding. Using the approximation of the prediction error from the incoming bitstream, the proposed transcoder is able to detect whether the performance of the incoming motion vector can be improved during the motion re-estimation process. By considering the diversity of the new set of the estimated motion vectors, the proposed video transcoder can control the decomposition of the motion vector dynamically, which

can achieve low computational complexity. Besides, the multiple reference frames estimation is applied to a region which contain higher prediction error to further reduce the computational complexity during the multiple reference frames estimation. Results of our experimental work show that the proposed architecture produces similar pictures quality with low computational complexity as compared with the conventional video transcoder.

7.2 Compressed domain DCT to integer transform conversion

7.2.1 Introduction

The H.264 video coding standard has been filed recently[46,47]. This new generation video standard, with its significant bandwidth savings, is expected to replace the H.263 for video compression in digital video systems[34]. Compared with the H.263 video format, the H.264 video format can provide equivalent video at 1/3 to 1/2 of the H.263 bitrates. In spite of its complexity, the actual bandwidth savings compared to the H.263 provides a big motivation to migrate to H.264 video coding. The standard is intended for use in low bitrate applications including mobile phones, moderate bitrate applications such as video conferencing, and high bitrate applications such as digital TV. The complete migration to the new video coding algorithm will take several years since H.263 and MPEG are widely used in many multimedia applications nowadays. This creates an important need for transcoding technologies[1,12-18,35,48,49,73] that convert the widely available H.263 compressed videos to H.264 compressed format and vice versa. However, given the significant differences between the H.263 and the H.264 algorithms, transcoding is much more complex. Figure 7.4 shows the conventional cascaded video transcoder. In [73], we have proposed a motion vector decomposition

algorithm to speed up the motion re-estimation during the H.263 to H.264 video transcoding process. The major concern is that the integer transform coefficients need to be re-computed[66]. In other words, high computational complexity and re-encoding errors are introduced during the re-estimation of the transformed coefficients. In this chapter, a new compressed domain video transcoder architecture is proposed to transcode the DCT coefficients to the integer transform coefficients in the compressed domain. Figure 7.5 shows the diagram of DCT coefficients to integer transform coefficients conversion for intra frame video transcoding. Figure 7.6 shows the conventional approach for transcoding the transformed coefficients in the pixel domain. The input is a block of 8x8 2D-DCT coefficients, A. An inverse DCT is applied to A to recover the 8x8 pixel block. The 8x8 pixel block is divided into four 4x4 blocks (pixel domain $data_i$ i=0,1,2,3). Each of the four blocks is passed to a corresponding Integer transform to generate four 4x4 blocks of transform coefficients. These four blocks of transform coefficients are combined to form a single 8x8 block. This process introduces high computational complexity as well as re-encoding errors. It is desired to perform the transcoding in the compressed domain to avoid the complete decoding and re-encoding process. For the interframe video transcoding, since H.264 support variable block sizes for motion compensation[67-72], the motion vector may be deviated from the original incoming bitstream as shown in Figure 7.7. Motivated by these problems, two sets of operators are derived to transcode the DCT coefficients to Integer transform coefficients without motion compensation and with motion compensation for intra frame and inter frame transcoding. To further speed up the transcoding process, a fast approximation of

the DCT coefficients is used. Since the operators contain floating point implementation, an integer approximation form of operators are derived for easy implementation.

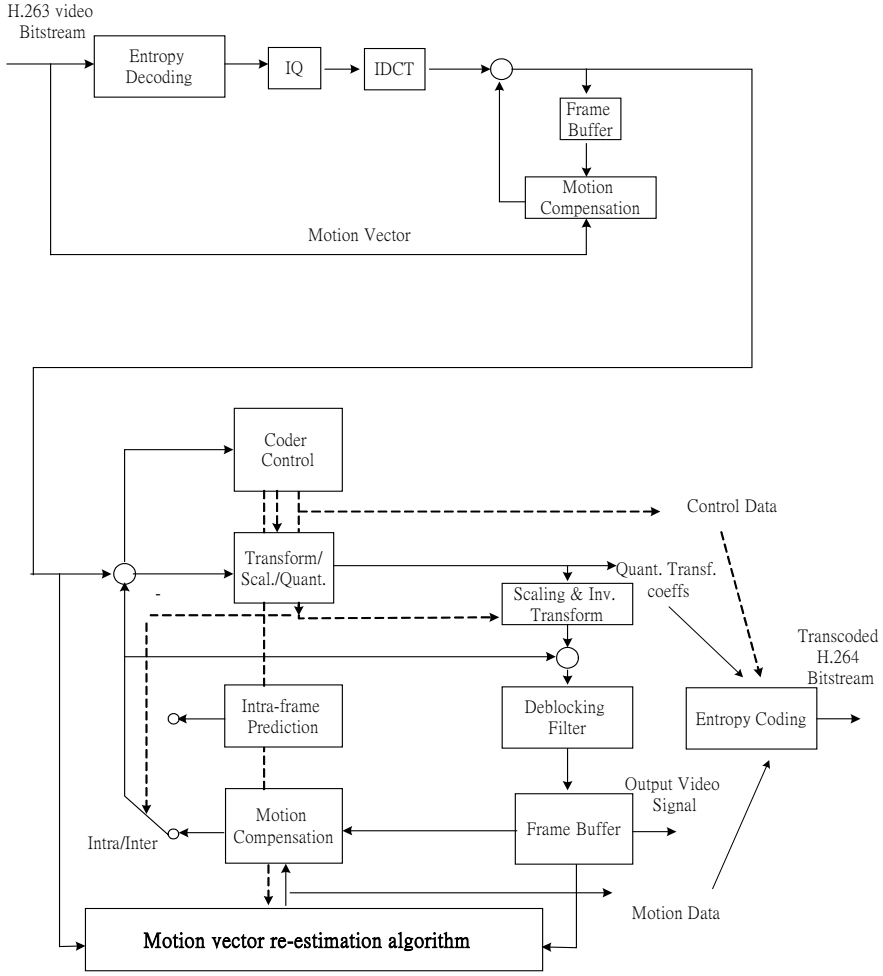


Figure 7.4. Conventional cascaded video transcoder for H.263 to H.264

Figure 7.5. Intra block transcoding from H.263 to H.264/AVC

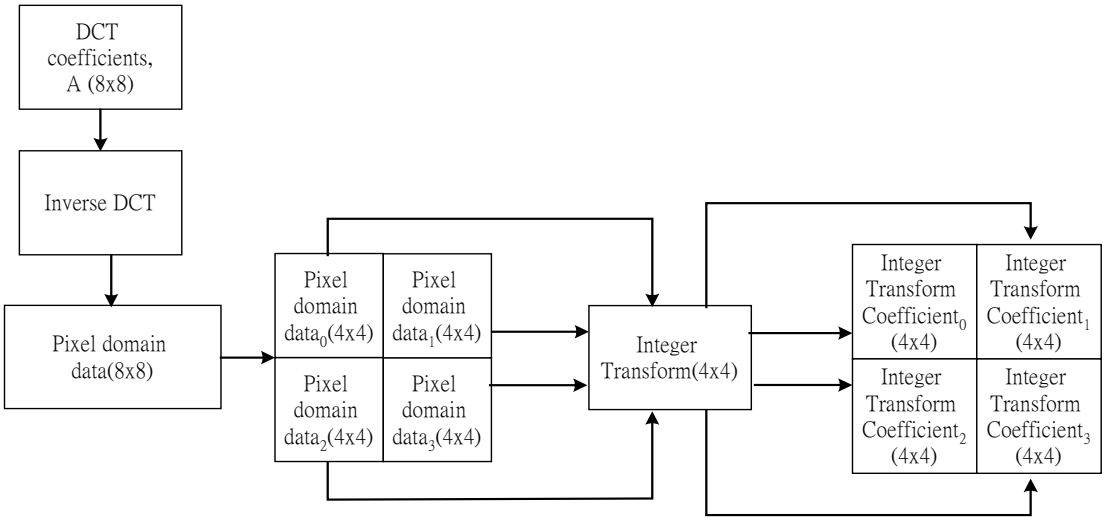


Figure 7.6. Conventional approach for transcoding the transformed coefficients in the pixel domain.

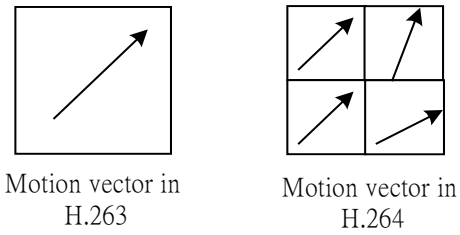


Figure 7.7. Motion vector in H.263 and H.264

7.2.2 Compressed domain video transcoder for H.263 to H.264

In [73], we have proposed a low complexity H.263 to H.264 video transcoding using motion vector decomposition to avoid full motion re-estimation process. In this section, we present a new H.263 to H.264 transcoding architecture which is an extension of the work of [73]. The new architecture has four new features:

(1) Formulation of the proposed operators for transcoding the DCT coefficients to Integer transform coefficients;

(2) Integer approximation form of operators for easy implementation;

(3) Motion compensated integer transform coefficients for inter frame video transcoding;

(4) Fast approximation of DCT coefficients for H.263 to H.264 video transcoding in low bitrate application.

The architecture of the proposed transcoder is as shown in Figure 7.8. In Figure 7.8, the video transcoder performs an entropy decoding to extract the header information, coding mode, motion vectors and quantized DCT coefficients from the incoming bitstream. The motion vectors information will be used to speed up the motion vector re-estimation process. For intra mode video transcoding, the DCT coefficients will be transcoded into the integer transform coefficients directly in the compressed domain using the proposed operators. For inter frame video transcoding, motion compensated integer transform module will be activated to transcode our targeted integer transform coefficients from the DCT coefficients. For low bitrate application, approximation of DCT coefficients will be activated to further speed up the transcoding process. For low processing power devices, integer approximation form of the operator will be activated for easy implementation. The formulation of the proposed operators, together with the details of other methods, are described in the following subsections.

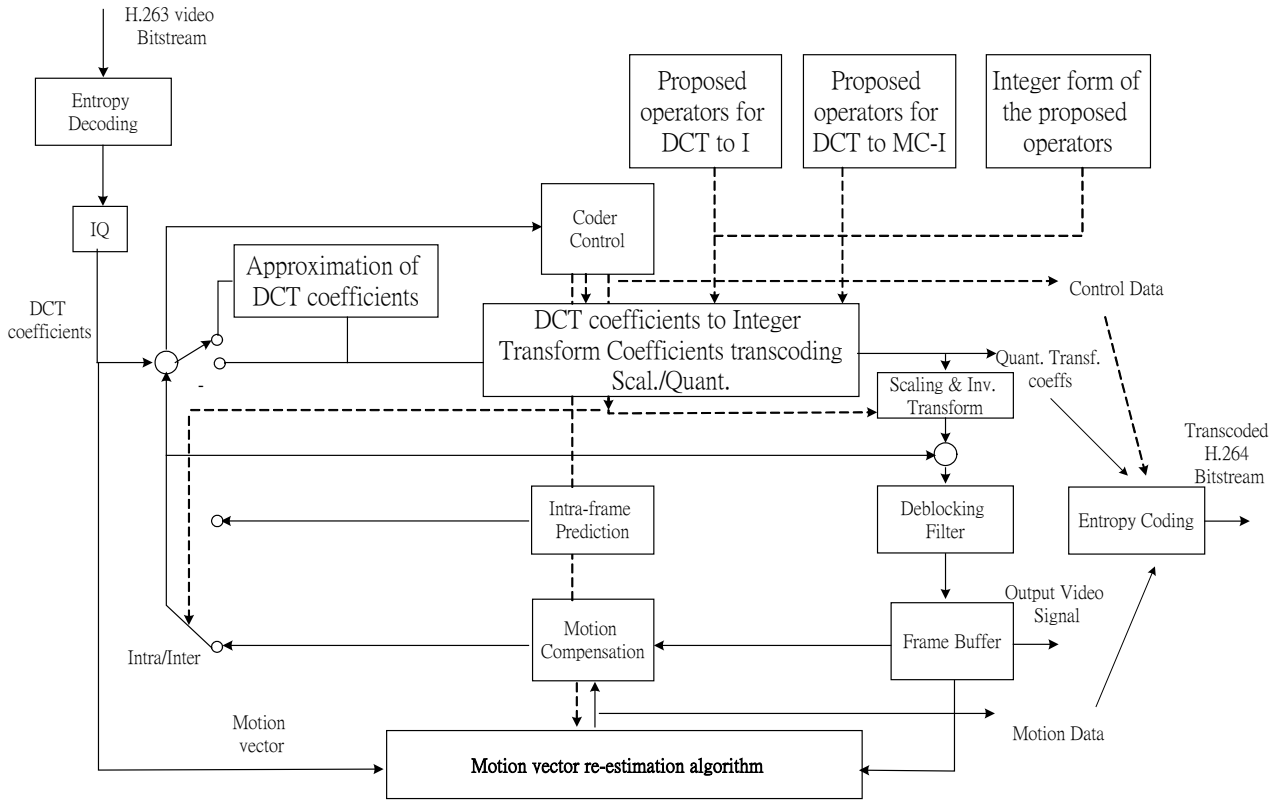


Figure 7.8. The proposed compressed domain video transcoder for H.263 to H.264

A. Formulation of the proposed operators for transcoding the DCT coefficients to Integer transform coefficients

In Figure 7.6, we have shown that high computational complexity is required for transcoding the DCT coefficients to Integer transform coefficients in the previous section. In this section, we propose some operators and reuse the incoming DCT coefficients such that we can transcode the integer transform coefficients in the compressed domain to reduce the computational complexity and re-encoding error. Let us recall the definition of the 2D-DCT of an 8x8 block, A, as shown below,

$$DCT(A) = \hat{A} = TAT^t \tag{7.5}$$

where T is an 8x8 DCT matrix with entries t(i,j) given by

$$t(i,j)=\frac{1}{2}k(i)\cos\frac{(2j+1)i\pi}{16}$$

i represents the row index, j represents the column index and

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

The 2D Integer Transform(IT) of a 4x4 block, B, can be defined as

$$IT(B)=IBI^t\otimes \text{ (scaling factor1)} \tag{7.6}$$

where I is a 4x4 IT matrix given by

$$I=\begin{bmatrix}1 & 1 & 1 & 1\\ 2 & 1 & -1 & -2\\ 1 & -1 & -1 & 1\\ 1 & -2 & 2 & -1\end{bmatrix}$$

$$scaling\quad factor1=\begin{bmatrix}a^2 & ab/2 & a^2 & ab/2\\ ab/2 & b^2/4 & ab/2 & b^2/4\\ a^2 & ab/2 & a^2 & ab/2\\ ab/2 & b^2/4 & ab/2 & b^2/4\end{bmatrix}=\begin{bmatrix}0.25 & 0.158 & 0.25 & 0.158\\ 0.158 & 0.1 & 0.158 & 0.1\\ 0.25 & 0.158 & 0.25 & 0.158\\ 0.158 & 0.1 & 0.158 & 0.1\end{bmatrix}$$

for a=0.5, $b=\sqrt{\frac{2}{5}}$ and

$\otimes$ represents the term by term multiplication.

Similarly, the inverse 2D transform is given by:

$$B'=J[I(B)\otimes \text{scaling factor2}]J^t \tag{7.7}$$

where J is a 4x4 inverse IT matrix given by

$$J=\begin{bmatrix}1 & 1 & 1 & 0.5\\ 1 & 0.5 & -1 & -1\\ 1 & -0.5 & -1 & 1\\ 1 & -1 & 1 & -0.5\end{bmatrix}$$

and $$scaling\ \ factor2=\begin{bmatrix}a^2 & ab & a^2 & ab\\ ab & b^2 & ab & b^2\\ a^2 & ab & a^2 & ab\\ ab & b^2 & ab & b^2\end{bmatrix}=\begin{bmatrix}0.25 & 0.316 & 0.25 & 0.316\\ 0.316 & 0.4 & 0.316 & 0.4\\ 0.25 & 0.316 & 0.25 & 0.316\\ 0.316 & 0.4 & 0.316 & 0.4\end{bmatrix}$$

In the following, we will derive a set of equations to transform the DCT coefficients, DCT(A), into the integer transform coefficients directly in the compressed domain. A similar formulation can be obtained for transcoding the integer transform coefficients to DCT coefficients.

Applying the inverse DCT to the DCT coefficients, DCT(A), we have

$$\text{pixel domain data}=T^t[DCT(A)]T \qquad \text{where T is an 8x8 DCT matrix} \qquad (7.8)$$

In order to satisfy the H.264 format, let us divide this reconstructed result into four groups. These four groups of data in the pixel domain can be extracted by using the proposed operators S_{i1} and S_{i2}, i.e., we have

$$\text{Pixel domain data}_i=S_{i1}T^t[DCT(A)]TS_{i2} \qquad \text{for i=0,1,2,3} \qquad (7.9)$$

where $$s_{01} = s_{11} = s_{02}^{\ T} = s_{22}^{\ T} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$s_{21} = s_{31} = s_{12}^{\ T} = s_{32}^{\ T} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

and i represents the index of the 4 groups of the integer transform coefficients.

Note that A is with a size of 8x8 and the integer transform coefficients is 4x4. Therefore, one set of DCT coefficients will give four sets of the integer transform coefficients.

Then we can apply the integer transform to the pixel domain $data_i$. We have

$$\text{Integer transform coefficients}_{\ i} = I\,(\text{pixel domain } data_i)\, I^t \otimes (\text{scaling factor 1}) \quad (7.10)$$

Combining equations (7.8-7.10), we have

$$\text{Integer transform coefficients}_{\ i} = I(S_{i1}T^t DCT(A) T S_{i2}) I^t \otimes (\text{scaling factor 1}) \quad (7.11)$$

Let us define the pre-computed operators, $P_{1i}=I*S_{i1}*T^t$ and $P_{2i}=T*S_{i2}*I^t$,

Therefore, we have

$$\text{Integer transform coefficients}_{\ i} = P_{1i}*[DCT(A)]*P_{2i} \otimes (\text{scaling factor 1}) \quad (7.12)$$

where $P_{10}=P_{11}=$ $\begin{bmatrix} 1.4142 & 1.2815 & 0 & -0.4500 & 0 & 0.3007 & 0 & -0.2549 \\ 0 & 0.9236 & 2.2304 & 1.7799 & 0 & -0.8638 & -0.1585 & 0.4824 \\ 0 & -0.1056 & 0 & 0.7259 & 1.4142 & 1.0864 & 0 & -0.5308 \\ 0 & 0.1169 & 0.1585 & -0.0922 & 0 & 1.0379 & 2.2304 & 1.9750 \end{bmatrix}$,

$$P_{12}=P_{13}=\begin{bmatrix} 1.4142 & -1.2815 & 0 & 0.4500 & 0 & -0.3007 & 0 & 0.2549 \\ 0 & 0.9236 & -2.2304 & 1.7799 & 0 & -0.8638 & 0.1585 & 0.4824 \\ 0 & 0.1056 & 0 & -0.7259 & 1.4142 & -1.0864 & 0 & 0.5308 \\ 0 & 0.1169 & -0.1585 & -0.0922 & 0 & 1.0379 & -2.2304 & 1.9750 \end{bmatrix},$$

$$P_{20}=P_{22}=\begin{bmatrix} 1.4142 & 0 & 0 & 0 \\ 1.2815 & 0.9236 & -0.1056 & 0.1169 \\ 0 & 2.2304 & 0 & 0.1585 \\ -0.4500 & 1.7799 & 0.7259 & -0.0922 \\ 0 & 0 & 1.4142 & 0 \\ 0.3007 & -0.8638 & 1.0864 & 1.0379 \\ 0 & -0.1585 & 0 & 2.2304 \\ -0.2549 & 0.4824 & -0.5308 & 1.9750 \end{bmatrix}$$

and $$P_{21}=P_{23}=\begin{bmatrix} 1.4142 & 0 & 0 & 0 \\ -1.2815 & 0.9236 & 0.1056 & 0.1169 \\ 0 & -2.2304 & 0 & -0.1585 \\ 0.4500 & 1.7799 & -0.7259 & -0.0922 \\ 0 & 0 & 1.4142 & 0 \\ -0.3007 & -0.8638 & -1.0864 & 1.0379 \\ 0 & 0.1585 & 0 & -2.2304 \\ 0.2549 & 0.4824 & 0.5308 & 1.9750 \end{bmatrix}$$

Note that P_{1i} and P_{2i} can be pre-computed. Therefore, low computational complexity can be achieved since the transformation is done in the compressed domain. In other words, no complete decoding and re-encoding process are required to obtain the integer transform coefficients from DCT coefficients.

B. Integer approximation form of operators for easy implementation

In the previous subsection, we have derived a set of operators to transcode the DCT coefficients to integer transform coefficients. The major concern is that floating point operations are more expensive to implement than integer operations in low processing power devices. Motivated by this, it is beneficial to make an approximation form of the derived operators in the previous subsection. It is obvious that we can

multiply the proposed operators by a large integer number such that the proposed operators become integer. If the selected integer is a power of 2, the resulting coefficients can be scaled down by a proper shifting operation. It implies that the shifting operations can be absorbed in the quantization process during the transcoding. i.e. no extra operations are required in this approximation process. Therefore, the larger number of the power of 2 integer is selected, the higher of the accuracy can be achieved. In general, 32 bit arithmetic is the capability of most of the signal processors. For DCT to Integer transform conversion, the DCT coefficients have a dynamic range of 4096(256(pixel intensity)x16(2D-DCT elements)). Hence a wordlength of 12 bits is required to represent it. The maximum sum of absolute values of our proposed operators is 6.44. So, the maximum dynamic range gain for the 2D operator is 41.47 which requires 6 bits to represent it. Therefore, the scaling factor we can select is the square root of $2^{32-6-12}=2^{7}$, i.e.128. Hence, the approximation form of the proposed operators becomes:

$$P_{10}{}^{I}=P_{11}{}^{I}=\begin{bmatrix} 181 & 164 & 0 & -58 & 0 & 38 & 0 & 33 \\ 0 & 118 & 285 & 228 & 0 & -111 & -20 & 62 \\ 0 & -14 & 0 & 93 & 181 & 139 & 0 & -68 \\ 0 & 15 & 20 & -12 & 0 & 133 & 285 & 253 \end{bmatrix},$$

$$P_{12}{}^{I}=P_{13}{}^{I}=\begin{bmatrix} 181 & -164 & 0 & 58 & 0 & -38 & 0 & 33 \\ 0 & 118 & -285 & 228 & 0 & -111 & 20 & 62 \\ 0 & 14 & 0 & -93 & 181 & -139 & 0 & 68 \\ 0 & 15 & -20 & -12 & 0 & 133 & -285 & 253 \end{bmatrix},$$

$$P_{20}{}^{I} = P_{22}{}^{I} = \begin{bmatrix} 181 & 0 & 0 & 0 \\ 164 & 118 & -14 & 15 \\ 0 & 285 & 0 & 20 \\ -58 & 228 & 93 & -12 \\ 0 & 0 & 181 & 0 \\ 38 & -111 & 139 & 133 \\ 0 & -20 & 0 & 285 \\ -33 & 62 & -68 & 253 \end{bmatrix}$$

and $$P_{21}{}^{I} = P_{23}{}^{I} = \begin{bmatrix} 181 & 0 & 0 & 0 \\ -164 & 118 & 14 & 15 \\ 0 & -285 & 0 & -20 \\ 58 & 228 & -93 & -12 \\ 0 & 0 & 181 & 0 \\ -38 & -111 & -139 & 133 \\ 0 & 20 & 0 & -285 \\ 33 & 62 & 68 & 253 \end{bmatrix}$$

The performance analysis of the integer approximation is given in the next section.

C. Motion compensated integer transform coefficients for inter frame video transcoding

For inter frame transcoding, the integer transform coefficients cannot be obtained directly from the DCT coefficients since the targeted integer transform coefficients may be shifted after the motion re-estimation process as shown in Figure 7.7. In other words, the shifted version of the integer transform coefficients is not available from the incoming bitstream. Figure 7.9 shows that the targeted integer transform coefficients are formed by using parts of four segments which come from its neighboring blocks. It is possible to perform inverse quantization, inverse DCT, forward integer transform to obtain the targeted integer transform coefficients. However, these processes require high computational complexity and introduce re-encoding errors. In the following, we propose to split the block into two regions: overlapping region and boundary region as

shown in Figure 7.10 to obtain the motion-compensated integer transform coefficient. In the overlapping region, we use the new motion vector obtained from the motion re-estimation process and propose to use some shifting operators for transcoding this region. By combining the proposed DCT to Integer Transform operators as described in the previous subsection and the shifting operators, we can obtain the motion compensated integer transform coefficients in the compressed domain. This is to achieve low computational complexity and reduce re-encoding errors.

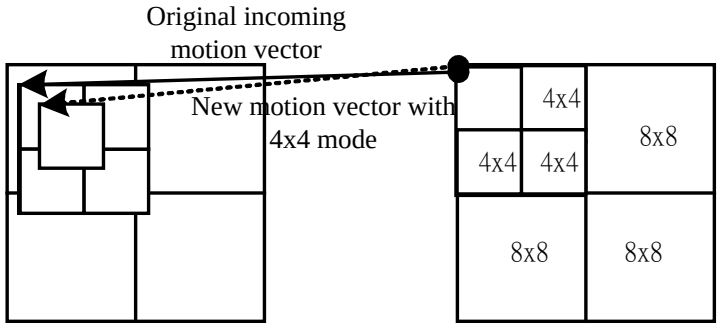


Figure 7.9. The targeted integer transform coefficients is formed by using parts of four segments which come from its neighboring blocks.

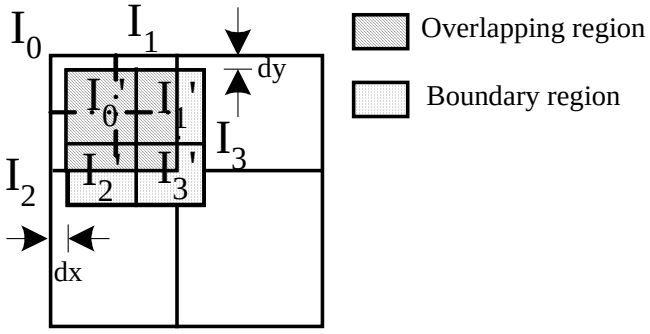


Figure 7.10. Overlapping region and the boundary region of MB_{t-1} in the transform domain.

Figure 7.10 shows the scenario that the overlapping region which is obtained from the original motion vector and the new re-estimated motion vector. The integer transform coefficients can be obtained from the incoming DCT coefficients as shown in the previous subsection. After the motion re-estimation process, the targeted integer

transform coefficients may be shifted. Using the proposed DCT to I operator, we can obtain the integer transform coefficients with the original motion vector as shown in Figure 7.9. By using the proposed shifted operator, we can obtain the motion compensated integer transform coefficients (MC-I) with the new motion vector which are obtained by motion re-estimation process. In the following of this section, a way of transcoding the overlapping region to obtain the MC-I is introduced and an adaptive approach for transcoding the MC-I boundary regions is discussed. The major idea to obtain the MC-I coefficients in the overlapping region is to represent it as the sum of all horizontally and/or vertically displaced anchor blocks. Then the integer coefficients of I_0', I_1', I_2' and I_3' are constructed using the precomputed transform operators and shift operators.

In the previous sub-section, we obtain the integer transform coefficients from the DCT coefficients. Therefore, our target is to represent the MC-I in terms of the transcoded integer transform coefficients, I_0, I_1, I_2 and I_3. In order to obtain the shifted version of integer transform coefficients , some horizontal and vertical shift operators are proposed. Since the integer transform coefficients is 4x4, the corresponding shifting operator is shown in the following. Using these operators, we can obtain the MC-I coefficients.

Consider that I_0' is the targeted integer transform coefficients. I_0, I_1, I_2 and I_3 are its four neighboring blocks in the transform domain, and the new motion vector is deviated from the original motion vector, (dx,dy). The shaded regions in I_0, I_1, I_2 and I_3 are moved by an amount, (dx,dy). Then I_0' can be represented by the following equation.

$$I_0' = \sum_{i=0}^{3} S_{i1} I_i S_{i2} \tag{7.13}$$

where S_{ij} are matrics with a size of 4x4 as shown in below.

$$S_{01}=\begin{bmatrix} 0 & I_{hd} \\ 0 & 0 \end{bmatrix}, S_{11}=\begin{bmatrix} 0 & I_{hd} \\ 0 & 0 \end{bmatrix}, S_{21}=\begin{bmatrix} 0 & 0 \\ I_{hb} & 0 \end{bmatrix}, S_{31}=\begin{bmatrix} 0 & 0 \\ I_{hb} & 0 \end{bmatrix}$$

$$S_{02}=\begin{bmatrix} 0 & 0 \\ I_{wd} & 0 \end{bmatrix}, S_{12}=\begin{bmatrix} 0 & I_{wb} \\ 0 & 0 \end{bmatrix}, S_{22}=\begin{bmatrix} 0 & 0 \\ I_{wd} & 0 \end{bmatrix}, S_{32}=\begin{bmatrix} 0 & I_{wb} \\ 0 & 0 \end{bmatrix}$$

Each I is an identity matrix of size hb=dy or hd=4-dy or wb=dx or wd=4-dx. As a pre-multiplication step, we have to shift the sub-block of interest horizontally while shift the sub-block vertically as a post-multiplication step.

Combining equation (7.12) and equation (7.13), we can obtain the MC-I directly from the DCT coefficients as shown in the following equation:

$$I_0' = \sum_{i=0}^{3} S_{i1} * P_{1i} * [\mathrm{DCT}(A)] * P_{2i} \otimes (\text{scaling factor } 1) * S_{i2} \tag{7.14}$$

Since the motion vector of I_0' may be different from other motion vectors in I_1', I_2' and I_3', decomposition of the block is needed. Using the properties of the integer transform, we have

$$I(A)+I(B)=I(A+B) \tag{7.15}$$

where A and B represent the pixels in the overlapping region and three boundary regions with a size of 4x4 respectively. We can split each of blocks I_1',I_2' and I_3'in two regions: overlapping region and boundary region as shown in Figure 7.10. From equation (7.15), we can use the integer transform coefficients in the overlapping region and in the boundary region to obtain the targeted integer transform coefficients. For

transcoding the boundary region, inverse DCT, motion compensation and forward integer transform process are required. Since zero values are in the overlapping region, low computational complexity for transcoding the boundary region can be achieved.

D. Fast approximation of DCT coefficients for H.263 to H.264 video transcoding in low bitrate application

To further speed up the transcoding process, an approximation form of the DCT coefficients is introduced. Since the energy distribution of the DCT blocks obtained from an incoming bitstream mainly concentrates on the low frequency region, it is beneficial to approximate the DCT coefficients using only a few significant DCT coefficients. The pattern of significant DCT coefficients for transcoding can be obtained by using the following equation which defines the energy of the DCT coefficients of a block with size NxN,

$$energy = \sum_{l=0}^{N-1}\sum_{m=0}^{N-1} B^2(l,m) \tag{7.16}$$

$$Approx.\,energy = \sum_{l=0}^{j}\sum_{m=0}^{k} B^2(l,m) \tag{7.17}$$

where $B(l,m)$ represents the l^{th} row and m^{th} column of the DCT coefficients. j and k represent the numbers of rows and columns to approximate the original DCT coefficients. Initially, j and k are set to zero. If the approximated energy is less than 0.95 time of the original energy, j or k will be increased by 1 until the desired approximation of the DCT coefficients is obtained. Note that the threshold 0.95 can be changed depending upon the application and the processing power of the transcoder. For the low bitrate video application, only a few DCT coefficients are needed to be transcoded. Hence, significant

computational savings can be achieved. The performance of using the approximation of DCT coefficients will be discussed in the next section.

7.2.3 Experimental results

Extensive simulations and performance comparison have been done with reference to a cascaded pixel-domain transcoder. These experiments have been performed to evaluate the overall efficiency of the proposed compressed domain video transcoding for H.263 to H.264. The cascaded transcoder was implemented using an H.263 video codec and an H.264 video codec. In the front encoder, the first frame of a sample sequence was encoded as an intraframe (I-frame), and the remaining frames were encoded as interframes (P-frames). Two situations were considered. In the first situation, variable block sizes motion estimation was not considered, whilst in the second situation variable block sizes motion estimation was used. In both situations, picture-coding modes were preserved during transcoding.

The first set of experiments aims at evaluating the performance of the proposed techniques including the proposed operators for transcoding the DCT to Integer transform coefficients in the compressed domain, integer approximation of operators and fast approximation of DCT coefficients. Results of our experimental work show that the proposed video transcoders outperform the conventional cascaded video transcoder in terms of computational complexity as shown in Table 7.2. Quality degradation I is used to indicate the case using the approximation of DCT coefficients in the proposed video transcoder as compared with the conventional cascaded video transcoder. Quality degradation II is used to indicate the case using the approximation of DCT coefficients and the integer approximation of operators in the proposed video transcoder as compared

with the conventional cascaded video transcoder. The speed up ratio is defined as the time for transcoding integer transform coefficients using the proposed transcoder with the approximation of DCT coefficients as compared with the time for conventional cascaded video transcoder. Theoretically, the speed up ratio for transcoding the DCT coefficients to Integer transform coefficients is about 2.8 times. (Assume that addition and multiplication operations take equal times. The conventional approach uses equation(7.6) and equation(7.8) while the proposed algorithm uses equation(7.12)) . Since our proposed operators have many zero values, 3.15 times speed up ratio can be achieved when these redundant operations are eliminated. If the approximation of DCT coefficients is used, the proposed video transcoding architecture can achieves a speed up of about 3.6 to 8.4 times for transcoding the DCT coefficients to integer transform coefficients. The results are more significant for sequences with low motion activities using the approximation form of the DCT coefficients for fast computation since only a few DCT coefficients are needed to be transcoded. In the second part of this experiment, variable block sizes motion estimation was employed. Two sets of the proposed operators for DCT coefficients to integer transform coefficients without motion compensation and with motion compensation are examined. Table 7.3 shows the performance of the proposed transcoder with variable block sizes motion re-estimation. For video sequences with low motion activity, the performance is very similar to the previous one since direct conversion of DCT coefficients to integer transform coefficients can be performed in most of the cases. For video sequences with high motion activities, decomposition of the block is required for transcoding the block with new motion vector obtained from the motion re-estimation. The result is expected since the transcoding process becomes more complex in this situation. If the approximation of DCT coefficients is used, the proposed video

transcoding architecture can achieve a speed up of about 3 to 8 times for transcoding the DCT coefficients to integer transform coefficients. In addition, the proposed architecture can combine with any existing fast searching algorithm for multiple reference frame and variable block sizes estimation to speed up the transcoding process.

Table 7.2. Performance of the proposed transcoder, where the frame rate of the incoming bitstream was 30 frames/s. H.263 was used as the front encoder for encoding "Salesman", "Miss_America", "Hall", "Tennis", "Football" and "Flower".

Sequences	Input bitrate	Quality degradation I(dB)	Quality degradation II(dB)	Speed up ratio
Salesman (352x288)	512k	0.03	0.05	8.2
	256k	0.05	0.06	8.4
Miss_ America (352x288)	512k	0.07	0.09	7.9
	256k	0.09	0.1	8.0
Hall (352x288)	512k	0.13	0.14	6.9
	256k	0.16	0.17	7.1
Tennis (352x240)	3M	0.2	0.22	6.3
	1.5M	0.25	0.27	6.4
Flower (352x240)	3M	0.27	0.29	4.4
	1.5M	0.3	0.32	4.5
Football (352x240)	3M	0.3	0.33	3.6
	1.5M	0.34	0.37	3.7

Table 7.3. Performance of the proposed transcoder with variable block size motion re-estimation, where the frame rate of the incoming bitstream was 30 frames/s. H.263 was used as the front encoder for encoding “Salesman”, “Miss_America”, “Hall”, “Tennis”, “Football” and “Flower”.

Sequences	Input bitrate	Quality degradation I(dB)	Quality degradation II(dB)	Speed up ratio
Salesman (352x288)	512k	0.05	0.07	7.8
	256k	0.07	0.08	8.0
Miss_ America (352x288)	512k	0.10	0.11	7.3
	256k	0.11	0.12	7.4
Hall (352x288)	512k	0.15	0.16	6.1
	256k	0.16	0.17	6.2
Tennis (352x240)	3M	0.22	0.24	5.5
	1.5M	0.27	0.29	5.6
Flower (352x240)	3M	0.28	0.31	3.8
	1.5M	0.31	0.34	3.9
Football (352x240)	3M	0.33	0.36	3.0
	1.5M	0.35	0.38	3.1

7.2.4 Conclusion

In this section, a new compressed domain video transcoder for H.263 to H.264 is proposed. We have derived two set of operators to transcode the DCT coefficients into Integer transform coefficients in the compressed domain for intra frame and inter frame video transcoding. Using the approximation of DCT coefficients, the proposed transcoder can further speed up the transcoding process significantly with similar transcoded video quality for low bitrate application. In addition, the integer approximation of the operators is proposed to avoid the floating point implementation for

the proposed transcoder. The proposed architecture can combine with any existing fast searching algorithm for multiple reference frames and variable block sizes estimation to speed up the transcoding process. Results of our experimental work show that the proposed architecture produces similar pictures quality with low computational complexity as compared with the conventional video transcoder.

Chapter 8

Conclusion and Possible future work

8.1 Conclusion of the present work

In this thesis, we give results for an investigation on homogeneous and heterogeneous video transcoding. The proposed DCT-based/IT-based techniques are very powerful for video transcoding which can improve the video quality as well as the computational speed of existing algorithms.

The investigation on the homogeneous video transcoding techniques based on the DCT domain manipulation and re-composition to achieve bitrate reduction has been presented in Chapter 3 and Chapter 4. In these two chapters, some possible adaptive solutions are provided to solve the computational complexity problems of video transcoding by performing the transcoding process in the DCT domain. These techniques include approaches that take into account of DCT properties and motion compensation in the DCT domain. In Chapter 3, the quality degradation in a transcoding process is addressed. It has been shown that during the transcoding process, the re-encoding error is introduced. Besides, it suffers from the intrinsic problem of double encoding in transcoding. Motivated by this, a new DCT-based transcoder is suggested. The proposed video transcoder employs the direct addition of DCT coefficients for macroblocks coded without motion compensation to avoid full decoding and re-encoding processes. Therefore, low computational complexity can be achieved and the quality degradation of the transcoded video can be avoided. For motion compensated macroblocks transcoding, we transcode the overlapping region and boundary region separately. Since the DCT coefficients from the incoming bitstream can be reused in the overlapping region, the

separation of the transcoding arrangement can provide computational saving and avoid the re-encoding error in the overlapping region. The proposed frame-skipping transcoder can be processed in the forward order when multiple frames are dropped. Thus, only one DCT-domain buffer is needed to store the updated DCT coefficients of all skipped frames. By using such a mechanism, a new frame-rate control scheme for the proposed transcoder is also suggested. Since the quality of the non-skipped frame impacts directly the motion smoothness of the transcoded sequence, it is beneficial to force the frame-rate control scheme to select frames which have good quality for reconstruction. The proposed scheme can dynamically adjust the number of skipped frames depending upon re-encoding errors as well as the accumulated magnitude of all of the motion vectors in the current frame.

In Chapter 4, we address the problems of video downscaling. For conventional downscaling video transcoder, a video server has firstly to decompress the video, perform downscaling operations in the pixel domain, and then recompress it. This is computational intensive. However, it is difficult to perform video downscaling in the DCT-domain since the prediction errors of each frame are computed from its immediate past higher resolution frames. It involves the re-estimation of motion vectors and DCT coefficients for the downscaled video. Since the incoming motion vectors have different directions and magnitudes, the resampled motion vector for the downscaled video cannot represent the original four motion vectors well. Therefore, a mismatch of the incoming DCT coefficients and the resampled motion vector will be occur except for the case that the four incoming motion vectors are the same. Motivated by this, we propose a video downscaling transcoder to minimize the mismatched region as little as possible. In other

words, re-encoding errors will be introduced only in these relatively small mismatched regions for our proposed split and merge algorithm. This arrangement for splitting the macroblock into two regions: overlapping region and boundary region, can provide the transcoder to re-use the incoming DCT coefficients as much as possible. As a consequence, computational complexity as well as re-encoding error can be minimized during the transcoding process.

In Chapter 5, the proposed technique is applied in the DCT domain and wavelet domain for the application of the video combiner in a video conferencing system. The proposed architecture guarantees a high video quality in the region of interest while reducing the overall bit rate and the computation time even under low bit rates.

An investigation on heterogeneous video transcoding techniques has been presented in Chapter 6 and Chapter 7. In Chapter 6, we address the issues of the forthcoming multimedia telecommunication services which are expected to use pre-encoded videos for storage and transmission. The heterogeneities of the present communication networks and the clients' devices demand a match of the encoding format of the video source to the constraints of the networks and clients' devices. One of the major difficulties for transcoding from MPEG format to H.263 format is that we need to convert a B-picture to a P-picture. However, the transcoding process involves high computational complexity as well as introduces re-encoding errors. Therefore, a new transcoder architecture to convert a B-picture into a P-picture by making use of the techniques of motion compensation in the DCT domain and indirect addition of DCT coefficients is proposed. We derive a set of equations and formulate the problem of how to obtain the DCT coefficients in this research work. The major saving in terms of

computational complexity and high quality of the transcoded video is achieved from the fact that the estimated DCT coefficients be obtained in terms of the incoming DCT coefficients and the proposed operators. Since the operators to perform the motion compensation in the DCT domain can be pre-computed and the DCT coefficients can be reused, the coefficients for B frame to P frame are obtained directly in the DCT domain without performing the full decoding and re-encoding process.

In Chapter 7, we give focus on the transcoding of a video sequence from the H.263 format to the H.264 format. The significant differences between the H.263 and the H.264 formats make the transcoding become a challenge task. In this chapter, the work gives focus on the conversion of DCT coefficients to Integer Transform(IT) coefficients. To simplify the conventional approach, we make use of the pre-computed operators to speed up the transcoding process. The pre-computed operator consists of the extraction of DCT coefficients operations, the inverse DCT and forward integer transform operations in one single step to achieve low computational complexity. In order to support the variable block sizes feature in the H.264 video standard, a set of operators is proposed to obtain the Motion-Compensated Integer transform coefficients(MC-I) from the incoming DCT coefficients directly. From the experimental results, we observe that a large number of DCT coefficients with zero value are generated by the incoming video especially for low bitrate video coding. Making use of this property, an approximation form of the DCT coefficients is proposed for low bitrate video transcoding to speed up the transcoding process with similar transcoded video quality. In addition, the integer approximation of the operators is proposed to avoid the floating point implementation for the proposed transcoder. Experimental results show that the proposed video transcoder

can reduce substantially the amount of computation and provide transcoded videos with similar quality as compared with those obtained from the conventional cascaded video transcoder.

In conclusion, the compressed domain techniques play an important role to speed up the transcoding process. Results of our investigation on using the adaptive techniques in the video transcoder in this thesis offer an effective video conversion. After all, we sincerely believe that the results obtained in this work are significant for an efficient realization of modern video transcoding.

8.2 Future Work

The continuous researches in theoretical and algorithmic adaptive techniques have greatly improved the performance, the practicality and the robustness of the video transcoder. These adaptive techniques can really upgrade the existing video conversion. There are a lot of successful applications appearing in the recent literature. The emphasis in this thesis is also based on an enhancement of the existing video transcoding techniques. However, with the rapid growth of the video technology which change everyday in terms of quality, speed and scalability, we give some opinions for possible future development of our related studies as shown below.

In the video transcoding, we have proposed the dynamic frame-skipping transcoder and video downscaling transcoder which can reduce the quality degradation significantly and reduce the computational complexity. In order to achieve a higher transcoding ratio, combining the spatial and temporal video transcoders may be a possible future study.

Besides, with the emerging of H.264 video format, this new generation video standard, with its significant bandwidth savings, is expected to replace the H.263 for video compression in digital video systems. A complete migration to the new video coding algorithm will take several years since H.263 and the MPEG are widely used in many multimedia applications nowadays. This creates an important need for transcoding technologies that convert the widely available MPEG/H.263 compressed videos to H.264 compressed format and vice versa. However, the significant differences between the MPEG/H.263 and the H.264 algorithms make the transcoding become more challenging. The new research areas of video transcoding may consists of the features of the codec conversion:

1. ***Various block sizes conversion***: Various block sizes video transcoding for H.263 to H.264 can be speed up by using diversity, importance and statistical measures. By an analysis of various block sizes and the distribution among the history of the transcoded frames, the statistical behavour of the block type does not vary too much among the frames in the corresponding locations. Using this correlation property, the transcoder can speed up the process of the block type decision for H.263 to H.264 video transcoding. Also, the importance measure of the various block sizes can help the transcoder to convert the H.264 to H.263 videos by identifying which block consists of the importance feature or smooth/background region to decide how to recompose the new motion vector and control the sizes of the refinement window adaptively.

2. ***Multiple reference frames conversion:*** Using the statistical information of motion vectors, the multiple reference frames can be projected to one single frame by considering the neighbor blocks motion to adjust the base motion vector and the sizes of the refinement window for transcoding H.264 to H.263 video. Also, the motion vector and the statistical information can be reused to speed up the multiple reference frames motion estimation process for transcoding the H.263 to H.264 videos.
3. ***Transform coefficients conversion:*** For conversion between the integer transform coefficients and DCT coefficients, fast algorithm and it's approximation form of realization can be derived using the bits shifting technique or absorb the multiplication operations inside the scaling factors.
4. ***Frame skipping and Arbitrary video downsizing:*** Since the H.264 will be widely used for multimedia application, arbitrary video downsizing and frame skipping technique is needed for the video adaptation. Frame skipping and arbitrary video downsizing in the H.264 will be more complicated since the number of reference frames is increased and various block sizes are employed. The dynamic of re-indexing the reference frame, projection of the motion vectors, recomposition of the transform coefficients become challenging tasks for spatial and temporal video transcoding.
5. ***Requantization and rate-distortion video transcoding***: Fast algorithms for the decision of the block sizes and quantization parameters may become the fundamental work for video transcoding in H.264 for bitrate reduction. It is similar to the work of the fast algorithm for encoding the H.264 video but the

motion vectors and the prediction errors are available for transcoding. The diversity and the importance of the motion vectors can be used to speed up the transcoding process to avoid the redundant operations for block size decisions and determine the requantization parameters.

6. ***Scalability and video transcoding tools in H.264***: Some of the primitive information can be added in H.264 video bitstream to enhance the video transcoding process for more flexible video adaptation such as spatial-temporal video transcoding. The primitive can be motion vector or residual errors for different video formats. If the bitrate is not going to be increased too much(<10%) when these primitive is added in the video bitstream, these transcoding tools may not be a burden and it allows the transcoder to achieve fast video adaptation and provide high quality transcoded videos for more diverse video applications.

References:

[1] Kai-Tat Fung, Yui-Lam Chan and Wan-Chi Siu, "New architecture for dynamic frame-skipping transcoder," IEEE Transactions on Image Processing, Vol.11, pp.886-900, Aug 2002

[2] Jeongnam Youn, Ming-Ting Sun and Chia-Wen Lin, "Motion vector refinement for high-performance transcoding," IEEE Transactions on Multimedia, Vol.1, pp.30 - 40, March 1999

[3] G. Keeman, R. Hellinghuizen, F. Hoeksema and G. Heideman, "Transcoding of MPEG bitstreams," Signal Processing: Image Communication, vol. 8, pp. 481-500, Sept. 1996.

[4] Shanableh, T.; Ghanbari, M., "Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding formats," IEEE Transactions on Multimedia, vol.2, pp.101-110, June 2000.

[5] Bo Shen, Ishwar K. Sethi and Bhaskaran Vasudev, "Adaptive motion-vector resampling for compressed video downscaling," IEEE Transactions on circuit and systems for video technology, vol.9, no.6, September 1999.

[6] H. Sun, W. Kwok and J.W. Zdepski, "Architectures for MPEG compressed bitstream scaling," IEEE Trans. Circuits Syst. Video Technol., vol. 6, pp. 191-199, Apr. 1996.

[7] T.Warabino, S. Ohta, D. Morikawa, M. Ohashi, H. Nakamura, H. Iwashita, F. Watanabe, "Video trasnscoding proxy for 3G wireless mobile Internet access," IEEE Commun. Mag., vol. 10, pp.66-71, 2000

[8] P.A.A. Assuncao, M. Ghanabari, "A frequency-domain video transcoder for dynamic bit-rate reduction of MPEG-2 bitstreams, IEEE Trans. Circuits Syst. Video Technol., vol.8, pp.953-967, 1998

[9] P.A.A. Assuncao, M. Ghanabari, "Fast computation of MC-DCT for video transcoding, Electron. Letter, vol.4, pp.284-286, 1997

[10] M.S.M. Lei, T.C. Chen, and M.T. Sun, "Video bridging based on H.261 standard," IEEE Transactions on circuit and systems for video technology, vol. 4, pp. 425-437, Aug. 1994.

[11] H.J. Stuttgen, "Network evolution and multimedia communication," IEEE Multimedia, vol. 2, pp. 42-59, Fall 1995.

[12] Yap-Peng Tan and Haiwei Sun, "Fast motion re-estimation for arbitrary downsizing video transcoding using H.264/AVC standard," IEEE Transactions on Consumer Electronics, Vol.50, no.3, pp.887-894, Aug. 2004

[13] Shizhong Liu and Bovik, A.C., "Local bandwidth constrained fast inverse motion compensation for DCT-domain video transcoding," IEEE Transactions on Circuits and Systems for Video Technology, Vol.12, no.5, pp.309-319, May 2002

[14] YongQing Liang, Lap-Pui Chau and Yap-Peng Tan, "Arbitrary downsizing video transcoding using fast motion vector reestimation" IEEE Signal Processing Letters, Vol.9, no.11, pp.352-355, Nov. 2002

[15] Haiyan Shu and Lap-Pui Chau, "An efficient arbitrary downsizing algorithm for video transcoding," IEEE Transactions on Circuits and Systems for Video Technology, Vol.14 , no.6, pp.887-891, June 2004

[16] Vetro, A., Hata, T., Kuwahara, N., Kalva, H. and Sekiguchi, S., "Complexity-quality analysis of transcoding architectures for reduced spatial resolution," IEEE Transactions on Consumer Electronics, Vol.48, no.3, pp. 515-521, Aug. 2002

[17] Mei-Juan Chen, Ming-Chung Chu and Chih-Wei Pan, "Efficient motion-estimation algorithm for reduced frame-rate video transcoder," IEEE Transactions on Circuits and Systems for Video Technology, Vol.12, no.4, pp.269-275, April 2002

[18] Kai-Tat Fung, Yui-Lam Chan and Wan-Chi Siu, "Low-Complexity and High-Quality Frame-Skipping Transcoder for Continuous Presence Multipoint Video Conferencing," IEEE Transactions on Multimedia, Vol.6, No.1, pp.31-46, Feb. 2004

[19] Ming-Ting Sun, Alexander C. Loui, and Ting-Chung Chen, "A coded-domain video combiner for multipoint continuous presence video conferencing," IEEE Transactions on Circuits and Systems for Video Technology, vol. 7, no. 6, pp. 855-863, December 1997.

[20] Ming-Ting Sun, Tzong-Der Wu, and Jenq-Neng Hwang, "Dynamic bit allocation in video combining for multipoint conferencing," IEEE Transactions on Circuits and Systems – II: Analog and Digital Signal Processing, vol. 45, no. 5, pp. 644-648, May 1998.

[21] Jenq-Neng Hwang, Tzong-Der Wu and Chia-Wen Lin, "Dynamic frame-skipping in video transcoding," 1998 IEEE Second Workshop on Multimedia Signal Processing, pp. 616 –621, 1998.

[22] Chia-Wen Lin, Te-Jen Liou, and Yung-Chang Chen, "Dynamic rate control in multipoint video transcoding," in Proc. IEEE Int. Symposium on Circuits and Systems'2000, vol. 2, pp. 17-20, May 28-31, 2000.

[23] K. T. Fung, Y. L. Chan and W. C. Siu, "Low-complexity and high quality frame-skipping transcoder," in Proc.IEEE Int. Symposium on Circuits and Syst.'2001, Sydney, Australia, pp. 29-32, May 6-9, 2001.

[24] Tap-Peng Tan; Haiwei Sun; YongQing Liang, "On the methods and applications of arbitrarily downsizing video transcoding," in Proc. IEEE International Conference on Multimedia and Expo, vol.1 , pp.609-612, 2002

[25] Zhijun Lei; Georganas, "H.263 video transcoding for spatial resolution downscaling," in Proc. International Conference on *N.D.* Information Technology: Coding and Computing, pp. 425-430, 2002.

[26] Shih-Fu Chang and David G. Messerschmitt, "Manipulation and compositing of MC-DCT compressed video," IEEE Journal on selected areas in communications, vol.13, no.1, pp. 1-11, Jan. 1995.

[27] Neri Merhav and Vasudev Bhaskaran, “Fast algortihms for DCT-domain image downsampling and for inverse motion compensation,” IEEE Trans. on circuits and systems for video technology, vol.7, no.3, June 1997.

[28] J. Song and B.L. Yeo, “A Fast Algorithm for DCT-Domain Inverse Motion Compensation Based on Shared Information in a Macroblock,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 10, no. 5, pp. 767-775, August 2000.

[29] U. Koc, K. Ray, “DCT-based motion estimation,” IEEE Trans. Image Process., vol.7 pp.948-965, 1998

[30] ISO/IEC 11172-2, “Information Technology -- Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to About 1,5 Mbit/s -- Part 2: Video,” 1993

[31] ISO/IEC 13818-2, “Information Technology -- Generic Coding of Moving Pictures and Associated Audio Information: Video,” 1996.

[32] ISO/IEC, “Test Model 5, TM5, ISO/IEC JTC/ SC29/ WG11/ N0400, MPEG93/457,”Apr. 1993

[33] ITU-T/SG15, “Video codec test model, TMN8,” June 97.

[34] *Video Coding for Low Bitrate Communication*, ITU-T Recommendation H.263, May 1997.

[35] A. Vetro, C. Christopoulos and H. Sun,, “Video transcoding architectures and techniques: An overview”, IEEE Signal Processing Magazine, pp.18-29, March 2003.

[36] Y. L. Chan and W. C. Siu, “New adaptive pixel decimation for block motion vector estimation,” IEEE Trans. Circuits Syst. Video Technol., vol.6, no.1, pp.113 -118, Feb. 1996.

[37] Y. L. Chan and W. C. Siu, “Edge oriented block motion estimation for video coding,” IEE Proceedings: Vision, Image and Signal Processing, vol. 144, no. 3, pp. 136-144, Jun. 1997.

[38] Y. L. Chan and W. C. Siu, “On block motion estimation using a novel search strategy for an improved adaptive pixel decimation,” Journal of Visual Communication and Image Representation, vol. 9, no. 2, pp. 139-154, Jun. 1998.

[39] Jo Yew Tham, Surendra Ranganath, Maitreya Ranganath, and Ashraf Ali Kassim, “A novel unrestricted center-biased diamond search algorithm for block motion estimation,” IEEE Trans. Circuits Syst. for Video Technol., vol. 8, pp. 369-377, Aug. 1998.

[40] Lai-Man Po and Wing Chung Ma, “A novel four-step search algorithm for fast block motion estimation,” IEEE Trans. Circuits Syst. for Video Technol., vol. 6, pp. 313-317, June 1996.

[41] J. Chalidabhongse and C.-C. Jay Kuo, “Fast motion vector estimation using multiresolution-spatio-temporal correlations,” IEEE Trans. Circuits Syst. Video Technol., vol. 7, pp. 477-488, June 1997.

[42] Gary J. Sullivan and Thomas Wiegand, “Rate-distortion optimization for video compression,” IEEE Signal Processing Magazine, pp. 74-90, Nov. 1998.

[43] Guy Côté, Berna Erol, Michael Gallant, and Faouzi Kossentini, "H.263+: video coding at low bit rates," IEEE Trans. Circuits Syst. Video Technol., vol. 8, pp. 849-866, Nov. 1998.

[44] L. Chiariglione, "The development of an integrated audiovisual coding standard: MPEG," Proceedings of IEEE, vol. 83, pp. 151-157, Feb. 1995.

[45] Y.-Q. Zhang and S. Zafar, "Predictive block-matching motion estimation for TV coding – Part II: Inter-frame prediction," IEEE Trans. Broadcast., vol. 37, pp. 102-105, Sept. 1991.

[46] Wiegand,T., Sullivan, G.J., Bjntegaard, G. and Luthra, A., " Overview of the H.264/AVC video coding standard" IEEE Transactions on Circuits and Systems for Video Technology, Vol.13, No.7, pp.560- 576, July 2003

[47] Ostermann, J., Bormans, J., List, P., Marpe, D., Narroschke, M., Pereira, F., Stockhammer, T.and Wedi, T., "Video coding with H.264/AVC: tools, performance, and complexity," IEEE Circuits and Systems Magazine, Vol.4, no.1, pp.7-28, 2004

[48] Xin, J., Lin, C.-W. and Sun, M.-T., "Digital Video Transcoding," Proceedings of the IEEE , Vol.93 , no.1, pp.84-97, Jan. 2005

[49] Wan-Chi Siu, Kai-Tat Fung and Yui-Lam Chan, "A compressed-domain heterogeneous video transcoder," IEEE International Conference on Image Processing(ICIP), pp.2761-2764, 2004.

[50] Ke Shen and Edward J.Delp, 'Wavelet-based Rate Scalable Video Compression', IEEE Trans. on Circuit and Systems for Video Technology, Vol. 9, No.1, pp. 109-122, February 1999.

[51] K. T. Fung, N. F. Law and W. C. Siu, "Region-based Object Tracking for Multipoint Video Conferencing Using Wavelet Transform," Proceedings, IEEE International Conference on Consumer Electronic(ICCE'2001), pp.268-269, June, 2001.

[52] Renxiang Li, Bing Zeng, and Ming L. Liou, "A new three-step search algorithm for block motion estimation," IEEE Trans. Circuits Syst. Video Technol., vol. 4, pp.438-442, Aug. 1994.

[53] ITU-T Study Group XV, Recommendation H.261, "Video codecs for audiovisual services at p × 64 kb/s," May 1992.

[54] ITU-T Study Group XV, Recommendation H.231, "Multipoint control units for audiovisual systems using digital channels up to 2 Mb/s," May 1992.

[55] ITU-T Study Group XV, Recommendation H.243, "Procedures for establishing communication between three or more audiovisual terminals using digital channels up to 2 Mb/s," May 1992.

[56] Y. Nakajima, H. Hori and T. Kanoh, "Rate conversion of MPEG coded video by re-quantization process," in IEEE International Conference on Image Processing, ICIP95, vol. 3, pp. 408-411, October 1995, Washington, DC.

[57] P. Assuncao and M. Ghanbari, "Post-processing of MPEG2 coded video for transmission at lower bit rates," in IEEE International Conference on Acoustic, Speech, and Signal Processing, ICASSP96, vol. 4, pp. 1998-2001, May 1996, Atlanta, GA.

[58] M.Yong, Q.-F. Zhu, and V. Eyuboglu, "VBR transport of CBR encoded video over ATM networks," in Proc. 6th Int. Workshop Packet Video, Portland, OR, Sept. 1994, pp. D18.1-D18.4.

[59] D.G. Morrison, M.E. Nilsson, and M. Ghanbari, "Reduction of the bitrate of compressed video while in its coded form," in Proc. 6th Int. Workshop Packet Video, Portland, OR, Sept. 1994, pp. D17.1-D17.4

[60] F.-H Cheng and S.-N. Sun, "New fast efficient two-step search algorithm for block motion estimation," IEEE Transactions on Circuits and Systems for Video Technology, vol. 9, no. 7, pp.977-983, October 1999.

[61] S.H. Kwok, W.C. Siu and A.G. Constantinides, "Adaptive temporal decimation algorithm with dynamic time window," IEEE Trans. on Circuits and Systems for Video Tech., vol.8, pp. 104-111, February 1998.

[62] S.H. Kwok, W.C. Siu and A.G. Constantinides, "A scaleable and adaptive temporal segmentation algorithm for video coding," Graphical Models and Image Proc., vol.59, pp. 128-138, May 1997.

[63] P. Yip and K.R. Rao, "The decimation-in-frequency algorithms for a family of discrete sine and cosine transforms," Circuits, Systems, and Signal Process., pp. 4-19, 1988.

[64] H.J. Stuttgen, "Network evolution and multimedia communication," IEEE Multimedia, vol. 2, pp. 42-59, Fall 1995.

[65] Y. Nakajima, H. Hori and T. Kanoh, "Rate conversion of MPEG coded video by re-quantization process," in Proc. IEEE Int. Conf. Image Processing'95, Washington, DC ICIP95, vol. 3, pp. 408-411, Oct. 1995.

[66] Henrique S. Malvar, Antti Hallapuro, Marta Karczewicz and Louis Kerofsky, "Low-Complexity Transform and Quantization in H.264/AVC," IEEE Transactions on Circuits and Systems for Video Technology, vol.13, no.7, pp.598-603, July 2003

[67] A.Ahmad, N.Khan, S.Masud and M.A. Maud, "Selection of variable block sizes in H.264," IEEE International Conference on Acoustics, Speech, and Signal Processing(ICASSP 2004), pp.173-176, 2004

[68] M.H. Chan, Y.B. Yu and A.G. Constantinides, "Variable size block matching motion compensation with applications to video coding," IEE Proceedings, vol.137, no.4, pp.205-212, Aug 1990.

[69] A. Ahmad, N. Khan, S.Masud and M.A. Maud, "Efficient block size selection in H.264 video coding standard," Electronics Letters, vol.40, no.1, Jan. 2004.

[70] X.Jing and L.-P. Chau, "Fast approach for H.264 inter mode decision," Electronic Letters, vol.40, no.17, pp.1050-1052, Aug 2004

[71] Zhi Zhou, Ming-Ting Sun and Yuh-Feng Hsu, "Fast variable block size motion estimation algorithms based on merge and split procedures for H.264/MPEG-4 AVC," IEEE International Symposium on Circuits and Systems (ISCAS 2004), pp.725-728, 2004

[72] Andy Chang, Peter H.W. Wong, Y.M.Yeung and Oscar C. Au, “Fast multi-block selection for H.264 video coding,” IEEE International Symposium on Circuits and Systems (ISCAS 2004), pp.817-820, 2004

[73] Kai-Tat Fung and Wan-Chi Siu, “Low-Complexity H.263 to H.264 video transcoding using motion vector decomposition,” paper accepted, to be published in the 2005 IEEE International Symposium on Circuits and Systems (ISCAS 2005)