INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Improved standard-conforming video transcoding techniques

Jun Xin

A dissertation submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington

2002

Program Authorized to Offer Degree: Department of Electrical Engineering

UMI Number: 3072156

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

Doctoral Dissertation

In presenting this dissertation in partial fulfillment of the requirements for the Doctoral degree at the University of Washington, I agree that the Library shall make its copies freely available for inspection. I further agree that extensive copying of the dissertation is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for copying or reproduction of this dissertation may be referred to ProQuest Information and Learning, 300 North Zeeb Road, Ann Arbor, MI 48106-1346, to whom the author has granted "the right to reproduce and sell (a) copies of the manuscript in microform and/or (b) printed copies of the manuscript made from microform."

Signature 

Date 11/26/2002

University of Washington
Graduate School

This is to certify that I have examined this copy of a doctoral dissertation by

Jun Xin

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

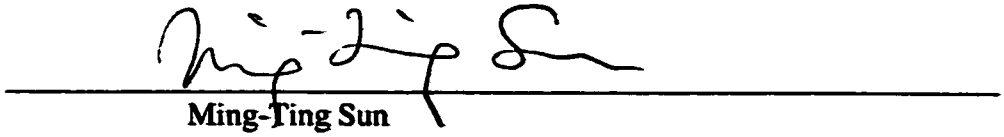
Chair of Supervisory Committee:

Ming-Ting Sun

Reading Committee:

Linda Shapiro

Eve Riskin



Ming-Ting Sun

Date: 11/26/2002

University of Washington

Abstract

Improved standard-conforming video transcoding techniques

Jun Xin

Chair of the Supervisory Committee

Professor Ming-Ting Sun
Department of Electrical Engineering

Many video coding standards have been established for various video applications, such as H.263 for low-bit-rate two-way video communications, MPEG-2 for broadcasting and general high-quality video applications, and MPEG-4 for streaming video and interactive multimedia applications. As digital video applications become more popular, increasing amounts of video content are encoded in different formats. Consequently there are increasing needs to dynamically convert the video between different formats. A video transcoder is a device that converts video from a compressed format into another compressed format. It is one of the enabling technologies for video delivery over heterogeneous networks. The goal of transcoding research is to achieve the best possible quality–complexity tradeoff given application constraints. The key to achieve this goal is to intelligently utilize the coding statistics of the input compressed video stream to facilitate the transcoding process.

In this dissertation, we propose several techniques to improve transcoding performance. First, we present an efficient motion re-estimation scheme that can handle

not only the temporal/spatial resolution conversion, but also the interlaced video processing. Exploiting the spatial and temporal correlations between the input motion vectors, we derive the motion vectors for the transcoded output video from appropriate input motion vectors. Compared with the straightforward, cascaded decoder-encoder implementation using full-search motion estimation, the proposed algorithm achieves virtually the same quality with greatly reduced computational requirement. We then present an adaptive picture-layer bit-allocation algorithm based on estimated output picture complexities. It achieves better bit-allocations to different types of pictures, and handles scene-changes better than conventional approaches. It the end, we propose an operational rate-distortion optimized macroblock-layer rate-control algorithm based on the Lagrange Multiplier (LM) algorithm. The LM algorithm is a quasi-optimal solution to this rate-control problem, but is computationally too expensive for practical applications. Taking advantage of the properties of the low-bit-rate video and the LM algorithm, we propose an approach to reduce its computational complexity. The proposed algorithm outperforms H.263 TMN8 in terms of video quality and buffer behavior with slightly increased computational requirement.

TABLE OF CONTENTS

List of Figures

Figure Number Page

List of Tables

Table Number Page

Glossary

CPDT: cascaded pixel domain transcoder

DCT: discrete cosine transform

DDT: DCT domain transcoder

MCP: motion compensated prediction

MSE: mean squared error

PSNR: peak signal-to-noise ratio

QP: quantization parameter

SAD: sum of absolute difference

VLC: variable length coding

Acknowledgements

I am extremely grateful to Professor Ming-Ting Sun, my advisor, for his invaluable guidance, consistent encouragement and support through the years. He has always been my mentor beyond the academic scope. Also I would like to thank Professor Eve Riskin and Professor Linda Shapiro for their inspiring suggestions.

I take this opportunity to thank many fellow students in the Information Processing Lab, Jeongnam Youn, Supavadee Aramvith, Dongxiang Xu, I-Ming Pao, Renjit Thomas, Tao Yang, Daniel GaticaPerez, Jian Zhou, Yeping Su, Jinhui Pan, and Zhi Zhou for various kinds of help. They have made my life and study at UW an enjoyable experience.

I would like to express my most sincere appreciation to Xuan, my wife, for her patience, understanding and support. Finally, I would like to thank my parents, who have provided tremendous support and great guidance since the first day I arrived in this world.

Chapter 1. Introduction

Digital video have received wide acceptance in the past decade. Digital TV broadcasting, Digital Video Disc (DVD), Digital Video Recorders, Distance Learning, Video on Demand, and Videoconferencing are some typical applications. A key technology that enables these applications is digital video coding. Digital video coding facilitates the storing and processing of digital video by greatly reducing the amount of data necessary to represent the digital video. Coded digital video representation using today's video coding standards is more efficient than its analog counterpart. For example, with digital video coding, one terrestrial NTSC analog TV channel can transmit four to six Standard Definition TV (SDTV) programs, each of which generally provides better video quality than the analog NTSC TV, or can transmit one High Definition TV (HDTV) program [1].

Currently many video coding standards have been established for various video applications, such as H.263 [2] for low-bit-rate two-way video communications, MPEG-1 [3] for storage media applications, MPEG-2 [4] for broadcasting and general high-quality video applications, and MPEG-4 [5] for streaming video and interactive multimedia applications. As the digital video applications become increasingly popular, there will be increasing amounts of video content encoded in various formats. Therefore, there are practical needs to dynamically convert the video between different formats.

Video transcoding is the operation of converting a video from a compressed format into another compressed format. A device that performs video transcoding is

called a video transcoder. One of the earliest applications of transcoding is to convert a compressed video into a lower bit-rate [25]. For example, a TV program may be compressed at a high bit-rate and stored in a server, but later needs to be transported over a network at a much lower bit-rate. A video transcoder can be used to achieve this purpose. Besides bit-rate adaptation, a transcoder can dynamically change any coding parameters (e.g., frame-rate and spatial resolution) and/or coding standards (e.g., MPEG-2 to MPEG-4) of the compressed video as illustrated in Figure 1.

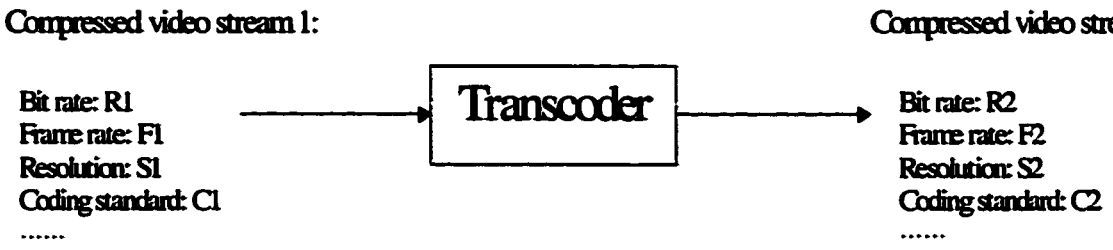


Figure 1. Video transcoder

With the capability of dynamically altering coding parameters of the compressed video, it is expected that video transcoding will play an important role for universal multimedia access by the Internet users with different access links and devices [6]. Different users may use different access networks to connect to the Internet, including LAN (Local Access Network), DSL (Digital Subscriber Line), Cable TV line, wireless LAN, ISDN (Integrated Services Digital Network), and Telephone modem. These different access networks have different channel characteristics: channel bandwidth, bit error-rate, packet loss-rate, etc. At the user end, the access devices are becoming more and more diversified. Network appliances including handheld computers, personal

digital assistants (PDA's), set-top boxes, smart cellular phones, are slated to replace personal computers to become the dominant access terminals for accessing the Internet. These network terminals vary a lot in resources such as computing power and display capability. In addition, users' interests may also differ from each other [7]. For multimedia data – including but not limited to coded video – to be intelligently delivered to users with different available resources, access networks, and interests, the multimedia content must be adapted dynamically according to the different user requirements. Transcoding serves as one of the key technologies to fulfill this challenging task. Envisioning the potential of transcoding, the emerging MPEG-7 standard [41], which standardizes a framework for describing audio-visual content, has defined "transcoding hints" to facilitate the transcoding of compressed video contents [42].

Dynamic change of limited coding parameters such as the bit-rate, the frame-rate, and the spatial resolution could also be achieved by scalable coding [8-9]. However, due to the inter-frame predictive coding nature of video coding, these scalable video coding schemes suffer severe quality degradation at bit-rates higher than the base-layer rate. Another disadvantage is that scalable video coding demands additional complexities at both video encoders and video decoders. These inherent weaknesses of scalable coding have kept it from being deployed in practical applications.

What differentiates the transcoding techniques from conventional encoding techniques is that the input coded video stream is readily available (at the input bit-rate) in the transcoding process. Valuable coding parameters and statistics of the video can be easily obtained from the input video stream, and be used to facilitate the transcoding

process by reducing its computational complexity and improving the coded video quality. In fact, since using a transcoder to adapt the bit-rate can be considered as a two-pass encoding process, it is possible to achieve better video quality than typical one-pass encoding of the video at the same bit-rate. This is the basic idea behind the new techniques proposed in this dissertation. In this dissertation, we discuss the new techniques for a video transcoding system that is capable of dynamically changing not only coding parameters, but also coding formats. The goal of the research is to achieve the best possible video quality and the least possible computational complexity.

In the first part of the dissertation, an efficient motion re-estimation scheme is presented for heterogeneous video transcoding. In the heterogeneous video transcoding, the output video differs from the input video not only in bit-rates, but also in spatial resolutions, temporal resolutions, and coded formats. Exploiting the spatial and temporal correlations between the input motion vectors, we compose the target motion vectors for the output video from appropriate input motion vectors. The derived motion vectors are then further refined using an efficient half-pixel refinement scheme. Compared with the straightforward, cascaded decoder-encoder implementation of the transcoder using full-search motion estimation, simulations show that the proposed motion re-estimation algorithm achieves virtually the same quality with greatly reduced computational requirement.

The coded information extracted from the input video stream is not only useful in reducing the computational complexity of the transcoder, but also can be used to improve the quality of the transcoded video. In the second part of the dissertation, we develop

rate-control algorithms that can improve the transcoded video quality under the given delay and channel constraint. The proposed rate-control includes a picture-layer bit-allocation algorithm and a macroblock-layer rate-control algorithm. The picture-layer bit-allocation is based on the picture complexities estimated from both the picture complexities from the input bit-stream and the previous picture complexities in the output bit-stream. It allocates proper number of bits to different types of pictures and handles scene-changes effectively. It outperforms previous bit-allocation approaches based only on picture complexities calculated from the input bit-stream or only on the previous picture complexities in the output bit-stream. For the macroblock-layer rate-control, we propose an operational rate-distortion optimization approach based on the Lagrange Multiplier (LM) algorithm. The LM algorithm is a quasi-optimal solution to this rate-control problem, but is computationally too expensive for practical applications. Taking advantage of the properties of transcoding, low-bit-rate video, and the LM algorithm, we propose a approach to reduce the computational complexity of the LM algorithm. Simulations show that the proposed LM-based algorithm outperforms the H.263 TMN8 rate-control algorithm [10] in terms of video quality and buffer control. The additional computational load is light compared with the whole transcoding process.

This dissertation is organized as follows. In Chapter 2, we give a brief overview of the video coding and transcoding techniques. In Chapter 3, we discuss the motion re-estimation algorithms for heterogeneous video transcoding. In Chapter 4, we present the proposed rate-control algorithm based on the combined input/output picture complexities for video transcoding. In Chapter 5, we present the proposed LM-based macroblock-

layer rate-control algorithm. Concluding remarks and suggestions for future work are given in Chapter 6.

Chapter 2. Overview of Video Transcoding Techniques

2.1. Standard Video Encoders

There are currently several established video coding standards, including MPEG-1 [3], MPEG-2 [4], MEPG-4 [5], H.261 [11], and H.263 [2]. They are all based on the same framework: hybrid DCT (Discrete Cosine Transform) [12] and MCP (Motion Compensated Prediction) coding. Video sequences usually contain various redundancies in both temporal and spatial dimensions. Video coding uses intra-frame coding techniques to remove the spatial redundancies within a video frame, and uses inter-frame coding techniques to remove the temporal redundancies between video frames. In intra-frame coding, block-based DCT is used to remove the redundancies between nearby pixels. In inter-frame coding, MCP is first used to remove the temporal redundancies between frames, then DCT is applied to remove the remaining spatial redundancies in the prediction errors.

In this section, we first describe the generic hybrid MCP/DCT coding framework in the context of MPEG-1 video. Then other major video coding standards are briefly discussed. At the end, we discuss the techniques that are critical to the performance of standard-conforming encoders.

2.1.1. Hybrid DCT/MCP Video Coding and MPEG-1 Video

The MPEG-1 video-coding algorithm operates on video frames represented in the YC_bC_r color space. Images usually stored in the 24-bit RGB format must first be

converted to the 24-bit YC_bC_r format [13]: 8 bits for the luminance signal (Y), and 8 bits each for the two chrominance signals (C_b and C_r). The chrominance signals are sub-sampled 2:1 in both horizontal and vertical dimensions. The sub-sampling does not affect the video quality visually because the human eyes are less sensitive to chrominance signals than they are to luminance signals. Through sub-sampling, the 24-bit per pixel RGB representation is reduced to 12-bit per pixel YC_bC_r representation, which gives a 2:1 compression. This sub-sampled representation is called 4:2:0 YC_bC_r.

Each video frame to be coded is divided into non-overlapping *macroblocks*, each of which covers a 16x16 pixel area. Each macroblock consists of four 8x8 luminance (Y) *blocks* and two corresponding 8x8 chrominance blocks (one C_b block and one C_r block). Macroblocks are the units for MCP, and blocks are the units for applying DCT. An MPEG-1 video frame may be coded in one of three types: intra-frames (I-frames), forward predicted frames (P-frames), and bi-directional predicted frames (B-frames).

An I-frame is encoded independently without referring to other frames. Each block in an I-frame is first DCT-transformed into the frequency domain. Most energy of the block is compacted into the DCT coefficients in the lower frequency bands. After this, the DCT coefficients are quantized. Quantization is the only lossy part of the whole compression process other than the sub-sampling. The resulting quantized DCT coefficients are then entropy coded: the quantized DCT coefficients are first run-length coded in a zigzag order and then variable-length coded (VLC). After encoding, the I-frame needs to be reconstructed and stored in the frame memory for future reference. In

the reconstruction, the quantized DCT coefficients are inverse quantized and inversed DCT (I-DCT) transformed.

A P-frame is encoded relative to its past reference frame. A reference frame can be a P-frame or an I-frame. A macroblock in a P-frame may be encoded as an intra-macroblock or an inter-macroblock. An intra-macroblock is encoded like a macroblock in an I-frame. An inter-macroblock is encoded as a 16x16 area of the past reference frame, plus the corresponding differences (prediction errors) between the area and the current macroblock. A motion vector is used to specify the 16x16 area of the past reference frame. Motion vectors may include half-pixel values, where pixels are interpolated for the calculation of the half-pixel motion vectors. The differences between the 16x16 area of the past frame and the current macroblock are encoded using DCT, quantization, run-length coding, and VLC similar to the encoding of the I-frame. A macroblock may also be skipped when it results in a (0, 0) motion vector and all-zero error terms. The search for the motion vector (the one that gives the smallest error terms) is the most computationally intensive part of an MPEG-1 video encoder and critically affects the resulting video quality. The P-frame is also reconstructed and stored for future reference. In the reconstruction, the quantized DCT coefficients are first inverse quantized, then IDCT is performed to reconstruct the prediction errors, and at the end, the prediction errors are added to the reference-block to reconstruct the P-frame.

A B-frame is encoded relative to its past reference frame and future reference frame. The encoding of a B-frame is similar to a P-frame, except that the motion vectors may refer to areas in the future reference frame. For macroblocks that use both past and

future reference frames, the two 16x16 areas are averaged and then used for the prediction of the current macroblock. A B-frame is not used as reference in any circumstances.

A typical coded video sequence uses a mix of all three types of frames. Each individual frame is allowed to be of any type. Often, however, a fixed frame-type mixing pattern is repeated throughout the entire video stream for simplicity. An example of MPEG-1 video frame-type mixing pattern is shown in Figure 2. In the figure, the arrows indicate the prediction directions. This pattern (except the last I-frame) repeats in the video sequence. The order of the frames in the coded video stream is rearranged in a way that an MPEG-1 decoder can decompress the frames with minimum frame buffering. For example, an input sequence encoded with an IBBPBBP pattern will produce a coded stream with the pattern IPBBPBB.

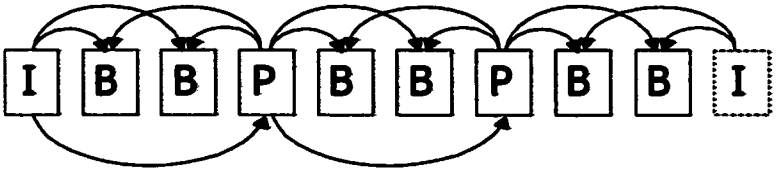


Figure 2. Example MPEG video frame-type mixing pattern.

A block diagram of a typical hybrid MCP/DCT video codec is shown in Figure 3. The encoding generates a variable bit-rate bit-stream. Therefore, an encoder-buffer is needed to buffer the bit-stream when the instantaneous output bit-rate is higher than the channel capacity. Basically, the encoder-buffer smoothes out the bit-rate so that the

averaged output bit-rate matches the channel bit-rate. To prevent the encoder-buffer from overflow or underflow, and to achieve the best overall video quality, a **rate-control** scheme is applied to adjust the quantization step-size so that consistently satisfactory video quality can be obtained without encoder-buffer overflow or underflow. The decoding is simply a reverse process of the encoding.

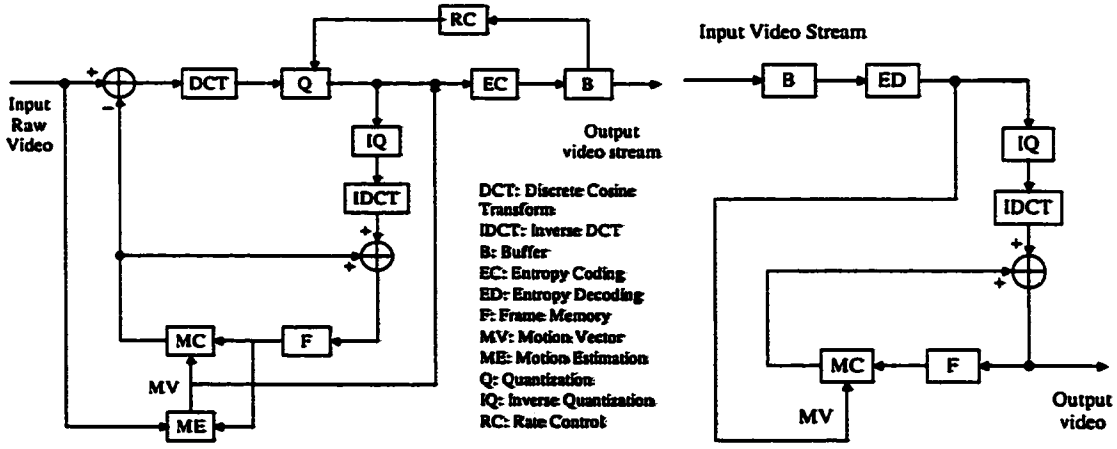


Figure 3. A hybrid MCP/DCT video codec.

A compressed MPEG-1 video stream is an ordered stream of bits, with a special bit-pattern conforming to the MPEG-1 standard syntax. The MPEG-1 video syntax structure is shown in Figure 4. Each video sequence is composed of a series of Group of Pictures (GOP). A GOP begins with an I-frame. An I-frame is an independently decodable unit and is intended to assist the random access into a video sequence. A video-frame consists of a series of slices. A slice is composed of a series of macroblocks, and a macroblock is composed of 6 blocks. The 6 blocks produce the entropy coded DCT coefficients. The slice structure allows decoding in the presence of transmission

errors. When a transmission error is detected, the decoder will search for the next slice header to recover the synchronization in the decoding. There can be one slice per frame, one slice per macroblock, or anything in between.

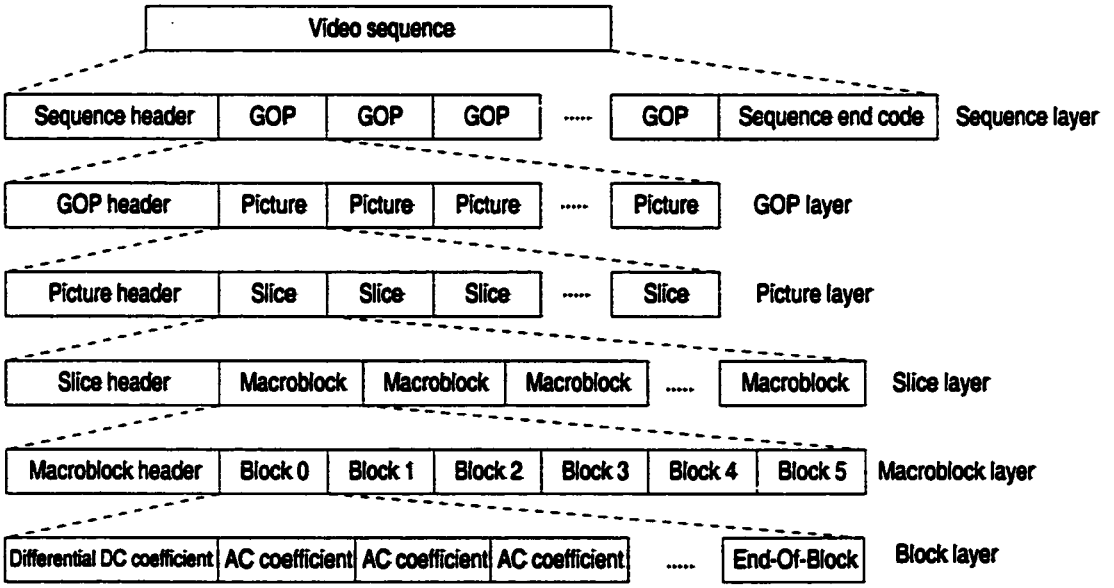


Figure 4. MPEG-1 video bit-stream syntax structure.

2.1.2. Overview of Other Major Video Coding Standards

H.261 was developed earlier than MPEG-1, in order to enable videoconferencing using ISDN channels. The major differences of H.261 from MPEG-1 are:

- H.261 only supports I- and P- frames due to the low-delay requirement.
- H.261 does not support half-pixel motion compensation.
- An optional loop filter [14] may be used to low-pass filter the reference frame, which usually decreases the prediction error and reduces the blockiness of the reference frame.

H.263 is an improved version of H.261, and is aimed at low-bit-rate (<64 kbps) video communications. The major improvements over H.261 include:

- Half-pixel motion compensation.
- Improved VLC coding.

In addition to these improvements, H.263 offers a list of optional features as annexes, including unrestricted motion vectors, advanced prediction mode, arithmetic coding, and deblocking filter. In order to enable the transport of H.263 video over unreliable networks, a set of tools have been developed for the purpose of error resilience, and have been included as additional annexes.

MPEG-2 enables MPEG-1 like functionality for interlaced video. The design target was to produce TV-quality pictures at data-rates of 4-8 Mbps and transparent quality pictures at 10-15 Mbps. MPEG-2 deals with high-quality coding of possibly interlaced video (e.g. SDTV or HDTV). A wide range of applications are addressed, including all forms of digital storage media, television broadcasting, and communications [16]. The major improvements of MPEG-2 over MPEG-1 are:

- MPEG-2 supports interlaced video sequences, and as a consequence, MPEG-2 allows additional scan patterns for DCT coefficients and motion compensation with blocks of 16x8 pixels.
- Additional techniques to improve the video coding efficiency, including ten-bit quantization for DC coefficients, non-linear quantization, and better VLC tables.

- MPEG-2 supports various modes of scalability, including spatial, temporal, and SNR scalability.

MPEG-4 is designed to address the requirement of the next generation interactive multimedia applications, while simultaneously supporting traditional applications. In MPEG-4, a Video Object (VO) corresponds to entities in the bit-stream that a user can access and manipulate. Instances of VO at a given time are called Video Object Plane (VOP). When the sequence has only one rectangular VOP of fixed size displayed at fixed interval, it corresponds to the frame-based coding technique, which is of our primary interest. Frame-based MPEG-4 video is similar to H.263, but supports extended tools. MPEG-4 video defines several profiles to address different applications. The major profiles are:

- Simple Profile (SP): It is based upon baseline H.263. MPEG-4 SP is targeted at low-bit-rate, low-delay video communications. It includes a set of error resilience tools.
- Advanced Simple Profile (ASP): It includes advanced coding tools such as Global Motion Compensation (GMC), ¼ pixel motion compensation, Bi-directional Video Object Plane (B-VOP), and interlaced video, etc, to provide better coding efficiency for low-bit-rate video communications.
- Streaming Profile: It uses the FGS (Fine Granularity Scalability) [9] encoding technique. An FGS video stream can be truncated at any point in its enhancement layers, and is particularly suitable for applications

involving streaming video over non-guaranteed Quality of Service (QoS) networks.

The complete description of MPEG-4 video profiles is referred to the MPEG-4 standards documents [5][17-18].

MPEG-4 AVC (Advanced Video Coding)/**H.264** [19]: It is the latest video coding standard and will be completed in the year 2003. It reflects the latest advances of video coding techniques. The major improvements over H.263 and previous MPEG-4 video coding profiles are: multiple reference frames, variable block-sizes (up to 16 MVs per MB), 4x4 integer DCT-like transform, improved intra prediction, arithmetic coding, 1/8 pixel interpolation, etc.

2.1.3. Video Coding Techniques

All these video coding standards only standardize the decoder, not the encoder. This allows the encoder performance to be further improved by technology advancements without affecting the interoperability. The technologies that make the most difference in the encoder performance include the motion estimation and the rate-control.

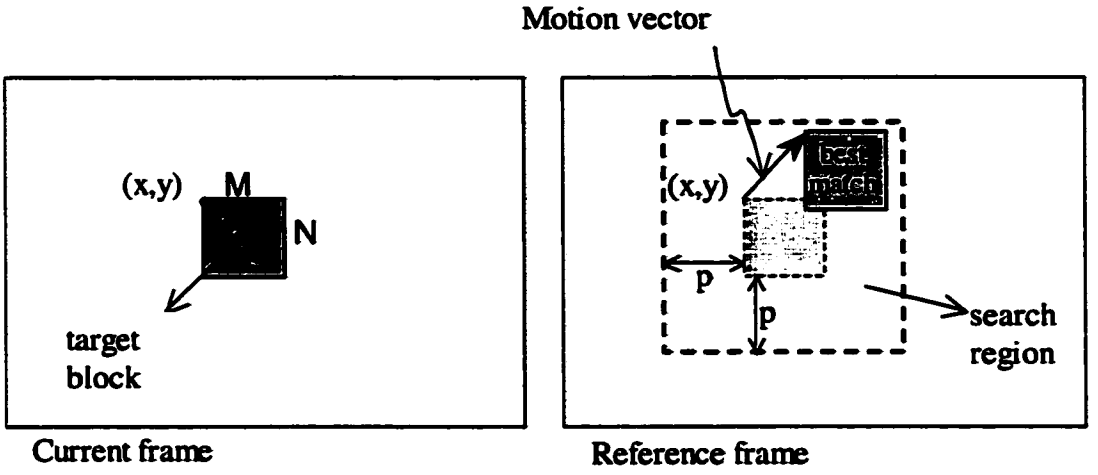


Figure 5. Block motion estimation.

Motion estimation is the technique to estimate the motion vectors in a video sequence. In most existing video coding standards, the motion estimation is block-based. Given a reference frame and an MxN macroblock in the current frame, the objective of motion estimation is to find the best-matched MxN block in the reference frame within a search region relative to the position of the current macroblock, as illustrated in Figure 5. SAD (Sum of Absolute Difference) is the most commonly used matching criterion, and it is defined as

$$SAD(i,j) = \sum_{k=0}^{M-1}\sum_{l=0}^{N-1}\left|P_C(x+k,y+l) - P_R(x+i+k,y+j+l)\right| \qquad (1)$$

where $P_C(x+k,y+l)$ are the pixels of the block in the current frame, $P_R(x+i+k,y+j+l)$ are the pixels in the reference frame, $-p \le i \le p$ and $-p \le j \le p$, and p determines the search-range.

Motion estimation is the most computation intensive operations in video coders. The exhaustive search (full-search) algorithm that searches every possible candidate block in the search range, gives the best performance. However, it is not suitable for many practical applications due to its high computational complexity. Because of this problem, various fast motion-estimation algorithms [20-21] have been developed, which trade off estimation accuracy for reduced computations. Three-Step Search [21] is one of the most popular fast motion-estimation algorithms. As shown in Figure 6, in each step, nine search-points are checked. After each step, the step-size is reduced by half, and the search ends with a step-size of one pixel. At each new step, the search center is moved to the best matching point from the previous step.

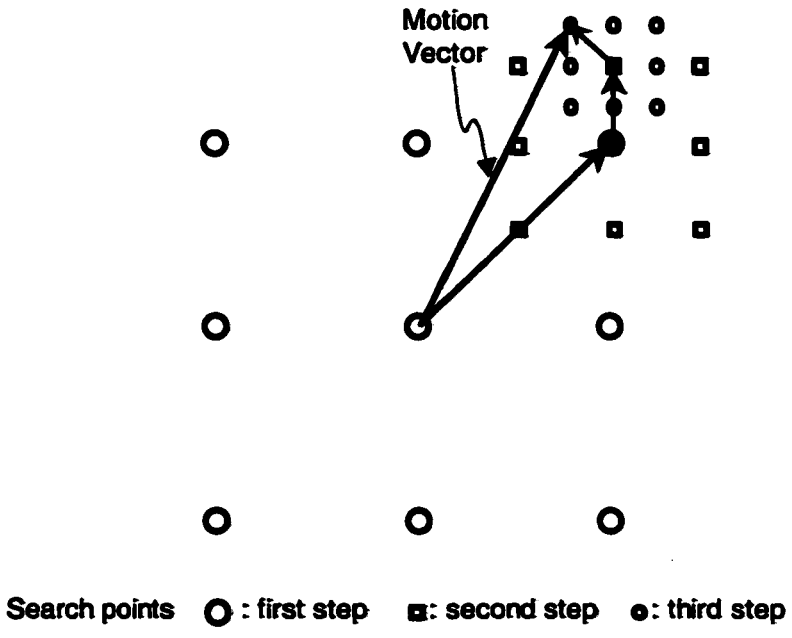


Figure 6. Three-step search method.

It has been shown that the half-pel-accuracy search can provide significant improvement over integer-pel-accuracy search [15], especially for the low-resolution video. In practice, half-pel positions around the best integer-pel are usually checked for improved performance.

For the rate-control, Figure 7(a) shows the variable bit-rate nature of the compressed video stream: using a fixed quantization parameter usually results in relatively constant video-quality and variable-bit-rate bit-streams. In order to transmit the video over a constant-bit-rate channel, such as the PSTN (Public Switched Telephone Network) or DSL lines, the encoder buffer is used to buffer the bit-stream when the instantaneous output bit-rate is over the channel capacity. To prevent the encoder buffer from overflow or underflow, the rate-control algorithm is used to adjust the quantization step-sizes to control the produced bit-rate. This results in variable Peak-Signal-to-Noise-Ratio (PSNR) as shown in Figure 7(b). Due to the rate-smoothing of the encoder buffer, the output of the buffer is at the same rate as the communication channel.

The challenge of the rate-control in video encoding is to determine the quantization step-sizes for the video frames to achieve the best video quality given the application constraints (channel bandwidth, delay, etc.). The rate-control, like the motion estimation, is not specified in the standards since they do not affect the interoperability of the encoder and the decoder. Currently, rate-control is still an active research area.

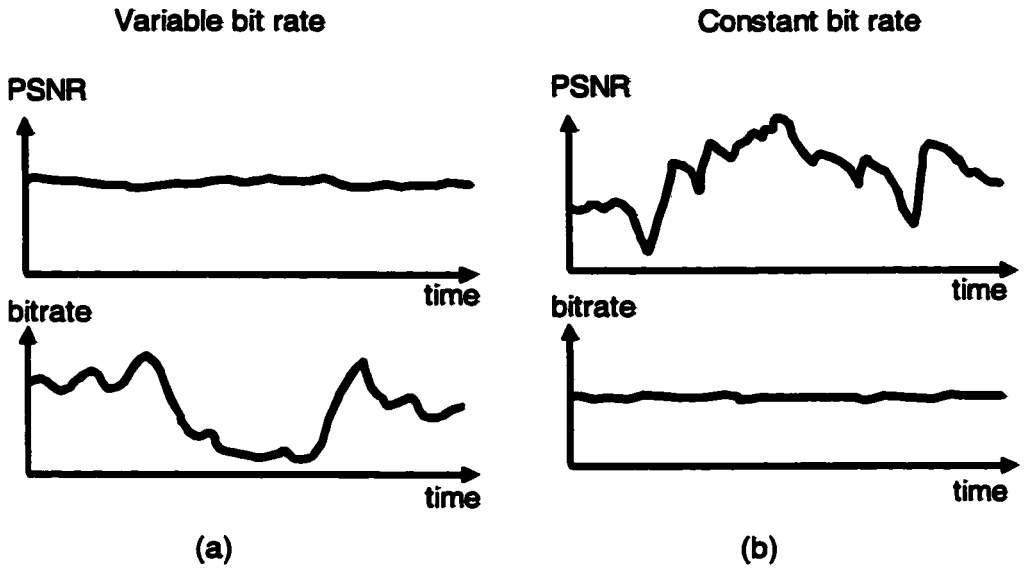


Figure 7. Variable bit-rate vs. constant bit-rate

2.2. Standard-Conforming Video Transcoding Techniques

A straightforward implementation of a video transcoder is to cascade a video decoder with a video encoder. The decoder decodes the input video into the pixel-domain, and then the encoder encodes the decoded frames into the desired compressed video formats. However, the computational complexity of this approach is very high since it requires to perform both the encoding and the decoding. In addition, decoding followed by encoding may introduce extra distortions to the video quality [36]. Thus, the challenge in the transcoding research is how to achieve the optimal tradeoff between the computational complexity, the video quality, and the flexibility. Different from conventional video coding, video transcoding has not only the decoded video frames, but also the coding statistics of the input compressed video. So, the key issue in transcoding is how to exploit this information to assist the video transcoding.

2.2.1. Bit-rate Transcoding

Bit-rate transcoding, which lowers the bit-rate of the compressed video without changing the frame-rate or the spatial resolution, has been studied extensively. The most studied topic in bit-rate transcoding is the efficient *transcoding architecture*. Basically, the architecture can be classified into three categories: open-loop transcoder [22-23], CPDT (Cascaded Pixel Domain Transcoder) [23-24] and DDT (DCT domain transcoder) [25-26].

Open-loop transcoder architectures include selective transmission or re-quantization. Selective transmission [22-23] discards high frequency DCT coefficients, and re-quantization [23][38] re-quantizes motion compensated residual errors to adapt the video to the bit-rate requirement. Both approaches operate on the DCT coefficients and are simple to implement. However, they suffer from the "drift" problem. In predictive coding, a coded video frame is predicted from other frames and only the prediction error (residual error) is coded. For the decoder to operate properly, the video frames reconstructed and stored in the decoder predictor must be exactly the same as those in the encoder predictor. Open-loop transcoder architectures change the residual errors and make the content in the decoder predictor different from that in the encoder predictor. This difference will accumulate with time until an intra-frame is received. The error accumulation resulted from the encoder/decode predictor mismatch is called "drift" and may cause severe degradations to video quality.

CPDT avoids the drift problem by compensating it in the architecture [23]. Figure 8 shows the block diagram of a CPDT transcoder. It is essentially a decoder followed by a simplified encoder. Rather than performing the motion estimation, as in a standalone video encoder, the simplified encoder reuses the motion vectors, along with other information, extracted from the input video bit-stream. Thus, motion estimation, the most time-consuming operation in the video encoding, which usually accounts for 60-70% of the encoder computation [27], is omitted. Its computational complexity can be further simplified by partially computing DCT and IDCT in the transcoding process [24].

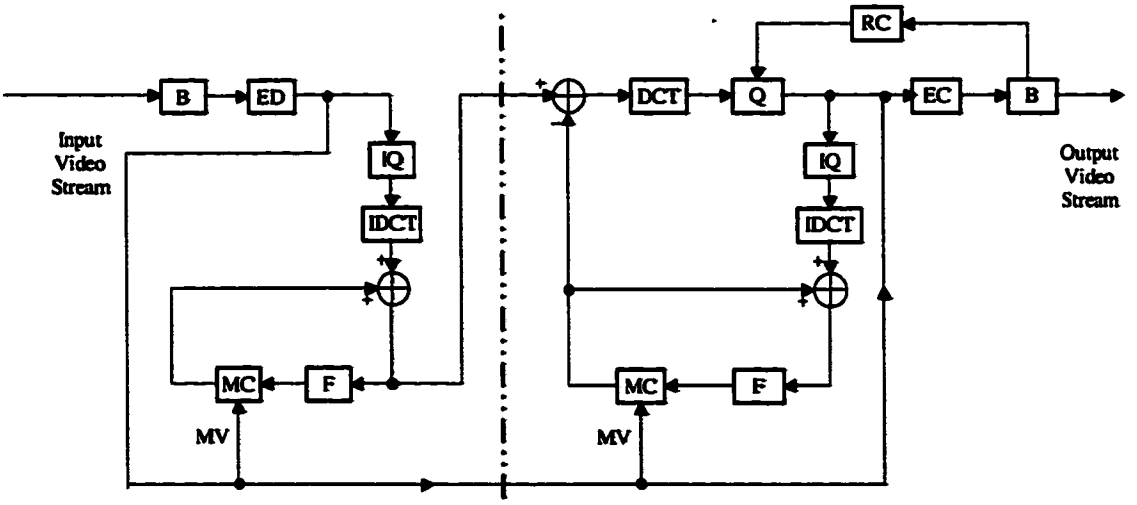


Figure 8. CPDT (Cascaded Pixel Domain Transcoder)

Assuming the linearity of DCT, IDCT, and MC, a structurally simpler but functionally equivalent **DDT** transcoder architecture was first derived in [25], and then further simplified in [26], as shown in Figure 9. The DCT-domain MC is the sole time-consuming operation in DDT [29], and several fast schemes have been

proposed in the literature [30-31]. It may require less computation than the CPDT. However, the assumptions on which the derivation based are not completely true, and thus, they may cause drift in the transcoded video [28]. In addition, it lacks the flexibility of CPDT: it can only be applied to bit-rate transcoding, it does not allow changes in spatial/temporal resolutions, motion vectors, or frame coding types.

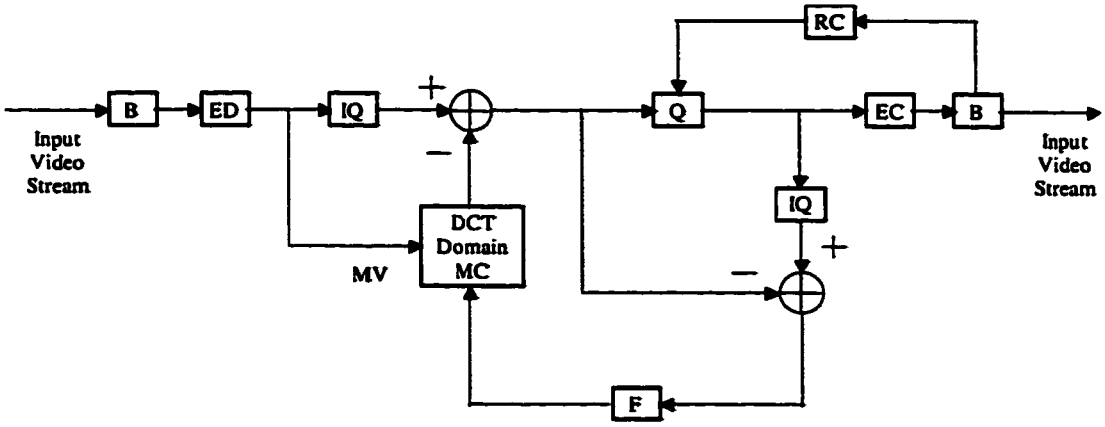


Figure 9. DDT (DCT Domain Transcoder)

2.2.2. *Spatial and Temporal Transcoding*

The heterogeneity of the present communication networks and the network access terminals demand the conversion of compressed video not only in the bit-rates, but also in the spatial/temporal resolutions and the coding formats.

In the spatial/temporal transcoding, the target video has different spatial/temporal resolutions from the input video, hence one challenging task is how to obtain the motion vectors for the target video without performing the computation-intensive motion

estimation. For the spatial transcoding, most previous works considered the 2:1 down-sampling. In 2:1 down-sampling, one target macroblock is down-sampled from four macroblocks in the corresponding input video frame. It has been found that the motion vector for this target macroblock is closely correlated to the motion vectors of the four input macroblocks, and different approaches have been proposed to compose the target motion vector using the four input motion vectors: any one (chosen randomly) [33-34], median [27], or weighted average [32]. Median is shown to give the best performance. Recently, the spatial transcoding for arbitrary down-scaling ratio is investigated in [43]. The challenge is how to incorporate the motion vectors with different contributions (correlations) to the target macroblock due to the non-integer-factor spatial down-sampling. In [43], the target motion vectors were composed as the weighted average of the motion vectors of all the input macroblocks from which the target macroblock is down-sampled. For transcoding with temporal resolution changes, one has to derive a new set of motion vectors that do not exist in the input video, and has to take into considerations the motion vectors of the dropped frames. A technique called Forward Dominant Vector Selection (FDVS) was proposed in [35]. The best-matched area in the dropped frame to the current macroblock overlaps with at most four macroblocks in the dropped frame. The motion vector of the macroblock with the largest overlapping portion is selected for the composition of the current motion vector. This process is repeated for all the frames dropped so that a new set of motion vectors is composed for the first frame after the frame dropping. A simpler method – Telescopic Vector Composition (TVC) [27] – accumulates all motion vectors of the corresponding

macroblocks of the dropped frames and adds each resultant composed motion vector to its correspondence in the current frame. It has been shown in [27] that the two methods in fact result in similar performance.

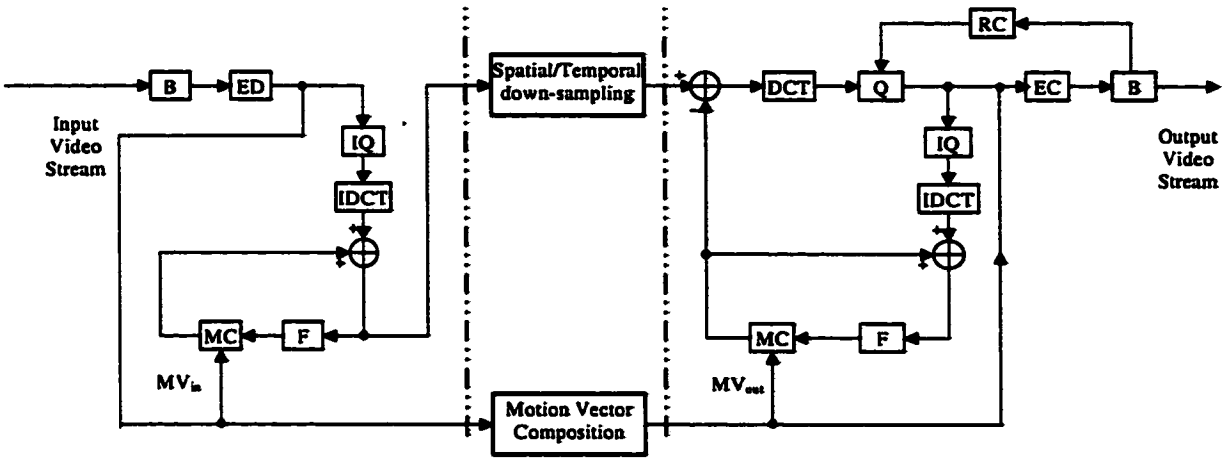


Figure 10. CPDT of spatial/temporal transcoding.

Due to the spatial/temporal resolution reduction, the drift problem is usually significant in the open-loop transcoding [39]. Therefore, drift-free CPDT architecture is more favorable in terms of quality. In Figure 10, with the CPDT for spatial/temporal transcoding, the motion vectors are obtained using the methods described in the previous paragraph, and the spatial/temporal down-sampling is performed on the reconstructed input video. Recently, several drift-compensation architectures were proposed in [40] to mitigate the drift problem for open-loop spatial transcoding. The intra-block-refresh architecture is claimed to provide good quality-computation tradeoff. However, this

architecture generates many intra-blocks and may perform poorly for low-bit-rate video applications.

2.2.3. Transcoding Techniques for Other Applications

Generally speaking, any operation that needs to change the compressed video bit-stream can be performed using a transcoder. Besides the bit-rate and spatial/temporal resolution reduction, transcoding is also useful in the following applications:

- Information insertion. For example, watermarks and company logos can be inserted into the compressed video using the transcoder [44] to protect the copyright.
- Transcoding for error resilience. When a pre-encoded video stream is delivered over a wireless channel, or a video is transmitted over combined wired and wireless channels, transcoder can be used before the video is sent over the wireless channel to insert error-resilience features into the video bit-stream to achieve robust video transmission [45-47].
- Syntax translation. Transcoding provides solutions for the incompatibility problem caused by the use of different video coding standards across different networking platforms. A very efficient syntax-translation between H.263 and MPEG-4 without bit-rate changes is discussed in [48]. Syntax translation with bit-rate and spatial/temporal resolution reduction [27][34] can be achieved by applying the techniques discussed in *2.2.1-2.2.2*.

2.3. Summary

In this chapter, we overviewed the coding techniques of video coding standards and the standard-conforming video transcoding techniques.

In Section 2.1, we gave an overview of the coding framework that all modern video coding standards are based upon: the hybrid MCP/DCT coding. Then, all major video coding standards were briefly described. At the end, the major video coding techniques to improve the performance of standard-conforming video encoders were discussed.

In Section 2.2, we reviewed the major research activities in the standard-conforming video transcoding area, including bit-rate transcoding, spatial/temporal transcoding, and other transcoding applications.

Chapter 3. Motion Re-estimation for Video Transcoding

3.1. Introduction

Practical applications often demand transcoding of video from one format into another. Two examples of such applications are: HDTV to SDTV transcoding and MPEG-2 to MPEG-4 transcoding.

HDTV broadcasting is expected to be increasingly popular in the coming decade. However consumers may not be willing to buy an expensive HDTV display device in the beginning. For these consumers to view HDTV programs on their Standard Definition TV (SDTV) screens, or to record the programs using their Digital Video Recorder, there is a need to convert the HDTV video streams into SDTV streams.

MPEG-2 video is by far the most successful digital video coding standard, and there are huge amount of MPEG-2 coded video contents. MPEG-4 Simple Profile (SP) video is suitable for low-bit-rate video communication over error-prone networks. For efficient delivery of pre-encoded MPEG-2 video over heterogeneous networks, it is often necessary to transcode the MPEG-2 video to the MPEG-4 SP format

HDTV and SDTV both use the MPEG-2 standard, but they have different spatial resolutions. Thus HDTV to SDTV transcoding needs to handle the spatial resolution reduction besides the bit-rate reduction. MPEG-2 to MPEG-4 video transcoding involves both spatial and temporal resolution reduction besides the bit-rate reduction, since they usually operate at different frame-rates, spatial resolutions, and bit-rates.

One of the most challenging tasks for spatial/temporal video transcoding is to efficiently and accurately re-estimate the motion vectors for the target video to avoid the computationally expensive full-search motion-estimation. In this chapter, we use the HDTV to SDTV transcoding as an example to present our motion re-estimation scheme for the spatial transcoding, and use the MPEG-2 to MPEG-4 SP transcoding as example to explain our scheme for joint temporal and spatial transcoding. These schemes can be easily extended to general transcoding scenarios.

3.2. Spatial Transcoding – HDTV to SDTV Transcoding

3.2.1. Problem Description

The challenges of the motion re-estimation for HDTV to SDTV transcoding include: non-integer factor spatial resolution down-scaling and interlaced video processing.

To support interlaced video, MPEG-2 video can be encoded in frame-pictures or field-pictures. For simplicity, in the following, we describe the proposed algorithm for the transcoding of P-frames. The extension to B-frames and field-pictures is straightforward. In the frame-picture coding, each macroblock in a frame-picture can use frame-prediction or field-prediction. In the frame-prediction, the macroblock is predicted from a block in the reference frame positioned by a motion vector. In the field-prediction, the macroblock is divided into two 16x8 blocks, one block belongs to the top-field, and the other block belongs to the bottom-field. Each 16x8 block has a field

selection bit which specifies whether the top or the bottom field of the reference frame is used, and a motion vector which points to the 16x8 pixel region in the appropriate field.

The problem of motion re-estimation for HDTV to SDTV transcoding is illustrated in Figure 11. Each target macroblock is down-sampled from a larger pixel block in the input picture, which is called the "mapped block" and is represented as the shadowed area in the figure. Macroblocks in the input picture overlapping with the mapped block are called the "candidate macroblocks". Each motion vector of a candidate macroblock is denoted in two characters: the first one is its index, and the second one specifies its reference frame/field selection. For example, the motion vector "1F" is the motion vector number 1, and uses the frame prediction. The next macroblock uses the field prediction, and has two motion vectors "2T" and "3B." "2T" is for the top field (since it is positioned in the upper-half of its macroblock) that is predicted from the top reference field, and "3B" is for the bottom-field that is predicted from the bottom reference field. In this figure, of the 16 candidate macroblocks, 12 are frame predicted and 4 are field predicted. There are 12 frame motion vectors and 8 field motion vectors.

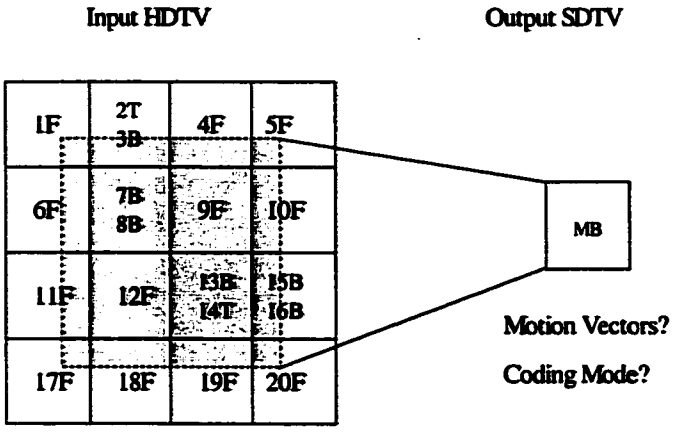


Figure 11. Motion re-estimation for HDTV to SDTV transcoding.

The problem we are investigating is how to efficiently derive the motion vectors of the target macroblock, given the input motion vectors, the coding modes, and the spatial-resolution reduction. An associated problem is how to determine the coding mode (inter or intra) for the target macroblock.

Most previous works on spatial transcoding, such as [27][32-33], were on 2:1 down-scaling of progressive video. Motion re-estimation for non-integer factor spatial resolution down-scaling was reported in [43], where the weighted-average was proposed for reusing motion vectors in progressive video transcoding. However, weighted-average would not produce accurate motion vectors when one or a few candidate motion vectors are not consistent with the majority. MPEG-2 interlaced video to H.263 video transcoding is discussed in [34]. However, it does not address the case when both the input and the output video could be interlaced. In addition, its motion vector reusing strategy is not reliable. In [34], one motion vector is arbitrarily chosen from a set of input motion vectors that are most relevant to the target motion vector. The arbitrarily chosen

single input motion vector characterizes the motion of only a small part of the target macroblock considering the spatial resolution reduction, thus, it is not a reliable estimation of the target motion vector.

The remaining part of this chapter is organized as follows. First, the transcoder architecture adopted for implementing the HDTV to SDTV transcoding is described. Then, the motion re-estimation and the mode decision schemes are presented. At the end, the simulation results are provided.

3.2.2. Transcoding With Spatial Resolution Reduction

Cascaded Pixel Domain Transcoder (CPDT) is adopted here as the transcoder architecture for the HDTV to SDTV transcoding, as shown in Figure 10. However, it should be noted that the motion re-estimation and mode decision algorithms presented in this section are also applicable to other transcoding architectures such as DCT-Domain Transcoder (DDT).

The down-sampling filter changes the picture from the HDTV resolution to the SDTV resolution. Using the conversion from 1920x1024i ("i" stands for "interlaced") to 720x384i as an example, the down-sampling factor is 1920:720 = 8:3. The same factor is applied vertically such that the resultant effective resolution is 720x384. Same down-sampling ratios are applied to both horizontal and vertical dimensions in order to preserve the aspect ratio. Each field is processed individually. The 8:3 down-sampling factor can be achieved by up-sampling by a factor of 3 then down-sampling by a factor of 8, as shown in Figure 12 (L=3, M=8), where $h(v)$ is a low-pass filter [49]. We adopt a similar

filter design as in [43]. The filter can be efficiently realized by a polyphase implementation.

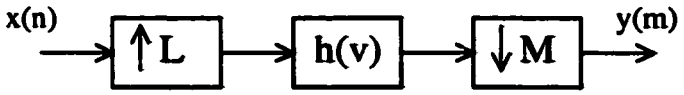


Figure 12. The interpolation-decimation routine for a rate-change of M:L.

3.2.3. Motion Re-estimation

For each target motion vector (frame vector or field vector), the proposed motion re-estimation scheme consists of two steps: first we find a set of candidate motion vectors that correlate with the target motion vector, then we compose a motion vector from the set of candidate motion vectors, and refine it.

One strategy of forming the candidate motion vector set for the target SDTV macroblock is to choose those input motion vectors that are most related to the target motion vector (i.e., using the same prediction mode) [34]. Here we call such motion vectors *normal candidate motion vectors*. In Figure 11, for the frame prediction, the target motion vector is for the target macroblock to predict from the reference frame, so, the normal candidate motion vectors are {1F, 4F, 5F, 6F, 9F, 10F, 11F, 12F, 17F, 18F, 19F, 20F}. In the field prediction, the normal candidate motion vectors are:

- {2T}, when the top field is predicted from the top reference field;
- {7B, 13B, 15B}, when the top field is predicted from the bottom reference field;
- {14T}, when the bottom field is predicted from the top reference field;

- {3B, 8B, 16B}, when the bottom field is predicted from the bottom reference field.

As can be noticed, when the top field is predicted from the top reference field, or when the bottom field is predicted from the top reference field, there is only one normal candidate motion vector available. Since one such motion vector characterizes the motion of only a small part of the mapped block, it is not reliable to serve as the motion vector of the target macroblock. Moreover, sometimes it is possible that no normal candidate vector exists.

The situation that few normal candidate motion vectors exist takes place more frequently in the frame-picture transcoding than in the field-picture transcoding. It is due to the following reason: In the frame-picture transcoding, for each macroblock in the output SDTV, five target motion vectors need to be estimated, each of which requires a prediction that has a different target reference field/frame combination – frame prediction; prediction of the top-field from the top reference-field; prediction of the top-field from the bottom reference-field; prediction of the bottom-field from the top reference-field; and prediction of the bottom-field from the bottom-reference field. However, in the field-picture transcoding, there are only two such types of predictions: prediction of the current field from the top reference-field and prediction of the current field from the bottom reference-field.

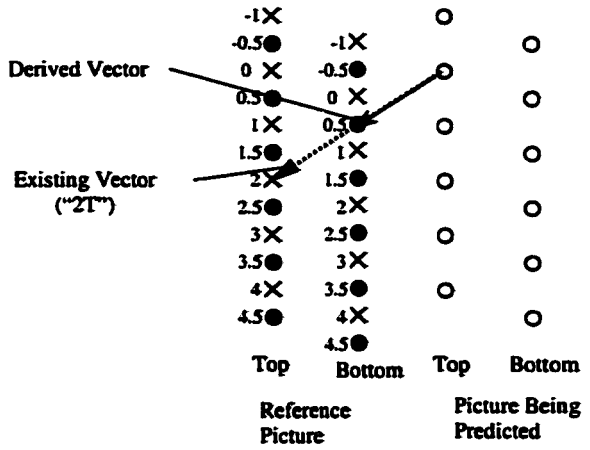


Figure 13. Derive a field motion vector for different reference fields.

To enhance the accuracy of the composed motion vectors, more proper candidate motion vectors are needed. Indeed, motion vectors referring to the field of the other parity can be used to derive such candidate motion vectors. Suppose the top field is being predicted from the bottom reference-field. Look at the second candidate macroblock in the first row in Figure 11. For its top field, the motion vector we have, "2T", refers to the top reference-field, and thus is not a normal candidate motion vector. However, if we assume that the motion is linear over a field interval, which is reasonable since a field interval is a short period of time, we can derive a motion vector for the bottom reference field by properly scaling the motion vector "2T". Figure 13 illustrates the derivation when the distance between the current frame and the reference frame is one frame. In general, the illustrated derivation can be expressed in the following equation:

$$mvY_{top->bot} = (\frac{2\times frame\_dist - 1}{2\times frame\_dist}) \times mvY_{top->top} - 0.5$$
$$mvX_{top->bot} = (\frac{2\times frame\_dist - 1}{2\times frame\_dist}) \times mvX_{top->top} \quad (2)$$

where *frame_dist* is the frame distance between the frame being predicted and the reference frame, $mvY_{top->bot}$ and $mvX_{top->bot}$ are the vertical and horizontal components of the motion vector of the top field referring to the bottom reference-field respectively, and $mvY_{top->top}$ and $mvX_{top->top}$ are the vertical and horizontal components of the motion vector of the top field referring to the top reference-field respectively. The result is rounded to the integer resolution and the refinement process is performed to achieve the half-pixel precision.

With a field motion vector, we can derive the field motion vector for the reference field of the other parity. Similarly, a frame motion vector can be formed from a field motion vector and vice versa. The motion vectors derived in this way, together with the normal candidate motion vectors, are called *extended candidate motion vectors*. It is worthwhile to point out the intra candidate macroblocks are not used in forming the extended candidate motion vectors since no motion information is associated with it.

With all these candidate motion vectors, the next problem is to compose a single motion vector from them.

It is proposed in [43] to use the weighted average of candidate motion vectors, with the weight of each motion vector being the overlapping area between its corresponding block (16x16 or 16x8) and the mapped block. The drawback of this approach is that the result is prone to impulse noise in candidate motion vectors.

On one hand, it is well known that median filter has good performance against impulse noises. On the other hand, the candidate motion vectors have different degrees of correlations with the target motion vector. So we propose to use weighted median to perform the composition. The weight is determined in the same way as in the weighted average.

For either algorithm, the composed motion vector needs to be scaled by the spatial resolution downscaling factor. The two approaches are expressed in the following two equations, where mv' denotes the estimated motion vector, R is the down-sampling ratio which generally is a vector: $[R_x, R_y]$, $\{mv_i\}$ is the set of candidate motion vectors, w_i is the weight of mv_i where $i=1,2,...,N$, with N being the total number of candidate motion vectors, and $\|\cdot\|$ denotes the distance measure where the absolute difference is used.

Weighted average:

$$mv' = \left(\frac{\sum_{i=1}^{N} mv_i w_i}{\sum_{i=1}^{N} w_i} \right) ./ R \tag{3}$$

Weighted median:

$$mv' = (mv_m)./ R \tag{4}$$

where mv_m is the weighted median of the candidate motion vectors:

$$\left\{ \sum_{i=1}^{N} w_i \| mv_m - mv_i \| \leq \sum_{i=1}^{N} w_i \| mv_j - mv_i \| \quad j = 1,2,...,N \right\}_{mv_m \in \{mv_i\}} \tag{5}$$

In the two equations, "./" is the "right array divide" operation. For example: $mv./R \equiv [mv_x/R_x, mv_y/R_y]$.

To further improve the accuracy, the estimated motion vector can be refined by searching the surrounding half-pixel positions.

3.2.4. Coding Mode Decision

Typically, in video encoding, mode-decision is performed after the motion estimation is finished. However, in video transcoding, it is possible to determine some of the macroblock modes directly from the mode decisions of the input video. This also saves the computations of the motion estimation.

The mode decision process can be broken into three serial decisions:

- Inter-intra decision;
- Prediction direction decision: forward, backward or bi-directional predicted; (This step does not exist in P pictures)
- Prediction mode decision: frame or field predicted for frame-pictures; field or 16x8 predicted for field-pictures.

The simple "Go-With-Majority" strategy is applied to the inter-intra decision. When the larger part of the mapped block is intra coded, the target macroblock is decided to be intra coded. Otherwise, it is inter coded.

When inter coding is used, the decision on prediction direction goes with the one that is used by the largest part of the mapped block. The final decision of prediction mode goes with the one that has the same prediction direction as the target direction (found through the second step) and is used by the largest part of the mapped block. It is

easy to see that the computation of the proposed decision strategy is very simple and is negligible compared with the complexity of other parts of the video transcoding.

With this proposed approach, when a macroblock is determined to be intra coded, there is no need to perform the motion estimation at all. Also, in a B picture, for macroblocks that are determined to be forward (backward) predicted, the backward (forward) motion estimation can be skipped. Similarly, for macroblocks in a frame-picture that are determined to be frame predicted, the motion estimation for field prediction can be avoided.

3.2.5. Simulation Results

In the simulations, the 1920x1024i HDTV sequences "Tennis" and "Basketball" are used. The implementation of the transcoder software is based on the MPEG-2 TM5 (Test Model 5 [50]) video codec developed by the MPEG Software Simulation Group (MSSG). It is available online at http://www.mpeg.org/MPEG/MSSG/.

The "Tennis" and the "Basketball" sequences are encoded using frame-pictures at 30 frames/s and 20 Mbps, with the GOP structure (15,3). They are transcoded to frame-pictures of SDTV resolution 720x384i at 4 Mbps. The frame-rate and the GOP structure are not changed.

We first compose motion vectors from only normal candidate motion vectors. The simulation results are shown in Figure 14. Figure 15 shows the performance of both composition strategies for the "extended" candidate motion vectors – WME (Weighted Median of Extended motion vectors) and WAE (Weighted Average of Extended motion

vectors). For comparison, simulation results for both normal and extended candidate motion vectors are also given in Table 1. The weighted median proves to be superior to the weighted average, and both strategies are improved when "extended" candidate motion vectors are used. The average performance gain of the weighted median over the weighted average for the "extended" candidate motion vectors is about 0.3~0.5 dB. For weighted median, the average gain of using "extended" candidates over "normal" candidates is also around 0.4 dB.

Table 1. Performance of WA and WM, WME and WAE.

Re-estimation Strategy	*Average PSNR (dB)*	
	Tennis	*Basketball*
WA	30.73	33.33
WM	30.92	33.57
WAE	30.97	33.36
WME	31.35	33.90

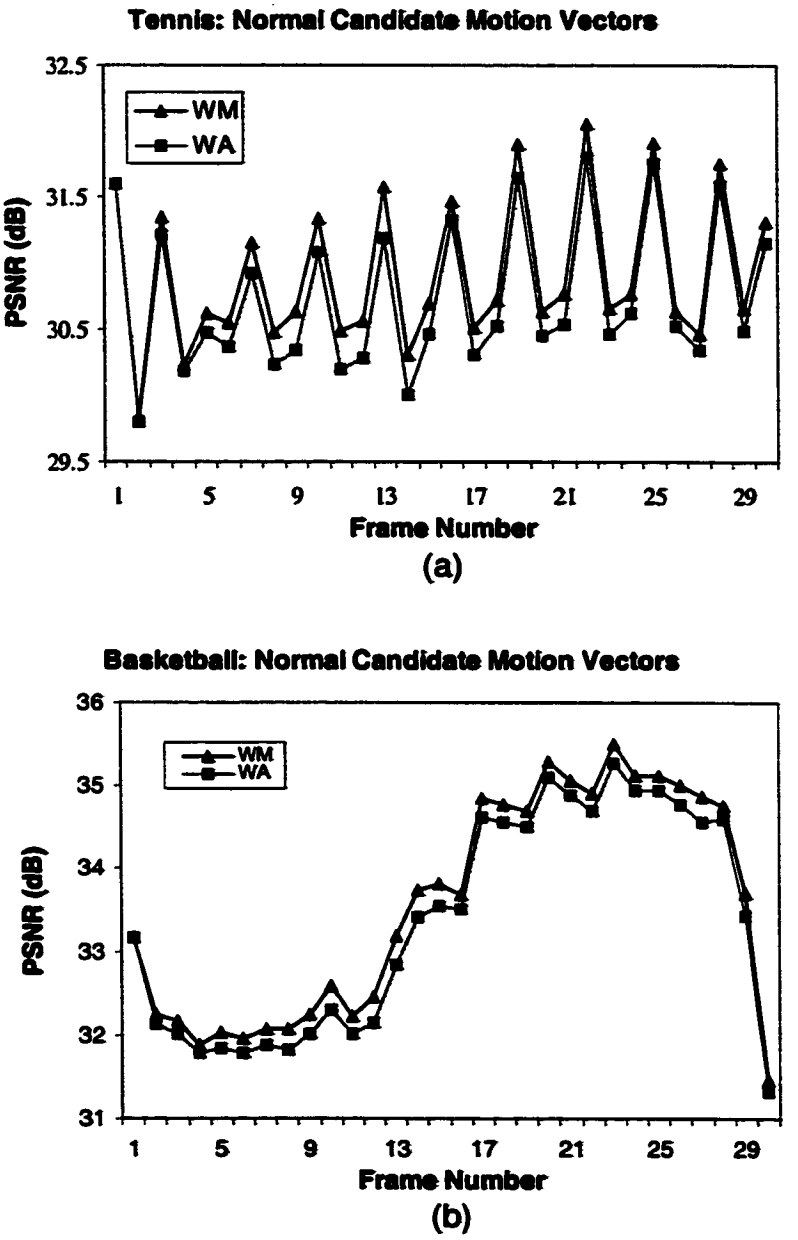


Figure 14. Performance of "normal" candidate motion vectors. (WA-weighted average, WM-weighted median): (a) Tennis sequence; (b) Basketball sequence.

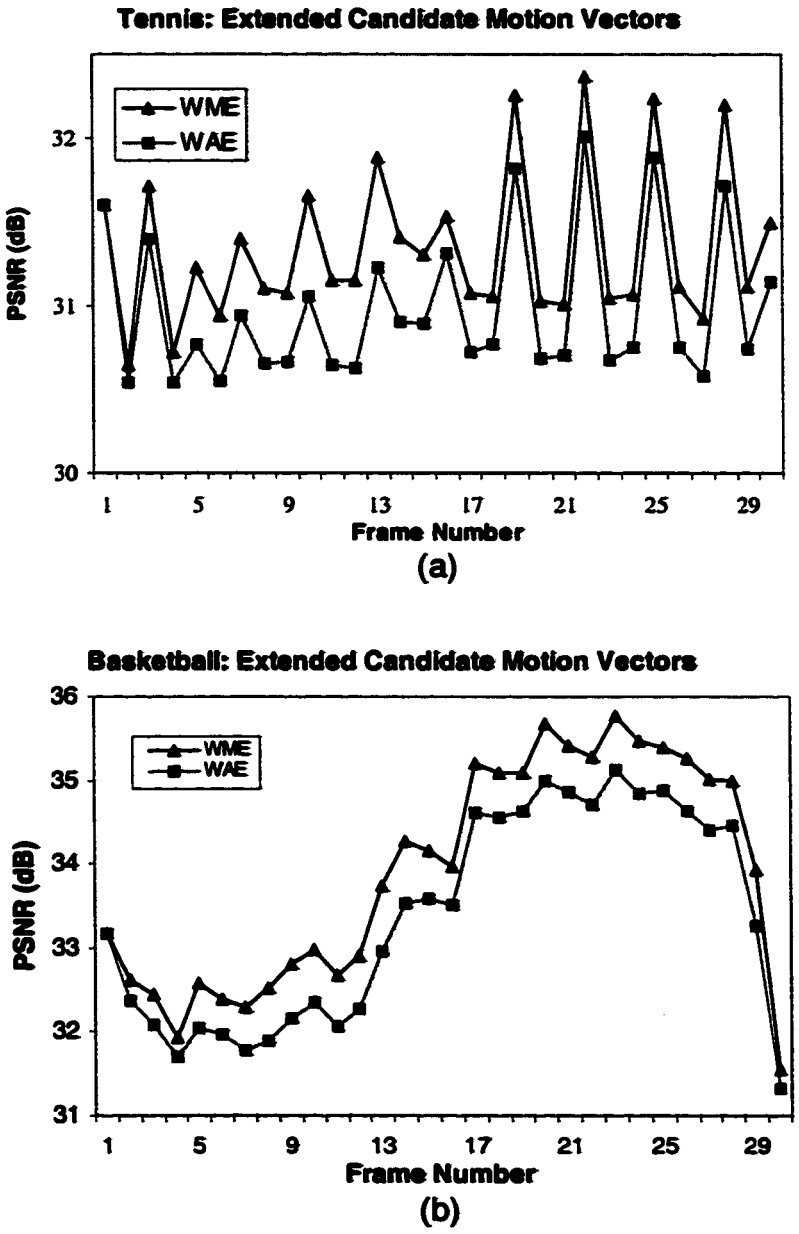
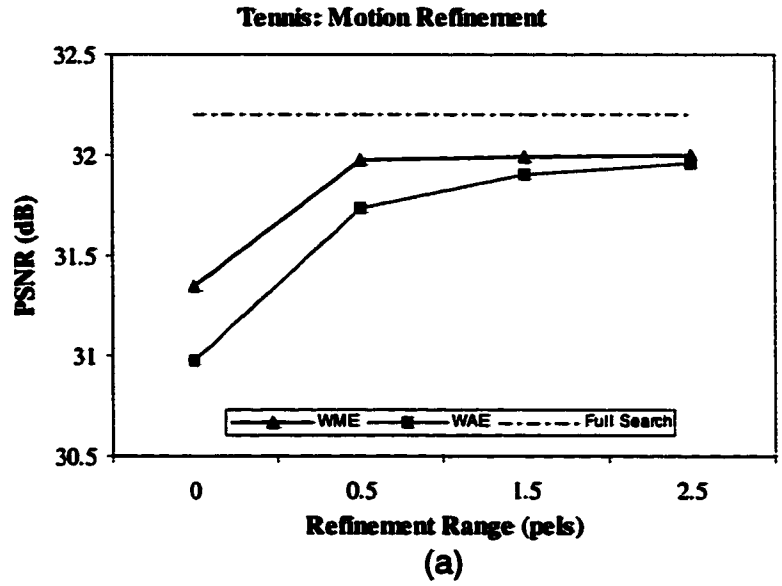
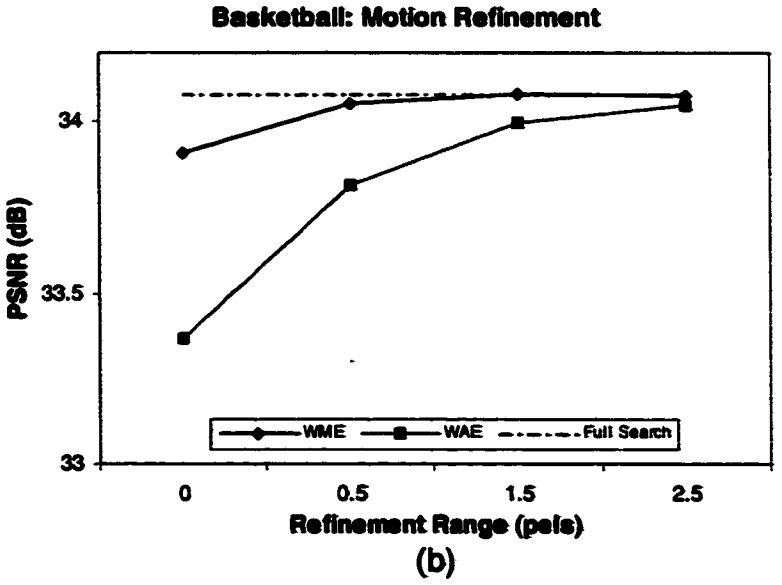


Figure 15. Performance of "extended" candidate motion vectors. (WME-weighted median of extended motion vectors; WAE-weighted average of extended motion vectors): (a) Tennis sequence; (b) Basketball sequence



(a)



(b)

Figure 16. Motion Vector Refinement. (a) Tennis sequence; (b) Basketball sequence.

The performance of the composed motion vector can be further improved by applying an additional refinement surrounding the composed motion vector. As shown in Figure 16, half-pixel refinement is found to be good enough; and further refinement does not improve performance significantly but consumes much more computation. The full-search algorithm finds motion vectors that are optimal in terms of minimizing the matching error, and are used for performance benchmarking. Figure 16 shows that the performance gap between the full-search and the proposed re-estimation algorithm with half-pixel refinement is less than 0.2 dB in terms of PSNR.

As for the computation, in our implementation (running on a PC with Pentium IV 1.2 GHz, 384 MB memory, Windows 2000), the transcoder using the weighted median motion re-estimation with half-pixel refinement takes only one third of the time needed by the transcoder that re-encodes the down-sampled video using the full-search motion estimation (the motion vector search range is set to ±8 pixels per frame, thus ±24 pixels for the P pictures). In other words, two thirds of the computation is saved with negligible quality loss. Note that for larger search ranges, the computational savings would be even greater.

To further save computation, some macroblock coding modes could be derived from the incoming macroblock modes instead of being re-decided using TM5. The impact of reusing incoming macroblock modes is shown in Figure 17. The four strategies shown in the figure are: 1) all modes are re-decided using TM5; 2) inter/intra decision is reused from the incoming macroblock modes, with other decisions using TM5; 3) inter/intra and prediction direction decisions are reused from the incoming

modes, with the field/frame decision using TM5; 4) Intern/intra, prediction direction and field/frame decisions are all reused from the input. The figure shows that the inter/intra decisions and the prediction direction can be reused from incoming modes with small sacrifice of quality, while the frame/field decisions reusing may cause quite significant quality loss and are not recommended. In this simulation, using the proposed mode-reusing strategy 3, on average about 30% of the macroblocks in B-pictures are decided not to be bi-directionally coded, thus for the macroblocks, half of the motion re-estimation computation is saved. That is about 15% saving in the motion re-estimation computation.

Based on the above simulation results and analyses, we recommend the following algorithms for the transcoding of HDTV to SDTV. Weighted median of extended candidate motion vectors gives accurate estimation of the motion vectors and is simple to compute, thus is regarded as a suitable approach to replace the full search re-estimation algorithm. The majority-based mode-reusing scheme is appropriate for the decision of the inter/intra type and prediction direction, but is not recommended for the frame/field coding-mode decision due to its inferior performance.

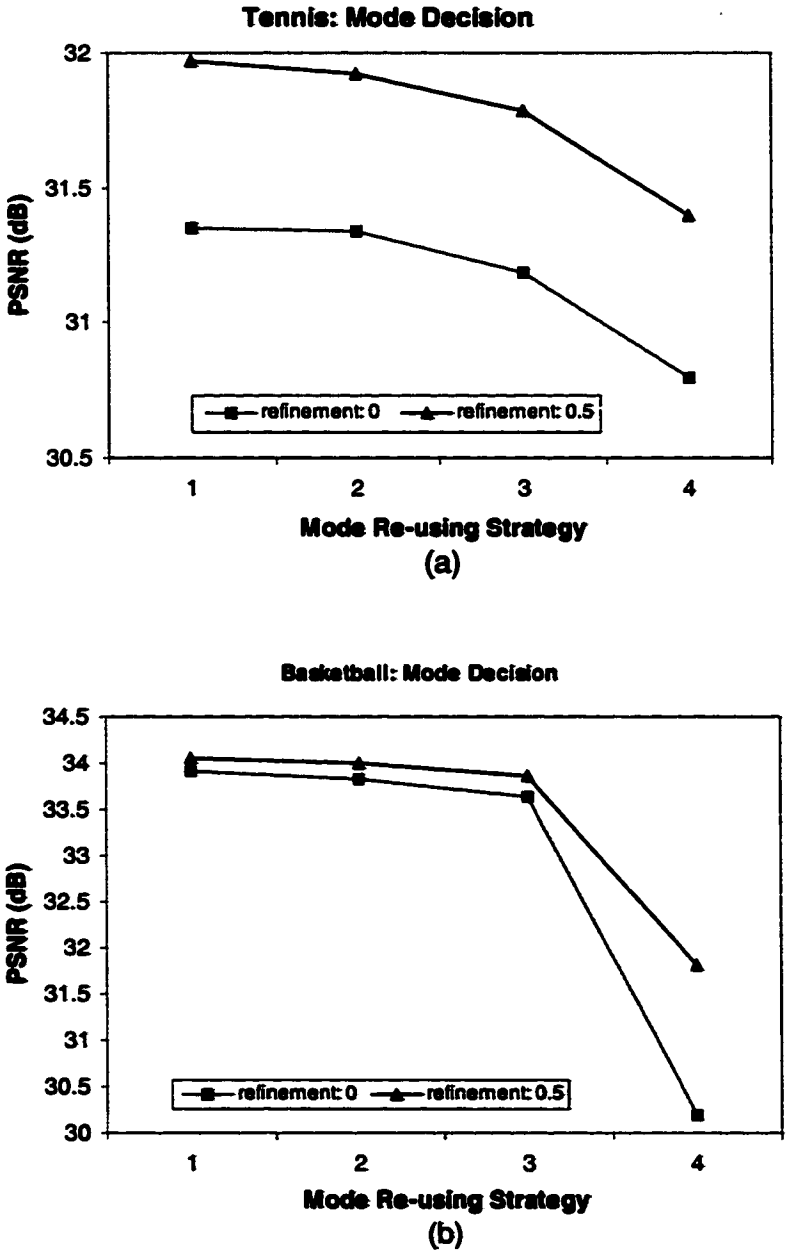
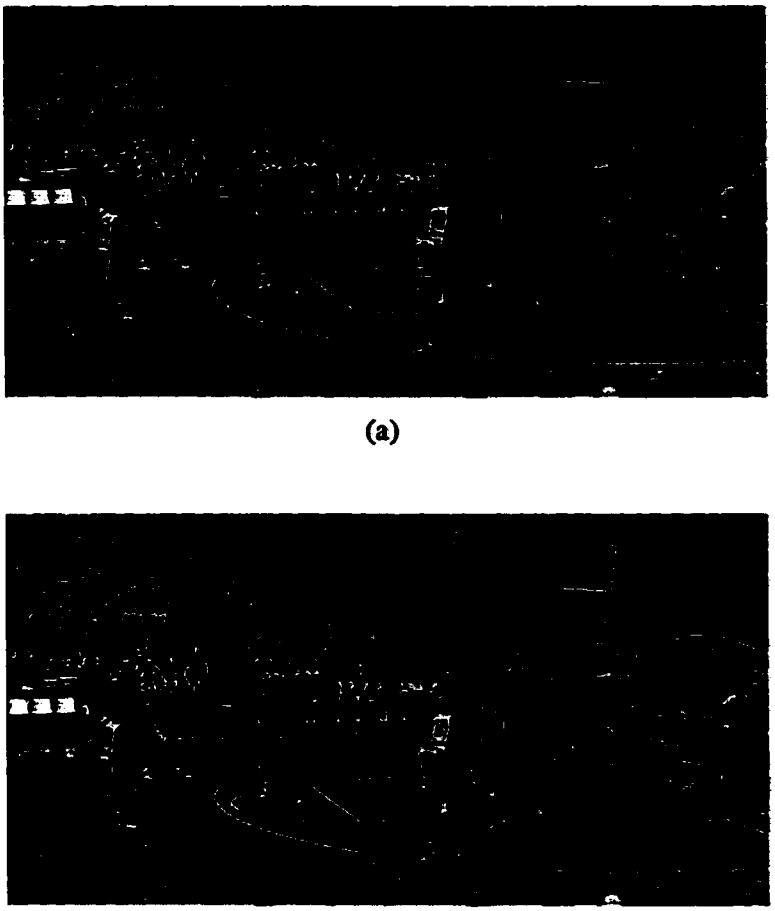


Figure 17. Performance of different macroblock mode re-using strategies: (a) Tennis sequence; (b) Basketball sequence.



(a)

(b)

Figure 18. Decoded frame using (a) the proposed transcoding algorithms; (b) the straightforward transcoding (decoder + encoder with full-search motion estimation).

The simulation results have shown that transcoding using the recommended algorithms can achieve similar performance to the straightforward transcoding (decoding followed by re-encoding) using full-search and TM5 coding-mode decisions with much reduced computational requirements. Figure 18 shows two sample corresponding frames of the basketball sequence generated by the two transcoding approaches. Subjective evaluations confirm that they exhibit virtually the same visual quality.

3.3. Joint temporal and spatial transcoding – MPEG-2 to MPEG-4 Transcoding

3.3.1. Problem Description

MPEG-4 Simple Profile (SP) video usually operates at not only lower spatial resolutions but also lower frame-rates than MPEG-2 video. In addition, MPEG-4 SP does not support the interlaced video. Therefore the MPEG-2 to MPEG-4 video transcoding has to handle the frame-rate reduction and the spatial resolution reduction at the same time. In addition, interlaced to progressive video processing also needs to be taken care of.

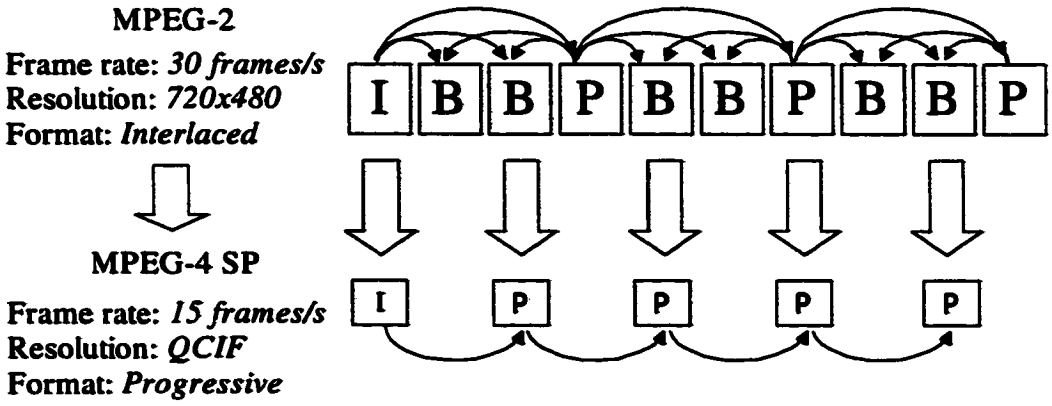


Figure 19. MPEG-2 to MPEG-4 transcoding

A typical scenario of the MPEG-2 to MPEG-4 transcoding is illustrated in Figure 19. Note that the frame-rate reduction brings up another problem: frame-type conversion. The new challenge is that the motion vectors of an incoming video frame may not use the same reference frame as the target frame. The FDVS [35] and TVC [27] techniques are proposed to form the target motion vectors when the input and output video use all P-

frames. The frame type conversion is addressed in [27], where a number of candidate motion vectors are formed for each target macroblock by using motion information from the current and adjacent frames, and the one giving the minimum matching error is taken. This approach is computational expensive since it tests all the candidate motion vectors for each target macroblock. To our knowledge, no work is reported to address the joint frame-rate reduction, frame-type conversion, and spatial-resolution reduction.

In our proposed approach, the frame-rate reduction and the spatial-resolution reduction are decoupled into two consecutive steps. We introduce an intermediate, virtual layer of video that has the same frame-rate and frame-type as the target video and the same spatial-resolution as the input video. In the first step, we form one or two motion vectors for each macroblock in the intermediate video frame using input motion information. In the second step, we compose the motion vectors for the target video from the motion vectors of the intermediate video. Effectively, the first step handles the frame-rate reduction and the frame-type conversion, and the second step handles the spatial-resolution reduction.

The remaining part of this section is organized as follows. First, the transcoding architecture is described. Then, the proposed motion re-estimation schemes for joint temporal and spatial resolution reduction are presented. At the end, the simulation results are provided.

3.3.2. *Transcoding for Joint Temporal and Spatial Resolution Reduction*

CPDT, as shown in Figure 10, is used as the transcoder architecture for the MPEG-2 to MPEG-4 SP transcoding as explained in previous sections. The motion re-estimation algorithm presented in this section is also applicable to all other transcoding architectures.

The temporal resolution reduction is achieved by discarding frames, as illustrated in Figure 19. The configuration in the figure is used throughout the following discussions: the input 720x480i MPEG-2 interlaced video has two B-frames between reference frames, coded at 30 frames/s, and the target QCIF MPEG-4 SP video is coded at 15 frames/s. The techniques discussed are general and can be applied to other situations. Although some P-frames in the input video stream do not exist in the output, they have to be decoded to decode its adjacent B-frames. The skipped B-frames can be simply discarded. We follow the recommendation in the MPEG-4 VM17 [51] to down-sample the video frames. The first field of a frame is omitted, then, appropriate down-sampling filters are applied to the physically centered 704x240 pixels to generate the smaller size pictures.

3.3.3. *Candidate Motion Vectors for Temporal Resolution Reduction*

As illustrated in Table 2, three different picture-type conversion patterns exist for such transcoding. The patterns repeat periodically due to the periodic input video GOP structure. The three patterns are classified according to the relative position of the input frame in its GOP, as will be discussed in the following in details.

Table 2. Picture type conversions in MPEG-2 to MPEG-4 SP transcoding.

Input frame index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Input	I	B	B	P	B	B	P	B	B	P	B	B	P	B	B	I	B
Ouput	I		P		P		P		P		P		P		P		P
Ouput frame index	0		1		2		3		4		5		6		7		8
Conversion Pattern			1		2		3		1		2		3		1		2

When there are frame-rate reductions, new motion vectors need to be estimated. For example, in Table 2, the input picture $B_{4,in}$ does not have motion vectors using $B_{2,in}$ as the reference frame, while such motion vectors are necessary for the transcoding ($B_{2,in}$ is input frame number 2, B picture, and $B_{4,in}$ is input frame number 4, B picture.) In this situation, not only the motion information of $B_{4,in}$, but also the motion information of $B_{2,in}$ will be used to derive the motion vectors for $P_{2,out}$.

We break this motion vector re-estimation problem into two steps: 1) frame-rate reduction, and 2) spatial resolution reduction.

In the following, we explain the general principles to form the motion vectors of the intermediate video frame for the three different patterns of picture type conversions.

The conversion of $B_{2,in}$ to $P_{1,out}$ belongs to pattern 1. The target motion vectors for the intermediate frame here are forward MV from $B_{2,in}$ to $I_{0,in}$. They are formed depending on the input macroblock type, as explained in Table 3, where $MV_{n->m}$ represents the motion vector going from frame n to frame m. The vector addition uses the TVC [27] approach, i.e., the motion vectors of macroblocks at the same spatial locations are added together.

Table 3. Forming motion vectors for picture-type conversion pattern 1: $B_{2,in}$ to $P_{1,out}$.

Input Macroblock Type	Derived Motion Vectors
Forward	$MV_{2\rightarrow 0}$
Backward	$MV_{3\rightarrow 0}+MV_{2\rightarrow 3}$
Bi-directional	$MV_{2\rightarrow 0}$ and $MV_{3\rightarrow 0}+MV_{2\rightarrow 3}$
Intra	NOMV

For the second type of picture-type conversion, i.e., $B_{4,in}$ to $P_{2,out}$, the target motion vectors are from $B_{4,in}$ to $B_{2,in}$. When the corresponding macroblocks in $B_{4,in}$ and $B_{2,in}$ both have backward motion vectors, the derivation is as follows:

$MV_{4\rightarrow 3}$-$MV_{2\rightarrow 3}$

$B_{4,in}$ and $B_{2,in}$ are just one frame away from frame $P_{3,in}$ in time, so, for most macroblocks $MV_{4\rightarrow 3}$ and $MV_{2\rightarrow 3}$ exist and the target motion vector can be formed. In case one of these two motion vectors does not exist, they can be formed as follows:

$MV4\rightarrow 3 = MV6\rightarrow 3 + MV4\rightarrow 6;$

$MV2\rightarrow 3 = MV2\rightarrow 0 - MV3\rightarrow 0.$

For the third type of picture-type conversion, i.e., $P_{6,in}$ to $P_{3,out}$, the target motion vectors are from $P_{6,in}$ to $B_{4,in}$. Motion vectors are formed using the following methods:

$-(MV_{4\rightarrow 6})$

$MV_{6\rightarrow 3} - MV_{4\rightarrow 3}$

Recall that the spatial resolution down-sampling process involves discarding the first field of the input frame. Suppose the top field comes first, then only the bottom field

is retained to form the target picture. Thus, the motion vectors we are trying to form are for the bottom field and use the bottom reference field as reference. The extended candidate motion vectors can be derived using the techniques discussed in the previous section.

Using the procedures described above, multiple motion vectors for a macroblock are formed. Note that these motion vectors are composed from the input motion vectors easily and stored for use in the second step, no motion estimation computation is performed.

It is likely that no motion vector is found for a macroblock through the above processes. For those situations, that macroblock is labeled as "NOMV", meaning that no motion vectors of that macroblock will be used in the second step of motion vector composition.

3.3.4. Spatial Resolution Reduction

The previous section shows that the motion re-estimation using weighted median of candidate motion vectors gives satisfactory performance for the HDTV to SDTV transcoding. However, when the target picture resolution becomes small, say the QCIF resolution, a large percentage of the macroblocks in a picture belong to object boundaries or picture boundaries. For macroblocks at picture boundaries, the motion vectors are not allowed to point to outside the reference picture in MPEG-2, while MPEG-4 supports unrestricted motion vectors that can point to the outside of the reference picture. So the problem is how to derive proper motion vectors from the restricted input motion vectors.

For a target macroblock belonging to the object boundaries, there might be two inconsistent motions associated with the macroblock. It is difficult to tell which motion vector performs better for the target macroblock unless they are explicitly tested. For these macroblocks with inconsistent input motion information, using a single composed target motion vector may not be adequate.

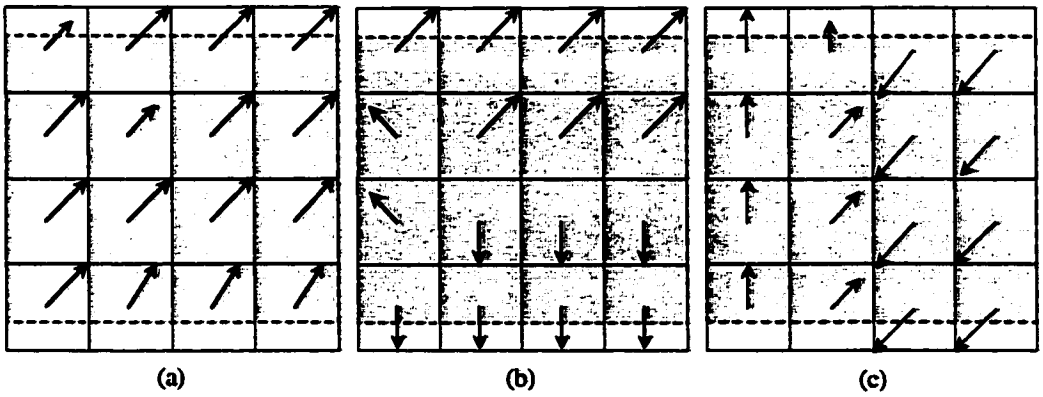


Figure 20. Candidate motion vectors for three macroblocks. (a) Consistent candidate motion vectors. (b) Two major motions exist in a macroblock. (c) Motion is inconsistent in the candidate macroblocks.

In order to result in a better motion vector for these macroblocks, we first study the candidate motion vectors for a target macroblock. Figure 20 shows three typical classes of target macroblocks according to the consistency of their candidates:

1) macroblocks that have consistent candidate motion vectors;

2) macroblocks at object boundaries which have two major groups of candidate motion vectors.

3) macroblocks at picture boundaries which do not have consistent candidate motion vectors;

When one motion is dominant, meaning that the majority of the candidate vectors are consistent, the result motion vector from the weighted-median of the candidate motion vectors is likely to be consistent with this motion. Thus, the weighted-median method should work well. When neither of the two motions is dominant, as shown in Figure 20-(b), it is difficult to tell which motion vector performs better. For this case, it is better to check the actual matching errors. The problem left is how to obtain the two candidate target motion vectors representing the two motions. One possibility is to use classical clustering algorithms to classify the candidate motion vectors, then, choose the medians of the two classes with the largest numbers of vectors. However, this method may be too complicated for use in the transcoder.

We propose a simple motion composition approach described as follows, given the candidate motion vectors derived in the frame-rate reduction stage.

1) Obtain the motion-vector set from the candidate macroblocks, down-scale the motion vectors and record their weights, with the weight being the related block area of the motion vector. Note that the down-scaled motion vectors are rounded to integer pixels;

2) Merge the down-scaled motion vectors that are the same, and sum the weights. Denote the resultant motion vector set as $C = \{MV_i\}$, and the weight as W_i, $i=1,2,..,N$, with N being the total number of motion vectors in the set. Let ϖ be the total weight of $\{MV_i\}$, i.e., $\varpi = \sum_{i=1}^{N} W_i$.

3) Calculate the weighted median of {MVi}. Denote it as MV_m, and its weight as W_m;

4) If $W_m >= \varpi/2$, go to Step 1 to process the next target macroblock and the MV_m is the result motion vector for the current macroblock. Otherwise continue;

5) Remove MV_m from C to get C', then calculate the weighted median of the vectors in C', denote it as MV_m';

6) The one of MV_m and MV_m' that results in lower SAD after motion compensated prediction is the final result motion vector, and the algorithm goes to Step 1 to process the next macroblock.

The basic idea is to check two target candidate motion vectors for those macroblocks that do not have consistent motions such as many of the macroblocks at object boundaries and picture boundaries. In this algorithm, the motion consistency is detected by checking if there is a single dominant motion in the candidate macroblocks, as described in Step 4. Although sometimes the consistent candidate motion vectors may not be identical, simulations show that the down-sampled candidate motion vectors most often are since they are rounded to integer pixels. This simple method can usually identify the dominant motion.

3.3.5. Efficient Half-pel Refinement

Half-pixel motion-vector refinement usually is performed around the composed motion vector to enhance the accuracy of the motion vector. The conventional half-pixel refinement checks the eight half-pixel positions around the composed motion vector, and

chooses the one yielding the minimum SAD as the refined motion vector. Through intensive simulations, we found that checking only the nearest four half-pixel points may be enough. The performance difference between the two refinement schemes is negligible (usually less than 0.05dB). The checking points for the two refinement processes are illustrated in Figure 21, where solid black circles are integer positions and the center one is the position to be refined. Conventional refinement checks all the eight half-pixel positions. We propose to check only the four nearest half-pixel positions indicated by the gray-shaded circles. Saving half the computation (four checking points) of the traditional refinement is trivial to a full-search motion estimation algorithm, but it is significant to the transcoding where the refinement accounts for the major portion of the motion re-estimation computation.

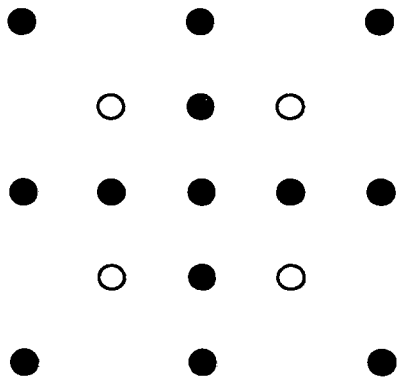


Figure 21. Checking points for the proposed half-pixel refinement.

3.3.6. Simulation Results

To evaluate the new re-estimation algorithm for spatial transcoding, the weighted median and the re-encoding are also simulated. By re-encoding we mean to use full-search to re-calculate the motion vectors as described in MPEG-4 VM 17.

Two test video sequences of resolution 720x480: "pigeon" and "flower garden" are used in the simulations. Figure 22 shows a snapshot of the two video sequences. They are encoded in the MEPG-2 format, with the following coding parameters: GOP length 30, I/P frame distance 3, and bit-rate 6 Mbps. It is transcoded to MPEG-4 at the QCIF resolution. The target bit-rate is set to be 96 kbps for "pigeon" and 192 kbps for "flower garden". The result frame-rate is 10 fps. Figure 23 shows the motion vector fields generated from the new and the reference algorithm for a frame of the "pigeon" sequence. Also shown is the difference between the two motion vector fields. As expected, the major difference between the two vector fields is at the object and picture boundaries. Similar conclusions can be drawn for the "flower garden" sequence. Figure 24 shows the PSNR plots of the new algorithm vs. the weighted median. The new algorithm consistently gives better performance. For both sequences, the maximum PSNR gain is more than 1 dB, and the average gain is around 0.5 dB.

Figure 22. Snapshot of the "pigeon" and "flower" sequences

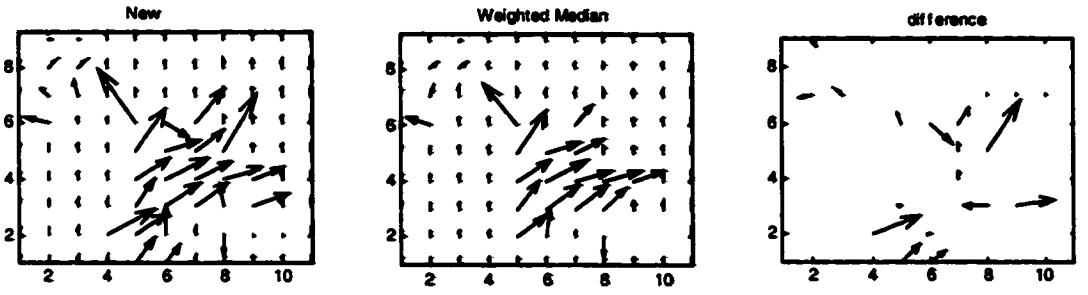
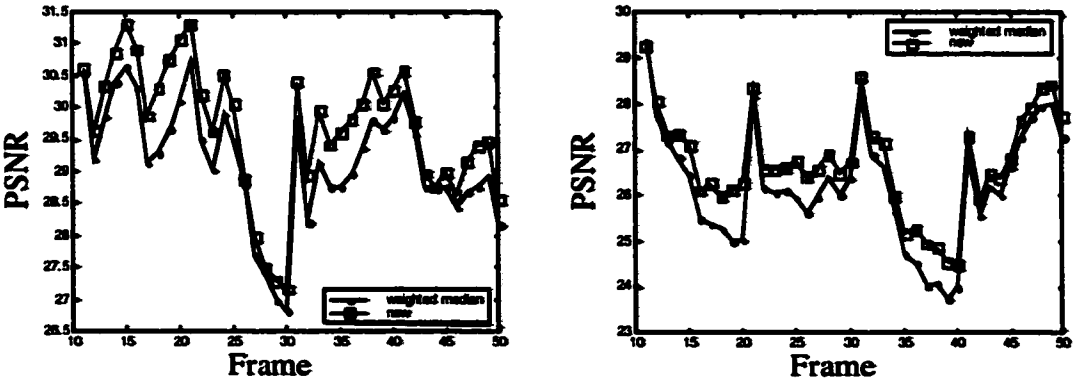


Figure 23. Motion vector fields of the new algorithm, the weighted median, and the difference between them for frame #37 of the pigeon sequence.



(a) Pigeon Sequence (b) Flower garden Sequence

Figure 24. Performance of the new and the weighted median algorithms for spatial transcoding

Table 4. Proposed vs. conventional half-pixel refinement (in dB).

Sequence	*Proposed refinement*	Conventional refinement
Pigeon	30.85	30.90
Flower Garden	28.31	28.36

Table 5. Proposed frame-rate reduction approach vs. re-encoding using full-search (in dB).

Sequence	Full Search (±8)	Proposed
Flower Garden	28.66	28.23
Football	32.26	32.34

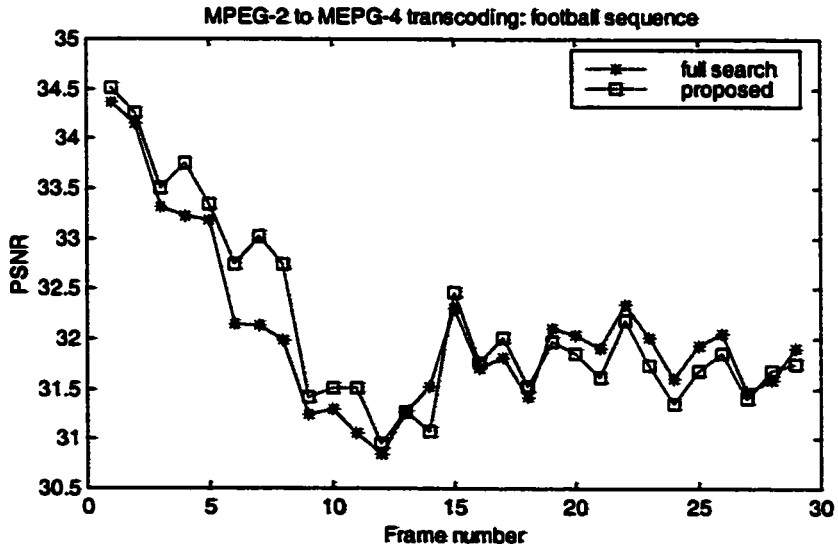


Figure 25. Full-search vs. proposed motion vector composition for frame-rate reduction

Similar to the observation in [27], half-pixel refinement is verified to be enough and further refinement does not give better performance. Table 4 gives the results of the proposed and the conventional half-pixel refinement. Compared with the conventional refinement, the proposed refinement reduces half of the search positions, while gives virtually the same performance.

For the frame-rate reduction, "Flower Garden" and "Football" sequences that have a resolution of 720x480i and 4:2:0 chrominance sampling are used. Both are encoded using MPEG-2 frame pictures at 5 Mbps and frame-rate of 30 fps. They are transcoded to MPEG-4 SP CIF resolution of 15 fps. The target bit-rate is 768 kbps. TM-5 rate-control is used. We simulated the full-search with search-range (-8,+8) to evaluate the performance of the proposed algorithm. Table 5 shows the simulations results. Figure 25 shows the PSNRs of the proposed algorithm and the full-search algorithm. The performance of the proposed algorithm approaches the full-search algorithm. Through our computer simulation, when only the computation of the motion estimation is compared, the proposed approach achieves a saving of more than 80%. In the "Football" sequence, it actually outperforms the full-search algorithm. Part of the reason is that the proposed weighted median approach produces smoother block motion vector field than the full-search algorithm, which may save bits in coding the differential motion vectors. Another reason is that the search-range of the full-search (-8,+8) is not large enough for some macroblocks, while the proposed approach is not restricted by this range. It can be expected the full-search approach can improve its performance by increasing the search range, however that will require more computational power.

3.4. Summary

In this chapter, we have presented our proposed motion re-estimation techniques for transcoding with spatial resolution, temporal resolution, and coding-format conversions.

Our proposed motion re-estimation possesses the following new features:

- Unique support of interlaced to interlaced video transcoding.
- Efficient frame-rate reduction and frame-type conversion.
- Improved performance against motion inconsistencies.
- More efficient half-pixel refinement.
- Simpler mode-decision algorithm.

Simulations have shown that the proposed approach achieved video quality comparable to the re-encoding for both HDTV to SDTV and MPEG-2 to MPEG-4 SP transcoding, but with greatly reduced computational complexity. Its extension to other transcoding scenarios involving spatial-resolution, temporal-resolution and coding-format conversions is straightforward.

Chapter 4. Bit-allocation for Video Transcoding

4.1. Introduction

The goal of the rate-control in video coding is to achieve a target bit-rate with good and consistent visual quality. Rate-control can be separated into two steps: picture-layer bit-allocation, which determines the target bit-budget for each picture, and macroblock-layer rate-control that determines the quantization parameters for coding the macroblocks. In this chapter, we discuss the bit-allocation for the picture-layer during the transcoding of a coded video bit-stream.

The most popular picture bit-allocation approaches for video coding are based on the concept of picture complexity defined in MPEG-2 Test Model 5 (TM5) [50]. It is defined to characterize the difficulty of coding a picture. Thus the target number of bits for coding a picture should be made proportional to its complexity. The complexity measure usually is not available (or very difficult to calculate) at the encoding time, therefore the current complexities are usually approximated using those of the previously encoded frames. In transcoding, intuitively we can compute the complexities of the pictures from the input bit-stream, then use these complexities for the bit-allocation in transcoding these pictures. In fact, most bit-allocation approaches for transcoding in the literature belong to this category [52-53][23]. However, as will be pointed out, the complexity measure is dependent on the coding bit-rate. Therefore, the complexity measures calculated from the input video bit-stream at the input bit-rate may not be suitable to directly serve as the complexities for coding the pictures at the output bit-rate.

In this chapter, we find the correlations between the picture complexities of different bit-rates, and use such correlations to estimate the picture complexities of the target video from the statistics extracted from the input video bit-stream. Based on the estimated complexities, a bit-allocation algorithm is presented to assign suitable target number of bits to each picture. The simulations results are also provided to show the improvements.

4.2. The Picture Complexity Measure

The proposed bit-allocation algorithm is based on the concept of picture complexity – the number of bits allocated to a picture is proportional to its complexity. We first review the definition of picture complexity in TM5, then, introduce the scheme to estimate the picture complexities at the output bit-rate for transcoding.

The conventional picture complexity is defined as in TM5:

$$\begin{aligned} X_I &= S_I Q_I \\ X_P &= \frac{S_P Q_P}{K_P} \\ X_B &= \frac{S_B Q_B}{K_B} \end{aligned} \tag{6}$$

where X_I, X_P, and X_B represent the complexities of I-, P-, and B-pictures respectively, S_I, S_P, S_B represent the numbers of bits generated by encoding these types of pictures respectively, Q_I, Q_P, and Q_B represent the average quantization parameters over all macroblocks for each type of pictures. Universal constants K_P and K_B are *1.0* and *1.4* respectively for the MPEG-2 default quantization matrices. Note that there is a slight

difference between this definition and that of TM5: the two universal constants are incorporated in the definition here, while in TM5 they are introduced in the bit-allocation.

For pre-encoded video streams, we can extract the average quantization parameter and the number of bits of each picture easily, with which we can calculate picture complexities as defined from the input video streams. This complexity will be called input complexity since it is computed with parameters obtained from the input video stream at the input bit-rate. The complexity computed from the transcoded output video stream at the output bit-rate will be called "output complexity", which is the complexity more suitable for the bit-allocation for video coding at the output bit-rate. Before the transcoding is performed, the output complexities actually are not available. We propose to use the input picture complexities, which are readily available from the input bit-stream, and the output complexity of the previous pictures coded at the output bit-rate, which are readily available from the output of the transcoder, to estimate the output complexity of the current picture to be coded.

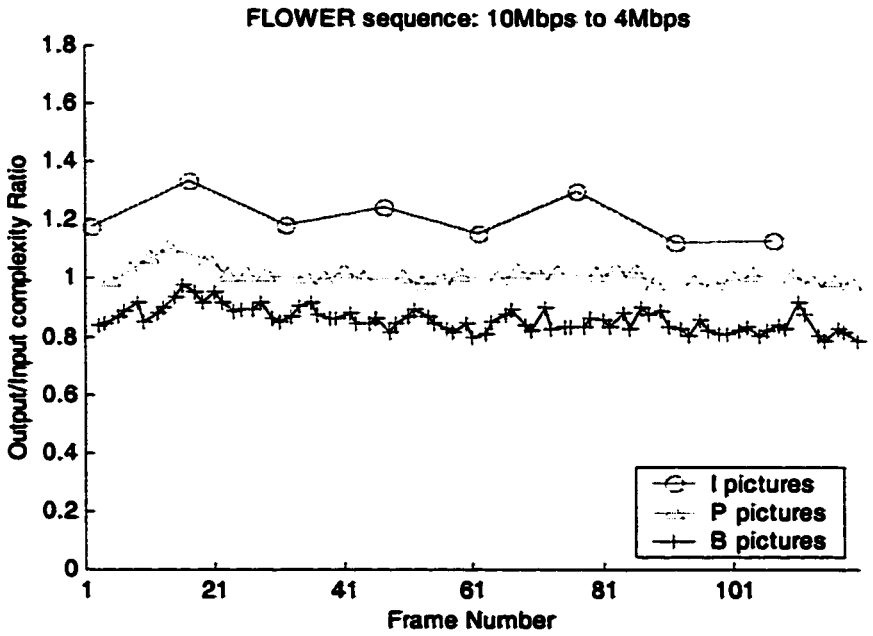


Figure 26. Output/input complexity ratios for different types of pictures.

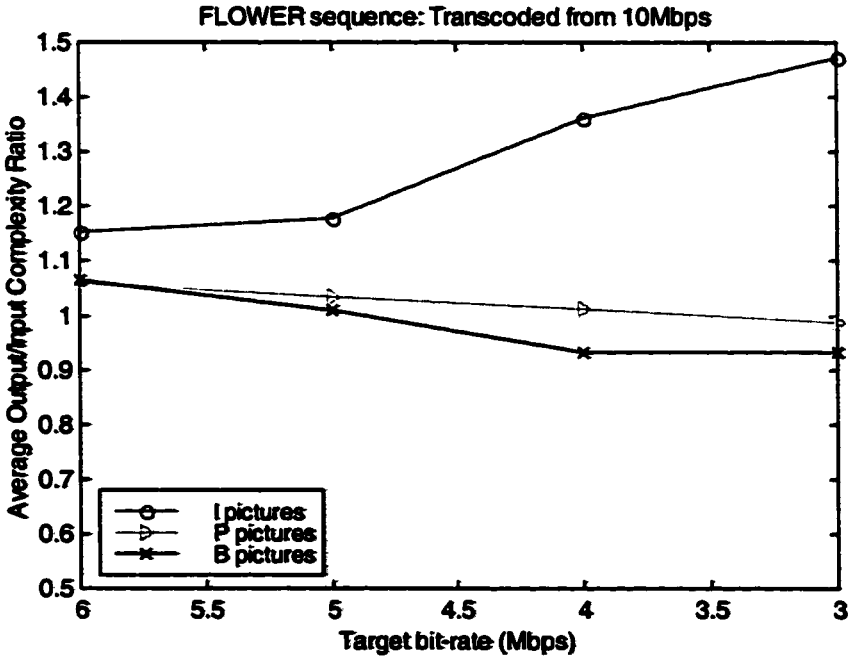


Figure 27. Average output/input complexity ratio for different types of pictures.

To obtain the correlations between the input and the output complexities for developing a scheme to estimate the output complexities from the input complexities, we conducted simulations on a *720x480* FLOWER video sequence. The sequence is encoded at 10 Mbps and is transcoded to several lower bit-rates (6 Mbps, 5 Mbps, 4 Mbps, and 3 Mbps) using the TM5 rate-control. The ratios of the output and the input complexities are calculated and the results are shown in Figure 26 and Figure 27. The observations are: 1) the ratios between the output complexities and the input complexities of the pictures of the same type are stable; and 2) the picture complexities are dependent on the output bit-rate, and when the transcoding target bit-rate decreases, the complexities of I-pictures tend to become relatively higher compared to the complexities of P- and B-pictures. Based on these observations, the following method is proposed to approximate the output complexities of I-, P-, and B-pictures.

$$\hat{X}_{out,I} = \frac{X'_{out,I}}{X'_{in,I}} \times X_{in,I}$$
$$\hat{X}_{out,P} = \frac{X'_{out,P}}{X'_{in,P}} \times X_{in,P} \qquad (7)$$
$$\hat{X}_{out,B} = \frac{X'_{out,B}}{X'_{in,B}} \times X_{in,B}$$

where $X'_{in,I}, X_{in,I}$ are input complexities of previous and current I-pictures, $X'_{out,I}$ is the output complexities of previous I-picture, and $\hat{X}_{out,I}$ is the estimated output complexity of the current I picture. Other notations follow the same convention, where the P and B in the subscripts represent P- and B- pictures respectively.

Thus, an output picture complexity can be estimated as a "weighted" input picture complexity, where the "weight" is the output/input picture complexity ratio of the previous picture of same type. When the weights are same for all pictures, this algorithm reduces to the one that directly scales the input complexity with a factor that captures the dependency of the complexity with the bit-rate.

An alternative interpretation is that the output picture complexity is the "weighted" output complexity of the previous picture of the same type:

$$\begin{aligned}
\hat{X}_{out,I} &= \frac{X_{in,I}}{X'_{in,I}} \times X'_{out,I} \\
\hat{X}_{out,P} &= \frac{X_{in,P}}{X'_{in,P}} \times X'_{out,P} \\
\hat{X}_{out,B} &= \frac{X_{in,B}}{X'_{in,B}} \times X'_{out,B}
\end{aligned} \tag{8}$$

where the weight is the ratio between the input complexities of the current picture and the previous picture of the same type. When the input information is not used, i.e., $X'_{in,I} = X_{in,I}, X'_{in,P} = X_{in,P}, X'_{in,B} = X_{in,B}$, this algorithm actually becomes TM5. Using the output complexities of the previous frames to do the rate-control (as in TM5) may also be able to take care of the dependency of the complexities with the resolution and bit-rate. However, it is well known that the bit-allocation of TM5 cannot handle scene-changes or large motion-changes well since its complexity estimation based on previous picture complexities would fail in these situations. Using the proposed complexity measure in (7), when scene-changes or large motion-changes happen, the input picture complexities will change accordingly reflecting the scene-changes or large motion-changes. Thus, the

proposed output complexity estimations can alleviate the scene-change or large-motion-change problem, and give better performance than TM5.

4.3. Proposed Algorithm

The proposed picture complexity estimations are used in the proposed bit-allocation for video transcoding. The target picture bit-allocation algorithm comprises two steps: GOP bit-allocation and picture bit-allocation.

1) The target number of bits for a GOP is the number of bits that can be transmitted at the output bit-rate within one GOP period:

$$T_{GOP} = \frac{N_{GOP}}{F} \times R_{CH} \tag{9}$$

where T_{GOP} is the target number of bits for the GOP; F is the frame-rate; N_{GOP} is the GOP size, and R_{CH} is the channel bit-rate.

2) Picture-layer target bit-allocation. We use the following notations: N_P and N_B are the number of P- and B- pictures in a GOP; $X_{in,I}, X_{out,I}$ are the input and output (estimated) complexities of the I-picture in the GOP; $X_{in,P}[i], X_{out,P}[i]$ ($i=1,2,...,N_P$) are the input and output complexities of P-pictures; and $X_{in,B}[j], X_{out,B}[j]$ ($j=1,2,...,N_B$) are the input and output complexities of B-pictures. The target number of bits for the I-picture is proportional to its output complexity:

$$T_I = \frac{\hat{X}_{out,I}}{\hat{X}_{out,I} + \sum_{i=1}^{N_P} \hat{X}_{out,P}[i] + \sum_{j=1}^{N_B} \hat{X}_{out,B}[j]} \times T_{GOP}$$
$$= \frac{X_{in,I} \times w_I}{X_{in,I} \times w_I + \sum_{i=1}^{N_P} X_{in,P}[i] \times w_P + \sum_{j=1}^{N_B} X_{in,B}[j] \times w_B} \times T_{GOP} \quad (10)$$

where w_I, w_P, and w_B are derived from (7):

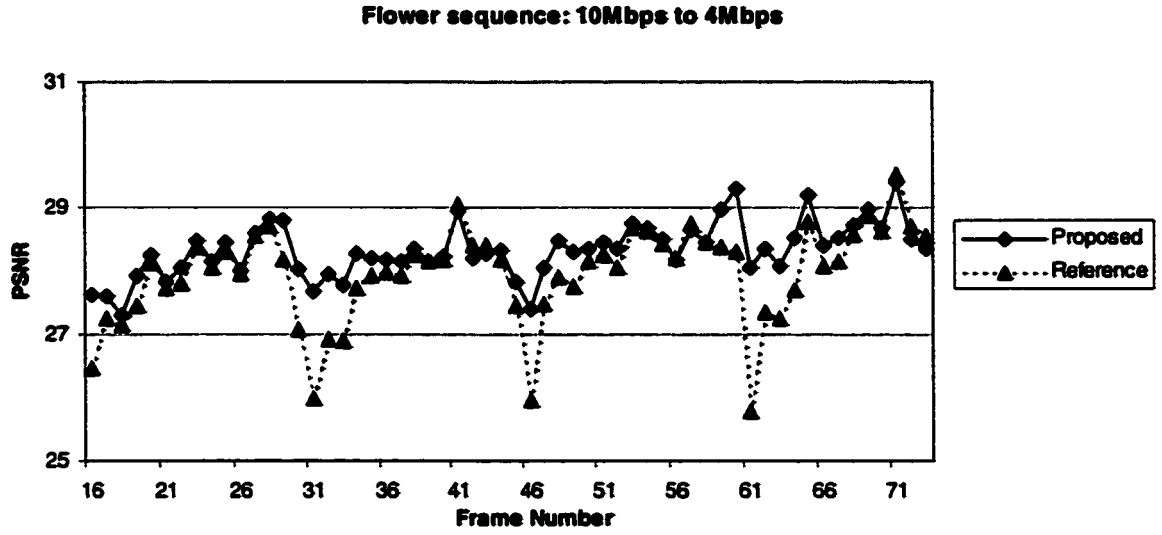
$$w_I = \frac{X'_{out,I}}{X'_{in,I}}, \quad w_P = \frac{X'_{out,P}}{X'_{in,P}}, \quad w_B = \frac{X'_{out,B}}{X'_{in,B}}.$$

The target number of bits for the P- and B-pictures can be computed in a similar fashion. For the first GOP there is no previous output-complexity available. So, the three weights need to be initialized. In our simulations {1.4, 1.0, 0.8} are used as initial values for $\{w_I, w_P, w_B\}$. These initial values are not important since they only affect the pictures in the beginning. Note that the weights $\{w_I, w_P, w_B\}$ are updated to the latest ones after encoding each picture.

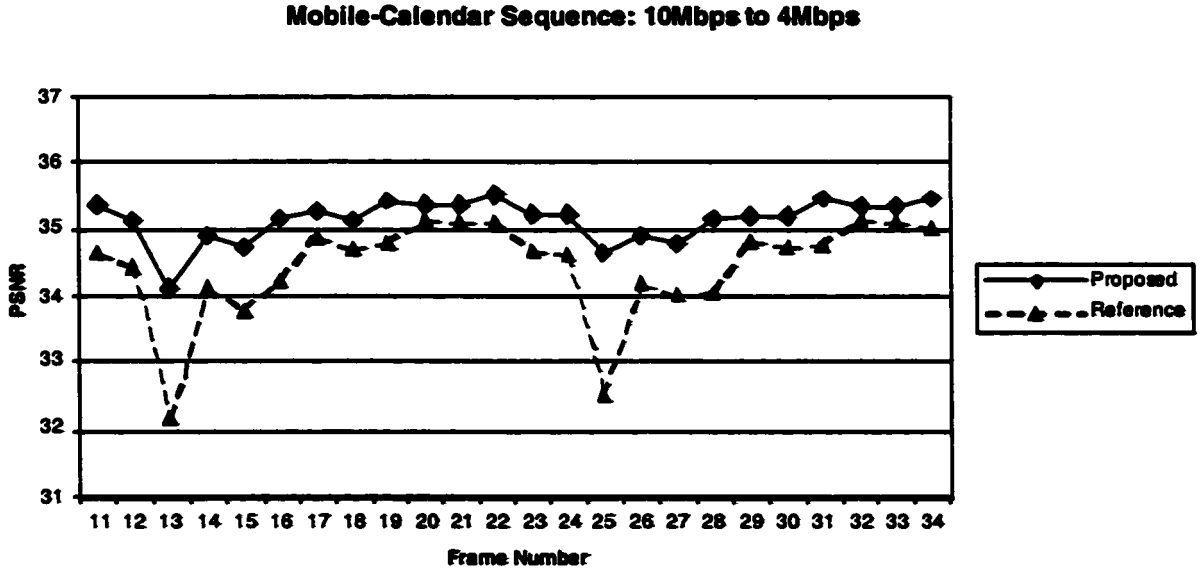
4.4. Simulations Results

Simulations are conducted using the "flower garden" sequence. It has a spatial resolution of 720x480 and YUV format 4:2:0. This sequence is encoded in MPEG-2 frame-pictures at 10 Mbps. Coding parameters are: I/P frame distance is 3, GOP-size is 15, and the frame-rate is 30 frames/s. This compressed sequence is used as the input to the transcoder. Figure 28-(a) shows the PSNR plot of the proposed algorithm and the reference algorithm using the input picture-complexities. The first GOP is not shown to omit the influence of the algorithm initialization. We have shown that the picture

complexities of different types of pictures change differently with bit-rates. That is why the bit-allocation algorithm based on the input complexities cannot assign bits properly between I-, P-, and B-pictures in the transcoding. For example, the picture complexities of I-pictures become relatively large compared to the complexities of P- and B-pictures when the video is transcoded to a lower bit-rate. Thus, in the transcoding, I-pictures need a greater portion of the available bits than they need at the input bit-rate. However, the bit-allocation using the input complexities cannot satisfy this requirement. It produces poor I-pictures, and the poor I-pictures affect their following pictures predicted from them. The bit-allocation algorithm using the estimated output complexities can allocate bits to I-, P- and B-pictures according to their complexities at the output bit-rate, and thus performs much better. The same two algorithms are simulated for a 40-frame "Mobile-Calendar" sequence. It has a spatial resolution of 720x576 and a frame-rate of 25 frames/s. It is coded using a GOP-size of 12. Other simulation configurations are the same as those of the "flower garden" sequence. The simulation results are shown in Figure 28-(b). Subjective evaluation is also given in Figure 29 to show the advantage of the proposed algorithm. The mean, standard deviation, and minimum PNSR values for both simulations are given in Table 6. It can be seen that that bit-allocation based on the estimated output picture complexities greatly improves the transcoding performance.



(a)

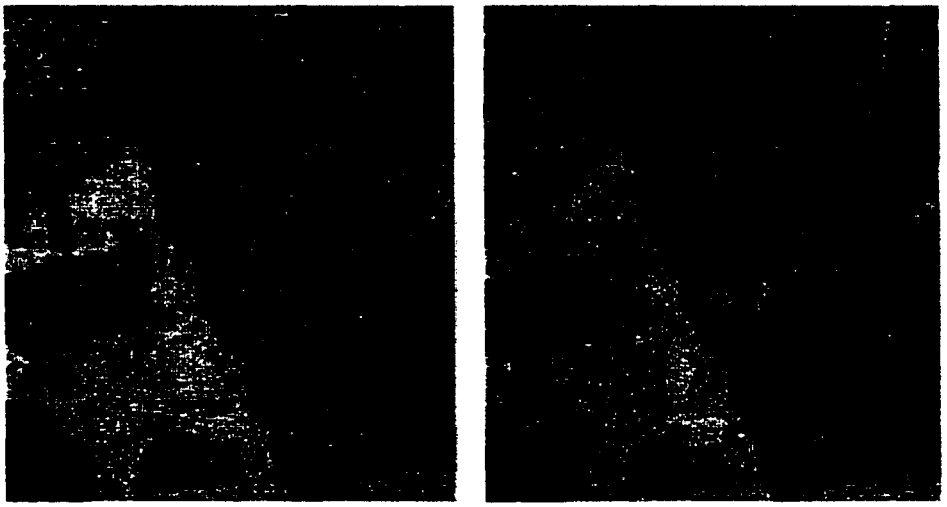


(b)

Figure 28. The proposed bit-allocation based on output picture-complexities vs. the reference bit-allocation based on input picture-complexities.

Table 6. Performance of the proposed bit-allocation based on output complexities and the reference based on input complexities. Unit: dB.

		Average	Minimum	Standard Deviation
Flower Garden	Proposed	28.40	26.93	0.57
	Reference	28.00	25.78	0.72
	Improvement	*0.40*	*1.15*	-
Mobile-Calendar	Proposed	35.03	34.11	0.57
	Reference	34.43	32.17	0.72
	Improvement	*0.60*	*1.94*	-



(a) (b)

Figure 29. Transcoded Frame #24 using (a) the proposed and (b) the reference bit-allocation.

To evaluate its scene-change handling capability, a combined test video sequence is used. The first 35 frames are from the "football" sequence and the remaining 40 frames are from the "flower garden" sequence. The combined sequence, called "football-flower", has the same parameters as those used in the previous simulations. It is also transcoded from 10 Mbps to 4 Mbps. The PSNR performance of the proposed bit-

allocation algorithm is shown in Figure 30. TM5 bit-allocation is also simulated for comparison. At the scene change (frame 36), TM5 performs poorly since it does not expect the abrupt picture complexity increase. In comparison, the proposed algorithm can take care of the scene change better since the input complexity also increases. In the same GOP where the scene-change happens, frames at and after the scene-change are more difficult to encode than the frames prior to the scene-change. This will be reflected by the input complexity increase, and the proposed algorithm will allocate more bits to these frames. The result is that the performance at the scene-change is significantly improved, so are the frames following the scene-change. The cost is the quality of the frames prior to the scene change is slightly lowered. However, this tradeoff is desirable since the worst-case video quality is improved, which usually determines the overall visual quality of video. Frame 46 is the I-frame of the first GOP following the scene-change, TM5 estimates its complexity from frame 31, the I-frame of the previous GOP, which occurs before the scene-change, and thus gets a poor estimation. Therefore, TM5 performs also poorly in that GOP. The proposed algorithm has a better estimation of complexities for that GOP, and thus performs better.

The computations involved in the proposed algorithm are the extracting of quantization parameters and the counting of the number of bits of each picture, which are simple to compute. For transcoding of stored video, it is possible to compute all the statistics of the video and store them along with the bit-stream. When the transcoder operates, it can read the stored statistics instead of extracting them from the compressed video to save computations.

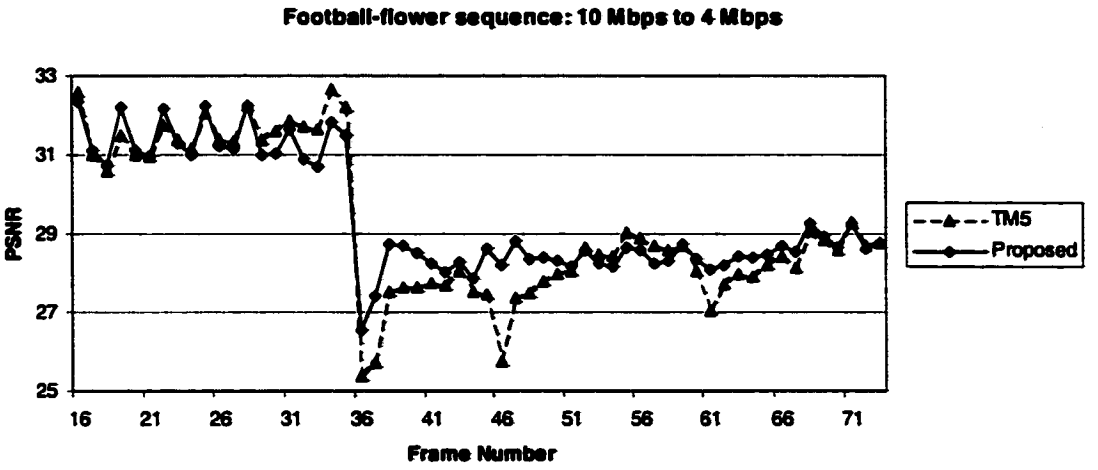


Figure 30. Scene-change handling: TM5 vs. the proposed bit-allocation.

4.5. Summary

In this chapter, we first showed that the picture complexities depend on the bit-rates. We further found the correlations between the output and input complexities at different bit-rates, and used the correlations to estimate the output picture-complexity. We then proposed a picture bit-allocation algorithm based on the estimated output picture complexity. Simulations show that the proposed algorithm can achieve better and smoother video quality than previously reported approaches.

Chapter 5. Rate-distortion Optimized Macroblock-layer Rate-control for Low-bit-rate Video Coding and Transcoding

5.1. Introduction

Rate-control for video coding usually includes frame-layer and macroblock-layer rate-control. The frame-layer rate-control allocates an appropriate number of bits to each video frame. The macroblock-layer rate-control determines the quantization parameters for macroblocks in a frame to meet the frame bit-target. In the previous chapter, we proposed an improved frame-layer rate-control scheme for video transcoding. In this chapter, we proposed an improved macroblock-layer rate-control scheme.

The macroblock-layer rate-control can be modeled as a constrained optimization problem: choosing quantization parameters to minimize the picture distortion under the given picture-rate constraint. The constrained optimization problem can be solved by dynamic programming [54]. Although it is the optimal solution to the problem, its enormous computational requirement prevents it from being applied to practical video coding applications. The operational Lagrange Multiplier (LM) algorithm [55] approaches the optimal solution with reduced computation. However, it is still considered to be too computationally expensive for practical video coding.

In this chapter, through analysis and simulations of the LM algorithm in the low bit-rate MPEG-4 video coding framework, we propose effective techniques to simplify its computational complexity to make it suitable for practical use.

5.2. The Lagrange Multiplier Algorithm for Rate Control

To apply the Lagrange Multiplier algorithm to the macroblock-layer rate-control, the following steps are performed [55].

The first step is to compute the Rate-Distortion (RD) points for each macroblock. An RD point of a macroblock consists of its Rate and Distortion values, and is determined by its quantization parameter. Each RD point is calculated by encoding the macroblock (including quantizing, zigzag scanning, entropy encoding, and de-quantizing) using a quantization parameter. For MPEG-4, there are 31 possible quantization parameters, thus, each macroblock needs to be encoded 31 times.

The next step is, for a given Lagrange Multiplier (λ) value and for each macroblock, finding the quantization parameter QP^* that minimizes the cost function $J=D+\lambda R$. Here D and R are the distortion (Sum of Squared Error) and the rate of encoding the macroblock using the quantization parameter. The D and R are functions of the quantization parameter QP. After the whole picture is processed, the rate and distortion of the picture can be obtained by accumulating the distortions and rates of all the macroblocks. If the rate is not close enough to the picture target bit-number, we need to vary λ and repeat the process to find for the picture a particular λ value that generates a bit-number close enough to the picture target bit-number.

A fast convex search algorithm has been proposed in the literature [56]. The algorithm first finds two boundary λ values $\lambda 1$ and $\lambda 2$ ($\lambda 1<\lambda 2$): $\lambda 1$ generates more bits than the target bit-number, and $\lambda 2$ generates fewer bits than the target bit-number. The

optimal λ^* will be between these two values, i.e., ($\lambda 1 < \lambda^* < \lambda 2$). The bi-sectional algorithm can then be used to search for the optimal λ^* [56].

The computational complexity of the algorithm consists of two parts: calculation of the RD points, and the search for the optimal λ^*. The RD point calculation requires encoding each macroblock 31 times for MPEG-4, while for each iteration of λ and each macroblock, the search needs to calculate the cost function for all the 31 RD points and find the minimum one. The overall computation of searching for the optimal λ^* depends on the number of iterations the algorithm takes to converge.

Both parts of the algorithm are computationally expensive. However, the computation load can be reduced significantly by taking advantage of the properties of the low-bit-rate MPEG-4 video coding as described later in this chapter.

5.3. Macroblock RD Point Calculation

The computation of RD points can be reduced by two means: reducing the number of RD points needed for the algorithm, and reducing the computation needed to calculate the RD points.

It has been shown that optimal quantization parameters of macroblocks in a picture tend to be close to each other for low-bit-rate video coding [10]. In fact, MPEG-4 syntax limits the change of quantization parameters from macroblock to macroblock to be within ±2. In our implementation, the average quantization parameter and the Lagrange Multiplier λ of the previous frame are used as the initial values in coding this picture. It

turns out for most cases only a small portion (about 4) of the 31 possible RD points of a macroblock need to be calculated.

At very-low-bit-rates, the quantization step-size is often several times greater than the standard deviation of the DCT (Discrete Cosine Transform) of a macroblock, thus the DCT coefficients of many macroblocks may be quantized to all zero. Such kind of macroblocks are called zero macroblocks. Table 7 shows the simulation results for two QCIF sequences of 10 frames/s coded at low-bit-rates. Typically more than 50% of all macroblocks will be quantized to zero at bit-rates lower than 32 kbps. The minimum quantization parameter that produces a zero macroblock can be calculated before any RD point is calculated. This quantization parameter is obtained from the maximum coefficient of the macroblock. Apparently when a quantization parameter is larger than the minimum quantization parameter, no computation is needed, and the macroblock can just simply be identified as a zero macroblock.

Table 7. Percentage of zero macroblocks in low-bit-rate video coding.

Sequence	Foreman		Carphone	
Bitrate (kbps)	20	32	20	32
Percentage of Zero Macroblocks	80.0%	57.9%	70.6%	57.6%

5.4. Search for the Optimal Lagrange Multiplier

The computation associated with the search includes: 1) Given a λ, for each macroblock, find the optimal quantization parameter. That is, for the i'th macroblock,

find QP_i^* that minimizes the cost function $J_i(QP)=D_i(QP)+\lambda R_i(QP)$. 2) The iterative search for the optimal Lagrange Multiplier λ^*.

If two boundary Lagrange Multipliers ($\lambda 1$, $\lambda 2$) have been found, with $\lambda 1$ generating more bits than the target and $\lambda 2$ generating fewer bits than the target, we know $\lambda 1<\lambda^*<\lambda 2$. Suppose for a macroblock, $QP1$ and $QP2$ are the two quantization-parameters optimal for $\lambda 1$ and $\lambda 2$, respectively. It can be shown that the optimal quantization-parameter QP^* must lie between $QP1$ and $QP2$ [55] provided that the macroblock rate-function $R_i(QP)$ is monotonously non-increasing, which is generally true in MPEG-4 video coding. For most macroblocks, simulations show that the differences between $QP1$ and $QP2$ are within 2, thus the cost calculations necessary to find the minimum-cost RD point are further reduced.

The search complexity increases with the number of iterations. For the purpose of macroblock-layer rate-control, we may not need to meet the target bit-number exactly. It is enough to generate a rate that is close to the target bit-number R_t. We can set a threshold to control how accurate the rate-control should be. With the above discussions, the proposed macroblock-layer rate-control algorithm is summarized in the following section.

5.5. The Proposed Macroblock-layer Rate-Control Algorithm

Step 1 – *Initialize the quantization-parameter QP and the Lagrange Multiplier* λ.

We initialize the QP and λ to the values of the previous picture, i.e., $QP=\overline{QP}^t$, $\lambda = \lambda^t$,

where λ' and $\overline{QP'}$ are the Lagrange Multiplier and the average quantization-parameter of the previous picture respectively.

Step 2 – *With the λ, vary QP to search for each macroblock the optimal RD point which minimizes the cost function $J=D+\lambda R$.* The search-range of the first macroblock is constrained to be within ±2 of the initial *QP*. The search-range for the remaining macroblocks is limited to be within ±2 of the *quantization parameter* of the previous macroblock. Note that as described in the previous section, after two boundary Lagrange Multipliers are found, the search range is further reduced. For each macroblock, the largest DCT coefficient is recorded when its first RD point is calculated such that no further computation is needed when the quantization parameter quantizes the largest coefficient to zero. Suppose when the coding of the picture is finished, the rate generated for the picture is *R*. If $0 \le |R_t - R| \le TH$ or a preset maximum number of iteration is reached, stop the search. Otherwise continue to the next step.

Step 3 – *Update the Lagrange Multiplier and the Quantization Parameter.* Denote *R1* and *R2* as the rates generated from the current and the previous iterations. *$(R1-R_t)(R2-R_t)<0$* means that two boundary λ have been found. We use the following pseudo C-code to explain the updating.

```
/* QP : average macroblock QP of the picture in the current iteration*/
K = λ/(QP * QP);
If ((R1-Rt)(R2-Rt)<0)
{
```

```
  /* Use bi-sectional algorithm to update the λ */
  /* (D1, R1) and (D2, R2) are the distortion-rate pairs associated with
     the two boundary λs */
  λ = -(D1-D2)/(R1-R2) ;
  QP =sqrt(λ/K);
}
else
{
  /* Update λ and QP using the following equations. */
  /* From the model QP*R=X , where X is the picture complexity*/
  dQP = -QP×dR/R;
  QP = QP + 1.2×dQP;
  λ=K×QP²;
}
```

In deriving the QP update formulation of the above pseudo-code, a simple picture rate-quantization model used in MPEG-2 TM5 is assumed: $QP*R=X$, where X is the complexity measure of the picture. In simulations, we use $1.2\times dQP$ as the increment of QP which can find the boundary λ faster. Then we increase the iteration number and go to the second step. Using this updating scheme, the boundary λs usually can be found in the first two or three iterations.

5.6. Simulation Results

We perform simulations to show the effectiveness of the proposed rate-distortion optimized macroblock-layer rate-control. QCIF sequences FOREMAN, CARPHONE, COASTGUARD, and SUSIE are used in the simulations. The frame-rate is 10 frames/second, coded at 32 kb/s. $TH=5\%R_t$, and the maximum iteration number is set to *8*. These two parameters are determined by experiments. Figure 31 shows how the algorithm approaches the picture target bit-number in a typical simulation. In this simulation, 20 iterations are performed without setting the threshold *TH* for a frame of the FOREMAN sequence. In this case, the algorithm converges to within 100 bits in the 3rd iteration and within 50 bits in the 5th iteration. Our simulations show that when the maximum iteration-number is set to 8, more than 95% of the time, the algorithm achieves within 5% of the target bit-number. Occasionally when the algorithm does not meet the *TH*, the quantization parameters are chosen from the iteration that achieves the smallest bit-difference from the target.

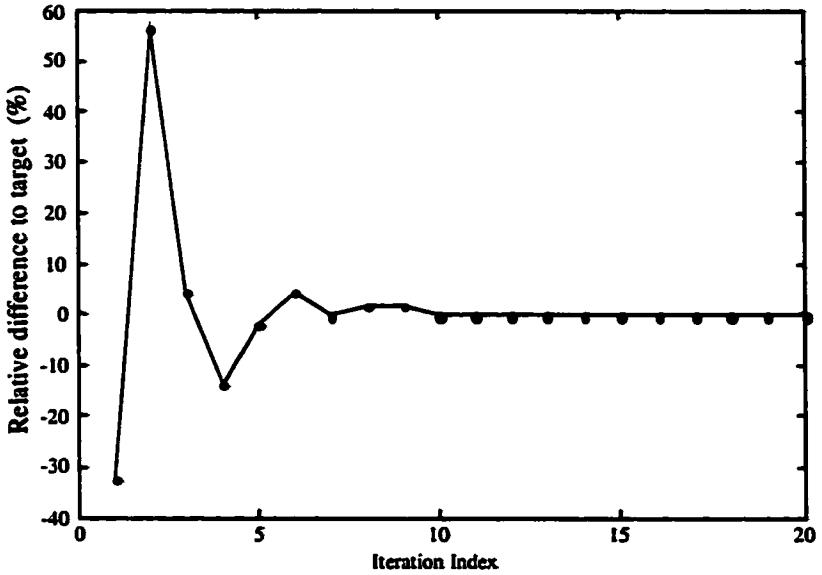


Figure 31. Evolution of the bit error.

Table 8. Average numbers of RD points calculated for a macroblock in the proposed algorithm.

Sequence	Foreman		Carphone		Coastguard		Susie	
Bitrate (kbps)	20	32	20	32	20	32	16	24
Average Number of RD Points	3.49	3.81	3.22	3.64	3.46	4.20	3.52	3.76

The average numbers of RD points needed to be computed are given in Table 8. Roughly, each macroblock needs to be encoded about four times. Compared with the total possible 31 RD points, more than 85% percent of the computation is saved. In our simulations, the computational requirement of the search algorithm (i.e., the cost function calculation) itself is rather low compared to that of the RD point calculation (about 10%). On average, for each macroblock, less than 15 cost-functions need to be calculated.

Simulations are also performed to evaluate the performance of the proposed rate-control algorithm compared to the H.263 TMN8 rate-control algorithm. The same frame-

layer bit-allocation is used and the same delay constraint (100 ms) is imposed. Figure 32 and Figure 33 show the encoder buffer behavior for FOREMAN and SUSIE at different bit-rates. The TMN8 frame-layer bit-allocation essentially tries to maintain the encoder buffer fullness at 10% of the encoder buffer-size to achieve low delay. The proposed algorithm achieves that goal much better than the TMN8 macroblock-layer rate-control. The inability of TMN8 to accurately meet the target bit-number often leads to skipped frames, especially at low bit-rates, as evident in Figure 32 (Frame #280~300).

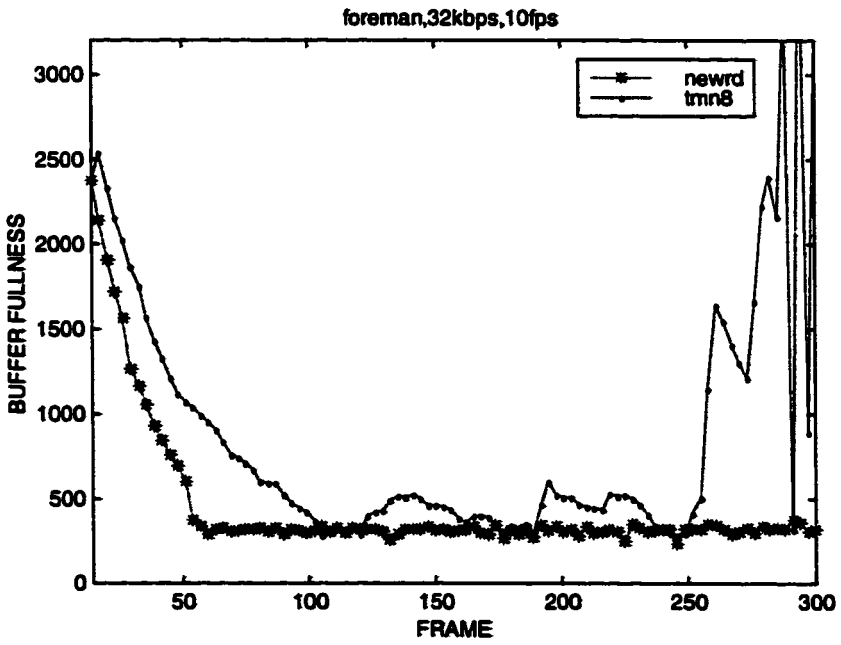


Figure 32. Encoder buffer fullness for the foreman sequence at 32 kbps. The encoder buffer size is 3200 bits.

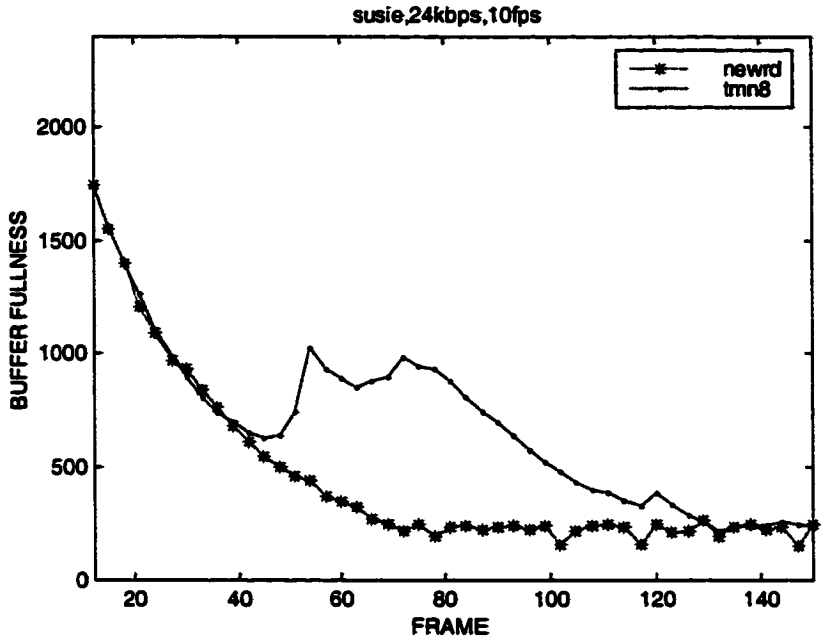


Figure 33. Encoder buffer fullness for the Susie sequence at 24 kbps. The encoder buffer size is 2400 bits.

To evaluate the coded video quality of the two algorithms, we adopt two quality measures: Mean PSNR (M-PSNR) and Total-sequence PSNR (T-PSNR) [57]. The M-PSNR is defined to be the mean of the PSNR of all pictures in the video sequence and is the most often-used measure. Suppose there are in total L pictures in the sequence:

$$\text{M-PSNR} = \frac{1}{L}\sum_{l=1}^{L} PSNR_l, \qquad \text{where } PSNR_l = 10\log_{10}\left(\frac{255^2}{MSE_l}\right).$$

$PSNR_l$ and MSE_l are the *Peak Signal to Noise Ratio* and *Mean Squared Error* of the l'th picture in the video sequence, respectively, where MSE_l is defined as follows:

$$MSE_l = \frac{1}{M \times N} \sum_{m=1}^{M} \sum_{n=1}^{N} \left| x_l(m,n) - \hat{x}_l(m,n) \right|^2$$

$x_l(m,n)$ and $\hat{x}_l(m,n)$ are the original and reconstructed pixel values at the spatial location (m,n) of the l'the picture. The picture has M lines, and each line has N pixels. T-PSNR is calculated using the MSE of the total sequence [57]:

$$\text{T-PSNR} = 10 \log_{10} \left(\frac{255^2}{TMSE} \right),$$

$$\text{where } TMSE = \frac{1}{L \times M \times N} \sum_{l=1}^{L} \sum_{m=1}^{M} \sum_{n=1}^{N} \left| x_l(m,n) - \hat{x}_l(m,n) \right|^2$$

T-PSNR weights the quality loss caused by frame skipping more, since a skipped frame causes larger changes in MSE than PSNR (due to the logarithmic operation). In other words, being more temporally smoother results in a higher T-PSNR. Therefore T-PSNR is a better quality measure in the sense that it also captures the impairment of visual smoothness [57]. It should be noted that the PSNR of a skipped frame is calculated by comparing the original frame with the previous reconstructed frame.

Table 9. Performance of the proposed and TMN8 rate-control: Mean PSNR (M-PSNR).

Sequence	Bitrate (kbps)	Proposed (M-PSNR: dB)	TMN8 (M-PSNR: dB)	*Gain (M-PSNR: dB)*	Proposed (Skipped Frames)	TMN8 (Skipped Frames)
Foreman	20	26.93	26.54	*0.39*	2	9
	32	29.13	28.96	*0.17*	0	2
Carphone	20	29.43	29.11	*0.32*	0	4
	32	31.37	31.19	*0.18*	0	0
Coastguard	20	26.24	25.84	*0.40*	0	3
	32	27.90	27.68	*0.22*	0	0
Susie	16	31.11	31.05	*0.06*	0	3
	24	33.19	33.02	*0.17*	0	0

Table 10. Performance of the proposed and TMN8 rate-control: total sequence PSNR (T-PSNR).

Sequence	Bitrate (kbps)	Proposed (T-PSNR: dB)	TMN8 (T-PSNR: dB)	*Gain (T-PSNR: dB)*
Foreman	20	25.95	24.08	*1.87*
	32	28.79	27.68	*1.11*
Carphone	20	28.54	27.93	*0.61*
	32	30.64	30.47	*0.17*
Coastguard	20	26.05	25.53	*0.53*
	32	27.71	27.52	*0.19*
Susie	16	30.15	29.31	*0.84*
	24	32.73	32.64	*0.09*

Table 9 compares the performance of the two rate-control algorithms in terms of M-PSNR. Test sequences are simulated at different bit-rates. The proposed algorithm achieves higher M-PSNR for all cases. The number of skipped frames is reduced as listed in the table. In Table 10, the T-PSNR is calculated as another quality measure. In this metric, the proposed algorithm shows significant advantage over TMN8, especially at very low bit-rates, which demonstrates that T-PSNR correlates well with video quality. The frame-by-frame PSNR comparison for foreman/carphone sequences at 20 kbps is given in Figure 34. The figure clearly shows that the proposed approach improves the video quality in terms of PSNR, and more importantly, improves the temporal smoothness of the video by reducing the number of skipped frames.

Finally, our run-time results (Pentium 4 1.3 GHz, 384 MB Memory, Windows 2000) show that the additional computation needed by the encoder using the proposed algorithm is about 6% on average, and less than 10% in the worst case, compared with

the encoder using the TMN8 algorithm. The additional computation is moderate and is reasonable for practical implementations.

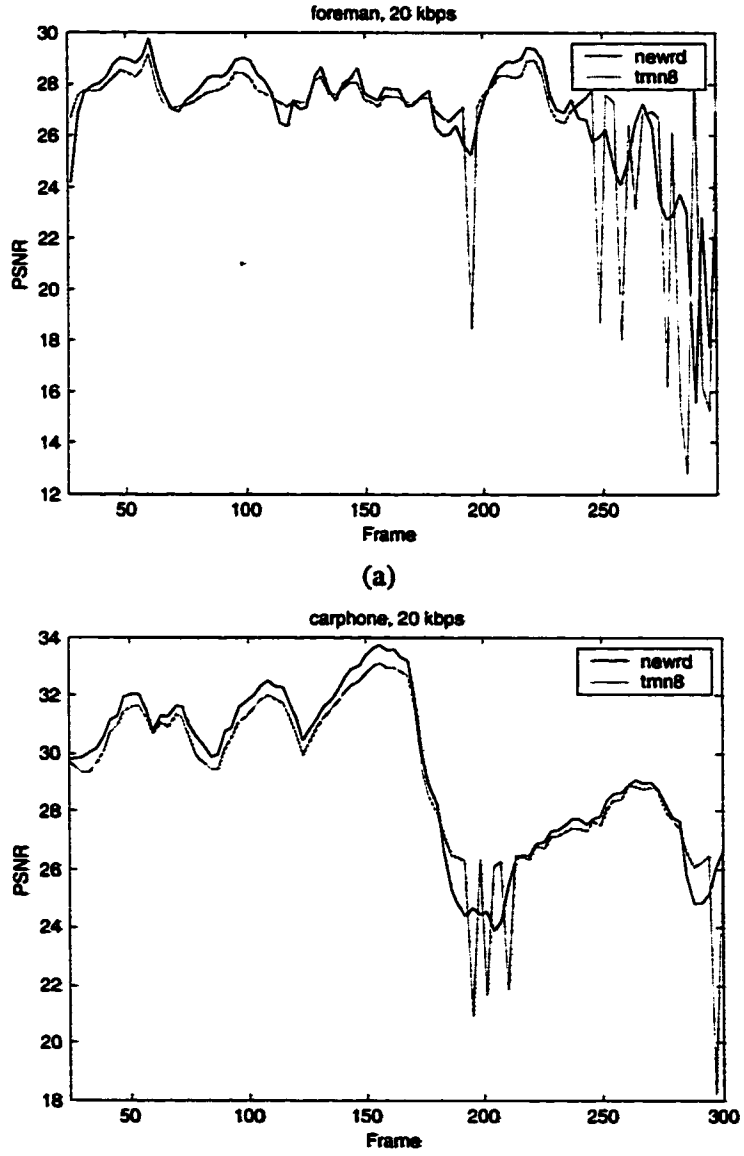


Figure 34. Performance comparison between the proposed ("newrd") and TMN8 rate-control. (a) foreman, 20 kbps. (b) carphone, 20 kbps

5.7. Summary

In this chapter we presented an efficient rate-distortion optimized macroblock-layer rate-control algorithm based on the Lagrange Multiplier algorithm. Taking advantage of the properties of low-bit–rate MPEG-4 video coding, the computation requirement for both macroblock RD-point calculations and the iterative search is greatly reduced. Compared to TMN8, simulations show that it can achieve more accurate buffer control and higher video quality with slightly increased computational complexity. The proposed techniques are general and can be used not only in transcoding, but also in general low-bit–rate video coding.

Chapter 6. Concluding Remarks

Video transcoding offers the capability of dynamically changing the coding format of a compressed video stream. It is one of the key technologies to enable the universal multimedia access. To approach the best quality-computation tradeoff of a video transcoder, we have presented several techniques to improve the quality of the transcoded video and to reduce its computational complexity. In this chapter, we first summarize the major contributions of this dissertation, and then provide a few suggestions for future research.

6.1. Summary of Major Contributions

After introducing the concept and applications of the video transcoding in Chapter 1, we gave an overview of the video coding and transcoding techniques in Chapter 2, where we pointed out the two major research issues in video transcoding: quality and complexity. In the remaining chapters, we focused on the two techniques that most critically affect the quality-complexity tradeoff: motion estimation and rate control.

In Chapter 3, we proposed several motion re-estimation techniques for video transcoding. We proposed to derive a set of motion vectors and use the median of these motion vectors to compose the target motion vector. One unique feature is the support of interlaced to interlaced/progressive video processing. In addition, the proposed techniques achieve efficient frame-rate reduction and frame-type conversion, improved performance against motion inconsistencies, more efficient half-pixel refinement, and

simpler mode decision. Simulations for HDTV to SDTV and MPEG-2 to MPEG-4 SP video transcoding showed the effectiveness of the proposed techniques.

In Chapter 4, we proposed a novel technique to estimate the current output picture complexities from the input complexities and the previous output picture complexities. We then presented a picture-layer rate-control (bit-allocation) algorithm using the estimated output picture-complexities. The proposed bit-allocation algorithm outperforms conventional bit-allocation algorithms by allocating more appropriate number of bits to different pictures. It also has better scene-change and motion-change handling capability.

In Chapter 5, we investigated the rate-distortion optimized macroblock-layer rate-control for low-bit-rate video based on the Lagrange Multiplier (LM) algorithm. Taking into consideration the properties of the LM algorithm-based rate-control for low-bit-rate video, we proposed techniques to significantly reduce its computational requirement. Simulations show the improved performance over commonly used H.263 TMN8. This algorithm can apply to both video coding and transcoding.

In summary, we presented several new techniques to approach the best quality-computation tradeoff for various video transcoding applications. Specifically, to reduce the computational requirement of video transcoding, we proposed efficient and accurate motion re-estimation algorithms that can handle the change of various coding parameters. To improve the video quality of the transcoder output, we proposed a frame-layer bit-allocation scheme for various video transcoding applications, and an efficient rate-

distortion optimized macroblock-layer rate-control for low bit-rate video transcoding and coding. The proposed techniques are practical, and the results obtained are promising.

6.2. Suggestions for Future Research

6.2.1. Rate-Distortion Optimization of Video Transcoding Operations

Operational rate-distortion (RD) optimization has become a powerful tool for enhancing video quality of low-bit-rate video coding. For example, the RD optimization has been applied to bit-allocation [54][59], rate-control [55], motion estimation [60-61], mode decision [58], and frame-type selection, [62] etc., for low bit-rate video coding. The disadvantage of the RD optimization is its high computational complexity. However, in the case of video transcoding, abundant coding statistics can be obtained from the input video stream and may be used to reduce the computational complexity of the RD optimization. For example, in the motion estimation of a macroblock, the smoothness of the input motion vectors of this macroblock and its neighboring macroblocks is an indication of the optimality of the input motion vector in the RD sense. If all neighboring motion vectors are similar and a single motion vector is different from its neighbors, it is likely that this motion vector is not the optimal one in the RD sense. The motion vector for this macroblock may be re-estimated using the RD optimization. And for those macroblocks whose motion vectors are consistent with their neighbors, they may simply reuse the input motion vectors.

6.2.2. *Transcoding to H.264*

H.264, a.k.a. MPEG-4 AVC, is the emerging video coding standard. It offers significant higher efficiency than earlier standards, including other profiles of MPEG-4 and MPEG-2. However, the improved compression efficiency comes at a price. The computational complexity of an H.264 encoder is much higher than other standards too. The increase in the computational complexity comes mainly from the new prediction modes introduced in H.264: multiple reference frames, variable (seven) block-sizes, and various intra prediction modes etc. For transcoding, e.g. from MPEG-2 to H.264, it is interesting to investigate how the input coding statistics can be intelligently utilized to calculate the new prediction modes with minimal computational requirement but can achieve optimal video quality. Intuitively, the various motion re-estimation techniques proposed in Chapter 3 could be extended to perform the multiple reference frame prediction in the H.264. Another example is that the input motion vector field can be used to guide the mode decision (block-size decision).

List of References

[1] Advanced Television Systems Committee (ATSC), "ATSC Document A/53: ATSC Digital Television Standard," Sep. 16, 1995, available on line at *http://www.atsc.org/Standards/A53/A-53_with_Amendment1.pdf.*

[2] ITU-T Recommendation H.263, "Video coding for low bit rate communication," 1998.

[3] ISO/IEC 11172-2, "Information Technology – Coding of moving pictures and associated audio for digital storage media at up to about 1.5Mbit/s – Part 2: Video," Edition 1, 1993. (MPEG-1 Video)

[4] ISO/IEC 13818-2, "Information technology – Generic coding of moving pictures and associated audio information: Video", Edition 2, 2000. (MPEG-2 Video).

[5] ISO/IEC 14496-2:2001, "Coding of Audio-Visual Objects - Part 2: Visual," 2nd Edition, 2001. (MPEG-4 Video).

[6] R. Mohan, J.R. Smith, and C.-S. Li, "Adapting multimedia internet content for universal access," *IEEE Transactions on Multimedia*, Vol. 1, No. 1, March 1999.

[7] N. Shacham, "Multipoint communication by hierarchically encoded data," in *Proc. IEEE INFOCOM'92*, May 1992, pp. 2107-2114.

[8] M. Ghanbari, "Two-layer coding of video signals for VBR networks," *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 5, June 1989, pp. 771-781.

[9] W. Li, "Fine granularity scalability in MPEG-4 for streaming video," *IEEE International Symposium on Circuits and Systems* 2000, vol. 1, Geneva, Switzerland, May 2000, pp. 299-302.

[10] J. Ribas-Corbera and S. Lei, "Rate control in DCT video coding for low-delay communications," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 9, No. 1, pp. 172-85, Feb. 1999.

[11] ITU-T Recommendation H.261, "CODEC for audio-visual services at p × 64 kbit/s," 1993.

[12] A. Ahmed, T. Natarajan, and K.R. Rao, "Discrete cosine transforms," *IEEE Transactions on Computers*, C-23: 90-93, 1974.

[13] Y. Wang, J. Ostermann, and Y.-Q. Zhang, "Video processing and communications," Prentice Hall, 2001.

[14] B. Girod, "The efficiency of motion-compensating prediction for hybrid coding of video sequences," *IEEE Journal on Selected Areas in Communications*, Vol. SAC-5, pp. 1140-1154, Aug. 1987.

[15] B. Girod, "Motion-compensating prediction with fractional-pel accuracy," *IEEE Transactions on Communications*, Vol. 41, No. 4, pp. 604-612, April 1993.

[16] B. G. Haskell, A. Puri, and A. N. Netravali, "Digital video: an introduction to MPEG-2," New York: Chapman & Hall, 1997.

[17] MPEG, "Studio profiles," Final Draft Amendment, *Doc. ISO/MPEG N3898*, Pisa MPEG Meeting, January, 2001.

[18] MPEG, "Streaming video profiles," Final Draft Amendment, *Doc. ISO/MPEG N3904*, Pisa MPEG Meeting, January, 2001.

[19] JVT of MPEG and ITU-T VCEG, "Committee Draft of Joint Video Specification," *Doc. MPEG02/N4810*, May 2002, Fairfax, MA.

[20] J. R. Jain, and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Transactions on Communications*, Dec. 1981, COM-29:1799-808.

[21] T. Koga et al., "Motion-compensated interframe coding for video conferencing," *National Telecommunication Conference*, Nov. 1981, G5.3.1-5, New Orleans, LA.

[22] A. Eleftheriadis and D. Anastassiou, "Constrained and general dynamic rate shaping of compressed digital video," *IEEE International Conference on Image Processing*, Washington, DC, Oct. 1995.

[23] H. Sun, W. Kwok, and J. W. Zdepski, "Architectures for MPEG compressed bitstream scaling," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 6, No. 2, April, 1996.

[24] J. Youn, M.T. Sun, and J. Xin, "Video transcoder architectures for bit rate scaling of H.263 bit streams," *ACM Multimedia* 1999, Orlando, Nov. 1999.

[25] G. Keesman, R. Hellinghuizen, F. Hoeksema, and G. Heideman, "Transcoding of MPEG bitstreams," *Signal Processing: Image Communication*, Vol. 8, No. 6, Sep. 1996, pp. 481-500.

[26] P.A.A. Assuncao and M. Ghanbari, "A frequency-domain video transcoder for dynamic bitrate reduction of MPEG-2 bit streams," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 8, pp. 953-967, Dec. 1998.

[27] T. Shanableh, and M. Ghanbari, "Heterogeneous video transcoding to lower spatial-temporal resolutions and different encoding froamts," *IEEE Transactions on Multimedia*, Vol. 2, No. 2, June 2000.

[28] J. Youn and M.T. Sun, "Video Transcoding with H.263 Bit Streams," *Journal of Visual Communication and Image Representation*, pp. 385-404, December 2000.

[29] S.-F. Chang and D. G. Messerschmitt, "Manipulation and compositing of MC-DCT compressed video," *IEEE Journal on Selected Areas in Communications*, vol. 13, pp. 1-11, Jan. 1995.

[30] J. Song and B.-L. Yeo, "A fast algorithm for DCT-Domain inverse motion compensation based on shard information in a macroblock," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 5, pp. 767-775, August 2000.

[31] S. Liu and A. C. Bovik, "Look-up-table based DCT domain inverse motion compensation," *IEEE International Conference on Image Processing*, vol. 2, pp. 965-968, 2001.

[32] B. Shen, I. K. Ishwar, and V. Bhaskaran, "Adaptive motion-vector re-sampling for compressed video downscaling," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, pp. 929-936, Sept. 1999.

[33] N. Bjork and C. Christopoulos, "Transcoder architecture for video coding," *IEEE Transactions on Consumer Electronics*, Vol. 44, pp. 88-98, Feb. 1998.

[34] Susie J. Wee, John G. Apostolopoulos, and Nick Feamster, "Field-to-frame transcoding with spatial and temporal downsampling," *IEEE International Conference On Image Processing*, 1999, Vol. 4, pp. 271-275.

[35] J. Youn, M.T. Sun, and C.W. Lin, "Motion vector refinement for high-performance transcoding," *IEEE Transactions on Multimedia*, Vol. 1, pp. 30-40, Mar. 1999.

[36] S.-F. Chang, and A. Eleftheriadis, "Error accumulation of repetitive image coding," *IEEE International Symposium on Circuits and Systems*, 1994, vol. 3, pp. 201-204.

[37] M. Pereira, and A. Lippman, "Re-codable video," *IEEE International Conference on Image Processing*, 1994, vol. 2, pp. 952-956.

[38] Y. Nakajima, H. Hori, and T. Kanoh, "Rate conversion of MPEG coded video by re-quantization process," *IEEE International Conference On Image Processing*, 1995, Vol. 3, pp. 408-411.

[39] P. Yin, M. Wu, and B. Liu, "Video transcoding by reducing spatial resolution," IEEE *International Conference on Image Processing*, 2000.

[40] P. Yin, A. Vetro, H. Sun, and B. Liu, "Drift compensation architectures and techniques for reduced resolution transcoding," *SPIE: Visual Communications and Image Processing*, 2002, San Jose, CA.

[41] ISO/IEC CD 15938-5, "Information technology -- Multimedia content description interface -- Part 5: Multimedia description schemes," Edition 1, 2001. (MPEG-7 MDS)

[42] P. M. Kuhn, T. Suzuki, A. Vetro, "MPEG-7 transcoding hints for reduced complexity and improved quality," *International Packet Video Workshop 2001*, Kyongju, Korea.

[43] G. Shen, B. Zeng, Y.-Q.Zhang, and M. L. Liou, "Transcoder with arbitrarily resizing capability," *IEEE International Symposium on Circuits and Systems*, 2001.

[44] J. Youn, J. Xin, and M.T. Sun, "Fast video transcoding architectures for networked multimedia applications," *IEEE International Symposium on Circuits and Systems*, 2000.

[45] R. Swann, and N. Kingsbury, "Transcoding of MPEG-II for enhanced resilience to transmission errors," *IEEE International Conference on Image Processing*, 1996, vol.2, pp. 813-816.

[46] G. de los Reyes, A. R. Reibman, S.-F. Chang, and J. C.-I. Chuang, "Error-resilient transcoding for video over wireless channels," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, June 2000.

[47] S. Dogan, et. al., "Error-resilient video transcoding for robust internetwork communications using GPRS," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 6, June 2002, pp. 453-564.

[48] S. Dogan, A.H. Sadka, and A.M. Kondoz, "Efficient MPEG-4/H.263 video transcoder for interoperability between heterogeneous multimedia networks," *IEE Electronics Letters*, 35, no. 11, 863-864, May 1999.

[49] R. E. Crochiere, L. R. Rabiner, "Multirate digital signal processing," *Prentice-Hall*, 1983.

[50] "Test Model 5," ISO/IEC JTC1/SC29/WG11, N0400, Apr. 1993.

[51] MPEG-4 Verification Model 17, July 2000 Beijing.

[52] P.N. Tudor and O.H. Werner, "Real-time transcoding of MPEG-2 video bit streams," *IEE International Broadcasting Convention*, Amsterdam, 1997, pp. 286-301.

[53] I. Koo, P. Nasiopoulos, and R. Ward, "Joint MPEG-2 coding for multi-program broadcasting of pre-recorded video," *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 4, 1999, pp. 2227-2230.

[54] A. Ortega and K. Ramchandran, "Rate-distortion methods for image and video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 23-50, Nov. 1998.

[55] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. 36, No. 9, pp. 1445-1453, Sep. 1988.

[56] R. Ramchandran and M. Vetterli, "Best wavelet packet bases in a rate-distortion sense," *IEEE Transactions on Image Processing*, Vol. 2, No. 2, pp. 160-175, Apr. 1993.

[57] J. Goshi, A.E. Mohr, R.E. Ladner, E.A. Riskin, and A. Lippman, "Unequal loss protection for H.263 compressed video," *Submitted to IEEE Transactions on Circuits and Systems for Video Technology*, June 2002.

[58] G. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 74-90, Nov. 1998.

[59] K. Ramchandran, A. Ortega, "Bit allocation for dependent quantization with applications to multi-resolution and MPEG video coders," *IEEE Transactions on Image Processing*, Vol. 3, No. 5, 1994.

[60] B. Girod, "Rate-constrained motion estimation," *SPIE: Visual Communications and Image Processing*, Nov. 1994, pp. 1026-1034.

[61] C. Chen, and A.N. Willson, Jr., "Rate-distortion optimal motion estimation algorithms for motion-compensated transform video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 8, No. 2, Apr. 1998.

[62] J. Lee, and B.W. Dickinson, "Rate-distortion optimized frame type selection for MPEG encoding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 3, June 1997, pp. 501-510.

Curriculum Vitae

Jun Xin was born in Pizhou, Jiangsu Province, China on January 10, 1973. He received his B.S. degree in Radio Engineering from Southeast University, Nanjing, China, in 1993, M.S. degree in Electrical Engineering from Institute of Automation, Chinese Academy of Sciences, Beijing, China in 1996, and Ph.D. degree in Electrical Engineering from University of Washington, Seattle, in 2002.

From 1996 to 1998 he was a software engineer at Motorola-ICT (Institute of Computing Technology) Joint R&D Lab, Beijing, China, participating in the development of MPEG-2 video codec products. From 1998 to 2002, he was a research assistant in the Department of Electrical Engineering at the University of Washington. He was an intern with Microsoft Research China in summer 1999, and with VideoTele in summer 2000, respectively.

His research interests include video coding and transcoding, multimedia signal processing and communication.

JOURNAL PUBLICATIONS

J. Xin, M.T. Sun, B. Choi, and K. Chun, "An HDTV to SDTV Transcoder", IEEE Transactions on Circuits and Systems for Video Technology, November, 2002.

J. Xin, M.T. Sun, and T.-D. Wu, "Optimization techniques for an MPEG-2 to MPEG-4 video transcoder," Submitted to the Journal of Visual Communications and Image Representation, October 2002.

CONFERENCE PUBLICATIONS

J. Xin, M.T. Sun, and K. Chun, "Motion Re-estimation for MPEG-2 to MPEG-4 Simple Profile Transcoding", Packet Video Workshop, Pittsburgh, PA, 2002.

J. Xin, M.T. Sun, and K. Chun, "Motion re-estimation for HDTV to SDTV transcoding", IEEE International Symposium on Circuits and Systems, Phoenix, AZ, 2002.

J. Xin, M.T. Sun, and K. Chun, "Bit allocation for transcoding of pre-encoded video streams", Proceedings of the SPIE: Visual Communications and Image Processing, San Jose, CA, 2002.

J. Xin, M.T. Sun, and T.-D. Wu, "Motion vector composition for MPEG-2 to MPEG-4 video transcoding", Workshop and Exhibition of MPEG-4, San Jose, CA, 2002.

J. Xin, M.T. Sun, and K.S. Kan, "Bit allocation for joint transcoding of multiple MPEG coded video streams", IEEE International Conference on Multimedia and Expo, Tokyo, Japan, 2001.

J. Youn, J. Xin, M.T. Sun and Y.Q. Zhang, "Video transcoding for multiple clients", Proceeding of the SPIE: Visual Communications and Image Processing, Perth, Australia, 2000.

J. Youn, J. Xin, and M.T. Sun, "Fast video transcoding architecture for networked multimedia applications", IEEE International Symposium on Circuits and Systems, Geneva, Switzerland, 2000.

J. Youn, M.T. Sun, and J. Xin, "Video transcoder architectures for bit rate scaling of H.263 bit streams," ACM Multimedia, Orlando, November 1999.