An Efficient Motion Estimation Method for H.264-Based Video Transcoding with Arbitrary Spatial Resolution Conversion

by

Jiao Wang

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2007

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.
I understand that my thesis may be made electronically available to the public.

Abstract

As wireless and wired network connectivity is rapidly expanding and the number of network users is steadily increasing, it has become more and more important to support universal access of multimedia content over the whole network. A big challenge, however, is the great diversity of network devices from full screen computers to small smart phones. This leads to research on transcoding, which involves in efficiently reformatting compressed data from its original high resolution to a desired spatial resolution supported by the displaying device. Particularly, there is a great momentum in the multimedia industry for H.264-based transcoding as H.264 has been widely employed as a mandatory player feature in applications ranging from television broadcast to video for mobile devices.

While H.264 contains many new features for effective video coding with excellent rate distortion (RD) performance, a major issue for transcoding H.264 compressed video from one spatial resolution to another is the computational complexity. Specifically, it is the motion compensated prediction (MCP) part. MCP is the main contributor to the excellent RD performance of H.264 video compression, yet it is very time consuming. In general, a brute-force search is used to find the best motion vectors for MCP. In the scenario of transcoding, however, an immediate idea for improving the MCP efficiency for the re-encoding procedure is to utilize the motion vectors in the original compressed stream. Intuitively, motion in the high resolution scene is highly related to that in the down-scaled scene.

In this thesis, we study homogeneous video transcoding from H.264 to H.264. Specifically, for the video transcoding with arbitrary spatial resolution conversion, we propose a motion vector estimation algorithm based on a multiple linear regression model, which systematically utilizes the motion information in the original scenes. We also propose a practical solution for efficiently determining a reference frame to take the advantage of the new feature of multiple references in H.264. The performance of the algorithm was assessed in an H.264 transcoder. Experimental results show that, as compared with a benchmark solution, the proposed method significantly reduces the transcoding complexity without degrading much the video quality.

Acknowledgements

First and foremost, I would like to express all my gratitude to my supervisor, Professor En-hui Yang, for his excellent guidance and valuable comments throughout my research, and for his support during my graduate studies at Waterloo. Without his help, the achievements in my research would have never been possible.

I would like to thank Professor George H. Freeman and Professor Oleg Michailovich for being the readers for this thesis and for their insightful comments and suggestion.

I would also like to thank all my colleagues in the multimedia communications group for their numerous stimulating conversations into my work. A special word of thanks goes out to Xiang Yu and Dr. Haiquan Wang for many valuable discussions related to this work.

I would like to express my appreciation to my friends for always providing me with joy and warmth throughout these years. I wish you all the best for your future endeavors.

My deepest gratitude and love belong to my parents for their constant support and continuous love throughout the past years.

Contents

1 Introduction **1**

1.1 Motivation and Applications 1
1.2 Research Challenges 2
1.3 Main Contribution 3
1.4 Thesis Outline 3

2 Video Transcoding Overview **4**

2.1 Video Transcoding Introduction 4
2.2 Video Transcoding Architectures 6
2.2.1 Frequency Domain Video Transcoder 6
2.2.2 Cascaded Pixel Domain Video Transcoder 7
2.3 Video Transcoding Functionalities 9
2.3.1 Bit Rate Reduction 10
2.3.2 Spatial Resolution Conversion 11
2.3.3 Temporal Resolution Conversion 15
2.3.4 Transcoding Between Multiple and Single Layers 17
2.4 Existing Motion Vector Re-Estimation Methods Overview 17
2.4.1 Average and Median Methods 19
2.4.2 Adaptive Motion Vector Resampling 20
2.4.3 Adaptive Motion Estimation 21
2.4.4 Predictive Motion Estimation 23
2.5 Summary 24

3 H.264-Based Video Transcoding with Spatial Resolution Conversion 26
3.1 H.264 Video Coding Standard . . . 26
3.1.1 Introduction . . . 26
3.1.2 The H.264 Codec . . . 27
3.1.3 The H.264 Structure . . . 29
3.1.4 H.264 Features . . . 32
3.2 Existing Research Work Overview . . . 33
3.2.1 Mode Mapping Method . . . 34
3.2.2 Area Weighted Vector Median . . . 35
3.2.3 Bottom-Up Motion Vector Re-Estimation . . . 36
3.2.4 A Fast Transcoding with Rate-Distortion Optimal Mode Decision . 38
3.3 Summary . . . 38

4 An Efficient Motion Estimation Algorithm 40
4.1 Problem Formulation . . . 40
4.2 Problem Solution . . . 42
4.2.1 Determining Reference Frame . . . 42
4.2.2 Searching for Motion Vectors . . . 43
4.3 Summary . . . 45

5 Experimental Results 46
5.1 System Setup and Test Condition . . . 46
5.2 Experimental Results . . . 47

6 Conclusion and Future Work 57
6.1 Conclusion . . . 57
6.2 Future Work . . . 58

Bibliography 60

List of Tables

5.1 Transcoding performance of seven methods with a down-sampling ratio of 3:2. 50
5.2 Comparative results over the "Benchmark" with a down-sampling ratio 3:2. 51
5.3 Transcoding performance of seven methods with a down-sampling ratio of 2:1. 52
5.4 Comparative results over the "Benchmark" with a down-sampling ratio 2:1. 53
5.5 Transcoding performance of seven methods with a down sampling ratio of 3:1. 54
5.6 Comparative results over the "Benchmark" with a down-sampling ratio 3:1. 55
5.7 Comparative results over the "Full Search" with a down-sampling ratio 2:1. 55

List of Figures

2.1 Video transcoding operations. 5
2.2 Frequency domain transcoder with re-quantization scheme. 7
2.3 Cascaded pixel domain transcoder. 8
2.4 Video transcoding functions. 9
2.5 Transcoding architecture with spatial/temporal resolution conversion. . . . 12
2.6 Motion vector correlation. 13
2.7 FGS transcoder with motion vector reused. 18
2.8 Scaled-down motion vector from four macroblocks. 19
2.9 Macroblock type. 22
2.10 Motion vector estimation by corresponding and neighboring macroblocks. . 23

3.1 H.264 encoder. 28
3.2 H.264 decoder. 29
3.3 Multiple macroblock partitions for motion compensated prediction. 31
3.4 Macroblocks positions for mode mapping. 35

4.1 Block diagram of video transcoding with spatial resolution conversion. . . . 41
4.2 Motion vector mapping. 42

5.1 RD performance comparison of “Akiyo” with down-sampling ratio 3:2. . . 49
5.2 RD performance comparison of “Akiyo” with down-sampling ratio 3:2. . . 51
5.3 RD performance comparison of “Akiyo” with down-sampling ratio 2:1. . . 53
5.4 RD performance comparison of “Akiyo” with down-sampling ratio 2:1. . . 55
5.5 RD performance comparison of “Akiyo” with down-sampling ratio 3:1. . . 56

5.6 RD performance comparison of “Akiyo” with down-sampling ratio 3:1. . . 56

Glossary

AME Adaptive Motion Estimation

AMVR Adaptive Motion Vector Resampling

ASO Arbitrary Slice Ordering

CABAC Context-Based Adaptive Binary Arithmetic Coding

CAVLC Context Adaptive Variable Length Coding

CBR Constant Bit Rate

CIF Common Intermediate Format

DCT Discrete Cosine Transform

DSL Digital Subscriber Line

DVD Digital Video Disc

FGS Fine Granularity Scalability

FMO Flexible Macroblock Ordering

GOP Group of Pictures

ISDN Integrated Services Digital Networks

ISO International Organization for Standardization

ITU-T International Telecommunication Unit-Telecommunication

JVT Joint Video Team

LAN Local Access Networks

MAD Mean Absolute Difference

MCP Motion Compensated Prediction

MPEG Moving Picture Experts Group

NAL Network Abstraction Layer

PME Predictive Motion Estimation

PSNR Peak Signal to Noise Ratio

QCIF Quarter Common Intermediate Format

RBSP Raw Byte Sequence Payload

RD Rate Distortion

VBR Variable Bit Rate

VLC Variable Length Encoding

VLD Variable Length Decoding

VCEG Video Coding Experts Group

VCL Video Coding Layer

WLAN Wireless Local Access Networks

Chapter 1

Introduction

1.1 Motivation and Applications

With the development of multimedia systems with diverse terminal device constraints, numbers of networks, network limitations or preferences of a user, interactivity and integration of different systems and different networks are becoming more and more important. In the universal multimedia access [1], [2], [3], on one hand, we have multimedia content that is growing fast in our daily life. On the other hand, there exists a variety of network terminals, such as full screen computers, laptops, personal digital assistants, smart cellular phones, etc. These network terminals have varying constraints and capabilities, including memory, computing power, display capability, etc. In addition, different terminals may have different access to the Internet, including local access networks (LAN), digital subscriber line (DSL), integrated services digital networks (ISDN), cable and wireless local access networks (WLAN). Furthermore, the networks used to connect those two entities may have different channel characteristics such as bandwidths and bit error rates.

In such multimedia systems, a big challenge is the great diversity of network devices from full screen computers to small smart phones. This leads to research on video transcoding, which involves efficiently reformatting compressed multimedia data from its original high resolution to a desired spatial resolution supported by the displaying device.

As we know, H.264 video coding standard is the newest video coding standard which has improved coding efficiency and simple syntax specification compared to previous video

coding standards. Motivated by the wide adoption of H.264 and the demand of universal multimedia data access over the expanding network with diverse devices, this thesis studies H.264-based video transcoding with spatial resolution conversion.

1.2 Research Challenges

A major issue for transcoding H.264-compressed video data from one spatial resolution to another is the computational complexity. Straightforwardly, a benchmark solution to H.264 video transcoding with spatial resolution conversion will be to first decode the bitstream, then down-sample the decoded video sequence, and finally re-encode the down-scaled video data with H.264 codec. The bottleneck, however, is the high complexity of the re-encoding procedure, as the H.264 decoding and the down-sampling are generally many-time faster than the H.264 encoding.

Specifically, the most time-consuming part is the MCP part in the re-encoding procedure. To achieve the best RD performance, a brute-force search within a given range is used to find the best motion vectors for MCP in the H.264 encoding procedure. In the scenario of transcoding with spatial resolution conversion, however, the MCP in the H.264 re-encoding may not necessarily count on the brute-force search due to an observation that the motion information obtained in the decoding procedure is highly correlated with the RD optimal motions in the re-encoding procedure.

To meet real time application requirement, it is beneficial to speed up the MCP in H.264 re-encoding by investigating and utilizing the correlation between full-scale motions in the original frames and that in the down-scaled images. Therefore, two important challenges for our work are:

- How to efficiently exploit the motion information from the original compressed video stream to speed up the MCP in the re-encoding procedure.
- How to keep the RD coding performance for the down-scaled video sequences as high as possible.

Our algorithm design is desirable to fulfill all of the above challenges.

1.3 Main Contribution

In this thesis, we propose an efficient and effective MCP method for homogeneous video transcoding from H.264 to H.264 with spatial resolution conversion. As discussed in the above, transcoding from H.264 features rich motion information in the full-scale scenes, which we plan to utilize for enhancing the MCP effectiveness in the following H.264 re-encoding procedure.

Specifically, we propose a motion vector estimation algorithm based on a multiple linear regression model, which systematically utilizes the motion information in the original scenes. We also propose a practical solution for efficiently determining a reference frame to take the advantage of the new feature of multiple references in H.264. Compared with other solutions, our proposed method significantly reduces the transcoding complexity while maintaining an RD coding performance very close to the best achievable.

1.4 Thesis Outline

The rest of the thesis is organized as follows. Chapter 2 introduces the video transcoding architectures, functionalities of homogeneous video transcoding and heterogeneous video transcoding. Different research issues and existing work are also described for each function of video transcoding. An overview of previous related work on motion vector composition for video transcoding from higher spatial resolution to lower spatial resolution in the literature is also presented. H.264 video coding standard and the existing research work on H.264-based video transcoding with spatial resolution conversion are introduced in Chapter 3. In Chapter 4, a formulation description of the motion re-estimation problem for H.264-based video transcoding with spatial resolution conversion is presented. The proposed transcoding method is also presented in this Chapter. Experimental results and analysis are presented in Chapter 5. Finally, conclusions are drawn in Chapter 6.

Chapter 2

Video Transcoding Overview

In this chapter, we first introduce the video transcoding architecture, including the frequency domain video transcoder and the pixel domain video transcoder. Next, we discuss the functionalities of homogeneous video transcoding and heterogeneous video transcoding. Different research issues related to each function of video transcoding are also described, including the research problems, existing solutions, their advantages and disadvantages. Based on the above discussion, the existing motion vector re-estimation methods for video transcoding with spatial resolution conversion are then introduced in details in section 2.4. Finally, a summary for this chapter is given.

2.1 Video Transcoding Introduction

A multimedia system may consist of various devices such as laptops, personal digital assistants and smart cellular phones interconnecting via wireline and wireless networks. In such a system, network terminals vary significantly in capabilities like display, memory, processing and decoder. In order to allow access by receiving devices with different available resources, the multimedia data originally compressed by means of a certain format may need format conversion or bit rate adjustment in order to make the content adaptive to the capabilities of diverse networks and terminal devices. Video transcoding is a key technology to enable multimedia devices with diverse capabilities to exchange video content via heterogeneous wireline and wireless networks.

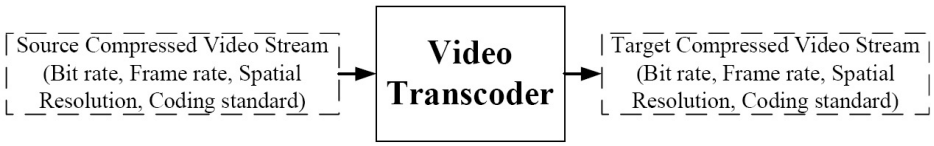


Figure 2.1: Video transcoding operations.

Video transcoding is a technique to convert a video sequence from one compressed stream into another, it may include bit rate conversion, spatial resolution conversion, temporal resolution conversion and/or coding syntax modification, as shown in Figure 2.1. One scenario is delivering multimedia data via heterogeneous wireline or wireless networks with diverse network bandwidths. Here, a video transcoder can perform dynamic adjustments in the bit rate of the video stream to suit available network bandwidth. Another scenario is exchanging multimedia content in a video conferencing system over the Internet in which the participants may be using different network terminals. Here, a video transcoder can perform frame size conversion or frame rate conversion to enable multimedia content exchange. Therefore, video transcoding is a flexible way to enable the interoperability between different systems and diverse networks.

Recently, several video coding standards have been developed for different multimedia applications. H.261, H.263, H.263+ video coding standards [4], aimed for low bit rate video applications such as videophone and videoconferencing, were defined by International Telecommunication Unit-Telecommunication (ITU-T) Video Coding Experts Group (VCEG). The MPEG video coding standards [5] were developed by International Organization for Standardization (ISO). MPEG-2 is mainly serves for high bit rate and high quality applications such as Digital Video Disc (DVD) and digital TV broadcasting. MPEG-4 is aimed for multimedia applications such as video streaming applications. The H.264 video coding standard [6], which is noted for achieving very high data compression, was jointly written by the ITU-T VCEG together with the ISO/IEC Moving Picture Experts Group (MPEG) as the product of a collective partnership effort known as the Joint Video Team (JVT).

As the demand of universal multimedia data access over the expanding network with diverse devices increases, interactivity between different systems is becoming highly de-

sirable. Video transcoding is needed to allow the inter-compatibility between different multimedia stream within or across different video coding standards [7], [8]. As shown in Figure 2.1, adjustment of bit rate and coding parameters of compressed video, spatial and temporal resolution conversions can all be done through video transcoding.

2.2 Video Transcoding Architectures

There are two common architectures can be used to implement video transcoding. One is the frequency domain video transcoder and another one is the cascaded pixel domain video transcoder.

2.2.1 Frequency Domain Video Transcoder

The frequency domain video transcoder operates directly in the frequency domain, as shown in Figure 2.2. In this architecture, the incoming compressed video stream is first variably length decoded, then inverse quantized to get the discrete cosine transform (DCT) coefficients. Those DCT coefficients are re-quantized. After the variable length decoding procedure, the decoded motion information is directly passed to the variable length encoding (VLD) process. Together with the re-quantized DCT coefficients, they are variably length encoded to get the target compressed video stream.

The frequency domain transcoder operates directly in the frequency domain and there is no need for an inverse DCT-based transform, thus it is very simple to implement and it is computationally efficient. However, the frequency domain video transcoder architecture is subject to drift error [2], which can be defined as the blurring or smoothing of successively predicted frames [7].

The reason for drift error is explained as follows. In predictive coding, each frame in a video sequence is predicted from its reference frames to get the predicted frame. The prediction residue errors between the original frame and the predicted frame are encoded. In order to make the decoder work properly with the encoder, the reconstructed frames used as reference frames in the decoding part must be exactly the same as those reference frames used in the encoding part. In the frequency domain transcoder, the prediction residue errors are changed due to the different quantizer used in the re-quantization process. Therefore,

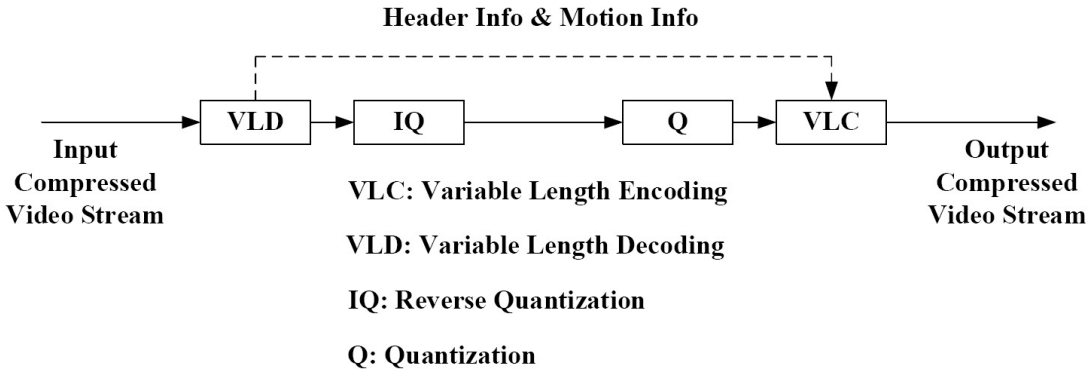


Figure 2.2: Frequency domain transcoder with re-quantization scheme.

the reconstructed reference frames used in the decoding part become different from those in the encoding process. As time goes on, the differences will accumulate through the whole group of pictures (GOP) and result in severely degraded reconstructed frames. This may finally cause a severe degradation in video quality [2], [9].

2.2.2 Cascaded Pixel Domain Video Transcoder

The video transcoder architecture shown in Figure 2.3 is a cascaded pixel domain video transcoder. This cascaded pixel domain video transcoder cascades the decoder and encoder directly.

In the decoding part, the incoming compressed video stream is fully decoded to the pixel domain by performing variably length decoding, inverse quantization, inverse DCT transform and MCP procedure. In the encoding part, the predicted video frames of decoded video frames are produced by the MCP procedure. The prediction residue errors between the original video frames and the predicted video frames are encoded by performing DCT transform, re-quantization process and variable length encoding procedure to get the target compressed video stream with a desirable bit rate or format.

Compared to the frequency domain transcoder, the cascaded pixel domain transcoder contains a feedback loop in the transcoding architecture. This feedback loop can correct the transcoding distortion by compensating the drift error in the transcoder [10]. However,

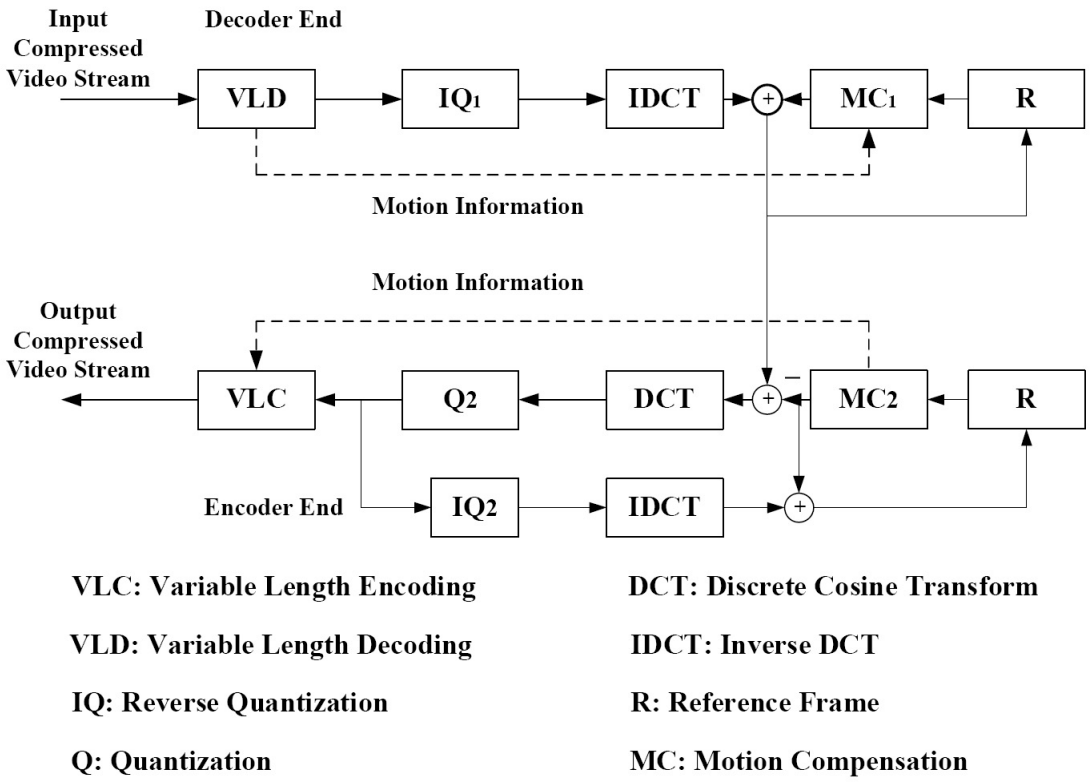


Figure 2.3: Cascaded pixel domain transcoder.

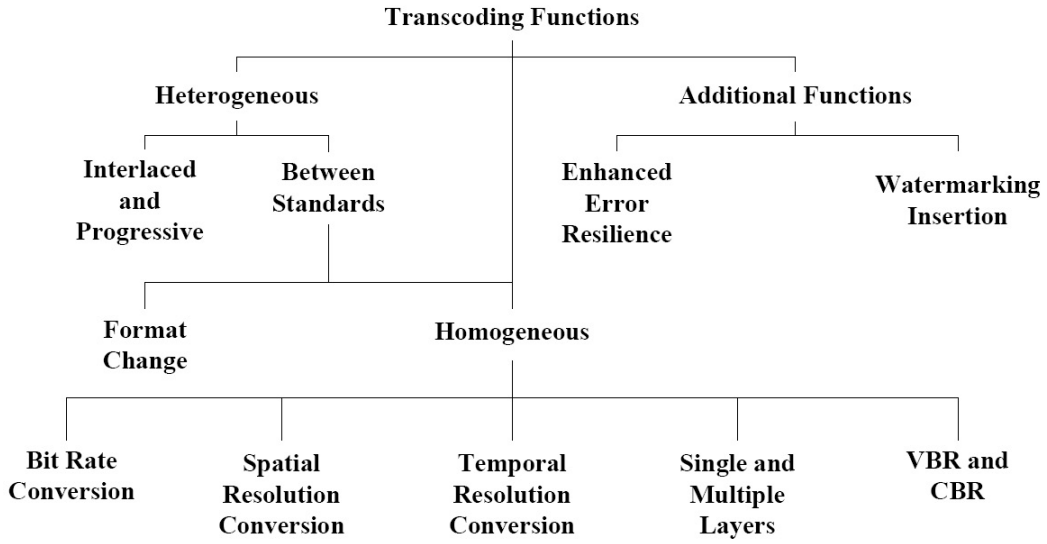


Figure 2.4: Video transcoding functions.

the frequency domain transcoder, though not drift error free, can still be quite useful due to its potentially low cost in computation and required frame memory while the degradation in the visual quality is acceptable. For the cascaded pixel domain transcoder, reducing the complexity of the implementation is a major driving force behind many research activities on video transcoding.

2.3 Video Transcoding Functionalities

Generally speaking, video transcoding is a technique to convert one compressed video stream into another. Different functions, such as adjustment of bit rate and format conversion, can be provided by a video transcoder. These functionalities [8] are illustrated in Figure 2.4. There exist two different scenarios for video transcoding: homogeneous video transcoding and heterogeneous video transcoding.

Homogeneous video transcoding re-encodes compressed video stream within the same video coding standard but with different parameters, such as from MPEG-1 to MPEG-1 or from H.263 to H.263 with different spatial/temporal resolutions. As shown in Figure

2.4, homogeneous video transcoding can provide several functions, including bit rate adjustment, spatial/temporal resolution conversion, transcoding between single and multiple layers, and constant bit rate (CBR) stream to variable bit rate (VBR) stream conversion.

On the contrary, heterogeneous video transcoding [11] extends this scenario to a conversion between different video coding standards, such as from MPEG-2 to H.263 and from MPEG-4 to H.264. As shown in Figure 2.4, heterogeneous video transcoding includes the coding syntax conversion between different video coding standards, such as the frame resolution, the frame type, frame rate and directionality of motion vectors [8]. Furthermore, a heterogeneous video transcoder provides the functionalities of a homogeneous video transcoder, including bit rate conversion, spatial resolution conversion and temporal resolution conversion, etc.

There are additional functions of video transcoding, such as enhanced error resilience for video over wireless channel and logo or watermarking insertion.

The following sections describe some of the research issues for different video transcoding functionalities, including bit rate reduction, spatial resolution conversion, temporal resolution conversion and transcoding between multiple and single layers.

2.3.1 Bit Rate Reduction

Video transcoding with bit rate reduction aims to reduce the bit rate while keeping the highest video quality possible and maintaining low complexity. Applications requiring this type of video transcoding include television broadcast and multimedia streaming. The video transcoding architectures shown in both Figure 2.2 and Figure 2.3 can be used to realize the bit rate reduction function of video transcoding.

For the bit rate reduction with fixed spatial and temporal resolution, there are two common techniques to reduce the bit rate of the output compressed video stream. One simple technique is called selective transmission [10], [12]. After variably length decoding and inverse quantization, it discards high frequency DCT coefficients from each macroblock as needed in order to obtain the target bit rate. In this technique, the re-quantizer may have the same or different quantizer level as the original quantizer which used in the coding process to produce the input compressed video stream. By this technique, the bit rate can be reduced while preserving an acceptable reconstructed video quality since most of the

energy is concentrated in the low frequency band of a frame. However, sometimes the blocking effects may be introduced in the resulting video.

Another technique to reduce the bit rate is to perform the re-quantization. In this type, the VLD part first decodes the incoming compressed video stream to extract the variable length code words corresponding to the quantized DCT coefficients. Meanwhile, the motion information data and other macroblock level information can also be extracted from the VLD part. The quantized coefficients are first inverse quantized using the original quantizer levels. Those coefficients are then re-quantized with a different quantizer in order to obtain the target video stream with lower bit rate [13], [14], [15]. Finally, the re-quantized coefficients and stored macroblock level information are encoded through the variable length coding (VLC) process. By increasing the quantization step for the re-quantization process, the number of nonzero quantized coefficients will decrease, which will result in reducing the amount of bits of the resulting bit stream. The re-quantization is a good method to control the bit rate. It also places a good balance between the complexity and reconstructed video quality.

Furthermore, the macroblock types extracted from the VLD part are not optimum for re-encoding procedure at the reduced bit rate by re-quantization. Thus, macroblock type decision for the output video stream should be taken into account. To solve this problem, Sun [10] proposed to always re-evaluate the macroblock type at the encoder part of the transcoder.

1. If it was encoded as intra, again encode it in intra mode.

2. If it was encoded as skipped, again code it as skipped mode.

3. If it was encoded in inter, check to see if all coefficients of the macroblock are zero and if they are coded as skipped mode, else check again whether the macroblock has to be coded in intra or inter mode.

2.3.2 Spatial Resolution Conversion

Spatial resolution conversion is needed when the terminal devices have constrained display capabilities. The cascaded pixel domain architecture used to implement the video

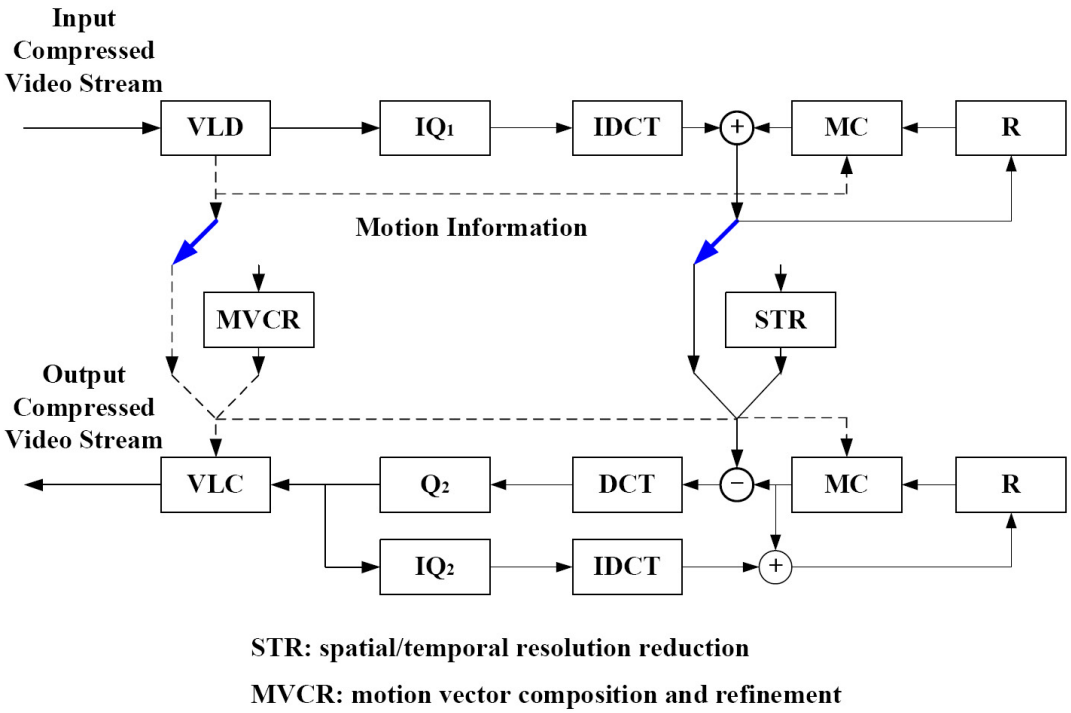


Figure 2.5: Transcoding architecture with spatial/temporal resolution conversion.

transcoding with spatial resolution conversion is to first fully decode the input compressed video stream to the pixel domain, then down-sample the decoded video frames in the pixel domain, and finally fully re-encode the down-scaled video frames to get the target compressed video stream with lower spatial resolution. The architecture is shown in Figure 2.5.

The best performance can be achieved by calculating new motion vectors and mode decisions in the re-encoding procedure for every macroblock of the down-scaled video frames. However, significant complexity saving can be achieved, while still maintaining acceptable video quality, by reusing motion information contained in the original input compressed video stream [16].

Here are some research problems which are involved in video transcoding with spatial resolution conversion.

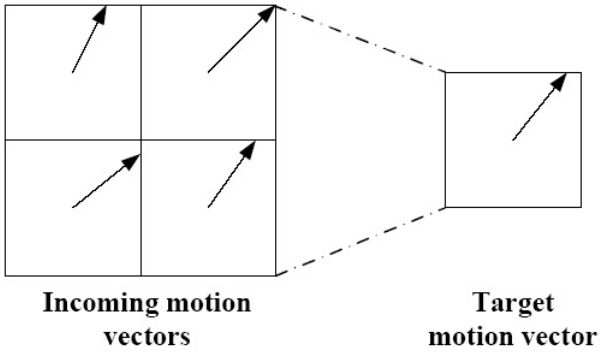


Figure 2.6: Motion vector correlation.

- ***Down-sampling:*** Down-sampling in the pixel domain consists of two steps, i.e., low-pass filtering and interpolation.

    1. ***Filtering:*** Theoretically, low-pass filtering for down-sampling is justified by the NyquistShannon sampling theorem for anti-aliasing.
    2. ***Interpolation:*** After low-pass filtering, image down-sampling in the spatial domain becomes a problem of estimating sample values at certain points, for which the interpolation theory has been established. Practically, there have been a wide range of interpolation methods with various complexity and quality being established in the literature, from the nearest neighbor interpolation, to the NyquistShannon interpolation. The NyquistShannon interpolation uses a sinc function, which achieves the theoretically optimal performance. The nearest neighbor interpolation, on the other hand, is the simplest interpolation method, which uses a square function.
    3. ***Pixel Averaging:*** Pixel averaging [11] is also a common technique used to reduce spatial resolution of video frames. In this technique, every $m \times m$ pixels are represented by a single pixel of their average value. This approach is the simplest method but the reconstructed pictures may become blurred.
    4. ***Discarding High Order DCT Coefficients:*** Discarding high order DCT coefficients is another technique to reduce the spatial resolution of video frames

[10], [11]. Discarding high order DCT coefficients method produces better image quality over the filtering and pixel averaging methods, but for large bit rate reduction greater than 25%, this method produces poor quality pictures [10].

- ***Motion Vector Composition:*** Motion vector composition is a challenging task in video transcoding with spatial resolution conversion. That is to compose new motion vectors for down-scaled video frames by using multiple decoded motion vectors when spatial resolution is reduced by a factor in each dimension. A variety of methods on the motion vector re-estimation for spatial resolution conversion consider the simple case of 2:1 down-sampling. Figure 2.6 shows the case for the motion vectors mapping with a down-sampling ratio of 2:1. The input macroblocks have four motion vectors and the target output macroblock has a single motion vector. Here are some common motion vector re-estimation methods in the literature for transcoding compressed video data from its original high resolution to a desired spatial resolution supported by the displaying device.

  1. ***Random:*** This method is to randomly choose one of the incoming decoded motion vectors [16]. The random method is fast but inefficient.
  2. ***Mean:*** The mean method takes the average of the incoming decoded motion vectors [11] to compose the corresponding new motion vector. This technique may yield poor results if the magnitude of one of the input motion vectors is significantly larger than the rest.
  3. ***Weighted Average:*** This approach takes the weighted average of the incoming decoded motion vectors [17], in which each decoded motion vector is weighted by the spatial activity of the corresponding residual macroblock. This method may bias the motion vector when original motion vectors are aimed in various directions.
  4. ***Median:*** The motion vector situated in the middle of the rest of the motion vectors is extracted by computing the Euclidean distances between each motion vector [11]. This method yields good performance, but determining the median motion vector requires substantial computation.

In all of those above methods, the magnitude of the new motion vector is scaled down by a factor in order to reflect the spatial resolution reduction.

- ***Mode Decision:*** How to decide the macroblock type for the down-scaled video is also a problem when performing the spatial resolution conversion. Mohan [3] adopted the following method to determine the macroblock type for down-scaled video.

    1. If there exists at least one intra type among the four original macroblocks, then set the type of corresponding macroblock in the down-scaled video frames as intra.
    2. Pass as inter type if there is no intra macroblock and at least one inter macroblock among the four original macroblocks.
    3. Pass as skip if all the four original macroblocks are of the skip type.
    4. Re-evaluate the macroblock type for each macroblock in the down-scaled video frames in the re-encoding procedure.

Our work is mainly focused on the motion vector composition method for H.264-based video transcoding with spatial resolution conversion. The above motion vector re-estimation methods will be introduced in section 2.4 in details.

2.3.3 Temporal Resolution Conversion

Applications of the temporal resolution conversion are required when the network terminal devices can only support the video sequence with a lower frame rate. The cascaded pixel domain architecture used to implement the video transcoding with temporal resolution conversion is to first fully decode the input compressed video stream to the pixel domain, then fully re-encode the decoded video frames to get the target compressed video stream with lower temporal resolution by dropping some frames in the re-encoding procedure. The architecture is shown in Figure 2.5.

A main research problem which is involved in video transcoding with temporal resolution conversion is the motion vector composition. With dropped frames, the incoming motion vectors become not valid because they point to the frames that have been dropped

in the transcoded video stream. Thus, deriving a new set of motion vectors has to be taken into account. Here are some common motion vector composition methods in the literature for transcoding compressed video data from its original high temporal resolution to a desired temporal resolution supported by the displaying device.

1. ***Bilinear Interpolation:*** The bilinear interpolation method is proposed by Hwang [18]. This method estimates the motion vectors from the current frame to the previous non-skipped frame. By using the bilinear interpolation from the motion vectors of the four neighboring macroblocks between every adjacent frame, it comes up with an approximation of the resulting motion vector. The bilinear interpolation method can partly solve the motion vectors reusing problem, however, further adjustment of the re-estimated motion vectors has to be performed by searching within a smaller range.

2. ***Forward Dominant Vector Selection:*** This method [19] selects the dominant motion vector from the four neighboring macroblocks, in which the dominant motion vector is defined as the motion vector of a macroblock that has the largest overlapping segment with the block pointed by the incoming motion vector. After a dropped frame, the area pointed by the motion vector of the current macroblock overlaps with at most four macroblocks in the previous dropped frame. The macroblock with the largest overlapping portion is selected and its motion vector is added to the current motion vector. Then repeat the process every time when a frame is dropped.

3. ***Telescopic Vector Composition:*** This approach accumulates all the motion vectors of the corresponding macroblocks of the dropped frames. Then adds each resultant composed motion vector to its correspondence in the current frame [11].

In the above motion vector composition methods, the bilinear interpolation method requires all motion vectors of the dropped frames to be stored, thus much extra memory is needed. Also unreliable motion vector can be produced sometimes when the four motion vectors are too divergent and too large to be described by a single motion vector. The forward dominant vector selection method has less computation than the bilinear interpolation method. Another merit of the forward dominant vector selection method is that it can

be processed in the forward order and only one table is needed for all the dropped frames when multiple frames are dropped. But the drawback of this method is that when the overlapping areas among the four macroblocks are very close, the motion vector decided by the forward dominant vector selection method may not be very effective. The telescopic vector composition approach needs less computation than forward dominant vector selection, while its video quality is little lower that of forward dominant vector selection method.

2.3.4 Transcoding Between Multiple and Single Layers

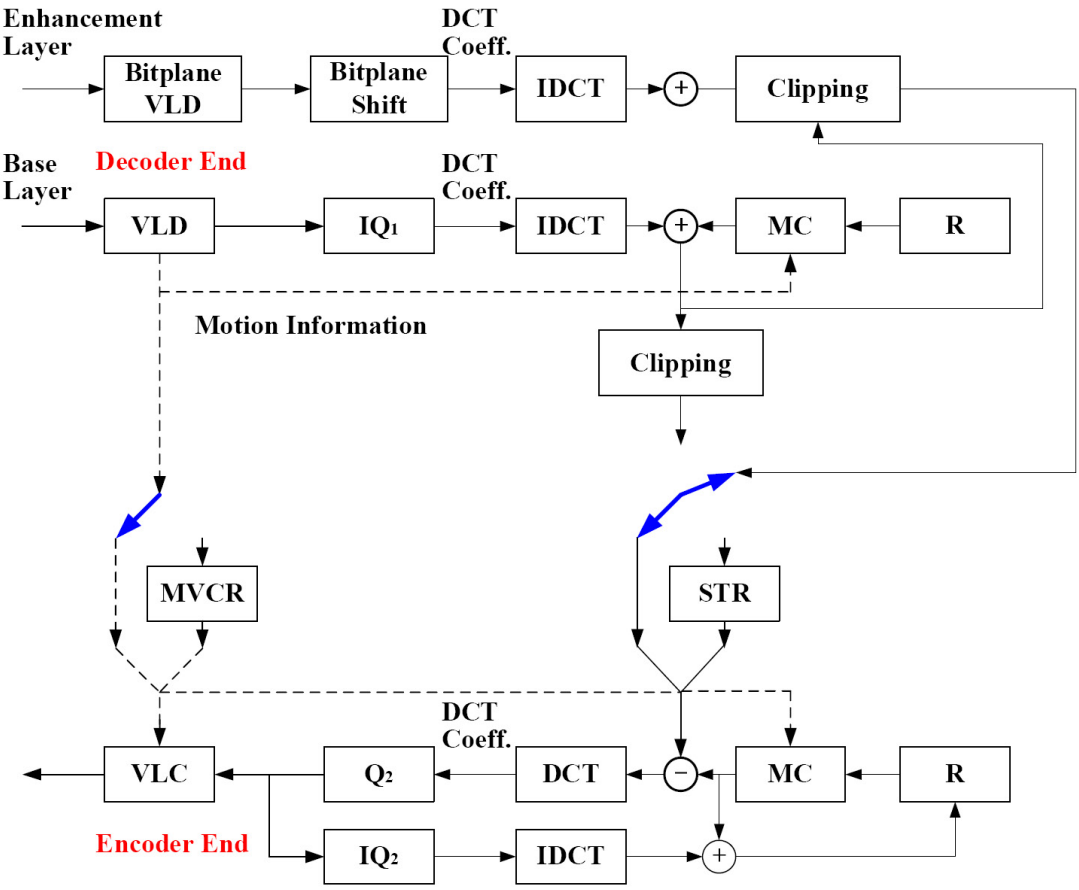
As we know, MPEG-4 coding standard provides a scalable coding scheme referred to as fine granularity scalability (FGS) [20]. In this scalable coding scheme, the video is encoded into a base layer and one or several enhancement layers. The base layer is compressed with the usual motion compensated DCT techniques. The enhancement layers are generated with a bit plane coding method by encoding the residual information between the base layer and the original input video. The key feature of the scalable video coding scheme is that it allows the enhancement layer bit stream to be truncated into any numbers of bits within each frame. Thus the quality of the reconstructed frame is proportional to the number of enhancement bits received [21]. Additional enhancement layers can add the residual information to enhance the video quality.

Figure 2.7 shows a cascaded pixel domain FGS transcoder which combine a FGS decoder and single layer encoder with motion vector reused.

2.4 Existing Motion Vector Re-Estimation Methods Overview

For transcoding from higher spatial resolution pictures to lower spatial resolution pictures, each motion vector of lower spatial resolution video frames is to be calculated from the corresponding motion vectors of higher spatial resolution video frames in order to speed up the motion estimation process in the re-encoding procedure.

The above idea of utilizing the full-scale motion vectors to speed up the MCP for the



STR: spatial/temporal resolution reduction

MVCR: motion vector composition and refinement

Figure 2.7: FGS transcoder with motion vector reused.

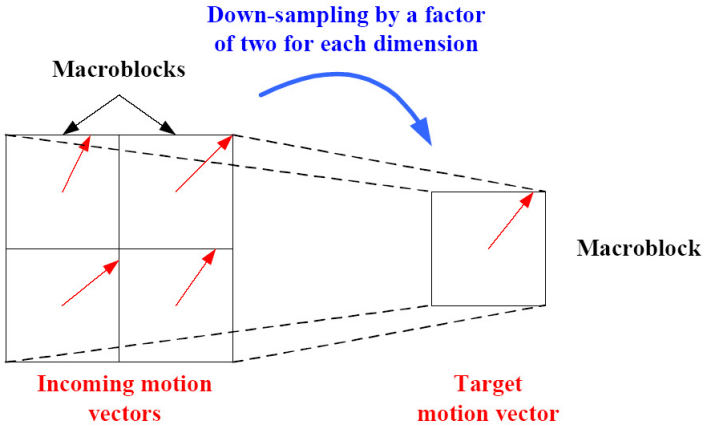


Figure 2.8: Scaled-down motion vector from four macroblocks.

re-encoding procedure has been discussed in the literature for homogeneous video transcoding, such as from MPEG-1 to MPEG-1 and from H.263 to H.263 with spatial resolution reduction [17], [22], and for heterogeneous video transcoding, such as from MPEG-2 to H.263 with spatial resolution conversion [11]. For instance, transcoding a higher spatial resolution input CIF format sequence into a lower spatial resolution QCIF format requires calculating a new motion vector from four input motion vectors. As shown in Figure 2.8, for a group of four 16×16 macroblocks of the original video frames, each macroblock has its motion vector. Various methods are proposed to compose the motion vector for the corresponding macroblock in the down-scaled video frames. These motion vector composition approaches are useful in reducing the computational complexity for motion estimation in video transcoding with spatial resolution conversion.

2.4.1 Average and Median Methods

Let $V = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3, \boldsymbol{v}_4\}$ to represent the four adjacent motion vectors associated with the four macroblocks in the original video frame and $\boldsymbol{v}$ to be the resulting motion vector in the down-scaled video frame. In [11], Shanableh et al. studied three methods for predicting motion vectors in the down-scaled scene based on motion vectors in the full-scale picture. They are referred to as median, simple average, same direction average methods.

In the first method, named the median method, a new motion vector in the down-scaled picture is estimated based on four adjacent motion vectors in the original frame. The sum of the Euclidean distance between each vector and the rest is calculated as follows:

$$d_i = \sum_{j=1, j\neq i}^{4} \|\boldsymbol{v}_i - \boldsymbol{v}_j\|$$

The median motion vector is defined as one of these vectors that has the least distance from all. This method chooses the motion vector situated in the middle of the rest of the motion vectors and then halves it to get the resulting median motion vector so that the resulting new motion vector can be associated with the macroblock in the down-sized video.

$$\boldsymbol{v} \;=\; \frac{1}{2}\,\boldsymbol{v}_k$$

such that

$$\min d_i = d_k$$

The second method, named the average method, takes the average of the four motion vectors associated with the four macroblocks in the original stream and halve it to get a new motion vector in the down-scaled scene.

$$\boldsymbol{v} = \frac{1}{2} \cdot \frac{1}{4} \sum_{i=1}^{4} \boldsymbol{v}_i$$

The third method, called the same direction average, calculates the average of those motion vectors with the same direction. Denote m as the number of vectors with the same direction. This method exploits the high motion correlation between the neighboring macroblocks.

$$\boldsymbol{v} = \frac{1}{2} \cdot \frac{1}{m} \sum_{i=1}^{m} \boldsymbol{v}_i, \quad m \leq 4.$$

2.4.2 Adaptive Motion Vector Resampling

The average method is a straightforward to compose the new motion vectors for down-scaled video frames, however, it may yield pool results if the magnitude of one of the four motion vectors is significantly larger than the rest or the four original motion vectors

are not well aligned. To deal with this problem, Shen et al. proposed a motion vector resampling method (AMVR), which takes into account the spatial activity measurement to generate a better prediction for the resulting 16×16 macroblock in the down-sampling video [17].

The motion vector resampling method by Shen et al. takes the weighted mean of original motion vectors to compose the new motion vector in a macroblock in the down-scaled video as follows

$$\boldsymbol{v} = \frac{1}{2} \frac{\sum_{i=1}^{4} A_i \boldsymbol{v}_i}{\sum_{i=1}^{4} A_i}$$

where $\boldsymbol{v}$ denotes the resulting motion vector in the down-scaled frame and $\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3, \boldsymbol{v}_4$ represent the four adjacent motion vectors associated with the four macroblocks in the original frame. A_i is defined as an activity measurement of the ith residual macroblock in the original scene, and it is computed with empirical formulas based on the DCT coefficients corresponding to related residual blocks, e.g., a simple solution is to use the number of nonzero DCT coefficients, other way is to calculate the sum of the absolute values of AC coefficients [17].

Besides estimating the new motion vectors for macroblocks in the down-scaled video sequence, the macroblock type of each macroblock in the down-scaled video sequence should be decided. As shown in Figure 2.9, if the four original macroblock are all of different types, how to decide the type of the resulting macroblock type is also a problem. Since the AMVR algorithm is used for MPEG-based and H.263-based video transcoders, Shen et al. proposed to use the rate control module of conventional encoder to determine the macroblock type for the output macroblock [23].

2.4.3 Adaptive Motion Estimation

To remove blocky artifacts of the AMVR method and to smooth the motion field, Yin et al. proposed a motion estimation scheme by taking into account motion vectors in neighboring blocks [22]. Let $\boldsymbol{v}$ denote the resulting motion vector in the down-scaled frame and $\boldsymbol{v}_i$, $i = 1, \cdots, 4$ represent motion vectors of the original four macroblocks. Let $\boldsymbol{v}_{nj}$, $j = 1, \cdots, 8$ denote the motion vectors from the eight neighboring macroblocks, as shown in Figure 2.10. The new motion vector for the macroblock in the down-scaled video

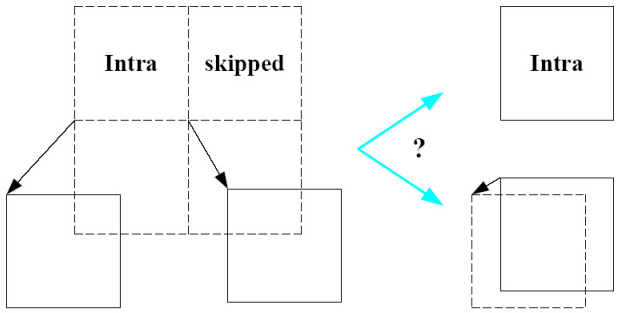


Figure 2.9: Macroblock type.

sequence is computed by

$$\boldsymbol{v} = \frac{r \cdot \sum_{i=1}^{4}(\omega_i \cdot \boldsymbol{v}_i) + (1-r) \cdot \sum_{j=1}^{8}(\omega_{nj} \cdot \boldsymbol{v}_{nj})}{2}$$

where the weighting factors ω_i, ω_{nj} and r are selected as follows [22]:

- If all of the four motion vectors $\boldsymbol{v}_i$ are equal, set $r = 1$, $\omega_i = \frac{1}{4}$.
- If the four motion vectors $\boldsymbol{v}_i$ are all different, set $r = 1$, ω_i proportional to the activity of residues, as in [17], [23].
- Otherwise, set $r = 0.8$, $\omega_i = \frac{1}{4}$ and $\omega_{nj} = \frac{1}{8}$

Taking into account neighboring blocks' motion vectors to calculate the motion vector of the corresponding new macroblock can skew the new motion vector to the moving object, thus smoothing the motion field and providing better resulting video quality [24].

For the macroblock type decision, Yin et al. proposed to intra-code the new macroblock if the four original macroblocks are all intra-coded. Otherwise to inter-code the new macroblock. For those intra-coded macroblocks which do not provide and motion information are viewed as inter-coded macroblocks with zero value motion vector [22].

Compared with the AMVR algorithm in [17], [23], this adaptive motion estimation (AME) method yielded a lower bit rate at the cost of a loss of the average PSNR. In general, it is desirable to have a more accurate model for composing new motion vectors efficiently from the original data.

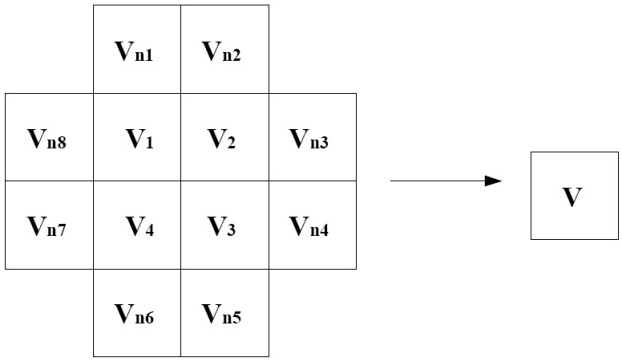


Figure 2.10: Motion vector estimation by corresponding and neighboring macroblocks.

2.4.4 Predictive Motion Estimation

By observing that in many cases the motion vector of a new macroblock in the down-scaled video found by full search is similar to one or more of the four original motion vectors reduced by half [25], Wong et al. proposed a motion estimation algorithm named predictive motion estimation (PME) to calculate the motion vector for downsized video sequence from the motion information of the original compressed video. Let $\boldsymbol{v}$ denote the resulting motion vector in the down-scaled frame and $\boldsymbol{v}_{o,i}$, $i = 1, \cdots, 4$ represent motion vectors of the original four macroblocks. This method can be illustrated as follows [25].

- Compute four candidate vectors as

$$\boldsymbol{v}_{r,i} = \frac{truncate(\boldsymbol{v}_{o,i})}{2}, \quad i = 1, \cdots, 4$$

  where "truncate" means to extract the integer part of the value of motion vector. The $\boldsymbol{v}_{r,i}$ will be in half pixel accuracy.

- Compute the mean absolute difference (MAD) in the downsized video for each of the four candidates $\{\boldsymbol{v}_{r,i},\ i = 1, \cdots, 4\ \}$. If one of the MAD value is equal to zero, Let $\boldsymbol{v}$=$\boldsymbol{v}_{r,i}$ and stop searching. Otherwise, go to next step.

- Compute a new candidate vector $\boldsymbol{v}_{r,5}$ by

$$\boldsymbol{v}_{r,5} = \frac{\sum_{i=1}^{4}[\boldsymbol{v}_{r,i} \cdot (\frac{1}{MAD_i})]}{\sum_{i=1}^{4}(\frac{1}{MAD_i})}$$

- If the $\boldsymbol{v}_{r,5}$ is not equal to one of the previous four candidate vectors, compute MAD for $\boldsymbol{v}_{r,5}$.

- The $\boldsymbol{v}_{r,i}$ $i = 1, \cdots, 5$ with the minimum MAD value is assigned as $\boldsymbol{v}$.

- For better performance, search the eight surrounding full or half pixel or $\boldsymbol{v}$. Compute the MAD for the surrounding eight full pixel or half pixels. Set the one with the minimum MAD will be chosen as the new motion vector $\boldsymbol{v}$.

2.5 Summary

This chapter first introduces video transcoding. Based on this introduction, two video transcoding architectures are then introduced, including the frequency domain transcoder and the pixel domain video transcoder. Section 2.3 discusses the functionalities of video transcoding. For different functions (including bit rate adjustment, spatial resolution conversion and temporal resolution conversion etc.) in video transcoding, the research problems and existing related work are also described.

Since our work is mainly focused on the motion vector composition problem for video transcoding with spatial resolution conversion, an overview of previous related work on motion vector composition for video transcoding from higher spatial resolution to lower spatial resolution in the literature is presented in details in section 2.4, including the average and median methods, adaptive motion vector resampling method, adaptive motion estimation algorithm and predictive motion estimation method.

The average and median methods [11] are easy to implement. The adaptive motion vector resampling method [17] proposed by Shen et al. takes the weighted mean of original motion vectors, which may produce blocking artifacts if without error compensation. To remove blocking artifacts and smooth the motion field, Yin et al. proposed the adaptive

motion estimation scheme [22] by considering motion vectors in neighboring blocks. Compared with the algorithm in [17], the method in [22] yields a lower bit rate at the cost of a loss of the average PSNR. The predictive motion estimation method [25] shows an average of 0.83 dB over AMVR [17].

The above motion vector estimation methods are easy to implement and have good performance on video stream with little motion. However, the incoming compressed video streams they considered are MPEG-1-compressed, MPEG-2-compressed, MPEG-4-compressed and H.263-compressed video stream. Compared to the incoming video stream compressed by those coding standards, an H.264-coded stream provides richer motion information in full-scale scenes, including the motion vector with quarter pixel accuracy, multiple references MCP, multiple macroblock partition prediction modes etc. These new features in H.264 lead to a fact that an H.264-coded stream provides richer motion information in full-scale scenes than streams coded in other formats such as MPEG-2, H.263. Intuitively, transcoding from H.264-coded streams should take advantage of the rich motion information. On the other side, the superior RD performance of H.264 highly relies on those new features, which result in effective MCP. Hence, for transcoding to H.264-coded streams, it is important to watch out the MCP accuracy while speeding-up the MCP procedure to tackle the complexity bottleneck. Overall, it is desirable to fully utilize the rich motion information obtained from H.264 decoding to build up an efficient and effective MCP procedure for the following re-encoding in H.264 syntaxes.

Chapter 3

H.264-Based Video Transcoding with Spatial Resolution Conversion

This chapter introduces the H.264 video coding standard and existing work on H.264-based video transcoding with spatial resolution conversion in the literature.

3.1 H.264 Video Coding Standard

3.1.1 Introduction

Early MPEG-1 video coding standard addresses coding of non-interlaced video at lower resolutions and bit rate. The MPEG-2 video coding standard, which was developed about 10 years ago, addressed coding of interlaced video at higher resolutions and bit rate enabling HDTV with commensurate video quality. The ITU-T H.261 and H.263 video coding standards were developed for telecommunication applications. The MPEG-4 coding standard is aimed for multimedia applications such as video streaming applications. The significant advance of MPEG-4 coding standard is achieved in the capability of coding of video objects.

The newest international video coding standard H.264 was jointly developed by the ITU-T VCEG and the ISO/IEC MPEG standards committees. The H.264 video coding standard was created to support the next generation of multimedia applications and is a

general purpose codec intended for applications ranging from low bit rate mobile video to high definition TV [26]. It aims to provide improved coding efficiency, improved network friendliness and simple syntax specification compared to any other previous video coding standard. It achieves clearly higher compression performance over the previous video coding standards.

3.1.2 The H.264 Codec

The H. 264 encoder includes two dataflow paths, a forward encoding path and a reconstruction path. In the following, we will describe the encoding and decoding dataflow paths of H.264 [27], [28].

- ***Encoder (Forward Path):*** Each macroblock in the current frame is encoded in intra or inter mode. For each block in the macroblock, a prediction P is formed based on reconstructed picture samples. In intra mode, P is formed from samples in the current frame that have previously encoded, decoded and reconstructed. In inter mode, P is formed by motion compensated prediction from the reference frames. The prediction P is subtracted from the current block to produce a residual block. Then transform the residual block and quantize it to get a set of quantized transform coefficients. The next step is to reorder those quantized transform coefficients and to entropy encode them. The encoded coefficients are passed to a Network Abstraction Layer (NAL) for further transmission or storage.

- ***Encoder (Reconstruction Path):*** The reconstruction process existing in the encoder to provide reference frames for further predictions. The coefficients are inverse quantized and inverse transformed to generate a residual block. Then add the prediction block P to the residual block to create a reconstructed block. A filter is applied to reduce the blocking distortion.

- ***Decoder:*** The decoder receives a compressed bitstream from the NAL and decodes the compressed data to get the quantized coefficients. These quantized coefficients are inverse quantized and inverse transformed. Using the header information decoded from the bitstream, the decoder creates a prediction block. Then the prediction block is added to a residual block to produce a reconstructed block.

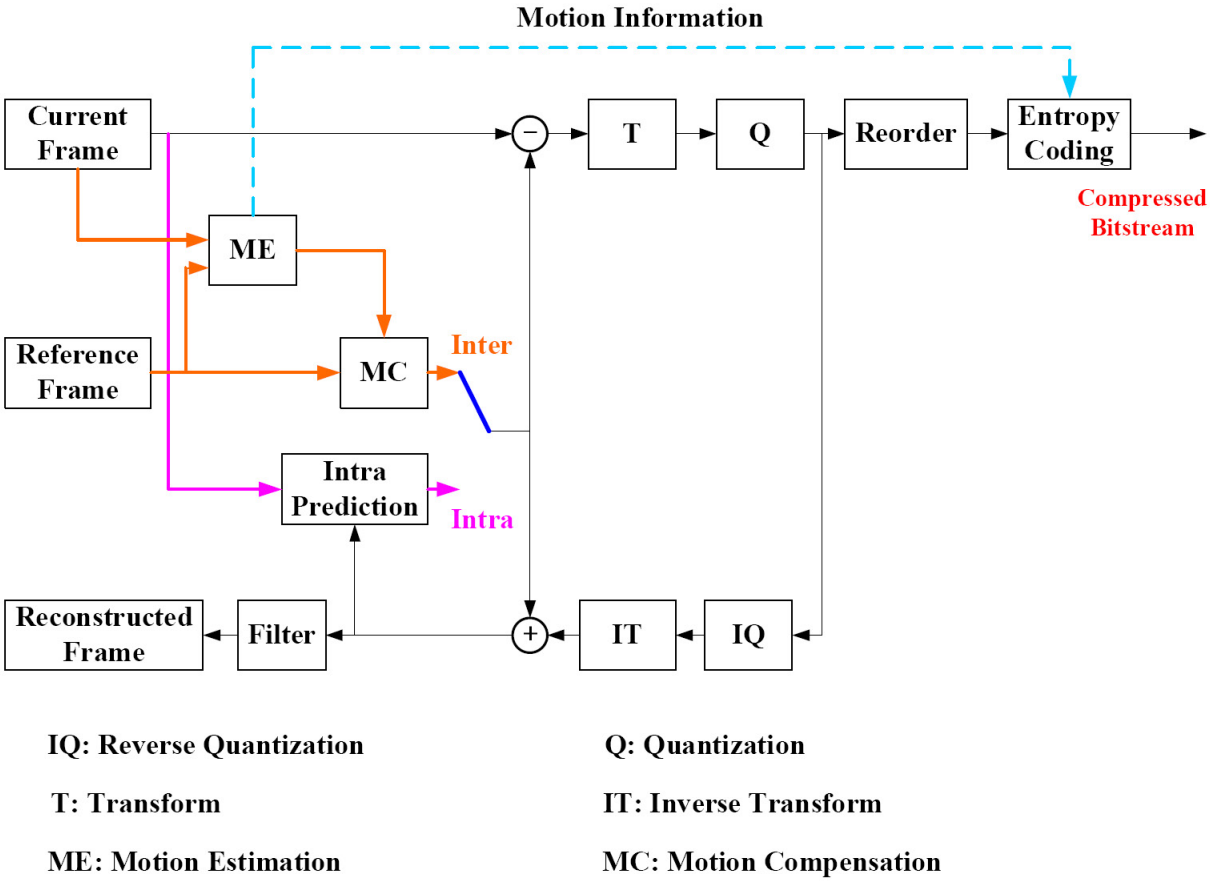


Figure 3.1: H.264 encoder.

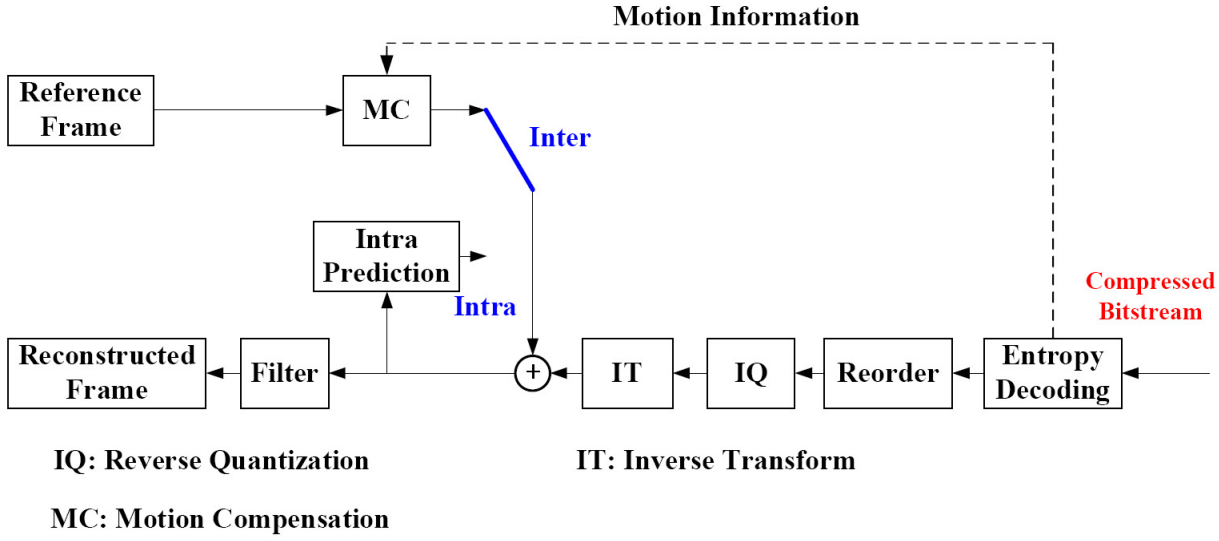


Figure 3.2: H.264 decoder.

3.1.3 The H.264 Structure

Profiles

H.264 defines three profiles, each profile supporting a particular set of coding functions and also specifying what is required of an encoder or decoder. The baseline profile supports intra coding, inter coding and context-adaptive variable-length codes (CAVLC). The main profile supports interlaced video, weighted prediction, and context-based arithmetic coding (CABAC). The extended profile supports the efficient switching between coded bitstreams and improved error resilience. Applications of the baseline profile include videotelephone, videoconferencing and wireless communications. The main profile can be applied to television broadcasting and video storage. Applications of extended profile may include multimedia streaming.

Reference Frames

For motion compensated prediction of each inter coded macroblock or macroblock partition, H.264 encoder uses one or two of a number of previously encoded pictures as reference frames. This encoder is able to search for the best match block for the current block partition from a set of reference pictures. Both the encoder and decoder maintain one or two lists of reference pictures.

Macroblock Prediction

A prediction for the current macroblock or block is created from frame samples that have already been encoded. This prediction is subtracted from the current macroblock or block and the residual is encoded and transmitted to the decoder, together with motion information (motion vector, prediction mode, etc.) used by the decoder to repeat the prediction process. The decoder performs the same prediction and adds this to the decoded residual.

Inter Prediction

In inter prediction mode, it generates a predicted version of a block, by choosing another similarly sized block from one or more previously decoded reference frames [27], [28]. In H.264, the blocks that are predicted using the motion compensation can have the following sizes: 16×16, 16×8, 8×16, 8×8, 8×4, 4×8 and 4×4.

- ***Multiple Macroblock Partition*** Each macroblock (16×16 samples) may be split up in four ways (Figure 3.3) and motion compensated either as one 16×16 macroblock partition, two 16×8 partitions, two 8×16 partitions or four 8×8 partitions. For the 8×8 mode, each of the four 8×8 submacroblocks within the macroblock may be split into one 8×8 submacroblock partition, two 8×4 submacroblock partitions, two 4×8 submacroblock partitions or four 4×4 submacroblock partitions, as shown in Figure 3.3. These partitions and submacroblock form a large number of possible combinations within each macroblock.

- ***Motion Vectors*** Each partition or submacroblock partition in an inter coded macroblock is predicted from an area of the same size in a reference frame. A motion

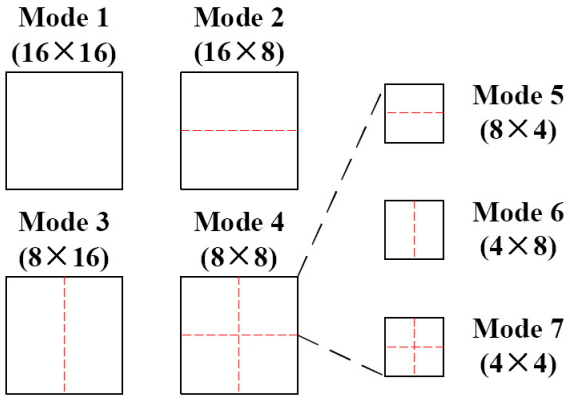


Figure 3.3: Multiple macroblock partitions for motion compensated prediction.

vector is defined as the offset between the two areas. Motion vector has quarter pixel accuracy for the luminance component and one eighth pixel accuracy for the chrominance components. Because the luminance and chrominance samples do not exist at sub pixel positions in the reference picture, it is necessary to create them by using interpolation from the neighbor encoded pixels.

Intra Prediction

For intra prediction mode, a prediction block is formed based on previously encoded and reconstructed blocks within the same frame. Then the prediction block is subtracted from the current block prior to encoding. For the luminance pixels, the prediction block is formed for each 4×4 block or for a 16×16 macroblock. There are a total of nine prediction modes for each 4×4 luminance block and four prediction modes for a 16×16 luminance block. The prediction mode that minimizes the difference between prediction block and the block to be encoded is selected for each block.

Transform

H.264 uses three integer transforms [27], [28] according to the type of residual data that is to be coded. A Hadamard transform is chosen for the 4×4 array of luminance DC

coefficients in intra macroblocks predicted in 16×16 mode. The Hadamard transform is also performed for the 2×2 array of chrominance DC coefficients. A DCT-based transform is performed for all other 4×4 blocks in the residual data.

Entropy Coding

The quantized data are encoded using either context adaptive variable length coding (CAVLC) or context-based adaptive binary arithmetic coding (CABAC) depending on the entropy encoding mode.

3.1.4 H.264 Features

It has improved performance over previous video coding standards, such as MPEG-2, H.263, and MPEG-4 part 2 due to its key features [27]:

- Variable block-size motion compensation: This standard supports more flexibility in the selection of motion compensation block size than any previous standard. The luminance component of each 16×16 macroblock can be split into one 16×16 partition, two 16×8 partitions, two 8×16 partitions and four 8×8 partitions. If necessary, the 8×8 can be further split into 8×4, 4×8 and 4×4 partitions, as shown in Figure 3.3.

- Quarter pixel accurate motion vectors: Most prior standards enable motion vectors with half pixel accuracy. The new coding standard H.264 uses quarter pixel motion vector accuracy, which enables more precise motion compensation.

- Multiple reference picture motion compensation: Unlike the previous coding standards which use only one previous picture to predict the values in an incoming picture, the H.264 coding standard exploits up to five previous coded pictures as references for inter prediction.

- In-the-loop deblocking filtering: Block-based video coding may introduce the blocking artifacts. In H.264, the deblocking filter is used to deal with the blocking artifacts, thus to improve the resulting video quality. Both subjective and objective video quality can be improved when the deblocking filter is designed well.

- 4×4 residual transform and quantization: Different from the prior video coding standards, H.264 allows the transform to operate on 4×4 blocks of residual data after MCP or intra prediction. With the integer transform, all operations can be carried out using 16-bit integer arithmetic and only a single multiply per coefficient, without any loss of accuracy.

- Context-based adaptive coding: There are two entropy coding methods applied in H.264, named CAVLC and CABAC, both of them use context-based adaptivity to improve the coding efficiency.

Other important features include weighted prediction, directional spatial prediction for intra coding, flexible macroblock ordering (FMO), arbitrary slice ordering (ASO), data partitioning, and so on.

H.264 provides mechanisms for coding video that are optimized for compression efficiency and aim to meet the needs of practical multimedia communication applications.

3.2 Existing Research Work Overview

H.264-based video transcoding, one of the homogeneous video transcoding technologies, provides conversions from H.264 to H.264. The same as other homogeneous video transcoders discussed earlier, the H.264-based video transcoder aims to reduce the bit rate, frame rate and frame resolution of the pre-encoded video stream. As shown in Figure 2.5, H.264-based video transcoding architecture can be formed by using a H.264 decoder and H.264 encoder in the decoding and encoding part, respectively.

These new features in H.264, such as motion vector with quarter pixel accuracy, multiple block size motion compensation and multiple reference frame selection, lead to a fact that an H.264-coded stream provides richer motion information in full-scale scenes than streams coded in other formats such as MPEG-2, H.263. Intuitively, transcoding from H.264-coded streams should take advantage of the rich motion information. On the other side, the superior RD performance of H.264 highly relies on those new features, which result in effective MCP. Hence, for transcoding to H.264-coded streams, it is important to watch out the MCP accuracy while speeding-up the MCP procedure to tackle the complexity

bottleneck. Overall, it is desirable to fully utilize the rich motion information obtained from H.264 decoding to build up an efficient and effective MCP procedure for the following re-encoding in H.264 syntaxes.

There are several methods presented in the literature focusing on homogeneous video transcoding from H.264 to H.264 with spatial resolution conversion [29], [30], [31], [32].

3.2.1 Mode Mapping Method

This work is focused on the mode decision part [29] of re-encoding process in the video transcoder. In the transcoding with spatial resolution reduced by a factor of 2, each macroblock in the down-scaled video frames is corresponding to four macroblocks in the original high resolution video.

They proposed two ways to determine the prediction mode for each macroblock in the down-scaled video. The first method is very simple and only uses the mode information of four macroblocks in the original video frame to estimate the mode of the corresponding macroblock in the down-scaled video frame.

- If the four pre-coded macroblocks are all intra coded, the corresponding macroblock in the down-scaled video frame is also selected to be intra coded.
- If more than three of the four pre-coded macroblocks are 16×16 inter coded, the corresponding macroblock in the down-scaled video frame is selected to be 16×16 inter coded; otherwise, it is selected to be 8×8 inter coded.

In the second method, the information of motion vectors of the four original macroblocks is also taken into account for estimating the mode of the corresponding macroblock in the downsized video frame. This method performs the same process as the first method, except that the inter coded with block size 8×8 is selected. If the inter coded with block size 8×8 is chosen, then the distances of motion vectors of the input four macroblocks are calculated. If all of the distances of motion vectors between the neighboring macroblocks among $\{MB_1, MB_2, MB_3, MB_4\}$ (Figure 3.4) are less than a threshold TH, inter 16×16 is selected. If both the distance of motion vectors between MB_1 and MB_3 and the distance of motion vectors between MB_2 and MB_4 are less than TH, inter 16×8, inter 8×16 or

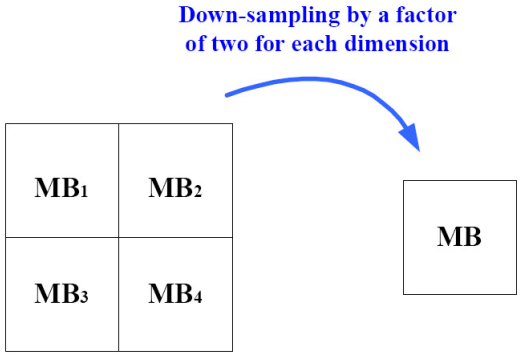


Figure 3.4: Macroblocks positions for mode mapping.

inter 8×8 will be selected as the prediction mode for the corresponding macroblock in the downsized video frame.

This method has higher computational complexity than the cascaded H.264-based transcoder using the full search scale motion search scheme but has about 3 dB PSNR losses.

3.2.2 Area Weighted Vector Median

Tan et al. has proposed to use a weighted vector median method to re-estimate the new motion vectors [30]. The motion vector of each corresponding macroblock is weighted by its portion area involved in composing the new block.

Denote $\boldsymbol{mv}_i$ as the motion vector and A_i as the area of the composing region of the ith corresponding macroblock. The area weighted vector median method is to replicate A_i times each motion vector $\boldsymbol{mv}_i$ and applying the normal median method as proposed in [11] over all the replicated motion vectors to get the median motion vector.

Assume the set of K corresponding motion vectors $V=\{\boldsymbol{mv}_1, \boldsymbol{mv}_1, \cdots, \boldsymbol{mv}_k\}$, since the distance between two same replicated motion vectors is zero, the area weighted vector median motion vector $\boldsymbol{mv}_{AWVM}$ can be calculated as

$$\boldsymbol{mv}_{AWVM} = \arg \min_{\boldsymbol{mv}_j \in V} \sum_{i=1}^{K} \parallel \boldsymbol{mv}_j - \boldsymbol{mv}_i \parallel_\gamma$$

$$\boldsymbol{mv}_{\prime} = S \times \boldsymbol{mv}_{AWVM}$$

where γ denotes the selected norm, S is the 2×2 down-sampling matrix, and $\boldsymbol{mv}_{\prime}$ is the new motion vector of macroblock in the down-scaled video frame.

Tan also proposed to intra code the new macroblock if all the corresponding partitions in the original high resolution video are intra coded. Also, if a corresponding partition is encoded with skip mode in a compressed H.264 video, then its motion vector is set to the one predicted from the motion vectors of previously coded partitions. After obtaining the composed new motion vector, a motion vector refinement by its eight surrounding pixels is performed to further refine the motion vector.

This proposed method performs about an average of 1 dB inferior to the cascaded H.264-based transcoder using the full search scale motion search scheme with a reduced computational complexity.

3.2.3 Bottom-Up Motion Vector Re-Estimation

Li et al. [31] proposed a bottom-up motion vector re-estimation, mode decision and adaptive motion search range to speed up the video transcoding with spatial resolution reduction. They adopted a 4:1 down-sampling scheme by using an average down-sampling operation.

For the motion vector re-estimation, they first derive the motion vector of each 4×4 block in the down-scaled frame by choosing the median motion vector [11] from the corresponding motion vectors in the original video. Then they combine the motion vectors to obtain the motion vectors of other modes with block size larger than 4×4. The motion vectors of blocks with size 8×4 and blocks with size 4×8 are derived as the average value of two motion vectors from the corresponding pair of 4×4 sized blocks. The motion vectors of blocks with size equal to or larger than 8×8 are set as the median value of motion vectors from a group of 4×4 blocks.

For the mode decision, they proposed to remove some modes that have lower probability of being the optimal coding mode. If the blocks in the original high resolution video are all intra coded, then the corresponding block in the downsized is also intra coded. By observations, they concluded that the 4×4 intra mode has higher probability to become

the best mode at higher bit rates and the 16×16 intra mode has higher probability to become the best mode at lower bit rates.

To determine the inter mode, they first defined the motion vector consistency as the difference between the motion vectors of four subblocks in blocks with size 8×8 and 16×16. The motion vector consistency is measured by

$$D_{i,j} = |\boldsymbol{mv}_i - \boldsymbol{mv}_j|, \quad i, j = 1, 2, 3, 4$$

Denote QP_o as the quantization step of the incoming bit stream and QP_r as the re-quantization step size in transcoding. For a 8×8 block, if the motion vector consistency $D_{i,j}$ of each 4×4 block within it is less than $Thr + \sqrt{QP_o - QP_r}$, where $Thr = Threshold\_b4$, three modes (mode 4×4, mode 4×8 and mode 8×4) will be eliminated from the RD calculation. For a macroblock, if the difference $D_{i,j}$ of each 8×8 block within it is less than $Thr + \sqrt{QP_o - QP_r}$, where $Thr = Threshold\_b8$, three modes (mode 16×8, mode 8×16 and mode 8×8) will be removed from the RD calculation. They set the $Threshold\_b8$ as 0 in their simulations. And $Threshold\_b4$ is calculated by an empirical formula based on observation.

They also define a search range and search center for further refine the composed motion vectors as

$$SearchRange = 1 + |\boldsymbol{mv}_p - \boldsymbol{mv}_r|$$

where $\boldsymbol{mv}_p$ is denoted as the prediction motion vector from the three surrounding blocks and $\boldsymbol{mv}_r$ is denoted as the re-estimated motion vector. The central location of the motion search is defined as

$$SearchCenter = \frac{\boldsymbol{mv}_p + \boldsymbol{mv}_r}{2}$$

The searching points used in their simulation for motion vector refinement are 49.

It turns out that, their method has an average of 8 times faster than the cascaded H.264-based transcoder using the full search scale motion search scheme with PSNR degradation about 0.2 to 2 dB.

3.2.4 A Fast Transcoding with Rate-Distortion Optimal Mode Decision

This method is focused on the mode decision and motion selection problems [32] for spatial resolution conversion with the down-sampling ratio 2:1.

This method is first to derive the partition mode of each 8×8 block in the downsized video by zooming out the partition mode of the corresponding macroblock in the original high resolution video. Also the motion vectors of the down-scaled video are generated by scaling down to the half value of the motion vectors of the corresponding block in the original video. A new mode decision method is proposed, which is free from computation of interpolation and SAD/SSD computation. Based on the down-scaled mode and motion vectors, the submacroblock mode decision is first performed by minimizing the proposed mode decision formula.

Along with the mode decision, the motion selection is also performed by minimizing the proposed mode decision formula. After submacroblock mode decision, the rest of modes including 8×16, 16×8 and 16×16, will be checked by the same proposed mode decision formula to determine the final partition mode of the macroblock in down-scaled video. Then the motion vector refinement process is processed to further refine the motion vectors obtained by the proposed mode decision method. Compared to the full search scale motion search scheme, this method has about 1 dB PSNR losses and the transcoding process is up to 15 times faster.

3.3 Summary

This chapter first introduces the H.264 video coding standard, including its codec, structure and main features. Based on the new features of H.264 coding standard, the H.264-based video transcoding is brought into highlight. Since our work is mainly focused on motion re-estimation problems for H.264-based video transcoding with spatial resolution conversion, the existing research works related to H.264-based video transcoding with spatial resolution conversion are illustrated in details in section 3.2. Also, the experimental results are mentioned for each work.

Among these existing work, the mode mapping method [29] has higher computational complexity than the cascaded H.264-based transcoder using the full search scale motion search scheme, but has about 3 dB PSNR losses. The area weighted vector median method [30] performs about an average of 1 dB inferior to the cascaded H.264-based transcoder using the full search scale motion search scheme with a reduced computational complexity. The bottom-up motion vector re-estimation method [31] has an average of 8 times faster than the cascaded H.264-based transcoder using the full search scale motion search scheme with PSNR degradation about 0.2 to 2 dB. Compared to the full search scale motion search scheme, the RD optimal mode decision method [32] has about 1 dB PSNR losses and the transcoding process is up to 15 times faster.

Chapter 4

An Efficient Motion Estimation Algorithm

This chapter is mainly focused on our proposed motion vector re-estimation algorithm. In the following, the problem of H.264-based video transcoding with spatial resolution conversion is described. Specifically, the coding features of H.264 are analyzed, bringing two main problems for H.264-based transcoding into the highlight. Then, the proposed motion re-estimation method is presented.

4.1 Problem Formulation

Figure 4.1 shows a typical homogeneous video transcoding system with spatial resolution conversion as a concatenation of decoding, down-sampling and re-encoding. In case of homogeneous transcoding from H.264 to H.264 with spatial resolution conversion, the motion information obtained from decoding original frames and the motion information required to effectively compress down-scaled frames are highly correlated. As discussed before, the bottleneck in the transcoding procedure is the MCP part. Hence, it is desired to utilize the motion information from the original compressed video to speed up the MCP part in the re-encoding procedure.

As the newest video coding standard, H.264 provides a whole set of new coding features, such as several prediction modes corresponding to various partitions from 16×16 to 4×4,

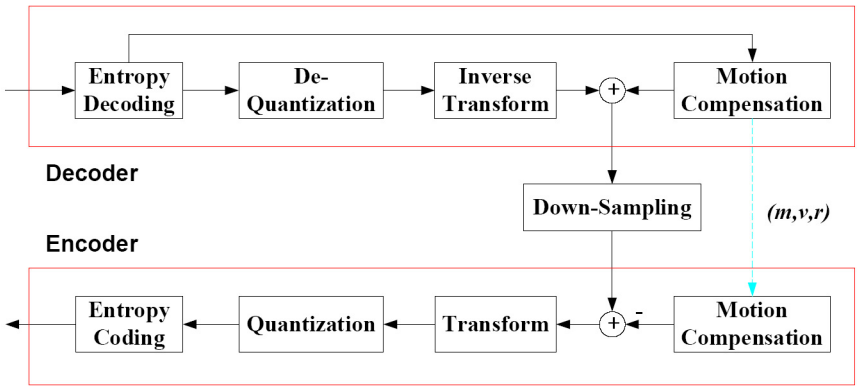


Figure 4.1: Block diagram of video transcoding with spatial resolution conversion.

multiple reference frames, quarter pixel accuracy, etc. Specifically, the MCP part in a standard H.264 coding procedure involves macroblock prediction mode selection, reference frame decision, and motion vector search. In our work, we focus on reference frame decision and motion vector search, while for prediction mode selection the standard RD optimization method is used.

Figure 4.2 illustrates the correlation between the motion information obtained from decoding original frames and the RD optimal motion information for compressing down-scaled frames. Specifically, each block in a down-scaled frame corresponds to a region in the original frame in the sense that the block is the down-scaled version of the region. Hereafter, we denote the block in the down-scaled frame as B, and the corresponding region in the original frame as R. As shown in Figure 4.2, the region R may contain some partial blocks. For simplicity, we prefer to extract and utilize the motion information for each pixel in R, which is also helpful for establishing an identical model for each specific block as discussed later in Section 4.2.

Now consider the mapping from a block B to a region R as shown in Figure 4.2. Note that the block partition in H.264 is determined by the prediction mode, which has 7 options for MCP in H.264. There are 7 types of mapping, i.e., from B_i to a region R_i with $i = 1, \cdots, 7$ corresponding to the 7 prediction modes. Denote the motion information obtained by decoding R_i from the original compressed data as $\boldsymbol{r}_i = (r_1, r_2, \cdots, r_{M_i})$ and

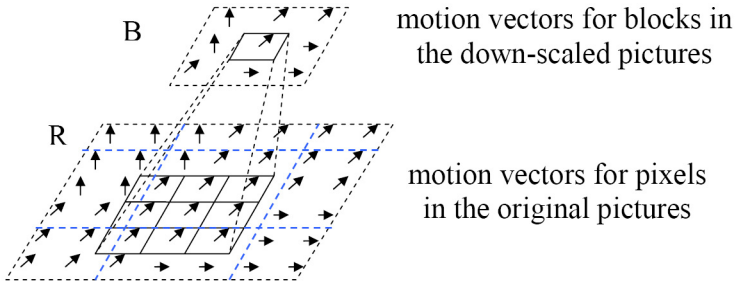


Figure 4.2: Motion vector mapping.

$\boldsymbol{v}_i = (\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_{M_i})$. M_i denotes the number of pixels in the region of R_i. Assume the RD optimal reference frame and motion vector for coding the block B_i are t_i and $\boldsymbol{y}_i$ respectively. The two main problems in our study on homogeneous video transcoding from H.264 to H.264 with spatial resolution conversion are as follows:

- Reference frame selection: Given the reference frame indexes $\boldsymbol{r}_i = (r_1, r_2, \cdots, r_{M_i})$, how to determine an RD optimal reference frame t_i.
- Motion vector search: Given $\boldsymbol{v}_i$ as motion vectors for R_i, how to find $\boldsymbol{y}_i$, which provides the RD optimal performance for coding B_i.

4.2 Problem Solution

4.2.1 Determining Reference Frame

The solution to the problem of reference frame selection is related to the total number of reference frames, which is selected to be equal to 5 in our work. Assume the set of reference frame indexes is $\{q_1, q_2, \cdots, q_5\}$. Correspondingly, define 5 counters $\{c_j(\cdot) : j = 1, \cdots, 5\}$. Given the reference frame indexes $\boldsymbol{r}_i = (r_1, r_2, \cdots, r_{M_i})$ from R_i, the reference frame index t_i is determined as follows

$$t_i = q_k \qquad \text{where} \qquad k = \arg \max_{j \in \{1, \cdots, 5\}} c_j(\boldsymbol{r}_i).$$

Essentially, this is an application of the majority rule for selecting an index which has been used most frequently.

4.2.2 Searching for Motion Vectors

While the motion vector search in a standard H.264 coding procedure involves in a time-consuming brute-force search within a large area, we propose an efficient motion search method for H.264-based transcoding, which consists of two steps. First, motion vectors obtained from decoding the original compressed data are used to compute an estimation. Second, a small surrounding area is searched to refine the estimated motion vector. In general, a precise estimation will result in a very small area for refinement, leading to an efficient motion vector search.

The proposed motion vector estimate method is developed based on a linear regression model. Consider 7 prediction modes. There is one linear regression model established corresponding to each prediction mode, i.e., $\boldsymbol{y}_i = f_i(\boldsymbol{v}_i)$ with $i = 1, \cdots, 7$. Specifically, the 7 linear models are defined as follows,

$$\boldsymbol{y}_i = \boldsymbol{\alpha}_0 + \sum_{m=1}^{M_i} \boldsymbol{\alpha}_m \odot \boldsymbol{x}_m, \qquad i = 1, \cdots, 7. \tag{4.1}$$

where i corresponds to a prediction mode, $\boldsymbol{\alpha}_0$, $\boldsymbol{\alpha}_1$, $\cdots$, $\boldsymbol{\alpha}_{M_i}$ are unknown parameters of the linear model, and $\odot$ stands for element-wise multiplication between two vectors.

Now for each linear model, it is off-line trained with a training set. For the ith model, the training set is generated by the following procedure:

1. Choose and decode several compressed video sequences. Then down-sample the decoded video data.

2. Encode the down-sampled video data with a standard H.264 encoder.

3. Check each block in the down-sampled video. Collect all blocks which are coded in the ith prediction mode at Step 2. Denote the number of collected blocks as N_i. Record their motion vectors $\{\boldsymbol{y}_n, n = 1, \cdots, N_i\}$.

4. For each block collected at Step 3, find the corresponding region in the original frame, and record corresponding motion vectors $\{\boldsymbol{v}_n, n = 1, \cdots, N_i\}$.

5. The training set for the ith model is formed as $S_i = \{(\boldsymbol{v}_n, \boldsymbol{y}_n), n = 1, \cdots, N_i\}$

Given a training set S_i, the ith linear model is trained using a multiple linear regression packaged program, which minimizes the sum of squares of deviations [33]. Consider the linear model in (4.1). Motion vectors are 2-dimensional and the two dimensions are independent, e.g., $\boldsymbol{y} = (y_1, y_2)$, $\boldsymbol{x}_m = (x_{1m}, x_{2m})$, $\boldsymbol{\alpha}_m = (\alpha_{1m}, \alpha_{2m})$. Therefore, the linear model in (4.1) contains two linear functions, each of which gives a weighted summation of variables for one dimension. For example,

$$y_{1i} = \alpha_{10} + \sum_{m=1}^{M_i} \alpha_{1m} x_{1m},$$

For simplicity to describe the training algorithm, hereafter we omit the subscripts for dimensions and the subscript of i.

$$y = \alpha_0 + \sum_{m=1}^{M} \alpha_m x_m, \tag{4.2}$$

Similarly, we can rewrite the training set by omitting the subscripts for dimensions and the subscript of i, and have a training set $S = \{(x_{1n}, x_{2n}, \cdots x_{Mn}, y_n), n = 1, \cdots, N\}$. Then, based on a data set of S, the model parameters for (4.2) are computed by

$$\boldsymbol{\alpha} = (\mathbf{X}\mathbf{X}^{'})^{-1}(\mathbf{X}\mathbf{Y}) \tag{4.3}$$

where

$$\mathbf{X} = \begin{pmatrix} 1 & 1 & \dots & 1 \\ x_{11} & x_{12} & \dots & x_{1N} \\ \vdots & \vdots & \vdots & \vdots \\ x_{M1} & x_{M2} & \dots & x_{MN} \end{pmatrix},$$

$\boldsymbol{\alpha}$=$(\alpha_0, \cdots, \alpha_M)^{'}$, and $\mathbf{Y}$=$(y_1, \cdots, y_N)^{'}$.

After the off-line training, the model parameters are stored in the transcoder. Finally, for a given block B to be coded with the ith mode, the motion vector search is performed as follow:

- Extract motion vectors for each pixel of the corresponding region R in the full-scale frame. Apply the ith linear model to compute an estimation of the motion vector.
- Refine the estimated motion vector by searching its surrounding 8 positions with quarter pixel accuracy.

4.3 Summary

This chapter first formulates motion re-estimation problem for H.264-based video transcoding with spatial resolution conversion. Then, a practical solution for efficiently determining a reference frame is proposed to take advantage of the new feature of multiple references in H.264. Also, a motion vector estimation algorithm based on a multiple linear regression model is proposed to utilize the motion information in the original scenes for efficiently predicting motion vectors in the down-scaled scene. The proposed motion re-estimation method has been implemented based on the H.264 reference software JM86. Experimental results are shown in next chapter.

Chapter 5

Experimental Results

This chapter presents the experimental results and analysis for the proposed motion vector re-estimation algorithm for H.264-based video transcoding with arbitrary spatial resolution conversion.

5.1 System Setup and Test Condition

Experiments have been conducted to evaluate the proposed approach for transcoding various H.264-compressed video sequences from higher spatial resolution to lower spatial resolution with different ratios. Seven typical CIF sequences with a resolution of 352×288 are selected for testing. We have tested three different down-sampling ratios, i.e., 2:1, 3:1, and 3:2. The frame pattern is the same for both the source compressed sequences and the transcoded sequences, i.e., "IPPP...", with only the first frame being intra-coded.

In this thesis, we implement and compare seven different transcoding methods. The first one, by the name "BestRD", uses a standard H.264 encoder with its RD optimization option enabled to re-encode the down-sampled video. It shows the best RD performance but with a high complexity. The second method uses the standard H.264 encoder but with its RD optimization option disabled. Disabling this option leads to a slightly compromised RD performance, yet a significantly reduced complexity. Thus, it is referred to as the "Benchmark" method for transcoding in this thesis. The third transcoder adopts the basic idea for motion estimation proposed in [11], which computes the mean. Hence, it is

named “Mean” in this thesis. The other four methods are all based on our proposed MCP algorithm, but with different settings for the RD optimal prediction model selection and for the refinement. Specifically, the fourth transcoder, by a name of “RGFR”, uses our proposed MCP algorithm with RD optimization over all possible Inter prediction modes, which are mode 1, 2, 3, and mode 8 with submode options of 4, 5, 6 and 7. The fifth transcoder, named “RGF”, is similar with the fourth one, except that the refinement step as discussed in the above section is omitted. The sixth transcoder, called “RGHR”, uses the proposed method for MCP with RD optimization over some Inter prediction modes, excluding the submode of 5, 6 and 7. The reason for testing this setting is because by observation submodes of 5, 6, and 7 are very rare to win the RD optimization for coding the down-scaled sequences. However, the complexity for evaluating the RD cost corresponding to these submodes is as high as that for other modes. Eliminating these submodes from the RD optimization procedure is expected to further accelerate the transcoding without compromising the RD performance. The seventh transcoder “RGH” is similar with the sixth one, but without the refinement step.

All the above transcoders employ the same down-sampling algorithms in the spatial domain. We use a concatenation of low-pass filtering and bilinear interpolation. Specifically, a Gaussian low-pass filter is used for anti-aliasing. As mentioned in the above, the complexity for down-sampling is very small compared to that for re-encoding. Simulations show that down-sampling contributes to 1% of the overall complexity for transcoding in the benchmark method.

5.2 Experimental Results

Tables 5.1, 5.3 and 5.5 show the RD performance and the complexity in terms of execution time for transcoding the testing sequences by seven transcoders with three different down-sampling ratios of 3:2, 2:1 and 3:1, respectively. For a clearer comparison, we choose the second transcoder as a benchmark, over which the relative gain of six other transcoders is computed and showed in Tables 5.2, 5.4 and 5.6. The average bit rate increase in Tables

5.2, 5.4 and 5.6 is calculated by

$$\frac{R(\mathrm{A}) - R(\mathrm{B})}{R(\mathrm{B})} \times 100\%$$

where $R(\mathrm{A})$ denotes the bit rate for one of the six transcoders, as "RGFR", "RGF", "RGHR","RGH" or "Mean". $R(\mathrm{A})$ represents the bit rate of the "Benchmark" method.

Tables 5.2, 5.4 and 5.6 show that the transcoders using the proposed MCP algorithm achieve a significant reduction of the complexity without compromising much the RD performance when compared with the benchmark method. Basically, the RD performance of the benchmark method is very close to the best achievable, which is obtained using the H.264 encoder for the re-encoding. This suggests that the proposed MCP algorithm work very well to provide accurate motion information for the re-encoding. Rather than the numbers for the bit rate and PSNR in these tables, the RD performance is also illustrated in Figures 5.1, 5.2, 5.3, 5.4, 5.5 and 5.6. Specifically, Figures 5.1 and 5.2 show the RD curves for transcoding "Akiyo" and "Paris" with a down-sampling ratio 3:2. Figures 5.3 and 5.4 show the result for a down-sampling ratio 2:1, while Figures 5.5 and 5.6 corresponds to a down-sampling ratio of 3:1. The proposed MCP algorithm yields to an RD performance very close to the optimal. The RD performance for the "Mean" method, however, has been severely compromised although its complexity is low.

As shown in Tables 5.2, 5.4 and 5.6, the transcoding performance of the four transcoders based on our proposed MCP is very similar. Overall, the "RGFR" method provides a little better RD performance with slightly higher complexity when compared with other three methods. In general, "RGFR" is the best choice when the RD performance is the highest priority, while "RGH" shall be employed if more preference is on the complexity.

Tables 5.7 shows the comparative results over "Full Search" for existing H.264-based video transcoding methods, in which the "Full Search" refers to the cascaded H.264-based transcoder using the full search scale motion search scheme. "Method 1" is the mode mapping method proposed by Zhang et al. [29]. "Method 2" refers to the area weighted vector median method proposed by Tan et al. [30]. The bottom-up motion vector re-estimation method [31] is denoted as "Method 3". And "Method 4" is the fast rate-distortion optimal mode decision proposed by Shen et al. [32]. In [29], they conclude that the "Method 1" has higher computational complexity than the "Full Search" method, but

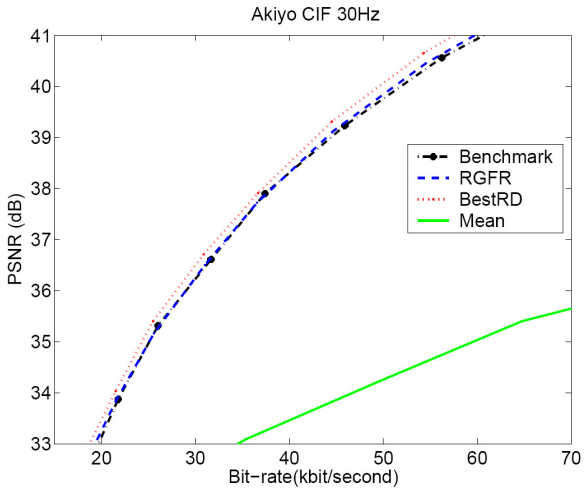


Figure 5.1: RD performance comparison of "Akiyo" with down-sampling ratio 3:2.

has about 3 dB PSNR losses. The proposed "Method 2" in [30] performs about an average of 1 dB inferior to the "Full Search" with a reduced computational complexity. "Method 3" has an average of 8 times faster than the "Full Search" with PSNR degradation about 0.2 to 2 dB. Compared to the "Full Search" scheme, "Method 4" losses up to 1 dB and the transcoding process is up to 15 times faster. Our proposed algorithm losses up to 0.3 dB while the transcoding process is up to 15 times faster compared to the "Full Search" method.

Sequence	Performance	BestRD	Benchmark	RGF	RGFR	RGH	RGHR	Mean
Akiyo	decoding time(sec)	5.21	5.03	4.89	5.04	5.09	5.07	5.06
	downsampling time(sec)	1.61	1.77	1.54	1.44	1.46	1.54	1.63
	encoding time(sec)	314.51	270.54	15.09	18.08	12.33	16.03	14.66
	total transcoding time(sec)	321.32	277.34	21.53	24.56	18.88	22.64	21.36
	psnr(dB)	40.65	40.56	40.44	40.45	40.41	40.41	37.82
	bitrate(kbps)	54.31	56.22	54.87	54.54	54.84	54.79	123.38
Bridge	decoding time(sec)	4.78	4.89	4.83	4.75	4.82	4.88	4.87
	downsampling time(sec)	1.51	1.46	1.47	1.41	1.36	1.49	1.35
	encoding time(sec)	306.84	269.16	14.52	17.30	11.84	15.83	13.76
	total transcoding time(sec)	313.13	275.50	20.81	23.46	18.02	22.20	19.98
	psnr(dB)	42.59	42.45	42.42	42.44	42.42	42.42	42.19
	bitrate(kbps)	17.99	18.57	17.92	17.90	17.92	17.92	17.48
Coastguard	decoding time(sec)	7.93	7.90	7.92	7.91	7.92	8.05	7.95
	downsampling time(sec)	1.49	1.60	1.61	1.44	1.46	1.41	1.37
	encoding time(sec)	354.00	299.56	20.50	24.05	16.08	20.54	17.51
	total transcoding time(sec)	363.42	309.07	30.02	33.39	25.46	29.99	26.84
	psnr(dB)	35.82	35.80	35.50	35.64	35.44	35.57	33.50
	bitrate(kbps)	273.08	284.94	336.00	303.39	345.65	310.85	1326.70
Mother & Daughter	decoding time(sec)	5.31	5.64	5.28	5.37	5.37	5.35	5.41
	downsampling time(sec)	1.67	1.99	1.60	1.58	1.50	1.65	1.39
	encoding time(sec)	316.75	279.21	15.50	18.42	12.44	16.13	14.14
	total transcoding time(sec)	323.74	286.84	22.38	25.38	19.31	23.12	20.94
	psnr(dB)	40.20	40.21	39.98	40.06	39.95	39.98	36.98
	bitrate(kbps)	67.37	70.73	74.10	71.52	74.39	72.17	175.74
News	decoding time(sec)	5.37	5.39	5.23	5.24	5.19	5.15	5.30
	downsampling time(sec)	1.52	1.57	1.56	1.52	1.38	1.39	1.45
	encoding time(sec)	319.86	276.59	16.31	19.09	13.19	16.76	16.25
	total transcoding time(sec)	326.75	283.56	23.09	25.85	19.76	23.30	22.99
	psnr(dB)	39.09	38.94	38.87	38.92	38.79	38.81	36.38
	bitrate(kbps)	108.56	114.23	126.23	122.65	128.53	125.39	302.09
Paris	decoding time(sec)	5.99	6.18	6.07	6.10	6.11	5.97	5.94
	downsampling time(sec)	1.80	1.66	1.47	1.59	1.54	1.54	1.53
	encoding time(sec)	334.23	281.47	17.87	20.86	14.03	17.56	17.95
	total transcoding time(sec)	342.02	289.31	25.41	28.54	21.68	25.07	25.42
	psnr(dB)	37.05	36.95	36.86	36.91	36.77	36.80	34.73
	bitrate(kbps)	205.46	214.32	222.98	215.70	231.55	223.47	695.79
Tempete	decoding time(sec)	7.38	7.57	7.55	7.53	7.50	7.38	7.30
	downsampling time(sec)	1.68	1.54	1.49	1.42	1.37	1.45	1.55
	encoding time(sec)	345.36	291.84	20.78	22.64	15.96	20.06	19.10
	total transcoding time(sec)	354.42	300.96	29.83	31.58	24.84	28.89	27.95
	psnr(dB)	36.09	36.15	35.80	35.86	35.72	35.80	32.61
	bitrate(kbps)	270.78	295.67	315.74	311.91	315.99	311.64	1066.94

Table 5.1: Transcoding performance of seven methods with a down-sampling ratio of 3:2.

Comparative Results	**BestRD**	**RGF**	**RGFR**	**RGH**	**RGHR**	**Mean**
Re-encoding Time for this divided by time for "Benchmark"	0.86	17	14	21	16	18
Total Transcoding Time for this divided by time for "Benchmark"	0.86	12	11	14	11	12
Average rate increase	-4%	5%	2%	7%	3%	182%
Average PSNR decrease(dB)	-0.06	0.17	0.11	0.22	0.18	2.40

Table 5.2: Comparative results over the "Benchmark" with a down-sampling ratio 3:2.

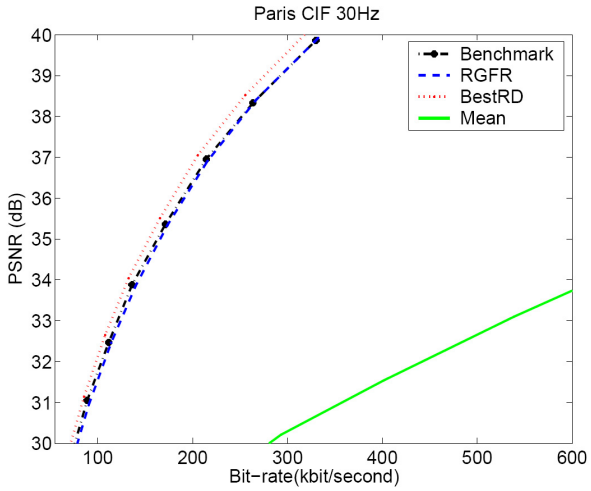


Figure 5.2: RD performance comparison of "Akiyo" with down-sampling ratio 3:2.

Sequence	Performance	BestRD	Benchmark	RGF	RGFR	RGH	RGHR	Mean
Akiyo	decoding time(sec)	5.15	5.02	5.01	5.01	5.18	4.86	5.02
	downsampling time(sec)	1.43	1.57	1.19	1.15	1.11	1.16	1.45
	encoding time(sec)	190.05	162.50	8.72	10.20	5.73	6.91	8.80
	total transcoding time(sec)	196.63	169.09	14.92	16.36	12.02	12.92	15.27
	psnr(dB)	39.56	39.52	39.40	39.41	39.35	39.35	37.07
	bitrate(kbps)	40.57	42.25	40.97	40.01	41.13	41.03	81.32
Bridge	decoding time(sec)	4.92	4.73	4.80	4.71	4.69	4.77	4.79
	downsampling time(sec)	1.43	1.41	1.13	1.16	1.15	1.20	1.16
	encoding time(sec)	183.58	161.11	8.20	9.59	5.47	6.66	7.80
	total transcoding time(sec)	189.93	167.25	14.13	15.46	11.30	12.63	13.75
	psnr(dB)	41.60	41.42	41.43	41.44	41.43	41.43	41.27
	bitrate(kbps)	13.93	14.24	13.94	13.94	13.75	13.76	13.70
Coastguard	decoding time(sec)	7.86	7.86	7.93	7.86	7.87	7.90	7.94
	downsampling time(sec)	1.38	1.60	1.16	1.19	1.38	1.06	1.13
	encoding time(sec)	208.43	176.37	11.89	13.76	8.16	9.22	10.51
	total transcoding time(sec)	217.67	185.84	20.98	22.80	17.41	18.18	19.58
	psnr(dB)	34.98	34.94	34.78	34.86	34.74	34.81	32.33
	bitrate(kbps)	185.24	192.52	216.01	205.58	218.22	207.45	881.03
Mother & Daughter	decoding time(sec)	5.34	5.47	5.35	5.36	5.37	5.29	5.37
	downsampling time(sec)	1.50	1.41	1.11	1.11	1.14	1.17	1.22
	encoding time(sec)	188.71	163.51	9.17	10.82	6.16	7.26	8.57
	total transcoding time(sec)	195.55	170.38	15.62	17.30	12.67	13.72	15.154
	psnr(dB)	39.03	39.00	38.83	38.85	38.79	38.80	35.95
	bitrate(kbps)	48.03	50.45	55.53	54.83	55.67	54.83	125.24
News	decoding time(sec)	5.34	5.32	5.19	5.17	5.19	5.23	5.33
	downsampling time(sec)	1.42	1.32	1.16	1.03	1.09	0.99	1.02
	encoding time(sec)	190.92	160.36	9.44	11.05	6.63	7.71	9.53
	total transcoding time(sec)	197.68	166.99	15.78	17.25	12.90	13.93	15.88
	psnr(dB)	38.04	37.92	37.90	37.91	37.80	37.82	35.68
	bitrate(kbps)	82.44	86.30	90.97	90.12	92.37	91.71	202.14
Paris	decoding time(sec)	5.94	6.02	5.95	5.94	5.97	5.97	5.99
	downsampling time(sec)	1.38	1.50	1.22	1.23	1.53	1.17	0.99
	encoding time(sec)	199.33	163.58	10.25	11.97	6.99	7.86	10.75
	total transcoding time(sec)	206.64	171.09	17.42	19.14	14.50	15.01	17.73
	psnr(dB)	36.16	36.05	36.02	36.05	35.88	35.92	33.75
	bitrate(kbps)	160.05	165.44	167.99	164.49	175.51	172.10	473.83
Tempete	decoding time(sec)	7.20	7.48	7.34	7.34	7.36	7.36	7.35
	downsampling time(sec)	1.55	1.42	1.20	1.16	1.14	1.25	1.21
	encoding time(sec)	207.36	169.41	12.18	13.85	8.53	9.30	11.52
	total transcoding time(sec)	216.10	178.30	20.72	22.35	17.02	17.91	20.07
	psnr(dB)	34.88	34.91	34.64	34.70	34.59	34.62	31.84
	bitrate(kbps)	211.75	230.10	241.05	235.55	242.40	239.44	677.57

Table 5.3: Transcoding performance of seven methods with a down-sampling ratio of 2:1.

Comparative Results	**BestRD**	**RGF**	**RGFR**	**RGH**	**RGHR**	**Mean**
Re-encoding Time for this divided by time for "Benchmark"	0.85	17	15	25	21	17
Total Transcoding Time for this divided by time for "Benchmark"	0.85	10	10	13	12	10
Average rate increase	-4%	4%	2%	5%	4%	158%
Average PSNR decrease(dB)	-0.07	0.10	0.06	0.17	0.14	2.20

Table 5.4: Comparative results over the "Benchmark" with a down-sampling ratio 2:1.

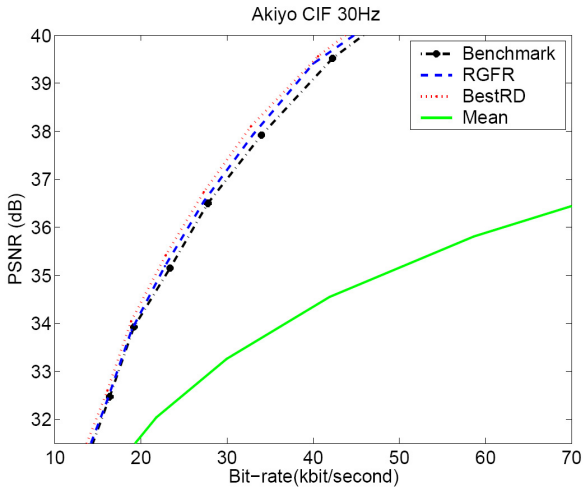


Figure 5.3: RD performance comparison of "Akiyo" with down-sampling ratio 2:1.

Sequence	Performance	BestRD	Benchmark	RGF	RGFR	RGH	RGHR	Mean
Akiyo	decoding time(sec)	5.22	5.01	4.97	5.00	5.10	5.01	4.99
	downsampling time(sec)	1.40	1.05	1.17	1.02	1.01	1.01	1.13
	encoding time(sec)	82.76	70.27	3.14	4.00	2.15	2.42	2.27
	total transcoding time(sec)	89.38	76.33	9.29	10.02	8.26	8.44	8.38
	psnr(dB)	37.88	37.81	37.75	37.76	37.70	37.71	35.94
	bitrate(kbps)	26.01	26.84	26.46	26.50	26.97	26.75	39.87
Bridge	decoding time(sec)	4.88	4.83	4.85	4.81	4.76	4.78	4.76
	downsampling time(sec)	1.11	1.11	1.16	1.02	1.06	0.94	1.04
	encoding time(sec)	79.61	70.06	3.03	3.81	1.77	2.26	2.82
	total transcoding time(sec)	85.61	75.99	9.04	9.64	7.59	7.97	8.62
	psnr(dB)	40.54	40.32	40.30	40.31	40.30	40.30	40.22
	bitrate(kbps)	9.06	9.13	8.99	8.98	8.99	8.98	8.93
Coastguard	decoding time(sec)	7.89	7.87	7.87	7.84	7.81	7.91	7.89
	downsampling time(sec)	1.11	1.11	1.08	1.03	1.13	1.22	0.95
	encoding time(sec)	89.45	75.73	4.55	5.26	2.88	3.18	3.66
	total transcoding time(sec)	98.45	84.71	13.50	14.13	11.82	12.30	12.50
	psnr(dB)	34.27	34.26	34.09	34.14	34.00	34.07	32.09
	bitrate(kbps)	91.13	95.42	104.78	100.04	108.51	102.76	321.93
Mother & Daughter	decoding time(sec)	5.43	5.30	5.36	5.30	5.42	5.28	5.33
	downsampling time(sec)	1.01	1.09	1.06	0.99	1.09	1.16	0.99
	encoding time(sec)	82.39	70.89	3.50	4.14	2.01	2.45	3.16
	total transcoding time(sec)	88.83	77.28	9.92	10.43	8.52	8.89	9.47
	psnr(dB)	37.51	37.52	37.33	37.40	37.28	37.30	35.14
	bitrate(kbps)	28.40	29.69	29.66	29.22	30.27	29.78	52.22
News	decoding time(sec)	5.15	5.24	5.19	5.06	5.27	5.24	5.28
	downsampling time(sec)	1.16	0.99	1.11	1.00	1.01	1.06	1.03
	encoding time(sec)	83.43	69.79	3.57	4.38	2.35	2.80	3.84
	total transcoding time(sec)	89.73	76.01	9.87	10.44	8.62	9.11	10.16
	psnr(dB)	36.89	36.68	36.76	36.78	36.61	36.64	34.87
	bitrate(kbps)	50.45	52.61	56.55	55.90	57.83	57.46	102.17
Paris	decoding time(sec)	5.91	6.03	6.01	5.89	5.96	6.03	5.93
	downsampling time(sec)	1.19	1.20	1.05	1.11	1.08	1.00	1.02
	encoding time(sec)	86.83	70.62	4.12	4.79	2.73	3.07	4.42
	total transcoding time(sec)	93.92	77.85	11.18	11.80	9.77	10.10	11.38
	psnr(dB)	35.16	34.99	35.03	35.04	34.86	34.89	32.52
	bitrate(kbps)	99.80	103.03	103.93	102.52	111.01	109.80	228.89
Tempete	decoding time(sec)	7.34	7.22	7.29	7.21	7.29	7.29	7.26
	downsampling time(sec)	1.23	1.26	1.17	0.92	1.07	1.03	1.06
	encoding time(sec)	90.81	73.45	4.86	5.67	3.29	3.70	4.66
	total transcoding time(sec)	99.38	81.93	13.32	13.80	11.65	12.02	12.97
	psnr(dB)	33.70	33.72	33.47	33.50	33.39	33.44	31.01
	bitrate(kbps)	134.15	145.07	144.41	142.85	144.43	143.64	294.93

Table 5.5: Transcoding performance of seven methods with a down sampling ratio of 3:1.

Comparative Results	**BestRD**	**RGF**	**RGFR**	**RGH**	**RGHR**	**Mean**
Re-encoding Time for this divided by time for "Benchmark"	0.84	19	16	30	26	21
Total Transcoding Time for this divided by time for "Benchmark"	0.85	7	7	9	8	8
Average rate increase	-4%	2%	0.5%	4%	3%	96%
Average PSNR decrease(dB)	-0.09	0.08	0.05	0.16	0.14	1.93

Table 5.6: Comparative results over the "Benchmark" with a down-sampling ratio 3:1.

Comparative Results	**Method 1** [29]	**Method 2** [30]	**Method 3** [31]	**Method 4** [32]	**Proposed**
Transcoding Time for this divided by time for "Full Search"	-	-	8	15	15
PSNR decrease(dB)	3.00	1.00	0.20-2.00	1.00	0.30

Table 5.7: Comparative results over the "Full Search" with a down-sampling ratio 2:1.

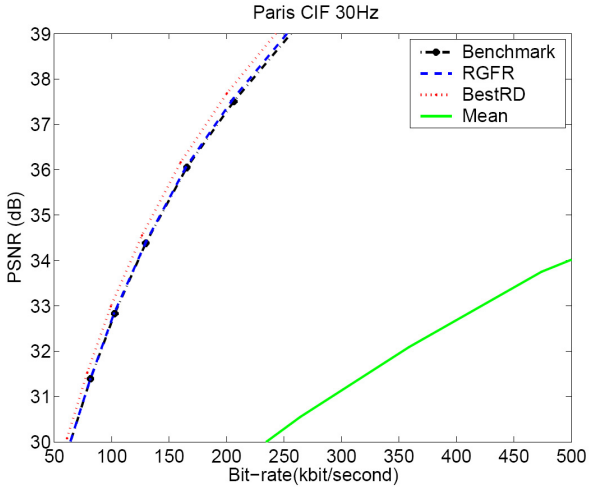


Figure 5.4: RD performance comparison of "Akiyo" with down-sampling ratio 2:1.

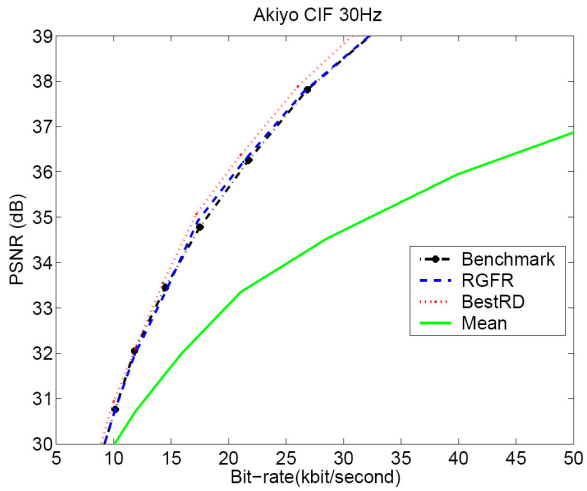


Figure 5.5: RD performance comparison of "Akiyo" with down-sampling ratio 3:1.

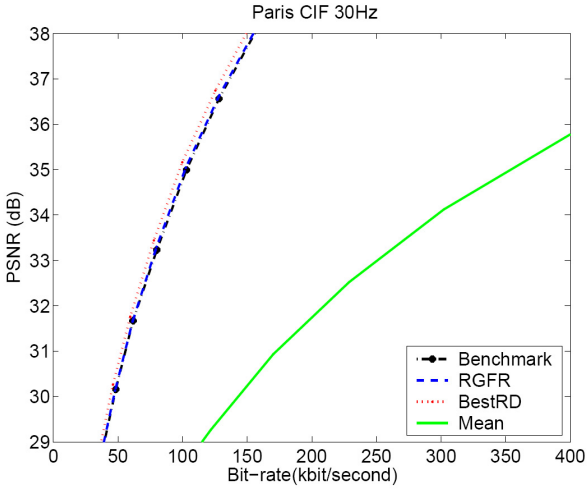


Figure 5.6: RD performance comparison of "Akiyo" with down-sampling ratio 3:1.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, we first introduce video transcoding. Two video transcoding architectures are described, including the frequency domain video transcoder and the pixel domain video transcoder. Section 2.3 discusses the functionalities of video transcoding. For different functions (including bit rate adjustment, spatial resolution conversion and temporal resolution conversion, etc.) in video transcoding, the related research problems and existing work are also described.

Since our work is mainly focused on the motion vector composition problem for video transcoding with spatial resolution conversion, the existing motion vector re-estimation methods are discussed in details in section 2.4, including the average and median methods, adaptive motion vector resampling method, adaptive motion estimation algorithm and predictive motion estimation method. However, the incoming compressed video streams they considered are MPEG-1-compressed, MPEG-2-compressed, MPEG-4-compressed and H.263-compressed video stream. Compared to the incoming video stream compressed by those coding standards, an H.264-coded stream provides richer motion information in full-scale scenes, including the motion vector with quarter pixel accuracy, multiple references MCP, multiple macroblock partition prediction modes etc. Thus in the homogeneous video transcoding from H.264 to H.264 with spatial resolution conversion, it is desirable to fully utilize the rich motion information obtained from H.264 decoding to build up an efficient

and effective MCP procedure for the re-encoding part.

Based on the above discussion, Chapter 3 introduces the H.264 video coding standard, including its codec, structure and main features. Based on the new features of H.264 coding standard, the H.264-based video transcoding is brought into highlight. Then, the existing research works related to the H.264-based video transcoding with spatial resolution conversion are illustrated in section 3.2. Also the experimental results are mentioned for each work.

Based on the prior systematic discussion, a formulation description of the motion re-estimation problem for H.264-based video transcoding with spatial resolution conversion is presented. An efficient method for transcoding one H.264-compressed video to a new H.264-compressed video with arbitrary spatial resolution conversion has been proposed. Particularly, First, a practical solution for efficiently determining a reference frame is proposed to take advantage of the new feature of multiple references in H.264. Then, a motion vector estimation algorithm based on a multiple linear regression model is proposed to utilize the motion information in the original scenes for efficiently predicting motion vectors in the down-scaled scene.

Experimental results show that the proposed video transcoding method reduces the computational complexity compared to a benchmark transcoding procedure using the standard H.264 encoding, while maintaining a comparable RD performance. Also, compared to the existing work on H.264 video transcoding with spatial resolution conversion, such as the mode mapping [29], the area weighted vector median [30], the bottom-up motion vector re-estimation method [31] and the fast RD optimal mode decision [32], our proposed method has reduced much of the computational complexity while keeping a better RD performance.

6.2 Future Work

Looking to the future of video transcoding, there are many research issues related to motion re-estimation problem of H.264-based video transcoding with spatial resolution conversion that can be further investigated.

- We have proposed an efficient motion re-estimation algorithm for transcoding from

one H.264-compressed video to a new H.264-compressed video with spatial resolution conversion based on a linear regression model to fully utilize the motion information obtained from decoding the original frames. In the future work, we should consider the real correlation between the motion vectors in the original high resolution video frames and the motion vectors in the downsized low resolution video frames from the aspect of theoretical analysis in order to obtain an optimal theoretical solution to this problem for video transcoding with spatial resolution conversion.

- We can also take into consideration the macroblock inter prediction modes (including 16×16 mode, 16×8 mode, 8×16 mode, 8×8 mode, 8×4 mode, 4×8 mode, and 4×4 mode) in the original high resolution video frames to predict the inter modes for corresponding macroblocks in the down-scaled video frames. In this way, the process time for the MCP part in re-encoding procedure can be further reduced.

Bibliography

[1] P. Yin, A. Vetro, B. Liu, and H. Sun, "Drift Compensation for Reduced Spatial Resolution Transcoding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, pp. 1009-1020, Nov. 2002.

[2] J. Xin, C.-W. Lin, and M.-T. Sun, "Digital Video Transcoding," *Proceedings of the IEEE*, vol. 93, pp. 84-97, Jan. 2005.

[3] R. Mohan, J. R. Smith, and C.-S. Li, "Adapting Multimedia Internet Content for Universal Access," *IEEE Transactions on Multimedia*, vol. 1, pp. 104-114, March 1999.

[4] *Video Coding for Low Bitrate Communication, Recommendation H.263 version 2*, ITU Telecom. Standardization Sector of ITU, Feb. 1998.

[5] *Coding of Audio-Visual Objects - Part 1: Systems, ISO/IEC 14496-1 International Standard*, MPEG98, Mar. 2000, ISO/IEC JTC1/SC29/WG11 N2501.

[6] T. Wiegand and G. Sullivan, "Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC)," JVT-G050, Pattaya, Thailand, Mar. 2003.

[7] A. Vetro, C. Christopoulos, and H. Sun, "Video Transcoding Architectures and Techniques: An Overview," *IEEE Signal Processing Magazine*, vol. 20, pp. 18-29 , March 2003.

[8] I. Ahmad, X. Wei, Y. Sun, and Y.-Q. Zhang, "Video Transcoding: An Overview of Various Techniques and Research Issues," *IEEE Transactions on Multimedia*, vol. 7, pp. 793-804, Oct. 2005.

[9] J. Youn, M.-T. Sun, and J. Xin, "Video Transcoder Architectures for Bit Rate Scaling of H.263 Bit Streams," *Proceedings of ACM on Multimedia*, pp. 243-250, Nov. 1999.

[10] 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, pp. 191-199, April 1996.

[11] T. Shanableh and M. Ghanbari, "Heterogeneous Video Transcoding to Lower Spatio-Temporal Resolutions and Different Encoding Formats," *IEEE Transactions on Multimedia*, vol 2, no 2, pp. 101-110, June 2000.

[12] A. Eleftheriadis and D. Anastassiou, "Constrained and General Dynamic Rate Shaping of Compressed Digital Video," *Proceedings of International Conference on Image Processing*, vol. 3, pp. 396-399, Oct. 1995.

[13] M.-T. Sun, T.-D. Wu, and J.-N Hwang, "Dynamic Bit Allocation in Video Combining for Multipoint Conferencing," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 45, pp. 644-648, May 1998.

[14] O. Werner, "Requantization for Transcoding of MPEG-2 Intraframes," *IEEE Transactions on Image Processing*, vol. 8, pp. 179-191, Feb. 1999.

[15] Y. Nakajima, H. Hori, and T. Kanoh, "Rate Conversion of MPEG Coded Video by Re-Quantization Process," *Proceedings of IEEE International Conference on Image Processing*, vol. 3, pp. 408-411, Oct. 1995.

[16] N. Bjork and C. Christopoulos, "Transcoder Architectures for Video Coding," *IEEE Transactions on Consumer Electronics*, vol. 44, pp. 88-98, Feb. 1998.

[17] B. Shen, I. K. Sethi, and B. Vasudev, "Adaptive Motion-Vector Resampling for Compressed Video Downscaling," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no 6, pp. 929-936, Sept. 1999.

[18] J.-N. Hwang and T.-D. Wu, "Motion Vector Re-estimation and Dynamic Frame-skipping for Video Transcoding," *32nd Asilomar Conference on Signals, Systems & Computer*, vol. 2, pp. 1606-1610, 1998.

[19] 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, March 1999.

[20] ISO/IEC 14496-2: 1999/FDAM4, ISO/IEC JTC1/SC 29/WG11 N3904, Jan. 2001.

[21] W. Li, "Overview of Fine Granularity Scalability in MPEG-4 Video Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no 3, pp. 771-775, Nov. 1994.

[22] P. Yin, M. Wu, and B. Liu, "Video Transcoding by Reducing Spatial Resolution," *IEEE international Conference on Image Processing*, vol. 1, pp. 972-975, Sept. 2000.

[23] B. Shen, I. K. Sethi, and V. Bhaskaran, "Adaptive Motion Vector Resampling for Compressed Video Down-Scaling," *Proceedings of IEEE International Conference on Image Processing*, vol. 1, pp. 771-774, Oct. 1997.

[24] M. T. Orchard and G. J. Sullivan, "Overlapped Block Motion Compensation: An Estimation-Theoretic Approach," *IEEE Transactions on Image Processing*, vol. 3, 1994.

[25] J. W. C. Wong, O. C. Au, P. H. W. Wong, and A. Tourapis, "Predictive Motion Estimation for Reduced-Resolution Video from High-Resolution Compressed Video," *Proceedings of International Conference on Image Processing*, vol. 2, pp. 461-464, Oct. 1998.

[26] H. Kalva, "The H.264 Video Coding Standard," *IEEE Transactions on Multimedia*, vol 13, pp. 86-90, Oct.-Dec. 2006.

[27] T. Wiegand and G. Sullivan, "Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC)," Mar. 2003.

[28] I. E. G. Richardson , *H.264 and MPEG-4 Video Compression : Video Coding for Next Generation Multimedia*, Chichester ; Hoboken, NJ : Wiley, 2003.

[29] P. Zhang, Y. Lu, Q. Huang, W. Gao, "Mode Mapping Method for H.264/AVC Spatial Downscaling Transcoding," *IEEE international Conference on Image Processing*, 2004.

[30] Y.-P. Tan and H. Sun, "Fast Motion Re-Estimation for Arbitrary Downsizing Video Transcoding using H.264/AVC Standard," *IEEE Transactions on Consumer Electronics*, vol. 50, pp. 887-894, Aug. 2004.

[31] C.-H. Li, C.-N. Wang, and T. Chiang, "A Fast Downsizing Video Transcoder Based on H.264/AVC Standard," *PCM*, pp. 215-223, 2004.

[32] H. Shen, X. Sun, F. Wu, H. Li and S. Li, "A Fast Downsizing Video Transcoder for H.264/AVC with Rate-Distortion Optimal Mode Decision," *Proceedings of IEEE International Conference on Multimedia & Expo*, pp. 2017-2020, July 2006.

[33] A. A. Afifi and S. P. Azen, *Satistical Analysis: A Computer Oriented Approach*, Academic Press, Inc, 1972.