Research on Transcoding of MPEG-2/H.264 Video Compression

WEI, Xianghui

Graduate School of Information, Production and Systems
Waseda University

February 2009
Abstract

Video transcoding performs one or more operations, such as bit-rate and format conversions, to transform one compressed video stream to another. It is one of the essential components for current and future multimedia systems that aim to provide universal access. Transcoding can enable multimedia devices of diverse capabilities and formats to exchange video content on heterogeneous network platforms. To suit available network bandwidth, a video transcoder can perform dynamic adjustments in the bit-rate and frame-rate of the video bit-stream without additional functional requirements in the decoder. In addition, transcoder provides functions of video format conversion to enable content exchange.

Currently, one of the biggest business drivers for transcoding is the increase in the number of service providers offering HD (High Definition) content. And the need for cost-effective yet high-quality transmission of HD content will grow significantly. One case of HD transcoding is for video storage. HD application requires huge storage space compared with other video format. Traditionally, HD content is compressed with MPEG-2 standard. If video content in MPEG-2 format can be transcoded to H.264/AVC standard, about 50% storage space can be saved.

This dissertation focuses on transcoding from MPEG-2 to H.264/AVC for HDTV application. The MPEG-2 video coding standard (also known as ITU-T H.262), which was developed about ten years ago primarily as an extension of prior MPEG-1 video capability with support of interlaced video coding, was an enabling technology for digital television systems worldwide. It is widely used for the transmission of standard definition (SD) and HDTV signals over satellite, cable, and terrestrial emission and the storage of high-quality SD video signals onto DVDs. ITU-T Recommendation H.264 and ISO/IEC MPEG-4 (Part 10) Advanced Video Coding (or
referred to in short as H.264/AVC) is the powerful and state-of-the-art video compression standard developed by the ITU-T/ISO/IEC Joint Video Team (JVT) consisting of experts from ITU-T's Video Coding Experts Group (VCEG) and ISO/IEC's Moving Picture Experts Group (MPEG). H.264/AVC represents a delicate balance between coding gain, implementation complexity, and costs based on state of VLSI (ASICs and Microprocessors) design technology. H.264/AVC design emerged with an improvement in coding efficiency typically by a factor of two over MPEG-2--the most widely used video coding standard today--while keeping the cost within the acceptable range.

Transcoding from MPEG-2 to H.264/AVC faces many challenges since the big syntax gap between them. In hardware design, one problem is how to apply data reuse methods to reduce bandwidth. Traditional search window reuse schemes rely on regular overlapping between successive search windows, which is guaranteed by full search block matching algorithms (FSBMA). In MPEG-2 to H.264/AVC transcoding, this regularity is broken up by MPEG-2 motion vector (MV), which is reused as search center in H.264/AVC encoder end. In this dissertation, two search window reuse methods, Level C and Level C+, are proposed for MPEG-2 to H.264/AVC transcoding to achieve various bandwidth levels. A hardware architecture suitable for the proposed Level C scheme is also presented. In addition, a motion vector prediction algorithm for transcoding is proposed to improve the precision of search window position.

This dissertation consists of 6 chapters which are as follows:

**Chapter 1 [Introduction]** introduces transcoding functions and existing architectures to implement transcoding systems. Transcoding functions are classified as: homogeneous, heterogenous and additional functions. The homogeneous transcoding performs conversion
between video bit-streams from the same standard; The heterogenous transcoding provides conversions between different video coding standards. Additional functions include error resilience and logo/watermarking insertion. The data reuse schemes are also introduced since it plays the most important role in bandwidth reduction for current video coding system design. Two kinds of data locality: locality in current frame and locality in reference frame. Locality in reference frame is further classified into five categories: Level A, Level B, Level C, Level C+ and Level D.

Chapter 2 [Level C Scheme for Transcoding] presents a Level C search window reuse scheme for MPEG-2 to H.264 transcoding, especially for HDTV application. The Level C scheme for transcoding is based on the fact that neighboring MPEG-2 MVs often have similar value. If the MV difference between successive MVs is less than a threshold, it is defined as smooth MV field and they are regularized to have fixed interval. That is successive two MBs have 16 pixel difference in x-coordinate. Therefore successive two MBs can share part of search window if MV field is smooth; otherwise search window should be flushed. Since most MB in sequence can be regularized based on the experimental results, a low bandwidth level can be achieved for transcoder combined with the smaller search range introduced the high accuracy by MPEG-2 MV. Experiment results show that the proposed method achieves average 93.1% search window reuse-rate in HDTV720p sequence with almost no video quality degradation. The bandwidth of the proposed scheme can be reduced to 40.6% of the transcoder without any data reuse scheme, which is almost equal to the bandwidth level of regular H.264/AVC encoder with Level C+ scheme.

Chapter 3 [Level C+ Scheme for Transcoding] proposes a search window reuse method
(Level C+) for MPEG-2 to H.264/AVC transcoding. The proposed method is designed for ultra-low bandwidth application, while the on-chip memory is not a main constraining factor. The ultra-low bandwidth ($R_\alpha < 2$) is required in some practical video transcoding system design because: 1) the strictly limited availability of bandwidth resource in these designs; 2) larger bandwidth also induces higher power dissipation, package cost and increase problems with skew; 3) the size of on-chip memory is not a constraining factor in these designs considering the continually decreased production cost of on-chip memory. Furthermore, from a systematic point of view, memory traffic introduced by other components such as variable-length-codec, DCT/IDCT, and so on must also be considered. Additionally, the motion-estimation process only uses the luminance pixel data, while the other components also use chrominance data thus increasing the importance of strong data-reuse level. By loading search window for the motion estimation unit (MEU) and applying motion vector clipping processing, each MB in MEU can utilize both horizontal and vertical search window reuse in the proposed method. An ultra-low bandwidth level ($R_\alpha < 2$) can be achieved with an acceptable cost of on-chip memory.

Chapter 4 [Hardware Architecture for Level C Scheme] proposes a low-bandwidth IME (Integer Motion Estimation) module for MPEG-2 to H.264 transcoder design. The Partial SAD architecture is adopted because partial SAD architecture has smaller gate count and suitable for medium and small resolution videos. Another advantage is that it has shorter critical path delay compared with SAD Tree because partial SAD is stored and propagated by propagation and delay register. Based on Level C bandwidth reduction method for transcoding, a modified ping-pong memory control scheme combined with Partial SAD VBSME architecture is realized. The memory control units must achieve two primary goals: 1) avoid memory input and output
confliction; 2) keep IME module to be fully utilized, which means the ME operation must has no stall. These objects are usually achieved by applying ping-pang strategy. That is when one SRAM is used, the other one is updated. The proposed architecture contains four memory banks (Mem 0-3) for storage of reference pixel. Two memories are involved to perform ME of 47×16 reference pixels. Reference pixel for ME operation of each MB is stored in three memory banks. In our design, Mem 0-2 are circularly accessed when MV field is smooth; Mem 3 is used when MV field is non-smooth. Experiment results show bandwidth of the proposed architecture is 70.6% of H.264 regular IME (Level C+ scheme, 2 MB stitched vertically), while the on-chip memory size is 11.7% of that.

**Chapter 5 [Motion Vector Prediction for Transcoding]** presents a hardware-oriented motion vector predictor (MVP) scheme for MPEG-2 to H.264/AVC transcoding. In transcoding, motion estimation is usually not performed in the transcoder because of its computational complexity. Instead, motion vectors extracted from the incoming bit-stream are reused. In many existing works, motion vector is improved by a procedure called motion vector refinement. This method is based on observation that the motion vector deviation in most macroblocks is within a small range and the position of the optimal motion vector will be near that of the incoming motion vector. But this method is difficult to be implemented in hardware. In this dissertation, we show that MVP from neighboring sub-blocks is more accurate than MPEG-2 MV as search center when MPEG-2 MV field is non-smooth. A criterion based on relative motion is proposed to evaluate smoothness of MPEG-2 MV field. And a hardware oriented MV prediction scheme is also proposed based on smoothness of MPEG-2 MV field. Experiment results show that the proposed MV prediction scheme with a relative small search range can approach the performance of full
search algorithm. Comparing with the method only utilizing MPEG-2 MV, the proposed approach can achieve significant improvement on accuracy of motion prediction, especially in sequences with fast motion and complicate background.

Chapter 6 [Conclusion] summarizes the results of my research, and indicates the future works.

Keywords
Transcoding, Bandwidth Reduction, MPEG-2, H.264/AVC, Motion Vector Prediction, Motion Estimation, Integer Motion Estimation (IME), Variable Block Size Motion Estimation (VBSME)
CONTENTS

3.3 The General Algorithm ................................................. 45
3.4 Performance Evaluation and Optimization of MEU Size .......... 47
3.5 Experimental Results .................................................. 49
  3.5.1 Experiment Result About Search Range ....................... 49
  3.5.2 Experiment Result About Video Quality ....................... 50
  3.5.3 External Bandwidth Comparison ............................... 54
3.6 Conclusion ............................................................... 55

4 Hardware Architecture for Level C Scheme ...................... 57
  4.1 Introduction ............................................................ 57
  4.2 The Top-Level Architecture ......................................... 65
    4.2.1 Performance Analysis ........................................... 65
    4.2.2 IME Architecture ............................................... 67
  4.3 Memory Input and Output Control Unit .......................... 70
  4.4 Experimental Results ............................................... 72
  4.5 Conclusion ............................................................. 74

5 A High-Precision Search Center Prediction Algorithm for Transcoding 75
  5.1 Introduction ............................................................ 75
  5.2 High-Precision Search Center Prediction Algorithm for Transcoding . 79
  5.3 Experimental Results ............................................... 83
    5.3.1 CDF Comparison ................................................ 83
    5.3.2 Image Quality Comparison ..................................... 84
  5.4 Conclusion ............................................................. 88

6 Conclusion ................................................................. 91
  6.1 Summary of results .................................................. 91
  6.2 Future work ............................................................ 93
List of Figures

1.1 Format conversion of video transcoding. ........................................... 9
1.2 The transcoding classification and functions. ..................................... 11
1.3 The applied transcoding architecture. ............................................... 13
1.4 Level A: local locality within reference block strip. Level B: local locality between adjacent reference strips. ......................................................... 23
1.5 Level C+ searching region data reuse for FSBMA, 2 MB are stitched vertically. 24

2.1 Irregular position of search window introduced by MPEG-2 MV and the regularized result. ................................................................. 30
2.2 Threshold value versus reuse-rate. ......................................................... 40
2.3 Video quality comparison on CIF. ......................................................... 41
2.4 Video quality comparison on HDTV720p. ............................................. 42

3.1 The flowchart of the proposed Level C+ method. ................................. 44
3.2 An example of MEU ($v = 3, h = 2$). .................................................. 45
3.3 An example of motion vector clipping. ............................................... 47
3.4 Bandwidth comparison under different $v$, $h$, $p_h$ and $p_v$. ..................... 48
3.5 Search range decision for CIF and HDTV720p. ...................................... 51
3.6 Video quality comparison in CIF. ......................................................... 52
3.7 Video quality comparison in HDTV720p. ............................................. 53
LIST OF FIGURES

4.1 The 1-D inter-level architecture proposed by [51]. . . . . . . . . . . . . . . . 59
4.2 The 2-D inter-level architecture proposed by [52]. . . . . . . . . . . . . . . . 60
4.3 The 2-D inter-level architecture proposed by [53]. . . . . . . . . . . . . . . . 61
4.4 The 2-D intra-level architecture proposed by [55]. . . . . . . . . . . . . . . . 62
4.5 The 2-D intra-level architecture proposed by [54]. . . . . . . . . . . . . . . . 63
4.6 The 2-D intra-level architecture proposed by [56]. . . . . . . . . . . . . . . . 64
4.7 The schema of transcoder IME architecture. . . . . . . . . . . . . . . . . . . 66
4.8 The Partial SAD architecture of VBSME. . . . . . . . . . . . . . . . . . . . . . 67
4.9 The architecture of PE array with a 1-D adder tree. . . . . . . . . . . . . . . 69
4.10 The detailed architecture of Partial SAD. . . . . . . . . . . . . . . . . . . . . . 69
4.11 An example of memory transition. . . . . . . . . . . . . . . . . . . . . . . . . 70
4.12 The finite state machine of memory control unit. . . . . . . . . . . . . . . . 70

5.1 Transcoding by the cascade of a decoder and an encoder. . . . . . . . . . . . . . 77
5.2 Transcoding by the cascade of a decoder and an encoder. . . . . . . . . . . . . . 77
5.3 Motion vector prediction for (a) 4x4 block, (b) 16x16 block in H.264 reference software. (c) Motion vector prediction for all block modes in hardware design. . . 82
5.4 $\Delta MV$ distribution under smooth MV field. . . . . . . . . . . . . . . . . . . 85
5.5 $\Delta MV$ distribution under non-smooth MV field. . . . . . . . . . . . . . . . . 85
5.6 Comparison of $\Delta MV$ distribution. . . . . . . . . . . . . . . . . . . . . . . . 86
List of Tables

1.1 Comparison of Different Data Reuse Schemes .......................... 21
1.2 Key Features of MPEG-2 and H.264/AVC Video Compression Standards. 25

2.1 Search window reuse-rate comparison ................................. 35
2.2 Bandwidth and on-chip memory comparison .......................... 38

3.1 Bandwidth and on-chip memory comparison .......................... 54

4.1 Memory update and motion estimation sequence ...................... 72
4.2 VLSI Implementation Result ............................................ 73
4.3 Hardware Cost Difference ................................................ 74

5.1 CDF under non-smooth MV field ....................................... 86
5.2 Image quality comparison ............................................... 87
5.3 Image quality comparison (cont.) ...................................... 88
LIST OF TABLES
Chapter 1

Introduction

1.1 Transcoder Architecture

Video transcoding performs one or more operations, such as bit-rate and format conversions, to transform one compressed video stream to another. Transcoding can enable multimedia devices of diverse capabilities and formats to exchange video content on heterogeneous network platforms such as the Internet. One application is delivering a high-quality multimedia source (such as a DVD or HDTV) to various receivers as PDA, Pocket PC, and fast desktop PC) on wireless and wireline networks. This application requires function of adjusting bit-rate of video bitstream. Another application is a video conferencing system on the Internet in which the participants may be using different terminals. Thus transcoder in this situation must offer two functionalities: provide video format conversion...
to enable content exchange, and perform dynamic bit rate adjustment to facilitate proper scheduling of network resources.

There are many other transcoding applications besides the universal multimedia access. In statistical multiplexing [1], multiple variable-bit-rate video streams are multiplexed together to achieve the statistical multiplexing gain. When the aggregated bit-rate exceeds the channel bandwidth, a transcoder can be used to adapt the bit-rates of the video streams to ensure that the aggregated bit-rate always satisfies the channel bandwidth constraint. A transcoder can also be used to insert new information including company logos, watermarks, as well as error-resilience features into a compressed video stream. Transcoding techniques are also shown useful for supporting VCR trick modes, i.e., fast forward, reverse play, etc., for on-demand video applications [2–4]. In addition, object-based transcoding techniques are discussed in [5] for adaptive video content delivery. A general utility-based framework is introduced in [6] to formulate some transcoding and adaptation issues as resource-constrained utility maximization problems. In [7], a utility-function prediction is performed using automatic feature extraction and regression for MPEG-4 video transcoding. Several rate-distortion models for transcoding optimization are introduced in [8] to facilitate the selection of transcoding methods under a rate constraint. Envisioning the need of transcoding, the emerging MPEG-7 standard [9], which standardizes a framework for describing audiovisual contents, has defined “transcoding hints” to facilitate the transcoding of compressed video contents [10, 11].

Currently, several video compression standards exist for different multimedia applications. Each standard may be used in a range of applications but is optimized for a limited range. H.261 [12], H.263 [13] designed by ITU (International Telecommunication Unit) are aimed for low-bit-rate video applications such as videophone and video conferencing. MPEG standards are defined by ISO (International Organization for Standardization). MPEG-2 [14] is aimed for high bit rate high quality applications such as digital TV broadcasting and DVD, and MPEG-4 [15] is aimed at multimedia applications
including streaming video applications on mobile devices. As the number of applications increases and various networks such as wireline and wireless integrate with each other, inter-compatibility between different systems and different platforms are becoming highly desirable. Transcoding is needed both within and across different standards to allow the interoperation of multimedia streams. As shown in Fig. 1.2, adjustment of coding parameters of compressed video, spatial and temporal resolution conversions, insertion of new information such as digital watermarks or company logos, and enhanced error resilience can also be done through transcoding.

![Transcoding Functions Diagram](image)

**Figure 1.2: The transcoding classification and functions.**

Scalable coding is another approach to enable bit-rate adjustment. Traditional scalability in video compression can be of three types: SNR scalability, spatial scalability, and temporal scalability. To achieve different levels of video quality, the video source is first encoded with low PSNR, low spatial resolution, or low frame-rate to form a base layer. The residual information between the base layer and the original input is then encoded to form one or more enhancement layers. Additional enhancement layers enhance the quality by adding the residual information. In many applications, the network bandwidth may fluctuate wildly with time. Therefore, it may be difficult to set the base-layer bit rate. If
CHAPTER 1. INTRODUCTION

the base-layer bit rate is set low, the base-layer video quality will be relatively low and the overall video quality degradation may be severe, since the prediction becomes less effective. On the other hand, if the base-layer bit rate is set high, the base-layer video may not get through the network completely. In general, the achievable quality of scalable coding is significantly lower than that of non-scalable coding. In addition, scalable video coding demands additional complexities at both encoders and decoders. Thus, pre-encoded video is used, scalable coding is inflexible since the number of different predefined layers is limited and the bit-rate of the target video cannot be reduced lower than the bit-rate of the base layer. Scalability alone does not solve the bit-rate adjustment problem.

The video transcoding functions are shown in Fig. 1.2. Transcoding functions are classified as: homogeneous, heterogenous and additional functions. The homogeneous transcoding performs conversion between video bitstreams from the same standard; The heterogenous transcoding provides conversions between different video coding standards, in which a syntax conversion between standards is necessary. The shared functions in transcoding include bit-rate change, spatial and temporal resolution conversion, etc. Additional functions include error resilience and logo/watermarking insertion.

Video transcoding architectures are categorized as: open-loop, closed-loop [16]. The closed-loop architecture directly cascades the decoder and encoder. The source video stream \( V_s \) is fully decoded and then re-encoded into the target video stream \( V_T \) with desirable bit-rate and format. Therefore no video quality degradation is involved. The open-loop architecture eliminates feed-back loop for compensating the drift-error in both decoder and encoder end. It aims to minimize transcoding complexity but cannot avoid drift-error.

The closed-loop architecture can be further classified as: spatial-domain, frequency-domain and hybrid-domain.

The definitions of spatial-domain and frequency-domain are shown in Fig. 1.3. Spatial-domain architecture (SDTA) can perform dynamic bit-rate adaptation via the rate-control
at the encoder side. This architecture is flexible since the decoder-loop and the encoder-loop can be totally independent of each other. This architecture is drift-free, but its computational complexity is high for real-time applications. Since a pre-encoded video stream arriving at a transcoder already carries useful information such as the picture type, motion vectors, quantization step-size, bit-allocation statistics, etc., it is possible to construct transcoders with different complexity and performance in terms of coding efficiency and video quality. Intuitively, most of the motion information and the mode decision information received in the video decoder can be reused in the video encoder without introducing significant degradation on visual quality. Thus, motion estimation, the most time-consuming operation in video encoding which accounts for 60%–70% of the encoder computation [17], is avoided. This leads to an SDTA that can reuse MVs. This architecture saves the motion estimation operation, which is the most time-consuming module. The
pre-encoded source video is decoded in the spatial-domain by performing variable length decoding (VLD), inverse quantization $Q_S^{-1}$, IDCT, and motion compensation. In the encoder, the motion compensated residue errors are encoded into frequency-domain through DCT, re-quantization $Q_T$, and variable length coding (VLC). The motion compensation operation at the encoding end is also performed in the spatial domain for the prediction operation. The MV reuse approach is useful in complexity reduction for motion estimation in video transcoding.

Exploiting the structural redundancy of the architecture in SDTA and the linearity of the DCT/IDCT, a structurally simpler but functionally equivalent frequency-domain transcoding architecture is possible [18], which can be further simplified [19–21]. In this architecture, only VLD and inverse quantization are performed to get DCT value of each block in the decoder end. At the encoder end, the motion compensated residue errors are encoded through re-quantization, and VLC. The reference frame memory in the encoder end stores the DCT values after inverse quantization, that are then fed to the frequency-domain MC module to reduce drift error. This is referred to as frequency-domain transcoding architecture (FDTA). In this architecture, motion compensation is performed in the frequency domain using a MV reusing algorithm. Detail frequency domain MC algorithm can be found in [19] and [22]. An FDTA may need less computation but suffer from the drift problem due to nonlinearity operations, which includes subpixel motion compensation, and DCT coefficients clipping during MC. FDTAs also lack flexibility and are mostly suitable for bi-rate transcoding.

Various transcoding algorithms provide tradeoff between the computational complexity and reconstructed video quality. In order to reduce the computational complexity while maintain the reconstructed video quality, ME should be omitted and DCT/IDCT should be avoided if possible. For example, the architecture in [23] uses MC for P frames only. I frames are intra coded, which need no ME and MC, and thus, IDCT/DCT for I frames can be omitted in principle. But since I frames are the anchors for subsequent P and B frames,
the IDCT at the decoder stage, inverse quantization and IDCT at the encoder stage for I frames are still needed to reconstruct the reference frames, while DCT at the encoder stage can be omitted. Since P frames are also the anchors for the following P and B frames, MC, DCT, and IDCT cannot be omitted. For B frames, which are not the reference frames for the subsequent frames, drift error generated in B frames would not propagated through the video sequence, so MC of B frames can be removed without introducing significant degradation on visual quality of reconstructed pictures. Thus, DCT/IDCT in all B frames can be omitted, and the transcoding of B frames can be directly done in the DCT domain. The transcoding delay can be further reduced without degrading the video quality in this architecture. P frames with frequent scene changes and rapid motion may contain a large number of INTRA blocks. One can further omit the IDCT/DCT and MC operation of these INTRA blocks in P frames. In other words, blocks of I and B pictures and INTRA blocks of P pictures are transcoded in frequency-domain, the spatial-domain motion compensation is done only when the block is inter block in P frames. This transcoding architecture is defined as hybrid domain transcoding architecture (HDTA).

This dissertation focus on heterogeneous transcoding from MPEG-2 to H.264/AVC for HDTV application. And only format conversion is taken into consideration. H.264 is a joint effort of MPEG and ITU with the first version finalized in May 2003 [24]. It aims to deliver a compression efficiency as twice as MPEG-2. The MPEG-2 standard [14] is widely used for the transmission of SD and HD TV signals over satellite, cable, and terrestrial emission and the storage of high-quality SD video signals onto DVDs. In H.264, video compression techniques are different from other stands that transcoding between H.264/AVC and other standards will face many difficulties. The syntax gap between MPEG-2 and H.264/AVC is listed in Table 1.2. The main difficulty is come from DCT transform and block size used in ME. The DCT in MPEG-2 is in $8 \times 8$ while that in H.264/AVC is $4 \times 4$. Thus transcoding in DCT domain will involve a procedure to transform DCT coefficients. MEPG-2 only has two block modes ($8 \times 8$, $16 \times 16$), while H.264/AVC has
seven block modes (16 × 16, 16 × 8, 8 × 16, 8 × 8, 8 × 4, 4 × 8, 4 × 4). Thus MV mapping and re-estimation with required precision is a challenge for transcoder design. Considering the big syntax gap between MPEG-2 and H.264/AVC as well as the video quality requirement in HDTV application, the closed-loop architecture with motion information reuse is chosen as the primary transcoding architecture as shown in Fig. 1.3.

1.2 Bandwidth Reduction Scheme for Video Coding System

Motion estimation has been widely employed in the H.264/AVC, MPEG-1, -2, and -4 video standards to exploit the temporal redundancies inherent among video frames. Block matching is the most popular method for ME. Pixel-by-pixel difference is central to the block matching algorithms and result in high computation complexity and huge memory bandwidth. Benefitting from the rapid progress of VLSI technology, computation complexity requirements can easily be satisfied by multiple PEs architecture, even for large frame sizes and frame rate. However, without enough data, these PEs can hardly be fully utilized and simply result in increased silicon area. The data rate is limited by available memory bandwidth. Therefore, straightforward implementation of motion estimation is an I/O-bounded problem rather than a computation-bound one. The memory bandwidth problem may be solved by careful scheduling of the data sequence and setting up appropriate on-chip memories. Meanwhile, a well-designed motion estimation architecture reduces the requirements of memory bandwidth and the I/O pin count, but still keeps high hardware efficiency.

Many algorithms and hardware have been dedicated to motion estimation. The full-search block-matching (FSBM) algorithm is one of these algorithms, which searches through every reference location to find the best match. To reduce the computational complexity
of FSBM, various fast algorithms were proposed [25–29] that searched fewer reference positions. However, these fast algorithms suffer from irregular control and lower video quality, and thus FSBM remains widespread owing to its simplicity, regularity, and superior video quality. Many hardware architectures have been proposed for implementing FSBM. These architectures use systolic array [30–33], tree structure [34], or 1-D structure [35] to solve computational problem by providing enough PEs. However, all of these architectures provide limited solutions to overcoming the memory bandwidth bottlenecks of high-quality video ME such as HDTV. Direct implementation is unrealistic without exploiting the locality or tricky designs. For example, the MPEG2 MP@ML format requires a memory bandwidth of tens of gigabytes per second, while the HDTV format with a large search range requires a terabytes per second bandwidth. This work only considers uni-directional ME, and bandwidth becomes even higher for bi-directional predictions. Redundancy relief is the solution to the huge bandwidth problem because many redundant accesses exist in the memory traffic of ME.

In [36], redundant memory access is considered as the main contributor to a large memory overhead. The excessive frame memory access result from each pixel having multiple accesses. Excessive memory access can be significantly reduced by exploiting data locality. There exist two levels of data locality: locality in current frame and locality in reference frame. Locality in reference frame can be further categorized as Level A, Level B, Level C, Level C+ and Level D.

In following discussion, search range is defined as $[-p_h, p_h]$ in horizontal direction and $[-p_v, p_v]$ in vertical direction; MB size is defined as $N \times N$; frame size is defined as $W \times H$. Accordingly, search window size is $(s_{rh} + N - 1)(s_{rv} + N - 1)$, where $s_{rh} = 2p_h$ and $s_{rv} = 2p_v$. 

17
1.2.1 Data Locality

Data locality refers to data invariability during motion estimation. In hardware architecture, it particularly refers to the data invariability in on-chip memory. It can be categorized as

Locality in Current Frame In FSBM, each MB is independent, i.e., pixels of one MB do not overlap with other ones. Consequently, the lifetime of one MB is just the time period of ME of this MB. Each MB pixel is used $sr_h \times sr_v$ times during this period, showing that pixels of current MB have good locality compared with pixels of reference frame. This characteristic allows the simple approach of keeping all $N \times N$ MB pixels locally, allowing the design to result in a $1/(sr_h \times sr_v)$ reduction in memory accesses of the current frame. Therefore, the additional on-chip memory for current MB reduces the access count of the current frame to just $W \times H$ for each frame, which is also the maximum possible saving. This idea is widely applied in many proposed ME architectures for minimizing current frame bandwidth. Consequently, the redundancy access of the current frame is as shown in the equation at the bottom of the page.

Locality in Reference frame Each search area in reference frame is a $(N + sr_h - 1) \times (N + sr_v - 1)$ rectangle centered around related current MB. Adjacent search areas thus overlap and are no longer independent. The locality of search area data has two types: local locality and global locality. Local locality covers data reuse within a single search area, regardless of overlapping among other search areas. Global locality refers to data reuse among different search areas. Five data-reuse levels are defined according to the degree of data reuse: from Level A (weakest reuse degree) to Level D (strongest reuse degree). The data-reuse level is an important factor in dealing with the memory bandwidth requirements of ME. Because stronger reuse level reduces redundant memory access, less memory bandwidth is required. The five reuse levels
1.2. BANDWIDTH REDUCTION SCHEME FOR VIDEO CODING SYSTEM

are defined below.

1. **Level A–Locality within Candidate Block Strip** Local locality can be further divided into locality within the same row and locality between adjacent rows. A row of reference blocks is called a reference block strip, as shown in upper half of Fig. 1.4. For the two consequent reference blocks, pixels are significantly overlapped. The pixel access count without any data reuse is $2 \times N \times N$ for calculating the two MVs, but only individual $N \times 1$ pixels are different. Reference block 1 and 2 in Fig. 1.4 demonstrates this situation. When dealing with reference block 2, the overlapped data of reference block 1 can be reused without accessing the frame memory again. The data-reuse approach can be extended to the entire reference block strip, i.e., pixels can be repeatedly used for adjacent MV calculations.

2. **Level B–Local Locality Between Adjacent Reference Block Strips** Vertically adjacent candidate block strips also overlap significantly, as demonstrated by reference block strips 1 and 2 in the lower half of Fig. 1.4. The size of the overlapping region of two reference block strips is $(N + sr_h - 1)(N - 1)$. The pixels in the overlapped region can be reused while processing the next reference block strip, which means exploiting the locality of adjacent reference block strips. This idea is extended to all reference block strips within the search area.

3. **Level C–Global Locality Within Search Area Strip** The global locality within the search area strip describes data reuse among different search areas corresponding to the location of their current blocks within the same search area strip. The search area strip is like an up-sampled version of the reference block strip, and is formed by entire rows of search area. The derivation resembles the scheme in Level A, and thus the formula and derivations are ignored herein.

4. **Level C+-Global Locality Between Search Window of MB Group** [37] proposed
another search window reuse scheme by exploiting global locality between MB group. That is, several successive current MBs in the vertical direction are stitched, and the searching region of these current MBs is loaded, simultaneously. Thus not only can the overlapped searching region in the horizontal direction be fully reused, but also the overlapped searching region in the vertical direction can be partially reused, as shown in Fig. 1.5.

Although the Level C+ scheme can save more external memory bandwidth with the less overhead of on-chip memory size, the Level C+ scheme with stripe scan conflicts with some coding tools such as Lagrangian mode decision and MV predictor. This is because, in general, the side information of left, top, and top-right MBs are necessary to encode current MBs. Therefore, the direct implementation of the Level C+ scheme with strip scan could lower the hardware efficiency of video coding systems with these functions and architectures.

In the previous works with raster scan, the data dependency of the left MB may not be satisfied, if there is a deep MB-pipelining architecture, where several current MBs are processed simultaneously in different functional modules. Therefore, the data dependency of the left MB is carefully designed to satisfy the side information in time in an MB-pipelining architecture. Hence, the hardware efficiency is high, and the required data buffer is small.

Compared to raster scan, satisfying the data dependency of the left MB is much easier in the stripe scan because there are some other MBs which are processed between the computations of left MB and current MB. Therefore, the side information of left MB will be generated earlier in the stripe scan than the raster scan. However, the data dependencies of the top and top-right MBs become a problem. In order to satisfy the requirements of side information in the MB-pipelining architectures, \textit{n-stitched zigzag scan} is also proposed in this work. In the \textit{n-stitched zigzag scan}, several successive MBs in a horizontal
1.2. BANDWIDTH REDUCTION SCHEME FOR VIDEO CODING SYSTEM

direction will be processed before the vertical scan starts, and \( n \) MB rows are processed alternately in the vertical scan. The former can guarantee that the data dependencies of neighboring MBs are satisfied, and the latter uses the concept of the Level C+ scheme to reduce the required memory bandwidth.

5. Level D—Global Locality Among Adjacent Search Area Strips Level D resembles Level B, except that it applies to the search area strips. Level D repeatedly reuses data already loaded from former search area strips for latter search area strips. Applying this level, one-access is achieved.

Among these methods, Level C and Level C+ can achieve a good balance between on-chip memory size and external bandwidth, the Level D locality requires relatively large on-chip memory, which makes it impractical under current memory technology.

In hardware implementation, the current MB buffer exploits locality in current frame to reduce bandwidth. The Level A and Level B locality are embedded in Level C implementation in current hardware design. This is achieved by parallelizing several PEG (Processing Element Group) and reference data scheduling.

1.2.2 Performance Evaluation

<table>
<thead>
<tr>
<th>Reuse Scheme</th>
<th>Redundancy access factor ( R_\alpha )</th>
<th>On-chip Memory Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level A</td>
<td>( \frac{sr_v \times N \times (N + sr_{h-1})}{N \times N} \approx sr_v (1 + \frac{sr_{h}}{N}) )</td>
<td>( N(N - 1) )</td>
</tr>
<tr>
<td>Level B</td>
<td>( \frac{(N + sr_{h} - 1) \times N \times (N + sr_{v} - 1)}{N \times N} \approx (1 + \frac{sr_{v}}{N})(1 + \frac{sr_{h}}{N}) )</td>
<td>( (N + sr_{h})(N - 1) )</td>
</tr>
<tr>
<td>Level C</td>
<td>( \frac{N \times (sr_{v} + N - 1)}{N \times N} \approx 1 + \frac{sr_{v}}{N} )</td>
<td>( (sr_{h} + N - 1)(sr_{v} + N - 1) )</td>
</tr>
<tr>
<td>Level C+</td>
<td>( \frac{N \times (sr_{v} + nN - 1)}{N \times N} \approx 1 + \frac{sr_{v}}{nN} )</td>
<td>( (sr_{h} + N - 1)(sr_{v} + nN - 1) )</td>
</tr>
<tr>
<td>Level D</td>
<td>( \frac{W \times H}{W \times H} = 1 )</td>
<td>( (sr_{h} + W - 1)(sr_{v} - 1) )</td>
</tr>
</tbody>
</table>
CHAPTER 1. INTRODUCTION

In the evaluation of bandwidth reduction schemes, two factors can be used to evaluate the performance of data reuse schemes: on-chip memory size for the reference frame and redundancy access factor $R_\alpha$. The on-chip memory size represents the required memory size to buffer the data of candidate blocks for data reuse. $R_\alpha$ is used to evaluate the external memory bandwidth and defined as

$$R_\alpha = \frac{\text{Total memory bandwidth for reference frame}}{\text{Minimum memory bandwidth}}$$  \hspace{1cm} (1.2.1)

In this definition, “Minimum memory bandwidth” in denominator refers to the situation that no reference data is needed to be loaded for ME. Thus only current MB data is needed to be loaded from external memory. Thus the definition of $R_\alpha$ actually evaluates how many reference pixels should be loaded for each MB pixel.

To realize data reuse, on-chip memory holding already loaded data for further access is required. The local memory required for the current frame, i.e., for storing one current block is $N \times N$. The local memory size for the previous frame equals the size of the overlapped region in each data-reuse level. Table 1.1 lists performance comparisons of the redundancy access factor $R_\alpha$ and on-chip memory between different data reuse schemes.
Figure 1.4: Level A: local locality within reference block strip. Level B: local locality between adjacent reference strips.
Figure 1.5: Level C+ searching region data reuse for FSBMA, 2 MB are stitched vertically.
1.2. BANDWIDTH REDUCTION SCHEME FOR VIDEO CODING SYSTEM

<table>
<thead>
<tr>
<th>Features</th>
<th>MPEG-2</th>
<th>H.264/AVC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profiles</td>
<td>Simple, Main, High,</td>
<td>Baseline/Main/Extended,</td>
</tr>
<tr>
<td></td>
<td>SNR/Spatially Scalable</td>
<td>HiP/Hi10P/Hi422P/Hi444PP,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High 10/4:2:2/4:4:4 Intra,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CAVLC 4:4:4 Intra, Scalable</td>
</tr>
<tr>
<td>Basic Approach</td>
<td>Block Based</td>
<td>Block Based</td>
</tr>
<tr>
<td>Processing Unit</td>
<td>Macroblock</td>
<td>Macroblock</td>
</tr>
<tr>
<td>Picture Types</td>
<td>I, P, B</td>
<td>I, P, B, SP, SI</td>
</tr>
<tr>
<td>I frames</td>
<td>More(random access)</td>
<td>Fewer(compression)</td>
</tr>
<tr>
<td>Entropy Coding</td>
<td>VLC</td>
<td>CAVLC or CABAC</td>
</tr>
<tr>
<td>ME Accuracy</td>
<td>$\frac{1}{2}$ pixel</td>
<td>$\frac{1}{4}$ pixel</td>
</tr>
<tr>
<td>Motion Vectors</td>
<td>Inside reference</td>
<td>Can point to outside</td>
</tr>
<tr>
<td></td>
<td>picture only</td>
<td>picture</td>
</tr>
<tr>
<td>Block Transform</td>
<td>$8 \times 8$ DCT</td>
<td>$4 \times 4$ DCT</td>
</tr>
<tr>
<td>Multi-Reference</td>
<td>Not Supported</td>
<td>Supported (up to 5)</td>
</tr>
<tr>
<td>Block size</td>
<td>$8 \times 8, 16 \times 16$</td>
<td>$16 \times 16, 16 \times 8, 8 \times 16$</td>
</tr>
<tr>
<td></td>
<td>$8 \times 8, 8 \times 4, 4 \times 8, 4 \times 4$</td>
<td></td>
</tr>
<tr>
<td>Intra Block Prediction</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Quantization</td>
<td>Linear or non-linear</td>
<td>Exponential</td>
</tr>
<tr>
<td>Rate-Distortion Optimization</td>
<td>Not Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Support Format</td>
<td>Progressive &amp; Interlaced</td>
<td>Progressive &amp; Interlaced</td>
</tr>
<tr>
<td>Prediction Modes</td>
<td>Field, frame</td>
<td>Field, frame</td>
</tr>
<tr>
<td>Support resolutions</td>
<td>From SQCIF to HDTV</td>
<td>From SQCIF to HDTV</td>
</tr>
<tr>
<td>Deblock Filter</td>
<td>No</td>
<td>Yes, $4 \times 4$ block boundary, in loop</td>
</tr>
</tbody>
</table>
CHAPTER 1. INTRODUCTION
Chapter 2

Level C Scheme for Transcoding

2.1 Introduction

Many works have been contributed to develop an efficient MPEG-2 to H.264/AVC transcoding algorithm. The key to reduce the complexity for inter transcoding is the motion re-estimation and mode decision, which typically accounts for more than 80% of a full H.264/AVC encoder. The motion re-estimation in MPEG-2 to H.264/AVC is often categorized in two kinds: in transform domain and in spatial domain.

[38–40] introduce an HDTV recording system that employs MPEG-2 to H.264/AVC transcoding to achieve efficient storage of broadcast streams. A transform-domain MPEG-2 to H.264/AVC intra video transcoder architecture is developed. Input DCT coefficients are first converted to H.264/AVC transform (HT) coefficients in the transform-domain. For intra coding, a fast rate-distortion optimized macroblock mode decision based on a simple cost function calculated in the HT-domain is then performed. Finally, the HT coefficients are coded using the selected modes to generate the output H.264 bitstream. This transcoder architecture reduces the complexity requirement about 50%, while maintaining virtually the same coding efficiency. The inter coding is performed in transform domain based on transform-domain cost function. Compared to spatial domain method, frequency domain
method is more efficient but will introduce more video quality reduction (about 1 db in [38]). [41] proposed compensation method for requantization errors and achieved about 5 db quality improvement over the open-loop transform-domain-based transcoding.

Spatial domain methods often employ MPEG-2 MV reuse technique to reduce computation for ME. In [42], the median predictors and MPEG-2 MV are be categorized as the most important predictors. The motion cost of the median predictor is first calculated, and if this cost is smaller then a pre-determined threshold, $T_{\text{median}}$, the motion search terminates. If not, a similar procedure is applied to MPEG-2 MV. When the cost of MPEG-2 MV is less than a second threshold $T_{\text{MPEG-2}}$, the motion search can again terminate, therefore avoiding any further computations. In [43], MV re-estimation for the 16 × 16 MB in H.264 encoder end follows the following procedure. The motion costs are compared for the MVPs from the MPEG-2 stream and for the H.264 MVP from the MBs neighbors, and the best one is selected as the search-center. Then the motion costs at the search center and the 4 pixel locations around the search-center with 1 pixel distance (top, bottom, left, and right) are compared to refine the search center. The search range is reduced because the MVPs from MPEG-2 MB are more accurate compared to the H.264 MVPs, especially when the MV field is not consistent. Furthermore, they are available at the picture boundary where some neighbor MBs are not available. The motion re-estimation for 7 sub-block $(16 \times 8, 8 \times 16, 8 \times 8, 8 \times 4, 4 \times 8, 4 \times 4)$ follows a top-down splitting procedure. [44] proposed an approach called Dynamic Search Range (DSR) based on the length of the motion vectors in the MPEG-2 video to reduce the search range adaptively. In [45], the correlation of MPEG-2 MV is used to evaluate the reliability of motion estimation for mapping from MPEG-2 inter mode to the H.264 intra mode.

For hardware design of transcoder, the MPEG-2 MV reuse will also bring many advantages. For example in the design of motion estimation engine in H.264 encoder, two buffer memories are required. One is for current MB, the other is for the search range data of current MB. If the MPEG-2 MV is reused, a relative small search range buffer is enough.
for motion estimation because of the precision of MPEG-2 MV. At the same time, the number of Processing Element (PE) can also be reduced, especially in HDTV application, because usually several PE units are parallelized to meet high performance requirement in such application. Thus the overall hardware cost and power dissipation will be reduced. In addition, the motion estimation procedure will also be accelerated because of smaller search range.

The MV reuse scheme is easily implemented in software. But in the hardware design for ME module, the introduction of MPEG-2 MV will bring some problems, one of which is the application of search window reuse scheme, which plays a critical role in current bandwidth reduction scheme in video coding systems. In traditional FSMBA algorithm, every frame is divided into MB; and each MB has a fixed position with regular interval to other MB. The center position of search window in reference frame is equal to the center of MB. Thus regular overlapping area of search window can be guaranteed. All currently existing data reuse scheme is based on such regularity. However, introducing of MPEG-2 MV as search center will break up this regularity, as shown in the upper part in Fig. 2.1. Irregular memory access will lower memory I/O efficiency and increase power dissipation. Thus a new framework is needed to handle this issue for transcoding while maintain simple architecture of hardware and good performance of external bandwidth.

In this dissertation, two data reuse methods are proposed for MPEG-2 to H.264 transcoding. The Level C method for transcoding utilize the similarity between successive MV to regularize the search window position for most MB. The proposed Level C+ method loads search window for a group of MB to achieve further bandwidth reduction. These two methods are shown in following chapters.

The Level C scheme for transcoding is based on the fact that neighboring MPEG-2 MV often have similar value. If the MV difference between successive MVs is less than a threshold, it is defined as smooth MV field and they are regularized to have fixed interval. That is successive two MBs have 16 pixel difference in x-coordinate. Therefore successive
two MBs can share part of search window if MV field is smooth; otherwise search window should be flushed. Since most MB in sequence can be regularized based on the experimental results, a low bandwidth level can be achieved for transcoder combined with the smaller search range introduced by MPEG-2 MV.

2.2 The General Algorithm

The Level C bandwidth reduction scheme for transcoding is shown in Algorithm 1. $\vec{mv}_i$ and $\vec{mv}_{i-1}$ are current and previous MPEG-2 MV in processing order; $(x_i, y_i)$ and $(x_{i-1}, y_{i-1})$ are their coordinates; $t$ is a predefined threshold; **MotionEstimation** represents the function of motion estimation performed within $SW_i$; **SearchWindow** is the function to determine search window based on $\vec{mv}_i$ as following

$$SW_i = \text{SearchWindow}(\vec{mv}_i) = \{(x, y) | x \in \mathcal{X}, y \in \mathcal{Y}\} \quad (2.2.1)$$

where $\mathcal{X}$ and $\mathcal{Y}$ are
\[ \mathcal{X} = [x_i - p_h, x_i + p_h + N - 1) \]
\[ \mathcal{Y} = [y_i - p_v, y_i + p_v + N - 1) \]  

(2.2.2)

The difference between our method and the standard Level C scheme is that if current MV is not similar with previous one, the search window buffer should be completely flushed, not just part of it. Experiment result shows that this kind of MB only occupies a small percent in sequences, especially in HDTV sequences. Thus most MB can utilize search window reuse under the proposed algorithm framework. The proposed method also requires two on-chip memories and associated control logic, which controls the memory flush based MPEG-2 MV difference.

<table>
<thead>
<tr>
<th>Algorithm 1: The Level C scheme for transcoding</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input</strong>: Current MB ( i ), MPEG-2 MV: ( \vec{\text{mv}}<em>i ) and ( \vec{\text{mv}}</em>{i-1} ) of MB ( i ) and MB ( i-1 ), reference data.</td>
</tr>
<tr>
<td><strong>Output</strong>: MV ( i ) of current MB ( i )</td>
</tr>
<tr>
<td>foreach MB ( i ) in sequence do</td>
</tr>
<tr>
<td>1 if (</td>
</tr>
<tr>
<td>2 ( x_i \leftarrow x_{i-1} + 16 )</td>
</tr>
<tr>
<td>3 ( y_i \leftarrow y_{i-1} )</td>
</tr>
<tr>
<td>4 ( SW_{\text{reuse}} \leftarrow \text{SearchWindow}(\vec{\text{mv}}<em>i) \bigcap \text{SearchWindow}(\vec{\text{mv}}</em>{i-1}) )</td>
</tr>
<tr>
<td>5 ( SW_{\text{loaded}} \leftarrow \text{SearchWindow}(\vec{\text{mv}}<em>i) - SW</em>{\text{reuse}} )</td>
</tr>
<tr>
<td>6 ( SW_i \leftarrow SW_{\text{loaded}} \bigcup SW_{\text{reuse}} )</td>
</tr>
<tr>
<td>8 else</td>
</tr>
<tr>
<td>9 ( SW_i \leftarrow SW_{\text{loaded}} \leftarrow \text{SearchWindow}(\vec{\text{mv}}_i) )</td>
</tr>
<tr>
<td>10 end</td>
</tr>
<tr>
<td>11 ( MV_i \leftarrow \text{MotionEstimation}(MB_i, SW_i) )</td>
</tr>
<tr>
<td>12 end</td>
</tr>
</tbody>
</table>
2.3 Performance Evaluation

Two factors must be taken into consideration to evaluate performance of data reuse scheme: on-chip memory size for reference data and redundancy access factor [37]. The on-chip memory size represent the required buffer of reference data. The redundancy access factor $R_\alpha$ evaluates external bandwidth and is defined as the number of reference pixel be loaded for each MB pixel. $R_\alpha$ of the proposed Level C method is calculated as the expectation of $R_\alpha$ of all MB, which is shown as following:

$$R_\alpha = p_{reuse} \cdot R_\alpha(\text{Level C}) + (1 - p_{reuse}) \cdot R_\alpha(\text{No Reuse})$$ (2.3.1)

where $p_{reuse}$ is the probability that a MB can reuse search window. $R_\alpha(\text{Level C})$ and $R_\alpha(\text{No Reuse})$ are calculated as following:

$$R_\alpha(\text{Level C}) = \frac{N(sr_v + N - 1)}{N^2}$$ (2.3.2)

$$R_\alpha(\text{No Reuse}) = \frac{(sr_h + N - 1)(sr_v + N - 1)}{N^2}$$ (2.3.3)

Based on the hardware design result, the on-chip memory is equal to $4/3$ times of the size of search window, which is $(sr_h + N - 1)(sr_v + N - 1)$. This is because reference pixel buffer must be flushed when the coordinate difference is larger than the threshold; and the next MB employs different control schedule to load reference pixel.

In Section 2.4.2, it is shown that more than 90% MB in HDTV720p sequence can reuse search window, while the average reuse rate is 84.2% in CIF sequence. It is also shown that video quality is almost not degraded although there exists MV modification.
2.4 Experimental Results

In our experiments, the following MV reuse scheme is adopted. First, the MPEG-2 MV is used as search center in ME module of H.264 encoder. In MPEG-2 sequence, only the MV of 16 × 16 MB is available which is used as search center for 16 × 16 MB mode and six sub-block modes in H.264. Secondly, a relative small search range is selected for ME part in H.264. After intensive experiments, we select [−8, 8) for CIF and [−16, 16) for HDTV720p sequence as search range on both horizontal and vertical direction.

Experiments are conducted in CIF (352 × 288) and HDTV720p (1280 × 720) sequence. The MPEG-2 reference software [46] and the H.264 reference software JM 11 [47] are used to construct the transcoding platform and to simulate our search window reuse algorithm. In MPEG-2 encoder, we set [−16, 16) and [−64, 64) as search range for CIF and HDTV720p sequence. All sequences are coded with IPPP pattern in MPEG-2 encoder. The MV of Intra MB in P-picture is set to the same value as previous MV; and the MV of P-Skip MB in P-picture is set to zero.

In practical applications, MPEG-2 video is usually coded in IBBP pattern. Transcoding such type of sequence involves motion and mode mapping between MPEG-2 and H.264. [48] proposed an motion and mode mapping algorithm for MPEG-2 to H.264 baseline profile transcoding. Coding pattern does not affect the high similarity among neighboring MV in the same frame, which is the foundation of the proposed method. Thus a simplified coding pattern IPPP of MPEG-2 sequence is employed to focus on testing the effectiveness of the proposed search window reuse scheme.

2.4.1 Experiment Result About Threshold Choice

The reuse-rate $p_{\text{reuse}}$ in the proposed method is defined as

$$p_{\text{reuse}} = \frac{n_{\text{reuse}}}{N_{\text{total}}}$$  \hspace{1cm} (2.4.1)
where \( n_{\text{reuse}} \) is the number of MB which can utilize the Level C search window reuse under the proposed algorithm framework. \( N_{\text{total}} \) is the total number of MB in the whole sequence.

Obviously \( p_{\text{reuse}} \) determines bandwidth level in the proposed algorithm, because higher \( p_{\text{reuse}} \) means more MB can utilize search window reuse. So we must test which threshold value can achieve a usable reuse-rate. Experiments are conducted in Foreman (CIF) and Crew (HDTV720p) sequence. Fig. 2.2 shows the experiment result. It is observed that about 80% (Foreman) and 90% (Crew) MB can be regularized if the threshold is set to 5. Further increasing the threshold will lead to slowly increase in reuse-rate; this is mainly due to the sparse distribution of MB whose MV difference is larger than the threshold. In the following experiments, the threshold is set to 5.

### 2.4.2 Experiment Result About Reuse-rate

In this section, the experiment results about reuse-rate is presented. Because if only a few MB in a sequence can use search window reuse, the bandwidth cannot be greatly reduced. In experiments, four CIF sequences and five HDTV720p sequences are tested. The results are shown in Table 2.1. The threshold is set to 5 to follow the discussion in section 2.4.1.

It is observed from Table 2.1 that most of MB both in CIF and HDTV720p sequences can utilize search window reuse under the proposed method. In general, the HDTV720p sequence achieves higher reuse-rate than CIF sequence. For the same video content, the HDTV720p sequence has higher resolution and thus a lower difference between pixels of neighboring MB. So the MV similarity in HDTV720p sequence is higher than CIF. In CIF sequence, the Canoa has the smallest reuse-rate because of the complicate background and fast motion in it, which will lead to larger difference between successive MV. And the Container achieves the highest reuse-rate in CIF sequence because of its slow motion.
2.4. EXPERIMENTAL RESULTS

Table 2.1: Search window reuse-rate comparison.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>S.R Reuse-rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDTV720p</td>
<td></td>
</tr>
<tr>
<td>Crew</td>
<td>90.7%</td>
</tr>
<tr>
<td>KnightShield</td>
<td>91.3%</td>
</tr>
<tr>
<td>City</td>
<td>92.8%</td>
</tr>
<tr>
<td>Harbour</td>
<td>93.2%</td>
</tr>
<tr>
<td>Parkrun</td>
<td>97.5%</td>
</tr>
<tr>
<td>CIF</td>
<td></td>
</tr>
<tr>
<td>Canoa</td>
<td>74.0%</td>
</tr>
<tr>
<td>Container</td>
<td>95.5%</td>
</tr>
<tr>
<td>Foreman</td>
<td>81.9%</td>
</tr>
<tr>
<td>Stefan</td>
<td>85.5%</td>
</tr>
</tbody>
</table>

2.4.3 Experiment Result About Video Quality

In this section, experiments have been conducted for testing whether the proposed search window reuse scheme will degrade video quality, because the regularized search window is not the same as the original one. The rate distortion (RD) curves are used for comparison. Two methods are compared, both of which use MPEG-2 MV as search center. And one method uses the proposed search window reuse scheme, the other does not. These two methods are indicated as “SW Reuse” and “MV Reuse” in Fig. 2.3 and Fig. 2.4. Search range is set to 8 and 16 respectively. And the threshold is set to 5 as shown in section 2.4.1 and their respective reuse-rate are shown in Table 2.1. Four CIF and four HDTV720p sequences are tested.

From Fig.2.3 and Fig. 2.4, it is observed that application of the proposed search window reuse scheme almost does not degrade video quality with the given threshold value; this means our search window reuse algorithm will leads to little degradation on video quality. The main reason for this is the similarity between successive MV and a relative small
threshold. If some small difference between successive search windows is ignored, the best matching position in reference frame for current MB will not be biased too much and video quality can be maintained.

As discussed in Section 2.4.2, further increasing the threshold would bring little difference in reuse-rate; thus video quality is not degraded too much because only a small part of MB would be affected by an increased threshold. But it is difficult to determine which threshold can achieve 100% reuse-rate for each sequence. As shown in later sections, the given threshold and associate reuse-rate already achieve a good level of external bandwidth; thus we set the threshold to 5 in all experiments.

### 2.4.4 External Bandwidth Comparison

Finally, the external bandwidths of three integer motion estimation (IME) modules are compared based on theoretical calculation. The test sequence is HDTV720p (1280 × 720), 30 fps. And one reference frame is used.

- **H.264 (Level C)** A regular H.264 integer motion estimation (IME) module with Level C search window. The search range is set to $p_h = 64, p_v = 48$.

- **H.264 (Level C+)** A regular H.264 IME module with Level C+ data reuse scheme, in which two MB are vertically stitched. The search range is set to $p_h = 64, p_v = 48$.

- **Transcoder (No Reuse)** An IME module of transcoder with MPEG-2 MV reuse but without any data reuse method. The search range is set to $p_h = 16, p_v = 12$.

- **Transcoder (Level C)** An IME module of transcoder with MPEG-2 MV reuse and Level C search window reuse scheme. The reuse-rate is set to 90% and the search range is set to $p_h = 16, p_v = 12$.

The redundancy access factor ($R_\alpha$) and on-chip memory size are compared in Table 2.2. The $R_\alpha$ of the proposed method is calculated as following.

36
\[ R_\alpha = p_{\text{reuse}} \cdot R_\alpha(\text{Level C}) + (1 - p_{\text{reuse}}) \cdot R_\alpha(\text{No Reuse}) \]
\[ = \frac{p_{\text{reuse}} N (sr_v + N - 1) + (1 - p_{\text{reuse}}) (sr_h + N - 1) (sr_v + N - 1)}{N \times N} \]
\[ \approx (v + 1) [p_{\text{reuse}} + (1 - p_{\text{reuse}})(h + 1)] \]
\[ = (v + 1)[1 + h(1 - p_{\text{reuse}})] \quad (2.4.2) \]

where \( p_{\text{reuse}} \) is reuse-rate, \( v = sr_v/N \), \( h = sr_h/N \), \( p_{\text{reuse}} \in [0,1] \). It is observed from this equation that if \( p_{\text{reuse}} \) is increased, \( R_\alpha \) will be decreased; thus external bandwidth will be decreased. This conclusion is in accord with our previous discussion in section 2.4.1.

From Table 2.2, it is observed that the required external bandwidth of the IME module with the proposed search window reuse scheme is only 40.6% of the transcoding algorithm without any data reuse method; and its bandwidth level is 41.9% of the standard IME module in H.264 with Level C scheme. The proposed method also achieves an equivalent performance compared with Level C+ method. This is because the IME module without search window reuse scheme should flush the whole buffer memory, which is \((32 + 15)(24 + 15) = 1833\) pixels, when a new MB is processed. The proposed scheme only needs to load \(0.9 \times 16 \times 39 + 0.1 \times 1833 = 744.9\) pixels. And the standard IME module of H.264 with Level C scheme needs to load \((96 + 15) \times 16 = 1776\) pixels. The Level C+ should load \((96 + 31) \times 16 \div 2 = 1016\) (2 MB stitched) and \((96 + 63) \times 16 \div 2 = 1272\) (4 MB stitched) pixels for each MB.

The size of on-chip memory is equal to search window size. According calculation equation mentioned in Section 2.3, the Level C needs \((128 + 15)(96 + 15) = 15873\) byte. The Level C+ needs \((128 + 31)(96 + 31) = 20193\) (2 MB stitched) and \((128 + 63)(96 + 63) = 30369\) (4 MB stitched) byte. The transcoding algorithm only with MV reuse needs \((32 + 15)(24 + 15) = 1833\) byte. The proposed method needs \(2 \frac{1}{3} (32 + 15)(24 + 15) = 2444\) byte, because two memories are needed, which are switched based on the coordinate difference between successive MPEG-2 MV. In our calculation, only luminance component is taken.
into consideration. Each reference pixel is assigned one byte to represent it.

<table>
<thead>
<tr>
<th>Table 2.2: Bandwidth and on-chip memory comparison.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rα</td>
</tr>
<tr>
<td>(Kbyte)</td>
</tr>
<tr>
<td>H.264 (Level C)</td>
</tr>
<tr>
<td>H.264 (Level C+)</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Transcoder (No Reuse)</td>
</tr>
<tr>
<td>Transcoder (Level C)</td>
</tr>
</tbody>
</table>

2.5 Conclusion

In this chapter, we propose a search window reuse scheme for MPEG-2 to H.264 transcoding. The existing data reuse methods rely on the regular position of search window, which is introduced by FSBMA. However in the ME module of H.264 encoder of transcoder, the introducing of MPEG-2 MV as search center will lead to irregular overlapping between neighboring search windows. In this chapter, we present method to regularize the overlapping area by utilizing the similarity between successive MV. Then the Level C data reuse scheme is available for most MBs. The proposed method is characterized as the judgement of smoothness of MV field. If MV field is smooth, search center of current MB is chosen as having a fixed interval with previous MB in processing order; Otherwise search center is maintained as the original one.

Experimental results shows that the external bandwidth of IME module using the proposed Level C search window reuse scheme can be reduced to 40.6% of transcoder without any data reuse method. And the bandwidth level is almost equivalent with H.264 IME module with Level C+ method. But the on-chip memory is much smaller.
The traditional data reuse methods utilize regular overlapping area of search window or candidate block. The proposed algorithm achieves a low external bandwidth by utilizing small search window together with the reuse of overlapping area among most MB. The small search window is caused by the high accuracy of MPEG-2 MV. And the overlapping area of most MB comes from the similarity between successive MV; this fact is obvious particularly in HDTV sequence as shown in experimental results.
Figure 2.2: Threshold value versus reuse-rate.
Figure 2.3: Video quality comparison on CIF.
Figure 2.4: Video quality comparison on HDTV720p.
Chapter 3

Level C+ Scheme for Transcoding

3.1 Introduction

In some practical video transcoding system designs, ultra-low bandwidth ($R_\alpha < 2$) is required because: 1) the strictly limited availability of bandwidth in these designs; 2) larger bandwidth also induces higher power dissipation, package cost and increase problems with skew; 3) the size of on-chip memory is not a constraining factor in these designs considering the continually decreased production cost of on-chip memory. Furthermore, from a systematic point of view, memory traffic introduced by other components such as variable-length-codec, DCT/IDCT, and so on must also be considered. Additionally, the motion-estimation process only uses the luminance pixel data, while the other components also use chrominance data thus increasing the importance of strong data-reuse level. Thus we can make a tradeoff between on-chip memory and bandwidth to meet the system requirement of ultra-low bandwidth.

To achieve this goal, a search window reuse scheme, called Level C+ scheme, is proposed in this paper for MPEG-2 to H.264/AVC transcoding. The fundamental idea of the proposed method is to load search window for a group of MB stitched together. MB of the same group can reuse search window horizontally and vertically. By this way, the
average loaded reference pixel for each MB can be further reduced. The proposed method is described as in Fig. 3.1. It is composed of four steps

1. Each frame is divided into motion estimation unit (MEU).

2. Search center and search window for each MEU is decided.

3. Motion vector clipping is performed on MV that goes beyond the search window boundary of MEU.

4. Motion estimation is performed for each MB in MEU.

\[ \text{Divide frame into MEU} \ (V \times H) \rightarrow \text{Decide search center for MEU} \rightarrow \text{Motion vector clipping} \rightarrow \text{Motion estimation for MEU} \]

Figure 3.1: The flowchart of the proposed Level C+ method.

### 3.2 Motion Estimation Unit (MEU)

The MB group in the proposed method is defined as Motion Estimation Unit (MEU), which is the unit of reference pixel loading and motion estimation. It is composed of \( v \) MB in vertical direction and \( h \) MB in horizontal direction as shown in Eq. 3.2.1. Fig. 3.2 shows an example of MEU.

To load search window for MEU, search center of MEU (\( \vec{m} \hat{v}_{MEU} \)) must be first determined, which is calculated as average \( \vec{m} \hat{v} \) of each MB in MEU. In this step, we make the assumption that each MB in MEU has the same MV as \( \vec{m} \hat{v}_{MEU} \); thus after the search
center of MEU is determined, the boundary of search window of MEU ($SW_{MEU}$) is also determined.

$$MEU = \{MB_{ij}| 0 < i \leq v, 0 < j \leq h\}$$

$$\vec{mv}_{MEU} = \frac{1}{v \cdot h} \sum_{i=1}^{v} \sum_{j=1}^{h} \vec{mv}_{ij}$$ (3.2.1)

### 3.3 The General Algorithm

The Level C+ bandwidth reduction method for transcoding is described in detail as shown in Algorithm 2. **SearchWindow** is the function to determine search window ($SW_{ij}$) based on $\vec{mv}_{ij}$ as shown in Eq. 2.2.1; **SearchWindowMEU** is the function to determine search window for MEU (Eq. 3.3.1); **MVClipping** represents the processing of motion vector clipping. It performs on $\vec{mv}_{ij}$ which goes beyond the boundary of $SW_{MEU}$ to make sure
that each MB in MEU can load reference pixels from the same on-chip memory. After that both horizontal and vertical search window reuse is available. Fig. 3.3 shows an example.

\begin{algorithm}
\textbf{Algorithm 2}: The Level C+ scheme for transcoding

\textbf{Input}: Current \( MB_{ij} \in \text{MEU} \), MPEG-2 MV \( \vec{mv}_{ij}, \vec{mv}_{MEU} \) and reference data.

\textbf{Output}: \( MV_{ij} \) of current \( MB_{ij} \)

1. \( SW_{MEU} \leftarrow \text{SearchWindowMEU}(\vec{mv}_{MEU}) \)

2. \textbf{foreach} \( MB_{i} \) in sequence \textbf{do}

3. \( SW_{ij} \leftarrow \text{SearchWindow}(\vec{mv}_{ij}) \)

4. \textbf{if} \( SW_{ij} \subset SW_{MEU} \) \textbf{then}

5. \( MV_{ij} \leftarrow \text{MotionEstimation}(SW_{ij}) \)

6. \textbf{else}

7. \( \vec{mv}'_{ij} \leftarrow \text{MVClipping}(\vec{mv}_{ij}) \)

8. \( SW'_{ij} \leftarrow \text{SearchWindow}(\vec{mv}'_{ij}) \)

9. \( MV_{ij} \leftarrow \text{MotionEstimation}(SW'_{ij}) \)

10. \textbf{end}

11. \textbf{end}
\end{algorithm}

\[ SW_{MEU} = \text{SearchWindowMEU}(\vec{mv}_{MEU}) = \{(x, y)|x \in X_{MEU}, y \in Y_{MEU}\} \quad (3.3.1) \]

where \( X_{MEU} \) and \( Y_{MEU} \) are

\[ X_{MEU} = [x_{MEU} - p_{h}, x_{MEU} + p_{h} + N \cdot h - 1) \]

\[ Y_{MEU} = [y_{MEU} - p_{v}, y_{MEU} + p_{v} + N \cdot v - 1) \quad (3.3.2) \]
3.4 Performance Evaluation and Optimization of MEU Size

The redundancy access factor $R_\alpha$ of the proposed Level C+ method is calculated as follows:

$$R_\alpha = \frac{(sr_h + h \cdot N - 1)(sr_v + v \cdot N - 1)}{v \cdot h \cdot N^2} \quad (3.4.1)$$

The size of $SW_{MEU}$ is $(sr_h + h \cdot N - 1)(sr_v + v \cdot N - 1)$, which is also equal to the size of on-chip memory ($M_{on-chip}$).

In Fig. 3.4, the relationship between $v$, $h$ and $R_\alpha$ is presented. It is observed that $R_\alpha$ will reach a very low level (approaching 1) if $v$ and $h$ are large enough. But too large $v$ and $h$ will lead to unacceptable size of on-chip memory. A proper size of MEU must strike a balance between external bandwidth and on-chip memory. This problem can be addressed...
by solving the following optimization problem.

\[ 
\min_{v,h} \quad z = f(v, h) = R_\alpha + \frac{1}{K} \cdot M_{on-chip} \\
\text{s.t.} \quad 0 < v \leq N_v, \\
\quad 0 < h \leq N_h, \\
\quad v > h. 
\]  

(3.4.2)

\( z \) is the cost function of variable \( v \) and \( h \), which is composed of redundancy access factor \( R_\alpha \) (Equation 3.4.1) and \( M_{on-chip} \) as mentioned in last section. \( K \) is the normalizer of
on-chip memory, which is set to the on-chip memory size of standard H.264 IME module (4 MB) as shown in Table 3.1. $N_v$ and $N_h$ is the number of MB in vertical and horizontal direction. $p_h = 16, p_v = 12$ are set for HDTV720p sequence and $p_h = 8, p_v = 6$ are set for CIF sequence.

$v > h$ is set because the horizontal variation of motion vector is larger than in vertical direction. The optimization result for HDTV720p is $v = 6, h = 4$ and for CIF is $v = 3, h = 2$.

### 3.5 Experimental Results

In experiments, MPEG-2 decoder and H.264/AVC encoder are directly cascaded to form a close-loop architecture. Video sequence is firstly encoded with MPEG-2 encoder and then decoded with MPEG-2 decoder. The decoded video sequence and MPEG-2 MV are inputted into H.264/AVC encoder.

Experiments are conducted in CIF (352 x 288) and HDTV720p (1280 x 720) sequence. The MPEG-2 reference software [46] and the H.264 reference software JM 11 [47] are used to construct the transcoding platform and to simulate our search window reuse algorithm. In MPEG-2 encoder, we set search range as $p_h = 32, p_v = 24$ for CIF and $p_h = 64, p_v = 48$ for HDTV720p. The search range in H.264/AVC encoder is set to $p_h = 8, p_v = 6$ for CIF and $p_h = 16, p_v = 12$ for HDTV720p. All sequences are coded with IPPP pattern in MPEG-2 encoder. The MV of Intra MB in P-picture is set to the same value as previous MV; and the MV of P-Skip MB in P-picture is set to zero.

#### 3.5.1 Experiment Result About Search Range

This section presents experiment results about search range selection for transcoding algorithm. R-D curves of two methods on are compared. The test sequence are two CIF and two HDTV720p sequences.
CHAPTER 3. LEVEL C+ SCHEME FOR TRANSCODING

- Full search algorithm (indicated as “FS” in Fig. 3.5) implemented in H.264 reference software. \( p_h = 32, p_v = 24 \) is set for CIF and \( p_h = 64, p_v = 48 \) for HDTV720p.

- Transcoding algorithm (indicated as “MPEG-2” in Fig. 3.5) reusing MPEG-2 MV as search center. \( p_h = 8, p_v = 6 \) is set for CIF and \( p_h = 16, p_v = 12 \) for HDTV720p.

Fig. 3.5 shows experiment results. It is observed that video quality achieved by these two methods have little difference. Thus \( p_h = 8, p_v = 6 \) and \( p_h = 16, p_v = 12 \) are proper search range setting for CIF and HDTV720p in transcoding application. In the following experiments, these settings are utilized.

### 3.5.2 Experiment Result About Video Quality

This section presents experiment results about video quality comparison since the proposed Level C+ search window reuse method applies motion vector clipping. Some motion information is lost in this processing. Experiments must be conducted to illustrate whether video quality is degraded by the proposed method. The test sequence are four CIF and four HDTV720p sequences. R-D curves of two methods are compared.

- Transcoding algorithm reusing MPEG-2 MV as search center. No search window reuse scheme is implemented. (indicated as “No Reuse” in Fig. 3.6 and Fig. 3.7)

- Transcoding algorithm reusing MPEG-2 MV as search center. The proposed Level C+ search window reuse scheme is implemented. (indicated as “Level C+” in Fig. 3.6 and Fig. 3.7)

Fig. 3.6 and Fig. 3.7 present little difference of video quality between the two compared methods. The proposed search window reuse method leads to negligible video quality degradation compared with regular transcoding algorithm. Therefore the video quality can be guaranteed by the proposed bandwidth reduction scheme.
Figure 3.5: Search range decision for CIF and HDTV720p.
Figure 3.6: Video quality comparison in CIF.
Figure 3.7: Video quality comparison in HDTV720p.
3.5.3 External Bandwidth Comparison

This section presents the bandwidth an on-chip memory comparison between five algorithms. The test sequence is HDTV720p (1280 × 720).

- **H.264 (Level C)** A regular H.264 integer motion estimation (IME) module with Level C search window. The search range is set to \( p_h = 64, p_v = 48 \).

- **H.264 (Level C+)** A regular H.264 IME module with Level C+ data reuse scheme, in which two MB are vertically stitched. The search range is set to \( p_h = 64, p_v = 48 \).

- **Transcoder (No Reuse)** An IME module of transcoder with MPEG-2 MV reuse but without any data reuse method. The search range is set to \( p_h = 16, p_v = 12 \).

- **Transcoder (Level C)** An IME module of transcoder with MPEG-2 MV reuse and Level C search window reuse scheme. The reuse-rate is set to 90\% and the search range is set to \( p_h = 16, p_v = 12 \).

- **Transcoder (Level C+)** An IME module of transcoder with the Level C+ search window reuse scheme \( (v = 6, h = 4) \) and the search range is set to \( p_h = 16, p_v = 12 \).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>( R_o )</th>
<th>On-chip Memory (Kbyte)</th>
</tr>
</thead>
<tbody>
<tr>
<td>H.264 (Level C)</td>
<td>6.94</td>
<td>15.5</td>
</tr>
<tr>
<td>H.264 (Level C+)</td>
<td>2 MB</td>
<td>3.97</td>
</tr>
<tr>
<td></td>
<td>4 MB</td>
<td>2.48</td>
</tr>
<tr>
<td>Transcoder (No Reuse)</td>
<td>7.16</td>
<td>1.8</td>
</tr>
<tr>
<td>Transcoder (Level C)</td>
<td>2.91</td>
<td>2.25</td>
</tr>
<tr>
<td>Transcoder (Level C+)</td>
<td>1.84</td>
<td>11.0</td>
</tr>
</tbody>
</table>
Table 3.1 shows the bandwidth and on-chip memory comparison results. The calculation method is based on [37], [49] and Section 3.4. It is observed from Table 3.1 that the redundancy access factor $R_\alpha$ of the proposed Level C+ method is 63.2% of the transcoder with Level C and 46.3% of H.264 IME with Level C+ (2 MB stitched). The $R_\alpha < 2$ means that it needs no more than two reference pixel for motion estimation of each MB pixel. The on-chip memory of the proposed method is 62.1% of H.264 IME with Level C+ (2 MB stitched) and 49.5% (4 MB stitched). The on-chip memory is 3.1 times of transcoder with Level C scheme. Therefore the proposed Level C+ scheme for transcoding makes a tradeoff between on-chip memory and bandwidth to achieve the design goal of ultra-low bandwidth.

3.6 Conclusion

In this chapter, an ultra-low bandwidth design method, called Level C+ scheme, is proposed for MPEG-2 to H.264/AVC transcoding. The proposed method is designed for applications with strictly limited bandwidth resource, while the size of on-chip memory is not a main design constraining factor. It makes a tradeoff between bandwidth and on-chip memory to achieve the design goal.

In the proposed method, motion estimation is performed in the unit of MEU. Search window position of MEU is determined by average MV of each MB in MEU. Motion vector clipping is applied to confine the search window of each MB within the search window of MEU. The proper size of ME is determined by an optimization-based method to strike a balance between $R_\alpha$ and on-chip memory. Through these processing, both horizontal and vertical search window reuse is available. And video quality is not affected by the proposed method because of motion vector smoothing. Experiment results show that the proposed Level C+ method can achieve a very low external bandwidth ($R_\alpha < 2$) with an acceptable on-chip memory.
CHAPTER 3. LEVEL C+ SCHEME FOR TRANSCODING

Two factors contribute to the ultra-low bandwidth achieved by the proposed method: the small search range introduced by high precision MPEG-2 MV; the horizontal and vertical search window reuse introduced by the proposed method. The difference between the proposed and Level C+ method for H.264/AVC encoder is that the inter search window reuse is not applied in the proposed method. Another difference is that both MB in horizontal and vertical direction are stitched as a unit of motion estimation in the proposed method.
Chapter 4

Hardware Architecture for Level C Scheme

4.1 Introduction

For video coding systems, motion estimation (ME) can remove most of temporal redundancy, so a high compression ratio can be achieved. Among various ME algorithms, a full-search block matching algorithm (FSBMA) is usually adopted because of its good quality and regular computation. In FSBMA, the current frame is partitioned into many small macroblocks (MBs) of size $N \times N$. For each MB in the current frame (current MB), one reference block that is the most similar to current MB is sought in the searching range of size $[-P, P]$ in the reference frame. The most common used criterion of the similarity is the sum of absolute differences (SAD)

$$\text{SAD}(m,n) = \sum_{i=1}^{N} \sum_{j=1}^{N} \text{Distortion}(i,j,m,n) \quad -P \leq m,n \leq P \quad (4.1.1)$$

$$\text{Distortion}(i,j,m,n) = |\text{cur}(i,j) - \text{ref}(i+m,j+n)| \quad (4.1.2)$$
where \( \text{cur}(i, j) \) and \( \text{ref}(i + m, j + n) \) are pixel values in the current MB (current pixel) and reference block (reference pixel), respectively, \((m, n)\) is one searching candidate in the search range, \( \text{Distortion}(i, j, m, n) \) is the difference between the current pixel and the reference pixel, and \( \text{SAD}(m, n) \) is the total distortion of this searching candidate. The row (column) SAD is the summation of \( N \) distortions in a row (column). After all searching candidates are examined, the searching candidate that has the smallest SAD is selected as the motion vector of the current MB. Although FSBMA provides the best quality among various ME algorithms, it consumes the largest computation power. In general, the computation complexity of ME varies from 50% to 90% of a typical video coding system. Hence, a hardware accelerator of ME is required.

Variable block-size motion estimation (VBSME) is a new coding technique and provides more accurate predictions compared to traditional fixed block-size motion estimation (FBSME). With FBSME, if an MB consists of two objects with different motion directions, the coding performance of this MB is worse. On the other hand, for the same condition, the MB can be divided into smaller blocks in order to fit the different motion directions with VBSME. Hence, the coding performance is improved. VBSME has been adopted in the latest video coding standards, including H.263 [13], MPEG-4 [15], WMV9.0 [50], and H.264/AVC [24]. For instance, in H.264/AVC, an MB with a variable block size can be divided into seven kinds of blocks including \( 4 \times 4, 4 \times 8, 8 \times 4, 8 \times 8, 8 \times 16, 16 \times 8, \) and \( 16 \times 16 \). Although VBSME can achieve a higher compression ratio, it not only requires huge computation complexity but also increases the difficulty of hardware implementation for ME.

Traditional ME hardware architectures are designed for FBSME, and they can be classified into two categories.

- **[Inter-Level Architecture]** Each processing element (PE) is responsible for one SAD of a specific searching candidate.
4.1. INTRODUCTION

- **[Intra-Level Architecture]** Each PE is responsible for the distortion of a specific current pixel in the current MB for all searching candidates.

[51] proposed the first VLSI motion estimator in the world, as shown in Fig. 4.1, which is a 1-D inter-level hardware architecture. The number of PEs is equal to the number of searching candidates in the horizontal direction, $2p_h$. Reference pixels are broadcasted into all PEs. By selection signals, the corresponding reference pixel is selected and inputted into each PE. Current pixels are propagated with propagation registers, and the partial SAD is stored in each PE. In each cycle, each PE computes the distortion and accumulates the SAD of a searching candidate. In this architecture, the most important concept is data broadcasting. With broadcasting technique, the memory bit width which is defined as the number of bits for the required reference data in one cycle is reduced significantly, although some global routings are required.

![Figure 4.1: The 1-D inter-level architecture proposed by [51].](image)

Fig. 4.2 shows a 2-D inter-level hardware architecture which is proposed by [52], which
consists of $2p_h \times 2p_v$ PEs and is similar to [51]. Reference pixels are broadcasted into PEs, and current pixels are propagated with propagation registers. The partial SADs are stored and accumulated in PEs, respectively. Because of broadcasting reference pixels in both directions, the number of PEs has to match the MB size. Hence, the search range is partitioned into $(2p_h/N) \times (2p_v/N)$ regions, and each region is computed by a set of $N \times N$ PEs. The characteristic of 2DInterYH is broadcasting in two directions at the same time, which can increase the data reuse.

[53] proposed another 1-D PE array that implemented a 2-D inter-level architecture with two data-interlacing reference arrays. The hardware architecture is shown in Fig. 4.3 and is similar to [52] except for two aspects. Reference pixels are propagated with propagation registers, and current pixels are broadcast into PEs. The partial SADs are still stored and accumulated in PEs. Besides, this architecture has to load reference pixels into propagation registers before computing SADs. The latency of loading reference pixels can be reduced by partitioning the search range in 2DInterLC. For example, the search

Figure 4.2: The 2-D inter-level architecture proposed by [52].
range can be partitioned into \((2p_h/N) \times (2p_v/N)\) parts for a shorter latency.

Figure 4.3: The 2-D inter-level architecture proposed by [53].

A 2-D intra-level architecture is proposed [55], as shown in Fig. 4.4, where the number of PEs is equal to the MB size. Each PE corresponds to a current pixel, and current pixels are stored in PEs, respectively. The important concept of [55] is the scanning order in searching candidates, which is known as the snake scan. In order to realize this, a great deal of propagation registers are used to store reference pixels, and the data in propagation registers can be shifted in upward, downward, and right directions. These propagation registers and the long latency for loading reference pixels are the tradeoffs for the reduction of memory usages. The computation flow is as follows. First, the distortion is computed in each PE, and \(N\) partial-row SADs are propagated and accumulated in the horizontal direction. Second, an adder tree is used to accumulate the \(N\)-row SADs to be SAD. The accumulations of row SADs and SAD are done in one cycle. Hence no partial SAD is required to be stored.

[54] proposed a detailed systolic mapping procedure by the dependence graph (DG). By using different DGs, including different scheduling and projections, different systolic hardware architectures can be derived. AB2 is a 2-D intra-level architecture, as shown in Fig. 4.5. Current pixels are stored in corresponding PEs. Reference pixels are propagated PE by PE in the horizontal direction. The \(N\) partial-column SADs are propagated and accumulated in the vertical direction first. After the vertical propagation, these \(N\)-column SADs are propagated in the horizontal direction. In each PE, the distortion of a current
pixel in the current MB is computed and added with the partial-column SAD which is propagated in PEs from top to bottom in the vertical direction. In the horizontal propagation, these $N$-column SADs are accumulated one by one by $N$ adders and $2N$ registers.

[56] proposed another 2-D intra-level hardware architecture with a search range buffer, as shown in Fig. 4.6. This architecture consists of PE arrays in the vertical direction, and each PE array is composed of PEs in a row. In [56], reference pixels are propagated with
propagation registers one by one, which can provide the advantages of serial data input and increasing the data reuse. Current pixels are still stored in PEs. The \( N \) partial-column SADs are propagated in the vertical direction from bottom to up. In each computing cycle, each PE array generates \( N \) distortions of a searching candidate and accumulates these distortions with \( N \) partial-column SADs in the vertical propagation. After accumulation in the vertical direction, \( N \)-column SADs are accumulated in the top adder tree in one cycle. The longer latency for loading reference pixels and large propagation registers are the penalties for the reduction of memory bandwidth and memory bit width.

For VBSME, [57] proposed two architectures. One is a 2-D intra-level architecture called the Propagate Partial SAD. The architecture is composed of \( N \) PE arrays with a

Figure 4.5: The 2-D intra-level architecture proposed by [54].
1-D adder tree in the vertical direction. Current pixels are stored in each PE, and two sets of $N$ continuous reference pixels in a row are broadcasted to $N$ PE arrays at the same time. In each PE array with a 1-D adder tree, $N$ distortions are computed and summed by a 1-D adder tree to generate one-row SAD. The row SADs are accumulated and propagated with propagation registers in the vertical direction. The other is SAD Tree architecture. The proposed SAD Tree is a 2-D intra-level architecture and consists of a 2-D PE array and one 2-D adder tree with propagation registers. Current pixels are stored in each PE, and reference pixels are stored in propagation registers for data reuse. In each cycle, $N \times N$ current and reference pixels are inputted to PEs. Simultaneously, $N$ continuous reference pixels in a row are inputted into propagation registers to update reference pixels. In propagation registers, reference pixels are propagated in the vertical direction row by row. In SAD Tree architecture, all distortions of a searching candidate
are generated in the same cycle, and by an adder tree, $N \times N$ distortions are accumulated to derive the SAD in one cycle.

In this dissertation, an IME module architecture is proposed for MPEG-2 to H.264/AVC transcoding. A modified ping-pang memory control scheme combined with Partial SAD VBSME are applied to realize the proposed Level C bandwidth reduction scheme for transcoding. The detailed contents are shown in the following sections.

### 4.2 The Top-Level Architecture

Fig. 4.7 shows top-level architecture of the proposed transcoder IME module for HDTV720p application. There exist four reference pixel memories, each of which is a $47 \times 16$ bytes single-port SRAM. Actually the search window is $47 \times 47$ bytes. The applied memories size is for an easy hardware implementation. The memory update and output is controlled by memory input and output control unit, which is controlled by MV smoothness decision unit. The IME module is implemented with Partial SAD architecture.

#### 4.2.1 Performance Analysis

Given the working frequency ($f$) of IME module, extra frequency ($f_{extra}$) for initialization latency and number of PEG ($m$), the processing ability of IME is expressed as $(f - f_{extra}) \times m$. The number of reference pixel to be processed is expressed as $(r \times n_{ref} \times W \times H \times sr_v \times sr_h)/256$, while $r$ is frame rate, $n_{ref}$ is number of reference frame, $W$ and $H$ is frame width and height. Therefore Equation 4.2.1 must be satisfied to process specific video sequence.

$$
(f - f_{extra}) \times m \geq \frac{r \times n_{ref} \times H \times W \times sr_v \times sr_h}{256} \tag{4.2.1}
$$

In the discussed transcoding application of HDTV720p, $f_{extra}$ is set to 16 clocks because this number of latency clocks are needed to produced SAD of the first reference position.
If one PEG is used to process HDTV720p video stream (1 reference frame, search range $[-16, 16]$), working frequency $f$ must satisfy Equation 4.2.2. Therefore the longest critical path delay must be less than 9.04$ns$.

\[
f \geq \frac{30 \times 1 \times 720 \times 1280 \times 32 \times 32}{256 \times 1} + 16 \simeq 110.6 M H z
\] (4.2.2)
4.2. THE TOP-LEVEL ARCHITECTURE

4.2.2 IME Architecture

In H.264/AVC, VBSME is adopted. For one MB, it has 4 block modes (16 × 16, 16 × 8, 8 × 16, 8 × 8). For 8 × 8 mode, it is further divided into 4 modes, namely 8 × 8, 8 × 4, 4 × 8, 4 × 4. For each MB, coding costs of all 7 block modes are calculated, and the block mode with the smallest cost is chosen as the MB mode. Compared with fixed block size ME algorithm, VBSME provides higher compression ratio, but it also puts heavy burden on the ME module. In hardware design, partial-SAD reuse methodology is adopted to reduce computation complexity, which means the SAD of smaller blocks are stored and accumulated to get the SAD of bigger ones.

The SAD Tree and Partial SAD architecture are proposed by [57] to realize VBSME. The SAD Tree is a 2-D intra-level architecture and consists of a 2-D PE array and one 2-D adder tree with propagation registers. Current pixels are stored in each PE, and reference pixels are stored in propagation registers for data reuse. In each cycle, $N \times N$ current and reference pixels are inputted to PEs. Simultaneously, $N$ continuous reference pixels in a row are inputted into propagation registers to update reference pixels. In propagation
registers, reference pixels are propagated in the vertical direction row by row. In SAD Tree architecture, all distortions of a searching candidate are generated in the same cycle, and by an adder tree, \( N \times N \) distortions are accumulated to derive the SAD in one cycle.

The SAD tree architecture is suitable for highly parallelized application and can share reference buffer between parallel PEG. But it has long critical path delay, which is 14.1\( \text{ns} \) based on our implementation. This delay cannot meet the performance requirement according to Section 4.2.1. To reduce delay, 16 12-bit registers can be inserted between SAD4 \( \times \) 4 and larger block’s SAD addition to form a 2-stage pipeline. When applying snake-scan processing order, one PEG (256 PE) of SAD Tree architecture needs \( 16 \times 12 + 16 \times 17 \times 8 = 2368 \) bit register.

The Partial SAD architecture is composed of \( N \) PE arrays with a 1-D adder tree in the vertical direction. Current pixels are stored in each PE, and two sets of \( N \) continuous reference pixels in a row are broadcasted to \( N \) PE arrays at the same time. In each PE array with a 1-D adder tree, \( N \) distortions are computed and summed by a 1-D adder tree to generate one-row SAD, as shown in Fig. 4.9. The row SADs are accumulated and propagated with propagation registers in the vertical direction, as shown in the right-hand side of Fig. 4.10. The reference data of searching candidates in the even and odd columns are inputted by Ref. Pixels 0 and Ref. Pixels 1, respectively. After initial cycles, the SAD of the first searching candidate in the zeroth column is generated, and the SADs of the other searching candidates are sequentially generated in the following cycles. When computing the last searching candidates in each column, the reference data of searching candidates in the next columns begin to be inputted through another reference input. Then, the hardware utilization is 100% except the initial latency. In Propagate Partial SAD, by broadcasting reference pixel rows and propagating partial-row SADs in the vertical direction, it provides the advantages of fewer reference pixel registers and a shorter critical path.

The Partial SAD architecture has smaller gate count and suitable for medium and small resolution videos. Another advantage is that it has shorter critical path delay compared
4.2. THE TOP-LEVEL ARCHITECTURE

Figure 4.9: The architecture of PE array with a 1-D adder tree.

Figure 4.10: The detailed architecture of Partial SAD.
with SAD Tree because partial SAD is stored and propagated by propagation and delay register. If one PEG (256 PE) of Partial SAD is used, it needs 1872 bit register. Therefore 496 bit register can be saved compared to SAD Tree architecture. In this paper, the Partial SAD architecture is chosen to implement IME architecture as shown in Fig. 4.8.

### 4.3 Memory Input and Output Control Unit

The memory input and output control units must achieve two primary goals: 1) avoid memory input and output confliction; 2) keep IME module to be fully utilized, which
4.3. MEMORY INPUT AND OUTPUT CONTROL UNIT

means the ME operation must has no stall. These objects are achieved by applying ping-pang strategy. That is when one SRAM is used, the other one is updated.

The proposed architecture contains four memory banks (Mem 0-3) for storage of reference pixel. Two memories are involved to perform ME of $47 \times 16$ reference pixels. Reference pixel for ME operation of each MB is stored in three memory banks. In our design, Mem 0-2 are circularly accessed when MV field is smooth; Mem 3 is used when MV field is non-smooth.

An example of memory transition is presented in Fig. 4.11 to show the generation of memory control signal. The associate FSM (Finite State Machine) is shown in Fig. 4.12. The FSM is composed of nine states, whose state labels indicate which memory to output and the output order. The decision about whether MV field is smooth is made in S01, S12 and S20, based on which next state is decided. For example, if MV field is smooth in S20, the next state is S20a. This means that Mem 2 and Mem 0 can be reused between search windows of successive two MBs. In S20a, Mem 1 must be updated since it will be accessed in next state S01. If MV field is non-smooth in S20, the next two states are S31 and S12. Mem 3 and Mem 1 is updated in S20; Mem 2 is updated in S31.

Table 4.1 shows the memory output, update and motion estimation sequence following the example shown in Fig. 4.11. In the Partial SAD architecture, $32 \times 16 + 15 = 527$ clocks are needed to process $47 \times 16$ reference pixels. 47 clocks are needed to update one memory bank. Therefore concurrent memory input and output do not lead to any memory accessing contention. For example in state S01a, Mem 2 is updated from clock 0 to clock 46 because it will be used in next state S12, while ME is performed by using Mem 0 and Mem 1 ranged from clock 0 to clock 526. Two memory banks should be updated when MV filed is non-smooth, such as in state S20 ranged from clock 1581 to 2107.

In HDTV720p application, more than 90% MB [49] just uses Mem 0-2. Therefore Mem 3 can be disabled to save power in this situation.
<table>
<thead>
<tr>
<th>Clock</th>
<th>State</th>
<th>Δmv</th>
<th>Mem 0</th>
<th>Mem 1</th>
<th>Mem 2</th>
<th>Mem 3</th>
<th>ME</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>S01a</td>
<td></td>
<td>Output</td>
<td>Output</td>
<td>Update</td>
<td></td>
<td>ME(0,1)</td>
</tr>
<tr>
<td>46</td>
<td></td>
<td></td>
<td>Output</td>
<td>Output</td>
<td>Update</td>
<td></td>
<td>ME(0,1)</td>
</tr>
<tr>
<td>526</td>
<td></td>
<td></td>
<td>Output</td>
<td>Output</td>
<td></td>
<td></td>
<td>ME(0,1)</td>
</tr>
<tr>
<td>527</td>
<td>S12</td>
<td>S</td>
<td>Output</td>
<td>Output</td>
<td></td>
<td></td>
<td>ME(1,2)</td>
</tr>
<tr>
<td>1053</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ME(1,2)</td>
</tr>
<tr>
<td>1054</td>
<td>S12a</td>
<td>Update</td>
<td>Output</td>
<td>Output</td>
<td></td>
<td></td>
<td>ME(1,2)</td>
</tr>
<tr>
<td>1100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ME(1,2)</td>
</tr>
<tr>
<td>1580</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ME(1,2)</td>
</tr>
<tr>
<td>1581</td>
<td>S20</td>
<td>NS</td>
<td>Output</td>
<td>Update</td>
<td>Output</td>
<td>Update</td>
<td>ME(2,0)</td>
</tr>
<tr>
<td>1627</td>
<td></td>
<td></td>
<td></td>
<td>Update</td>
<td>Output</td>
<td>Update</td>
<td>ME(2,0)</td>
</tr>
<tr>
<td>2107</td>
<td></td>
<td></td>
<td></td>
<td>Output</td>
<td>Output</td>
<td></td>
<td>ME(2,0)</td>
</tr>
<tr>
<td>2108</td>
<td>S31</td>
<td></td>
<td>Output</td>
<td>Update</td>
<td>Output</td>
<td></td>
<td>ME(3,1)</td>
</tr>
<tr>
<td>2154</td>
<td></td>
<td></td>
<td>Output</td>
<td>Update</td>
<td>Output</td>
<td></td>
<td>ME(3,1)</td>
</tr>
<tr>
<td>2634</td>
<td></td>
<td></td>
<td>Output</td>
<td></td>
<td></td>
<td></td>
<td>ME(3,1)</td>
</tr>
<tr>
<td>2635</td>
<td>S12</td>
<td>S</td>
<td>Output</td>
<td>Output</td>
<td></td>
<td></td>
<td>ME(1,2)</td>
</tr>
<tr>
<td>3161</td>
<td></td>
<td></td>
<td>Output</td>
<td>Output</td>
<td></td>
<td></td>
<td>ME(1,2)</td>
</tr>
<tr>
<td>3162</td>
<td>S12a</td>
<td>Update</td>
<td>Output</td>
<td>Output</td>
<td></td>
<td></td>
<td>ME(1,2)</td>
</tr>
<tr>
<td>3208</td>
<td></td>
<td></td>
<td></td>
<td>Output</td>
<td>Output</td>
<td></td>
<td>ME(1,2)</td>
</tr>
<tr>
<td>3688</td>
<td></td>
<td></td>
<td></td>
<td>Output</td>
<td>Output</td>
<td></td>
<td>ME(1,2)</td>
</tr>
<tr>
<td>3689</td>
<td>S20</td>
<td>S</td>
<td>Output</td>
<td>Output</td>
<td></td>
<td></td>
<td>ME(2,0)</td>
</tr>
<tr>
<td>4215</td>
<td></td>
<td></td>
<td>Output</td>
<td>Output</td>
<td></td>
<td></td>
<td>ME(2,0)</td>
</tr>
</tbody>
</table>

S: smooth  NS: non-smooth  ME(Memory number 1, Memory number 2)

### 4.4 Experimental Results

The proposed architecture is realized with Verilog HDL and synthesized by Synopsys Designer Compiler with TSMC 0.18µm technology. Working condition is set to the worst
(1.62V, 125°C). In the following discussion, experiment data is the simulation result from Synopsys Designer Compiler.

The implementation conditions and hardware cost of the proposed transcoding architecture are summarized in Table 4.2. The critical path delay in worst condition is 4.67ns, which meets the frequency requirement discussed in Section 4.2.1. It is observed that the proposed design can achieve 110.6MHz with 98.2K NAND gate. In Table 4.2, “Current MB” is the buffer needed to storage MB data; “Partial SAD” is the Partial SAD architecture; “Min SAD” is the hardware module to calculate minimum SAD and MV; “Control” is the the control unit; “Total” is the hardware cost of the whole architecture.

Table 4.3 shows hardware cost difference between the proposed architecture and a H.264/AVC IME module [58]. It is observed that the number of PE and search window size of transcoder IME is smaller than H.264/AVC IME. This reduction is attributed to the precise search center indicated by MPEG-2 MV in H.264 encoder end of transcoder. Thus a smaller search window and less number of PE can meet the performance requirement of HDTV application. This result proves that the MV reuse scheme applied in transcoding will introduce less hardware cost compared to H.264/AVC IME module.

<table>
<thead>
<tr>
<th>Table 4.2: VLSI Implementation Result</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Technology</strong></td>
</tr>
<tr>
<td>Operating Conditions</td>
</tr>
<tr>
<td>Frequency</td>
</tr>
<tr>
<td>Current MB</td>
</tr>
<tr>
<td>Partial SAD</td>
</tr>
<tr>
<td>Min SAD</td>
</tr>
<tr>
<td>Control</td>
</tr>
<tr>
<td>Total</td>
</tr>
</tbody>
</table>
### Table 4.3: Hardware Cost Difference

<table>
<thead>
<tr>
<th></th>
<th>Ref. [58]</th>
<th>Proposed</th>
</tr>
</thead>
<tbody>
<tr>
<td># of PE</td>
<td>128 × 8</td>
<td>256</td>
</tr>
<tr>
<td>Process</td>
<td>0.18µm</td>
<td>0.18µm</td>
</tr>
<tr>
<td>Gate Count</td>
<td>305K</td>
<td>98.2K</td>
</tr>
<tr>
<td>Frequency</td>
<td>81/108MHz</td>
<td>110.6MHz</td>
</tr>
<tr>
<td>Application</td>
<td>SDTV/HDTV720p</td>
<td>HDTV720p</td>
</tr>
<tr>
<td>Search Window</td>
<td>128 × 64</td>
<td>16 × 16</td>
</tr>
<tr>
<td># of Ref.</td>
<td>4/1</td>
<td>1</td>
</tr>
</tbody>
</table>

### 4.5 Conclusion

In this chapter, an IME architecture for MPEG-2 to H.264/AVC transcoding is proposed for HDTV720p application. To achieve the goal of bandwidth reduction, a Level C bandwidth reduction scheme for transcoding is implemented. Considering the characteristics of Level C method, a modified ping-pong memory control unit is adopted. In the proposed IO control unit, an additional memory band is added to handle the situation of non-smooth MV field. By circularly accessing memory banks, the IME module is kept fully utilize without stall. The Partial SAD architecture is adopted after the hardware cost comparison with SAD Tree architecture.

The implementation results show a reduced hardware cost compared with H.264/AVC IME module. This reduction is attributed to the MPEG-2 MV reuse in the H.264 encoder end of transcoder. MV reuse will lead to less number of PE and smaller buffer for reference data.
Chapter 5

A High-Precision Search Center
Prediction Algorithm for Transcoding

5.1 Introduction

In transcoding, motion estimation is usually not fully performed in the encoder end of transcoder because of its computational complexity. Instead, motion vectors extracted from the incoming bit stream are reused. However, this simple motion-vector reuse scheme may introduce considerable quality degradation in many applications [59, 60]. Although an optimized motion vector can be obtained by a full-scale motion estimation, this is not desirable because of its high computational complexity.

Current video compression techniques exploit mainly two types of redundancies in the uncompressed video signal to achieve the desired compression gain [61]. First, preserving only significant DCT coefficients can considerably eliminate the spatial redundancy between pixels within a single frame because of the energy compaction property of the DCT. Furthermore, the motion-compensated predictive coding scheme is used to remove the temporal redundancy between frames. In other words, a motion-compensated block in the previous reconstructed reference frame is subtracted from the current macroblock. The
residual signal is encoded using DCT to further remove the spatial redundancy. To find the motion vector for a macroblock in the current frame, a best matching macroblock is searched within a predefined search window in the previous reconstructed reference frame. The motion vector is defined as the displacement of the best matching block from the position of current macroblock. The motion estimation is performed on the luminance macroblocks and is usually based on the sum of absolute differences (SAD) of the respective pixels. A block with the minimal SAD is considered the best matching block. To obtain the motion vector for the current macroblock

\[
(I_x, I_y) = \arg \min_{(m, n) \in S} \text{SAD}_f(m, n) 
\]

\[
\text{SAD}_f(m, n) = \sum_{i}^{M} \sum_{j}^{N} |P_c^f(i, j) - R_p^f(i + m, j + n)|
\]

where \(m\) and \(n\) are the horizontal and vertical components of the displacement of a matching block, \(P_c^f(i, j)\) and \(R_p^f(i + m, j + n)\) represent a pixel in the current frame and in the previous reconstructed reference frame, respectively. The superscript \(c\) and \(p\) denotes the “current” and “previous” frame, respectively, and the subscript \(f\) is used to indicate the “front-encoder” as shown in Fig. 5.1. In the following discussion, the subscript \(s\) will be used to denote the “second-encoder” in the transcoder.

Fig. 5.2 details the structure of the cascaded transcoder. The motion estimation block is omitted for simplicity. Since the output bitrate is lower than the input bitrate, the quantization step size in Q2 in the transcoder is usually much coarser than the quantizer step size in Q1 in the front encoder.

In the transcoder, an optimized motion vector for the outgoing bit stream can be obtained by applying the motion estimation such that
5.1. INTRODUCTION

Figure 5.1: Transcoding by the cascade of a decoder and an encoder.

\[
(O_x, O_y) = \arg \min_{(m,n) \in S} \text{SAD}_s(m, n) \quad (5.1.3)
\]

\[
\text{SAD}_s(m, n) = \sum_{i}^{M} \sum_{j}^{N} |P^p_s(i,j) - R^p_s(i+m,j+n)| \quad (5.1.4)
\]

From Fig. 5.2, since the reconstructed picture in the front encoder is the same as the current input frame to the second encoder, \( R^p_f(i,j) = P^p_s(i,j) \) and \( R^c_f(i,j) = P^c_s(i,j) \). Thus,
CHAPTER 5. A HIGH-PRECISION SEARCH CENTER PREDICTION ALGORITHM FOR TRANSCODING

\[
SAD_s = \sum_{i} \sum_{j} |P_{c}^s(i, j) - R_{p}^s(i + m, j + n)|
\]

\[
= \sum_{i} \sum_{j} |P_{f}^s(i, j) - R_{p}^f(i + m, j + n) + \Delta_{c}^f(i, j) - \Delta_{p}^s(i + m, j + n)|
\]

where \(\Delta_{c}^f(i, j) = R_{c}^f(i, j) - P_{c}^f(i, j)\) and \(\Delta_{p}^s(i, j) = R_{p}^s(i, j) - P_{p}^s(i, j)\).

In Eq. 5.1.5, \(P_{f}^s(i, j) - R_{p}^f(i + m, j + n)\) is the term used to compute the SAD in the front-encoder in Eq. 5.1.1 to give the incoming motion vector. \(\Delta_{c}^f(i, j)\) represents the reconstruction error of the current frame in the front-encoder due to the quantization Q1, while \(\Delta_{p}^s(i, j)\) represents the reconstruction error of the previous frame in the second-encoder due to the quantization Q2. From Eq. 5.1.5, when the effect of the term \(\Delta_{c}^f(i, j) - \Delta_{p}^s(i + m, j + n)\) is negligible, performing a new motion estimation will give the same motion vector as the incoming motion vector (i.e., the incoming motion vector is optimal). However, since in general there is no guarantee that the effect is negligible all the time, there are nonzero probabilities that the quantization errors may cause the incoming motion vector to be non-optimal.

Although the optimized motion vector can be obtained by a new motion estimation, it is not desirable because of its high computational complexity. The reuse of the incoming motion vectors has been widely accepted because it was generally thought to be almost as good as performing a new fullscale motion estimation and was assumed in many transcoder architectures [62–64]. However, as discussed in the previous section, simply reusing the incoming motion vectors is not optimal. Our simulation results (which will be presented in the later sections) show that its performance may be considerably worse than that can be achieved with a new motion estimation. In the analysis in the previous section, it is shown that the differential reconstruction error causes incoming motion vectors to deviate from optimal values. In most macroblocks the deviation is within a small range and the position of the optimal motion vector will be near that of the incoming motion vector. Therefore, the optimal motion vector can be easily obtained by refining the incoming motion vector...
within a small range as opposed to applying a full-scale motion estimation [59, 60]. For the refinement of incoming motion vectors, we define a base motion vector \((B_x, B_y)\) as the motion vector obtained from the incoming bit stream. A delta motion vector \((D_x, D_y)\) can be estimated within a new search window \(S_R\), around the point indicated by the base motion vector:

\[
(D_x, D_y) = \arg \min_{(m,n) \in S_R} \text{SAD}_R
\]

\[
\text{SAD}_R = \sum_i \sum_j |P^{c}_{i,j} - R^{p}_i(i + B_x + m, j + B_y + n)|
\]

The new search window \(S_R\) can be set much smaller than the full-scale window \(S\) (e.g., a search range of ±2 pixels instead of 15 pixels or larger) and still produce almost the same video quality as the full-scale motion estimation.

5.2 High-Precision Search Center Prediction Algorithm for Transcoding

In hardware design, the MVP selection and MV refinement processing is difficult to implement. MVP selection involves the load of reference pixels from external memory. Because MVP from neighboring MB and MPEG-2 MV is always different from each other, reference pixels for each MVP cannot be loaded simultaneously from external memory based on memory I/O structure. Thus the parallel processing of MVP selection is impossible, which will decrease hardware efficiency. In hardware design, we need an algorithm to determine an accurate search center without any motion cost comparison; Utilizing MVP of neighboring MB and MPEG-2 MV is possible to improve the precision of search center prediction.

In the design of MPEG-2 to H.264/AVC transcoder, an other obstacle is how to generate
the motion vector of seven sub-block modes in H.264. Direction motion vector mapping from MPEG-2 motion vector to H.264 motion vector is difficult to achieve acceptable accurate motion vectors. The adopted scheme in this chapter is to perform a new motion estimation in H.264 encoder end. In that, the search center is indicated by a refined MPEG-2 motion vector.

The smoothness of motion vector field is often applied to evaluate the precision of motion vector. [65] proposes an enhanced predictive motion vector field adaptive search technique (E-PMVFAST). It points out that a smooth motion vector field can represent object motion more accurate. [66] also follows this assumption to develop a fast block-matching algorithm. [67] employs an adaptive motion vector smoothing method based the theory of vector median filter. In [68], a new motion vector field smoothing filter is proposed to aid conventional block matching algorithm (BMA) in tracking the real motion of video sequence.

The smoothness of motion vector field can also be applied in evaluation of MPEG-2 MV accuracy for transcoding. The proposed criterion is that if the coordinate difference between current and previous MPEG-2 MV is less or equal than a threshold, MPEG-2 MV field is smooth and current MPEG-2 MV is accurate. Otherwise, MPEG-2 MV field is non-smooth and current MV is inaccurate. The detailed criterion is defined as follows:

**Algorithm 3**: The decision criterion for MPEG-2 MV field smoothness.

```
1 if |x - x_{pre}| < T and |y - y_{pre}| < T then
2 MPEG-2 MV field is smooth.
3 else
4 MPEG-2 MV field is non-smooth.
5 end
```

(x, y) and (x_{pre}, y_{pre}) are coordinates of current and previous MPEG-2 MV. T is a predefined threshold which is set to 8 in our experiments on CIF sequence. And we make
assumption that the processing order of MB is raster scan.

In hardware design, the formation of MVP is different from that in software. MVP selection from multiple candidates is not acceptable, because such selection often introduces RD comparison which leads to load reference pixels several times. It will impose heavy burden on the external bandwidth, which is limited resource in hardware. For the same reason, MV refinement is also not applicable. So one-pass MV prediction algorithm is required for hardware design, which means choosing optimal MVP with only one definite algorithm step.

The MVP candidates for hardware is also different from the definition in H.264, in which MVP candidates of a sub-block is come from top, top-right and left neighboring sub-block as shown in Fig. 5.3 (a) and (b). But in parallel hardware computation architecture, the coding mode of sub-block in current MB is not available. So the MVP candidate should be modified as in [58], which uses the MVs of sub-block in neighboring MB instead of that in current MB. This is shown in Fig. 5.3 (c).

Based on previous discussion about the relationship between smoothness of MPEG-2 MV filed and its accuracy in prediction of object motion, a hardware-oriented MV prediction algorithm is proposed. The fundamental idea is that if MPEG-2 MV field is smooth, current MPEG-2 MV is considered to be accurate and selected as search center. Otherwise, the median value of MVs from sub-blocks in neighboring MBs is chosen as search center. The Detailed algorithm is defined as follows:

**Algorithm 4**: The hardware-oriented MVP algorithm.

```plaintext
1 if MPEG-2 MV field is smooth then
2   Current MPEG-2 MV is set as MVP.
3 else
4   Median(MV₀, MV₁, MV₂) is set as MVP.
5 end
```
$MV_0$, $MV_1$ and $MV_2$ is the MV of $4 \times 4$ sub-block in top-left, top and top-right MB, as shown in Fig. 5.3 (c). *Median* is a function which returns median value of the three MVs.
5.3 Experimental Results

This section presents simulation results with the proposed MV prediction method. The experimental transcoding platform is constructed with MPEG-2 reference software [14] and the H.264 reference software JM11 [24]. The MPEG-2 decoder is cascaded with H.264 encoder to form the transcoding system. The test sequence is first compressed by MPEG-2 encoder; then it is passed to transcoder. The simulation condition is shown as following.

- **MPEG-2 encoder**
  - Search range: [-32, 32).
  - Sequence format: I-P-P-P.
  - Picture type: frame picture.

- **H.264 encoder**
  - Search range: [-8, 8).
  - Sequence format: I-P-P-P.
  - MB mode: 16×16, 16×8, 8×16, 8×8, 8×4, 4×8, 4×4.

5.3.1 CDF Comparison

Firstly, the accuracy of MVP formed by the proposed MV prediction scheme is compared with MPEG-2 MV. A criterion called cumulative distribution function (CDF) [69] is used. It first calculates \( \Delta MV \) which is summation of difference between the predicted MV and the optimal MV obtained by full search for seven modes. And CDF is defined as the probability of \( \Delta MV \) less or equal than threshold \( d \). Its detailed definition is as follows:

\[
P(\Delta MV \leq d) = P \left( \bigcup_{i=1}^{7} (|MVP_{ix} - MV_{ix}| + |MVP_{iy} - MV_{iy}|) \leq d \right)
\]  

(5.3.1)
where \((MVP_{ix}, MVP_{iy})\) and \((MV_{ix}, MV_{iy})\) are coordinates of MVP and MV obtained by the full search algorithm. The \(i\) indicates block mode and \(d\) is the threshold. \(\Delta MV\) measures distance between MVP and optimal matching position for each sub-block mode. If CDF function can achieve high value with small \(d\), it means a small search range can be achieved when MVP is set as search center.

Fig. 5.4, Fig. 5.5 and Fig. 5.6 show the distribution of \(\Delta MV\) under smooth and non-smooth MPEG-2 MV field. The test data is the first one hundred frames of Football (CIF) sequence. Fig. 5.4 and Fig. 5.5 are under smooth and non-smooth MV field respectively. It is observed that under non-smooth MV field \(\Delta MV\) spread over a larger range of \(d\) compared with that under smooth MV field. This means some optimal matching positions is located far away from search center and cannot be found with a given search range. Fig. 5.6 shows the comparison of distribution under non-smooth MV field. The solid line represents distribution when MPEG-2 MV is set to search center. The doted line represents distribution when search center is produced by the proposed algorithm. It is observed that the doted line is more centralized around origin point than the other, which means more optimal position can be found with a smaller search range.

Table 5.1 shows CDF under non-smooth MV field. It is observed that the proposed algorithm can achieve higher CDF compared with the method using MPEG-2 MV as search center, which means MVP produced by the proposed algorithm is more accurate than MPEG-2 MV as search center.

### 5.3.2 Image Quality Comparison

Table ?? shows image quality comparison of two transcoding algorithm: one uses MPEG-2 MV as search center, the other uses search center produced by the proposed algorithm. The benchmark is a full search algorithm simulated by H.264 reference software, in which search range is set to \([-32, 32)\). The search range for transcoding algorithms is set to \([-8,
5.3. EXPERIMENTAL RESULTS

![Football CIF (Smooth MV field)](image1)

**Figure 5.4**: $\Delta MV$ distribution under smooth MV field.

![Football CIF (Nonsmooth MV field)](image2)

**Figure 5.5**: $\Delta MV$ distribution under non-smooth MV field.

8). The type of test sequence is CIF. The comparison method can be found in [70].

The bitrate and PSNR difference is compared. It is observed that bitrate increment
of the proposed method is smaller than that using MPEG-2 MV. In sequences with fast motion and complicate background, such as in Canoa and Stefan, the improvement is more significant. And the overall performance of the proposed method can approach the
5.3. EXPERIMENTAL RESULTS

Table 5.2: Image quality comparison.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>QP</th>
<th>PSNR (db)</th>
<th>Bitrate (Kbps)</th>
<th>ΔPSNR</th>
<th>ΔBitrate</th>
<th>ΔPSNR</th>
<th>ΔBitrate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bus</td>
<td>24</td>
<td>37.64</td>
<td>2034.92</td>
<td>-0.02</td>
<td>+2.33%</td>
<td>-0.00</td>
<td>+0.23%</td>
</tr>
<tr>
<td></td>
<td>28</td>
<td>34.66</td>
<td>1194.52</td>
<td>-0.03</td>
<td>+3.32%</td>
<td>-0.01</td>
<td>+0.57%</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>31.45</td>
<td>660.24</td>
<td>-0.06</td>
<td>+4.47%</td>
<td>-0.02</td>
<td>+1.14%</td>
</tr>
<tr>
<td></td>
<td>36</td>
<td>28.66</td>
<td>361.25</td>
<td>-0.11</td>
<td>+5.85%</td>
<td>-0.06</td>
<td>+1.63%</td>
</tr>
<tr>
<td>Canoa</td>
<td>24</td>
<td>38.08</td>
<td>2908.88</td>
<td>-0.06</td>
<td>+25.95%</td>
<td>-0.01</td>
<td>+0.82%</td>
</tr>
<tr>
<td></td>
<td>28</td>
<td>35.05</td>
<td>1912.00</td>
<td>-0.05</td>
<td>+29.56%</td>
<td>-0.00</td>
<td>+0.55%</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>31.95</td>
<td>1179.95</td>
<td>-0.02</td>
<td>+33.83%</td>
<td>-0.01</td>
<td>+0.13%</td>
</tr>
<tr>
<td></td>
<td>36</td>
<td>29.07</td>
<td>672.05</td>
<td>-0.06</td>
<td>+41.04%</td>
<td>-0.02</td>
<td>+0.89%</td>
</tr>
<tr>
<td>Football</td>
<td>24</td>
<td>39.18</td>
<td>2304.08</td>
<td>-0.00</td>
<td>+2.43%</td>
<td>-0.03</td>
<td>+1.03%</td>
</tr>
<tr>
<td></td>
<td>28</td>
<td>36.40</td>
<td>1490.84</td>
<td>-0.03</td>
<td>+3.59%</td>
<td>-0.05</td>
<td>+2.05%</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>33.70</td>
<td>928.33</td>
<td>-0.03</td>
<td>+5.39%</td>
<td>-0.11</td>
<td>+2.72%</td>
</tr>
<tr>
<td></td>
<td>36</td>
<td>31.28</td>
<td>558.32</td>
<td>-0.02</td>
<td>+6.82%</td>
<td>-0.12</td>
<td>+3.68%</td>
</tr>
<tr>
<td>Foreman</td>
<td>24</td>
<td>39.02</td>
<td>757.64</td>
<td>-0.04</td>
<td>+2.43%</td>
<td>-0.03</td>
<td>+1.03%</td>
</tr>
<tr>
<td></td>
<td>28</td>
<td>36.57</td>
<td>402.54</td>
<td>-0.06</td>
<td>+3.59%</td>
<td>-0.05</td>
<td>+2.05%</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>34.19</td>
<td>226.01</td>
<td>-0.09</td>
<td>+5.39%</td>
<td>-0.11</td>
<td>+2.72%</td>
</tr>
<tr>
<td></td>
<td>36</td>
<td>31.96</td>
<td>136.25</td>
<td>-0.12</td>
<td>+6.81%</td>
<td>-0.12</td>
<td>+3.68%</td>
</tr>
<tr>
<td>Husky</td>
<td>24</td>
<td>36.13</td>
<td>6613.19</td>
<td>+0.02</td>
<td>+0.88%</td>
<td>+0.01</td>
<td>+0.06%</td>
</tr>
<tr>
<td></td>
<td>28</td>
<td>32.68</td>
<td>4648.14</td>
<td>+0.01</td>
<td>+1.24%</td>
<td>+0.00</td>
<td>+0.17%</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>29.10</td>
<td>2946.90</td>
<td>-0.01</td>
<td>+1.76%</td>
<td>-0.01</td>
<td>+0.32%</td>
</tr>
<tr>
<td></td>
<td>36</td>
<td>25.67</td>
<td>1664.61</td>
<td>-0.02</td>
<td>+2.26%</td>
<td>-0.01</td>
<td>+0.52%</td>
</tr>
</tbody>
</table>

full search algorithm. It is also observed that bitrate increment for sequence with fast motion and complicated background is still higher than other sequence.
### Table 5.3: Image quality comparison (cont.).

<table>
<thead>
<tr>
<th>Sequence</th>
<th>QP</th>
<th>PSNR (db)</th>
<th>Bitrate (Kbps)</th>
<th>∆PSNR</th>
<th>∆Bitrate</th>
<th>∆PSNR</th>
<th>∆Bitrate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stefan</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>38.54</td>
<td>2120.56</td>
<td></td>
<td>-0.05</td>
<td>+21.40%</td>
<td>-0.01</td>
<td>+3.96%</td>
</tr>
<tr>
<td>28</td>
<td>35.44</td>
<td>1174.90</td>
<td></td>
<td>-0.09</td>
<td>+31.07%</td>
<td>-0.03</td>
<td>+8.28%</td>
</tr>
<tr>
<td>32</td>
<td>32.13</td>
<td>579.00</td>
<td></td>
<td>-0.09</td>
<td>+46.62%</td>
<td>-0.03</td>
<td>+1.23%</td>
</tr>
<tr>
<td>36</td>
<td>29.15</td>
<td>300.65</td>
<td></td>
<td>-0.12</td>
<td>+60.78%</td>
<td>-0.04</td>
<td>+2.56%</td>
</tr>
<tr>
<td>Table</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>38.31</td>
<td>1325.79</td>
<td></td>
<td>-0.04</td>
<td>+5.47%</td>
<td>-0.02</td>
<td>+1.77%</td>
</tr>
<tr>
<td>28</td>
<td>35.48</td>
<td>721.48</td>
<td></td>
<td>-0.07</td>
<td>+7.73%</td>
<td>-0.04</td>
<td>+2.78%</td>
</tr>
<tr>
<td>32</td>
<td>32.79</td>
<td>377.41</td>
<td></td>
<td>-0.09</td>
<td>+9.95%</td>
<td>-0.06</td>
<td>+3.67%</td>
</tr>
<tr>
<td>36</td>
<td>30.60</td>
<td>194.18</td>
<td></td>
<td>-0.09</td>
<td>+12.99%</td>
<td>-0.06</td>
<td>+5.43%</td>
</tr>
</tbody>
</table>

### 5.4 Conclusion

In this chapter, a motion vector prediction algorithm for transcoding is proposed based on smoothness of MPEG-2 MV field. The proposed method is characterized as the decision of smoothness of MV field. If the MPEG-2 MV field is smooth, current MPEG-2 MV is reused as MVP; otherwise the median MV from sub-block of neighboring MB is set as MVP.

Experiment results show that the proposed method can approach the performance of full search. The performance improvement is significant compared with the method only using MPEG-2 MV, especially in sequences with fast motion and complicate background. The proposed motion vector prediction scheme can also be applied to software algorithm. The accuracy evaluation procedure of MPEG-2 MV can replace the MVP selection step. The computation for this part will be saved.

It is observed from experiment data that motion prediction accuracy for sequence with fast motion and complicate background is still lower than other sequence. Our future work
will involve improvement on this type of sequence.
CHAPTER 5. A HIGH-PRECISION SEARCH CENTER PREDICTION ALGORITHM FOR TRANSCODING
Chapter 6

Conclusion

6.1 Summary of results

This dissertation addressed the bandwidth reduction problem in MPEG-2 to H.264/AVC transcoding. Data reuse schemes play a critical role for bandwidth reduction in video coding system design. In MPEG-2 to H.264/AVC transcoding, the main difficulty of applying data reuse methods comes from the irregular search center position introduced by MPEG-2 motion vector. In this dissertation, two bandwidth reduction schemes, Level C and Level C+, are proposed for MPEG-2 to H.264/AVC transcoding, in which different levels of bandwidth can be achieved after addressed the irregularity problem. The hardware architecture of the proposed Level C scheme is also proposed. In addition, a hardware-oriented motion vector prediction method is proposed for MPEG-2 to H.264/AVC transcoding to improve the prediction accuracy.

By utilizing the similarity between successive MV, the proposed Level C scheme for transcoding can regularize the overlapping area and then search window reuse is available for most MB. Experimental results show that the bandwidth level is able to be reduced to 40.6% of that without any bandwidth reduction scheme. This level is almost equal to standard H.264 IME with Level C scheme. But the on-chip is much smaller (only 12.7%
CHAPTER 6. CONCLUSION

(2 MB) and 10.1% (4 MB) of standard H.264 IME). An IME architecture is proposed to implement the Level C scheme. The modified ping-pong memory control logic is the key to realized it. The synthesis results shows an signification reduction in hardware cost compared to regular H.264 IME module. And the cost to achieve the same bandwidth level is much smaller.

An ultra-low bandwidth design method, called Level C+ scheme, is also proposed for MPEG-2 to H.264/AVC transcoding. The proposed method is designed for applications with strictly limited bandwidth resource, while the size of on-chip memory is not a main design constraining factor. It makes a tradeoff between bandwidth and on-chip memory to achieve ultra-low bandwidth. In this method, motion estimation is performed in the unit of MEU. Search window position of MEU is determined by average MV of each MB in MEU. Motion vector clipping is applied to confine the search window of each MB within the search window of MEU. Through these processing, both horizontal and vertical search window reuse is available for each MB in MEU. Experiment results show that the proposed Level C+ method can achieve a very low external bandwidth \((R_a < 2)\) with an acceptable on-chip memory.

A motion vector prediction algorithm for transcoding is proposed based on smoothness of MPEG-2 MV field. The smoothness of MPEG-2 MV field is utilized as a criterion to evaluate the precision of MPEG-2 MV. Experiment results show that the proposed method can approach the performance of full search. The performance improvement is significant compared with the method just using MPEG-2 MV as search center, especially in sequences with fast motion and complicate background. The proposed motion vector prediction scheme can also be applied to software algorithm. The accuracy evaluation procedure of MPEG-2 MV can replace the MVP selection step. The computation for this part will be saved.
6.2 Future work

In this dissertation, the bandwidth reduction schemes and its associated hardware architecture are shown for MPEG-2 to H.264/AVC transcoding. The possibility of improving motion vector prediction accuracy is also discussed. Further reducing the irregular memory access, especially in external memory, is an interesting problem. This improvement will reduce the power dissipation the whole system and improve the memory I/O efficiency.

Frequency domain transcoding further reduce the complexity of the whole system because the DCT/IDCT is saved. Motion estimation and motion compensation are completely performed in frequency domain. In MPEG-2 to H.264/AVC transcoding, the DCT coefficient transformation and frequency domain motion estimation are interesting research topic. In addition, its associated hardware design will attract more attention in further research.
Acknowledgment

First and foremost, I would like to appreciate Professor Satoshi Goto of Waseda University for his constant encouragement, guidance and support during my research. He advised me enthusiastically to aim a world’s highest research goal. Working on my research with him must be the most invaluable asset in my life. I also express my appreciation to Professor Takeshi Ikenaga and Professor Shinji Kimura for their teaching and guidance in hardware design technologies.

I also want to express my thanks to Mr. Shen Li, Mr. Yang Song, Mr. Zhen-yu Liu and Mr. Ling-Feng Li for their insightful discussion and the inspiration got from them. At the same time, I should thanks the other members of Professor Goto’s lab for their help and support during the whole procedure of my research.

I also thanks funds from the Ministry of Education, Culture, Sports, Science and Technology (MEXT) via Kitakyushu innovative cluster project, “Ambient SoC Global COE Program of Waseda University” and CREST project.

Finally, I thank my families for their kind support over the years.
List of Publications

Journal Paper


International Conference Paper


**Award**

1. Okawa Information and Telecommunication Award, Waseda University 2007.

Bibliography


BIBLIOGRAPHY


107
BIBLIOGRAPHY


