國 立 交 通 大 學

電子工程學系 電子研究所

博 士 論 文

適用於 H.264/MPEG4-AVC 及其可調式視訊編碼之高效頻寬移動估測研究

The Study of Bandwidth Efficient Motion Estimation for H.264/MPEG4-AVC Video Coding and Its Scalable Extension

研 究 生：李國龍

指導教授：張添烜教授

陳美娟教授

中 華 民 國 一百 年 八 月

適用於 H.264/MPEG4-AVC 及其可調式視訊編碼之高效頻寬移動估測研究

The Study of Bandwidth Efficient Motion Estimation for H.264/MPEG4-AVC Video Coding and Its Scalable Extension

研 究 生：李國龍　　Student：Gwo-Long Li

指導教授：張添烜　　Advisor：Tian-Sheuan Chang
陳美娟　　Mei-Juan Chen

國 立 交 通 大 學

電子工程學系 電子研究所

博 士 論 文

A Dissertation
Submitted to Department of Electronics Engineering and
Institute of Electronics
College of Electrical and Computer Engineering
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in
Electronics Engineering

August 2011
Hsinchu, Taiwan, Republic of China

中華民國一百年八月

適用於 H.264/MPEG4-AVC 及其可調式視訊編碼之高效頻寬移動估測研究

研究生：李國龍　　　　指導教授：張添烜　博士
陳美娟　博士

國　立　交　通　大　學
電機學院　電子工程學系　電子研究所

摘　要

移動估計演算法的高計算複雜度與資料頻寬存取量，一直以來是視訊編碼研究領域中非常重要的研究議題。然而，隨著高解析度視訊影像應用需求的增加，使得移動估計演算法的高資料頻寬存取量，成為影響整體視訊編解碼器效能的重要關鍵。此外，為了達到應用可調性，可調式視訊編碼所採用的層間預測模式，亦為移動估計帶來更高之資料頻寬存取量及計算複雜度。為了解決視訊編碼之高資料頻寬存取量所造成編碼系統效能下降之問題，本論文提出若干適用於整數點及分數點移動估計演算法之資料存取及計算量減少演算法。

針對整數點移動估計資料頻寬之問題，本論文提出一位元失真率最佳化的頻寬有效率之移動估計演算法。在此演算法中，本論文提出一數學模型以描述位元失真率與頻寬之關係。藉由此模型的建立，進而發展出一低頻寬需求之移動估計演算法。此外，經由此數學模型的幫助，本論文亦發展出一頻寬感知移動估計演算法。此演算法可以在頻寬有限的條件之下，適當的分配頻寬資源給移動估計演算法，進而達到更佳的失真位元率表現。透過本論文所發展之方法，可以達到 78.82%頻寬節省。

在可調式視訊編碼中，由於其額外採用的層間預測方式，使得原本整數點移動估計之高頻寬存取所帶來的問題更加嚴重。因此，本論文提出若干個適用於可調式視訊編碼之高效頻寬層間預測移動估計演算法。透過利用畫面層間之高相關性，達到較高的資料

共用性進而減少多餘的資料存取。實驗結果顯示，本論文提出之適用於可調式視訊編碼之高效頻寬層間預測演算法，可至少達到 50.55%的資料頻寬之節省。

除了可調式視訊編碼中額外採用的層間預測方法所帶的高資料頻寬存取量之問題外，分數型移動估計演算法的高計算複雜度，亦大幅度的造成計算複雜度的增加進而影響可調式視訊編碼器的系統效率。因此，本論文提出一分數型移動估計模式預先選擇演算法，以預先過濾掉潛在可忽略之預測模式。藉由觀察不同模式之整數型移動估計成本及分數型移動估計成本之間之關係，進而提出若干個模式過濾機制。透過本論文提出之演算法，在不造成太多位元失真率效能失真的情況之下，平均可達到 65.97%模式減少。

對於分數型移動估計演算法因硬體設計考量，導致參考資料未被從外部記憶體下載進來所造成的效能下降而言，本論文提出了搜尋範圍重新決定演算法用以減少視訊編碼器之效能下降。在此演算法中，本論文透過觀察移動向量子與非重疊區域尺寸之關係，提出一數學公式予以描述。因此，透過移動向量子的大小，即可計算出非重疊區域尺寸，進而重新計算所需之搜尋範圍尺寸。此外，本論文亦提出一搜尋範圍長寬比決定演算法，透過移動向量子與非重疊區域尺寸關係之數學求解，可得到較佳之非對稱式搜尋範圍長寬比。經由本論文所提出之搜尋範圍與長寬比決定之演算法，可達到 90%的位元率下降改善。

整體而言，透過本論文所提之演算法，除了可大幅度的減少資料頻寬之存取外，亦可減少整數及分數點的計算複雜度，進而達到更佳視訊編碼效能之改善。

The Study of Bandwidth Efficient Motion Estimation for H.264/MPEG4-AVC Video Coding and Its Scalable Extension

Student：Gwo-Long Li　　Advisors：Dr. Tian-Sheuan Chang
Dr. Mei-Juan Chen

Department of Electronics Engineering
Institute of Electronics
National Chiao Tung University

ABSTRACT

In the video coding system, the overall system performance is dominated by the motion estimation module due to its high computational complexity and memory bandwidth intensive data accesses. Furthermore, with the increasing demands of high definition TV, the system performance drop caused by the intensive data bandwidth access requirement becomes even more significant. In addition, the additional adopted Inter-layer prediction modes of scalable video coding also significant increase the data access bandwidth overhead and computational complexity. To solve the high computation complexity and intensive data bandwidth access problems, this dissertation proposes several data access bandwidth and computational complexity reduction algorithms for both of integer and fractional motion estimation.

First, this dissertation proposes a rate distortion bandwidth efficient motion estimation algorithm to reduce the data bandwidth requirements in integer motion estimation. In this algorithm, a mathematical model is proposed to describe the relationship between rate distortion cost and data bandwidth. Through the modeling results, a data bandwidth efficient motion estimation algorithm is thus proposed. In addition, a bandwidth aware motion estimation algorithm based on the modeling results is also proposed to efficiently allocate the data bandwidth for motion estimation under the available bandwidth constraint. Simulation results show that our proposed algorithm can achieve 78.82% data bandwidth saving.

In scalable video coding standard, the additional included Inter-layer prediction modes significantly deteriorate the video system coding performance since much more data have to be accessed for the prediction purpose. Therefore, this dissertation proposes several data efficient Inter-layer prediction algorithms to lighten the intensive data bandwidth requirement problem in scalable video coding. By observing the relationship between spatial layers, several data reusing algorithms have been proposed and thus achieve more data bandwidth requirement reduction. Simulation results demonstrate that our proposed algorithm can achieve 50.55% data bandwidth reduction at least.

In addition to the system performance degradation caused by intensive data bandwidth access problem, the high computational complexity of fractional motion estimation also noticeably increases the system performance drop in scalable video coding. Therefore, this dissertation proposes a mode pre-selection algorithm for fractional motion estimation in scalable video coding. In our proposed algorithm, the rate distortion cost relationship between different prediction modes are observed and analyzed first. Based on the observing and analytical results, several mode pre-selection rules are proposed to filter out the potentially skippable prediction modes. Simulation results provide that our proposed mode pre-selection algorithm can reduce 65.97% prediction modes with ignorable rate distortion performance degradation.

Finally, for the video coding system performance drop problem caused by the fractional motion estimation process skipping due to hardware implementation consideration, this dissertation proposes a search range adjust algorithm to adjust the search range for the motion estimation so that the new decided search range can cover the absent reference data as much as possible for fractional motion estimation. By mathematically modeling the relationship between motion vector predictor and non-overlapping area size, the new search range can thus be adjusted. In addition, a search range aspect ratio adjust algorithm is also proposed in this dissertation by means of solving the mathematical equations. Through the proposed search range adjust algorithm, up to 90.56% of bitrate increasing can be reduced when compared to fractional motion estimation skipping mechanism. Furthermore, the proposed search range aspect ratio adjust algorithm can achieve better rate distortion performance when compared to the exhaustive search method under the same search range area constraint.

In summary, through the algorithms proposed in this dissertation, not only the data access bandwidth but the computational complexity of integer and fractional motion estimation can be reduced and thus improve the overall video coding system performance significantly.

Outline

List of Tables

List of Figures

Chapter 1

Introduction

1.1. Introduction

With rapid development of information technology, the information technology products have been widely used in our daily life. In the early information technology era, the information communication is just happened in text form. However, thanks the contribution of advanced information technology, our communication behavior has been significantly changed. In nowadays, the popularity of multimedia communication allows us to send the multimedia information including text, voice, picture, and video to our friends through electronic devices. As a result, our friends can real-time obtain our latest visual information wherever they are.

Videos, a form of visual information, are composed of a number of successive still images [1]. By rapid displaying the consecutive still images, the effect of visual illusion originally existed in our human visual system will leaded us to see a moving picture [1]. However, from the ingredients of video, we can find that the video is composed of many still images and each image is composed of a number of basic elements called pixels. For gray images, each pixel only requires one byte to represent it. However, for the color images, each pixel needs several bytes to represent depending on the color space they used. Quantitatively, for a video sequence which has the specifications of 352×288 image resolution, RGB color space, and 30fps frame rate. It consumes 352×288×3(RGB)×30=9123940Bytes=9810Kbytes data bandwidth per second. However, for the larger image resolution, the data bandwidth issue becomes much more difficult to deal with [1].

To deliver the video information through the bandwidth limited transmission channel, the video coding technique has been developed in the decades to reduce the data amounts of video signal and thus lighten the overheads of transmitting noticeable amounts of video signal. As a result, many video coding standards [2]-[8] have been standardized to unify the encoded video signal format in order to increase the compatibility between the video application products of different companies. In these video coding standards, the state-of-the-art video coding standard called H.264/AVC [7] standardized by Joint Video Team (JVT) can achieve unbelievable video compression ratio when compared to previous video coding standards due to the adoption of a number of advanced optimization techniques. However, the adoption of the advanced technologies also significantly results in the design difficulty and implementation cost increasing [8]. In addition, to support application diversities, a video coding standard called Scalable Video Coding (SVC), as an extension of H.264/AVC, has been standardized recently to satisfy the requirement of end user heterogeneities. As a result, not only the benefit of outstanding compression ratio of H.264/AVC has been inherited but the significant computational complexities have been also accepted without question. Therefore, this dissertation focuses on the issues of high computational complexity and data access bandwidth of H.264/AVC and SVC. In the following subsections, the H.264/AVC and SVC will be introduced briefly and the issues we faced will also been described in detail.

1.2. Introduction of H.264/AVC

Fig. 1-1 shows the overall architecture of H.264/AVC and the detailed explanations can be found from [9]. The operation of H.264/AVC is briefly described as follows. First, the input image is decomposed into the non-overlapping basic coding unit called macroblock (MB) with the size of 16×16. For the first incoming image, it usually has been recognized as Intra (I) prediction frame and the *Intra Prediction* is applied for the intra frame to obtain the intra prediction results. In H.264/AVC, three block sizes of 16×16, 8×8, and 4×4 have been adopted in intra prediction mode and each block size individually supports four, nine, and nine prediction modes from different prediction angles. Afterwards, the prediction results will be subtracted from the current MB to derive prediction errors (residuals) and the derived prediction errors will be fed into *Transform* and *Quantization* to obtain quantized coefficients that will be used to generate coded bitstream. In H.264/AVC, two entropy coding approaches called Context-Adaptive Variable Length Coding (*CAVLC*) and Context-Adaptive Binary Arithmetic Coding (*CABAC*) have been adopted to achieve entropy coding. Simultaneously, the quantized coefficients will be inverse quantized and transformed to derive the lossy prediction errors. The obtained lossy prediction errors will be therefore added to prediction pixels to get the reconstructed images. Hence, the reconstructed images will be deblocking filtered to remove the blocking effects and the filtered images will be stored into *Reconstructed Frame Buffer* for the following prediction usage. Once all MBs of the first incoming frame (I frame) have finished the coding process, the second incoming frame will be recognized as inter prediction frame (P or B frame) and both of intra and *Inter Prediction* operations will be applied for the inter frame. Like the intra frame, the intra prediction in inter frame is the same as the intra prediction in I frame. However, the Inter prediction should be additional applied for inter frame. In the *Motion Estimation*, it consists of Inter Motion Estimation (IME) and Fractional Motion Estimation (FME). For IME, H.264/AVC supports seven block size motion estimation including the size of 16×16, 16×8, 8×16, 8×8, 8×4, 4×8, and 4×4. In addition, up to five reference frames can be searched in the motion estimation process in H.264/AVC. Literature [10] reported that 0.5dB to 1dB PSNR improvement can be achieved by searching five reference frames. For FME, the half-pixel position motion estimation is first applied to find out the best outcome surrounding the best result of IME. Afterwards, the quarter-pixel position motion estimation is executed to obtain the final predictions surrounding the best result of half-pixel position. Once the best results of intra and Inter prediction have been decided, the prediction mode with smallest rate distortion cost will be selected as the best result. Henceforth, the reconstruction process is the same as the intra prediction except that the prediction mode and motion vectors should be involved in the inter frame reconstruction process.

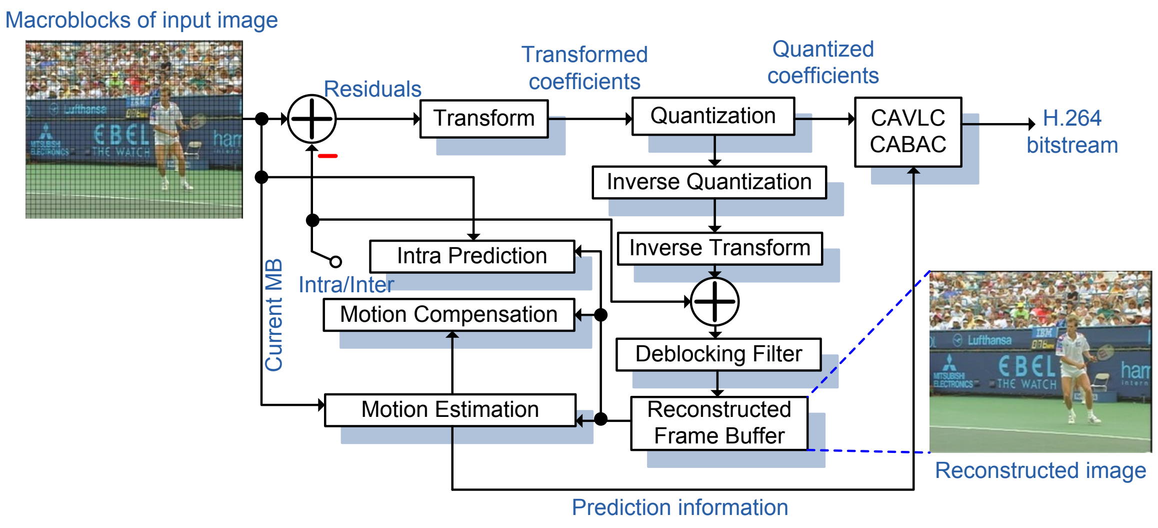


Fig. 1-1 Overall architecture of H.264/AVC

1.3. Introduction of H.264/AVC Scalable Extension (SVC)

To satisfy the application diversities, the SVC can achieve image resolution, frame rate, and bitrate adaptation by using the spatial, temporal and quality scalability. The spatial scalability is aimed at by coding several different image resolutions and the temporal scalability is achieved by hierarchical B frame concept. For quality scalability, two approaches called coarse grain scalability (CGS) and medium grain scalability (MGS) have been adopted to support the bitrate adaptation. However, as mentioned early that the SVC is an extension of H.264/AVC. The SVC inherits all computational modules from H.264/AVC including its coding efficiency as well as coding complexity. Fig. 1-2 exhibits an overall architecture of SVC with three spatial layers. However, the detailed explanations for SVC can be found in [11]. To achieve H.264/AVC compatibility, the SVC spatial base layer is completely composited of all H.264/AVC computations and therefore the SVC spatial base layer is also called H.264/AVC bitstream. However, to fully exploit the relationship between successive spatial layers, the SVC additional supports Inter-layer prediction (ILM) to achieve better rate distortion performance. The operation of SVC is briefly described as follows. At the begging, the higher resolution input image is scaled down into several lower resolution images depending on how many spatial layers that the application needed. Afterwards, the lowest resolution image is firs encoded by H.264/AVC compatible SVC encoder to generate the bitstream of SVC spatial base layer. Once all MBs have been encoded in spatial base layer, the fine resolution image is encoded by SVC spatial enhancement layer coding approach. Similar to the operations defined in H.264/AVC, the SVC spatial enhancement layer coding approach executes all operations specified in H.264/AVC except the additional operations of Inter-layer predictions and all operations required for generating quality scalability. To support Inter-layer prediction in SVC enhancement layer, some information including texture, motion information, and residuals are scaled up to the target size according to the image resolution ratio between two consecutive spatial layers. In general, the dyadic spatial layer relationship is commonly used due to its implementation simplicity. However, for non-dyadic spatial layer relationship situation, the extended spatial scalability (ESS) [12] should be applied to derive the spatial layer relationship with the expense of complex computations. Once all required images have been successfully encoded, a multiplexer is used to pack all generated bitstreams from different spatial layers to derive a single SVC bitstream.

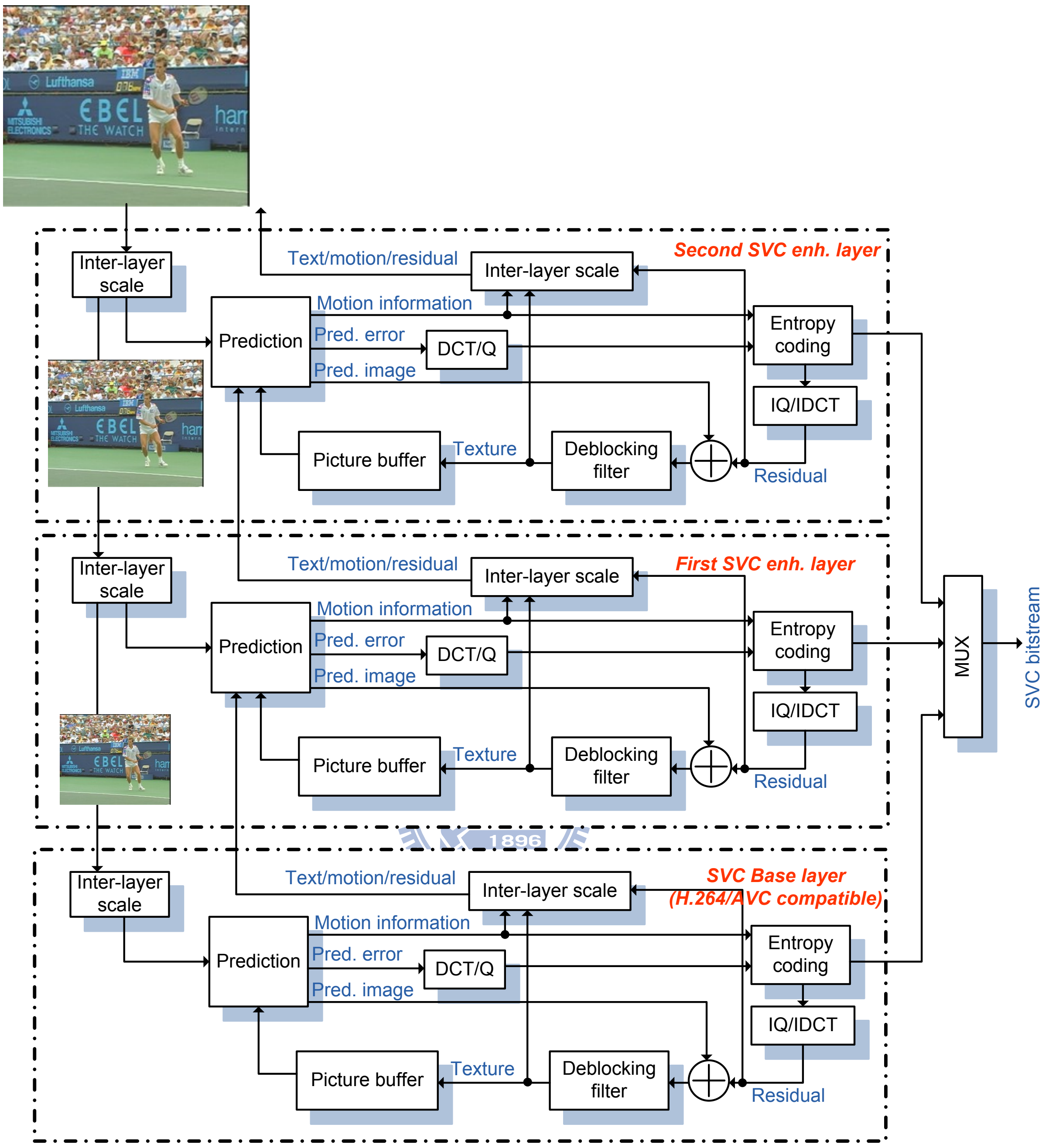


Fig. 1-2 Overall architecture of SVC with three spatial layers

1.4. Analysis for H.264/AVC and Its Scalable Extension

In this subsection, we will first analyze the computational complexity and data access bandwidth requirements for H.264/AVC and SVC encoders and try to find out the most computation and data access intensive parts which dominate the overall video coding system performance. If the system performance dominating parts can be found, we can use as much as effort to optimize them and thus to increase the video coding system performance.

1.4.1 Computational Complexity Analysis for H.264/AVC

As mentioned above that the H.264/AVC is organized by the hybrid motion compensated DCT encoding scheme. Therefore, the main functions can be classified as motion estimation (*ME*), motion compensation (*MC*), transform and inverse transform (*DCT/IDCT*), quantization and inverse quantization (*Q/IQ*), entropy coding (*Entropy*), interpolation for fractional motion estimation (*Interpolation*), rate control (*RC*), reconstruction (*REC*), and other terms such intra prediction (*Other*). Fig. 1-3 shows the CPU occupancy of H.264/AVC main functions [13]. From this figure, we can observe that the *ME* and *MC* occupy 41% of CPU usage in total. However, since theu *Interpolation* is also used during the motion estimation process to derive the fractional motion estimation results, the portion of *Interpolation* should be also included into *ME* and *MC*. Therefore, the total portion of 54% is occupied by the *ME*, *MC*, and *Interpolation*.

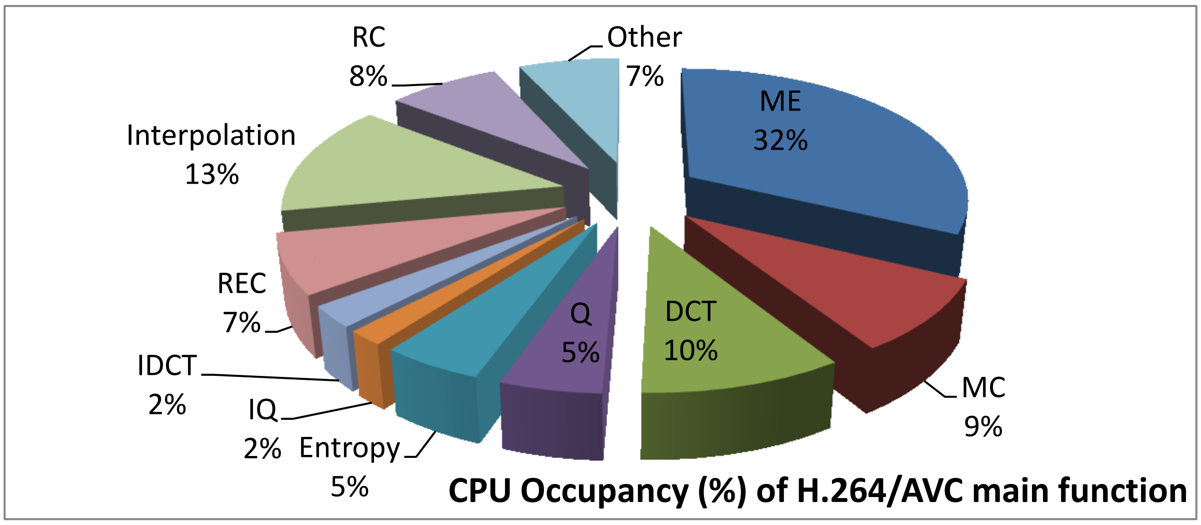


Fig. 1-3 CPU occupancy of H.264/AVC main function

1.4.2 Memory Access Requirement Analysis for H.264/AVC

Fig. 1-4 exhibits the memory access requirements of H.264/AVC [14]. From this figure, it can be seen that the *IME* part dominates up to 77.54% memory access bandwidth. However, if the *FME* part is further included, up to 98.51% memory access bandwidth will be occupied by the Inter prediction.

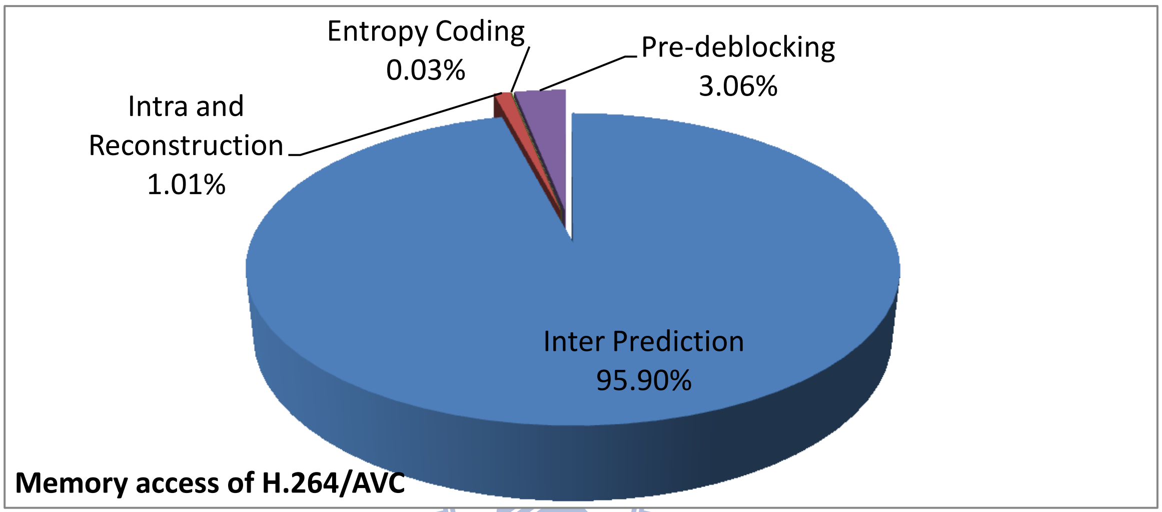


Fig. 1-4 Memory access occupation of H.264/AVC

From the above analyses, we can conclude that the Inter prediction including *IME* and *FME* is the system performance dominating part. Therefore, we can put a lot of efforts on optimizing the performance of Inter prediction to improve the overall video coding system performance.

1.4.3 Memory Access Requirement Analysis for SVC

Fig. 1-5 shows the percentage of external memory bandwidth requirements of SVC analyzed in [15]. In this figure, the components of all operations are roughly classified into *Pre-deblocking filter*, *Entropy coding*, *Intra and Reconstruction*, and *Inter and Inter-layer prediction* parts. From this figure, it can be seen that the *Inter and Inter-layer prediction* part occupies more than 67% of external memory access in SVC.

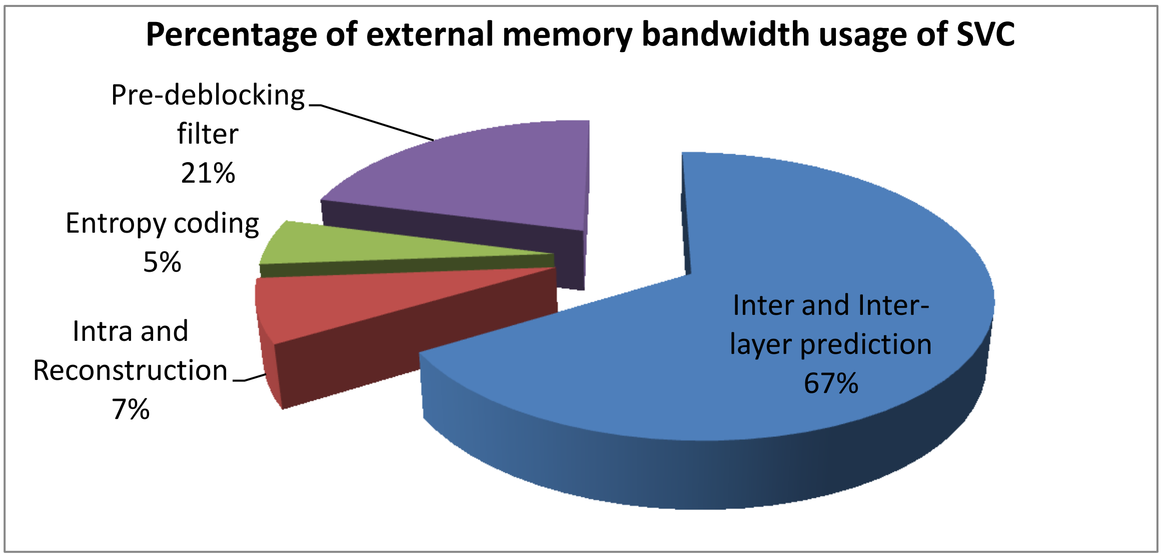


Fig. 1-5 Percentage of external memory usage of each component in SVC

1.4.4 Problem Statement

As analyzed before that the most computation and memory access intensive portions of overall video coding system are the Inter prediction parts including IME and FME. However, with the rapid demands of high definition TV, the problem of video coding system performance will be getting bad since much more video data need to be processed. Fortunately, thanks the advancing of VLSI technology, the computation speed can be increased significantly and thus lighten the coding speed problem. Fig. 1-6 reveals the CPU performance improvement year by year. From this figure, it is very obvious that the CPU performance is linearly improved year by year. This performance improvement of CPU implies that the computation will be the problem no more since we can use the devices with higher computation speed to accelerate the computation and thus increase the overall video coding system performance. However, unfortunately, the memory performance improvement seems not good sufficiently as Fig. 1-6 shown. From this figure, we can find that the memory performance improve is increased very slightly year by year due to the growth of memory spaces to support data intensive applications. Fig. 1-7 shows the memory bandwidth requirements of motion estimation part for different resolution video sequences. The horizontal axis indicates the frame resolution and the numbers inside the brackets are the search range size. In addition, the vertical axis is the memory bandwidth requirements in the unit of mega bytes per second. This figure is resulted by using 30fps frame rate and Level C data reuse full search motion estimation scheme [16]. From this figure, it can be seen that the memory bandwidth requirements of motion estimation are significantly grew up with the increase of frame resolution.

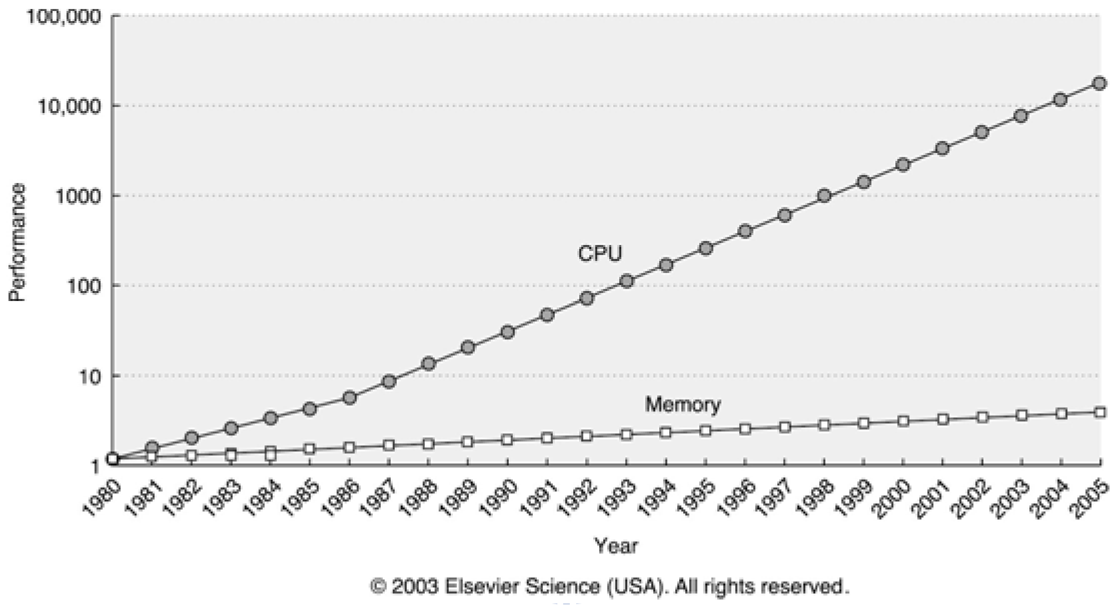


Fig. 1-6 CPU and memory performance gap comparison

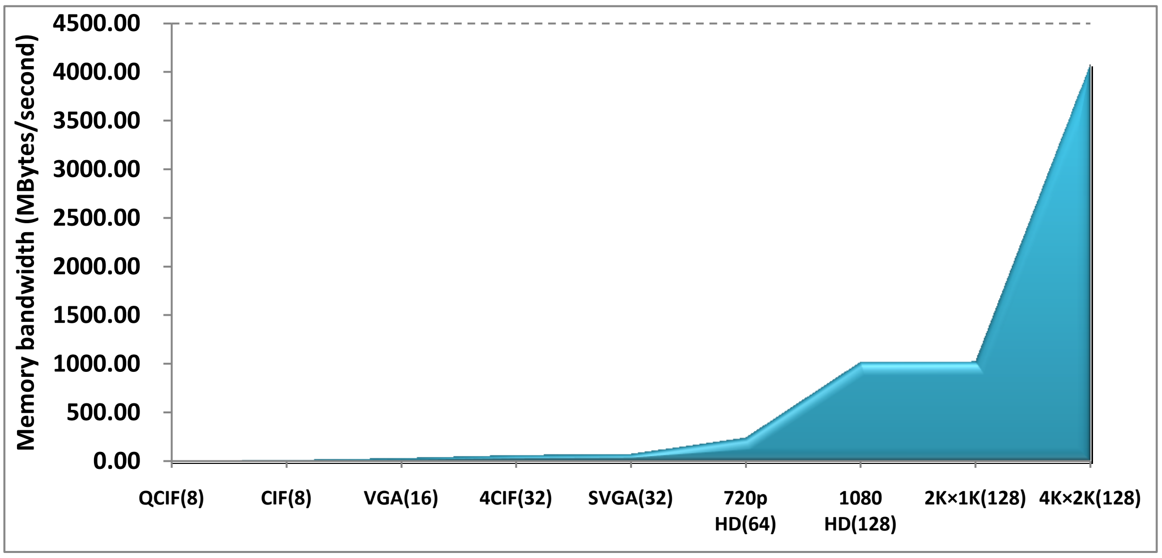


Fig. 1-7 Memory bandwidth requirements of different video resolution

From above analyses, we know that the memory performance hasn't been improved efficiently and the memory bandwidth requirements have been increased significantly. Therefore, although the computation speed improvement can release the burdens of heavy computations of video coding system, the intensive memory bandwidth issue will finally become the most critical part in video coding system and thus significant affect the overall video coding performance. Therefore, this dissertation pays the most attentions on the memory bandwidth issues. In addition, some optimization techniques are also proposed in this dissertation to further improve the coding speed of video coding systems.

1.5. Organization of this Dissertation

This dissertation is organized as follows. In Chapter 2, we propose a rate distortion bandwidth efficient motion estimation algorithm to solve the intensive data access problem of motion estimation. In addition, a bandwidth aware motion estimation algorithm is also proposed in this chapter to efficiently allocate the memory bandwidth for motion estimation according to the available bandwidth constraints. For SVC, some data efficient Inter-layer prediction algorithms have been proposed in Chapter 3 to increase the data reusability between spatial layers and thus increase the SVC system performance. In Chapter 4, a mode pre-selection algorithm is proposed to skip the potentially ignorable prediction modes before entering the fractional motion estimation so that the heavy computational complexity of fractional motion estimation can be released. In addition, to further increase the motion estimation performance, a search range adjust algorithm and search range aspect ratio decision algorithm with considering both of IME and FME information is proposed in Chapter 5 to improve the coding performance of motion estimation. Finally, some conclusions and future works are given in Chapter 6.

Chapter 2

RD Bandwidth Efficient Motion Estimation and Its Hardware Design with On-Demand Data Access

2.1. Introduction

Motion estimation (ME) not only contributes to the most of coding efficiency but also is the most computational intensive as well as data bandwidth intensive component in modern video encoding design [9]. ME finds out the best matching block by matching its current coding macroblock (MB) (a block with 16×16 pixels) with search candidates within the search range. A direct approach called full search, which searches all candidates, has been widely used in hardware implementations [17] due to its simplicity and regularity. With its regularity, it is easy to fully reuse the overlapped search range data between different searches [16] to reduce the required access and several hardware architectures were proposed in [18]-[20] to take the advantages of search range data overlapping. However, the bandwidth requirement is still high due to its content independent data loading. Other approaches like fast ME algorithms [21]-[27] can reduce the computational complexity but they do not help a lot to reduce the required bandwidth for irregular search pattern.

To deal with data bandwidth problem, works in [28]-[33] proposed various efficient ME architectures to reduce data bandwidth requirements. In [28], the authors proposed a modified MB processing order to further increase the data reuse of Level C data reuse scheme [30]. For multiple reference frames motion estimation in H.264, [29] proposed a single reference frame multiple current macroblocks (MBs) scheme to reuse the reference data. An alternative direction to reduce the data bandwidth requirement was introduced in [30], [31] by decreasing the size of search range centered at the motion vector predictor (MVP). In [30], the authors efficiently selected the search area to reduce the data bandwidth requirement by tracing the motion vectors. [31] used a two-step windowing approach to dynamically decide whether to load more reference data for motion estimation. In hardware design, [32], [33] proposed the hardware architectures to reduce the memory bandwidth overhead by adopting the binary motion estimation mechanism which executed the matching process on the generated bit map instead of on the pixels. However, these works still require large and constant bandwidth support from a system because of the adopted full search algorithm. Large bandwidth requirement increases the cost as well as power consumption. Besides, assumption of constant bandwidth support is not practical for a modern complex system-on-a-chip due to high varieties of processing tasks. In addition, they do not consider how to efficiently use the available bandwidth or adapt the search process according to available bandwidth, which would be vital to portable video applications like video phones due to costly DRAM power consumption. As a result, above bandwidth policy implies two consequences: either over design to meet the demands or design unchanged with insufficient bandwidth supply that results in quality degradation or coding time increase.

To solve above problems without the side effects, this paper presents an on-demand data access efficient ME and its hardware realization under the rate distortion (RD) framework. Based on the introduced bandwidth-rate-distortion model, the proposed approach can minimize and predict the data bandwidth requirement, and access required data on demand while maximizes the rate-distortion performance within available bandwidth constraint. The on-demand data access can reduce the unnecessary data access and thus minimize the search range buffer. The simulation results show that the proposed algorithm can effectively allocate and reduce the required data bandwidth with negligible quality loss. Such efficiency also brings the low cost benefits for the resulted hardware implementation for its smaller search range buffer than other designs.

The organization of this section is as follows. Section II first briefly reviews the previous work and modeling the memory bandwidth of ME. Section III presents the proposed bandwidth aware ME algorithm and the simulation results are shown in Section IV. The hardware design and its implementation results are exhibited in Section V to demonstrate the efficiency of our proposal. Finally, a conclusion is given in Section VI.

2.2. Data Bandwidth Modeling for ME

2.2.1. Search Algorithms and Its Memory Bandwidth Modeling

In ME search algorithms, the full search ME loads all pixels inside the search area from external memory to find out the best matching MB by exhaustively checking every candidate position. The data bandwidth per frame for full search ME (DB_{FS}) can be calculated as follows.

$$DB_{FS} = [(SR_V \times 2 + 16) \times (SR_H \times 2 + 16)] \times \#MBs_{frame} \tag{2-1}$$

where SR_V and SR_H indicate the search range in the vertical and horizontal direction, r espectively and $\#MBs_{frame}$ refers to the number of MBs per frame.

Due to the high computational complexity and data access of full search ME, several fast algorithms [21]-[24] tried to reduce the number of candidate positions instead of exhaustive checking to decrease the computational complexity of full search ME and thus lessen the bandwidth requirements with less data to be loaded. However, these fast algorithms suffer from the irregular data access and complex data access control and consequently result in the difficulty of hardware realization and ill data reuse.

In addition to check point reduction approach, search range data can be reused to decrease the bandwidth requirement since the adjacent MBs will share a large portion of overlapped search range as shown in Fig. 2-1. A systematic analysis for data reuse in full search algorithm has been proposed in [28]. In which the Level C data reuse scheme as shown in Fig. 2-1 is widely used for ME design due to its high data reuse property. Hence, the data bandwidth per MB for full search with Level C reuse (DB_{FS_LevelC}) can be computed as follows.

$$DB_{FS\_LevelC} \cong \begin{cases} (SR_V \times 2 + 16) \times (SR_H \times 2 + 16) \ldots\ldots \forall MB \in left\ most\ MB\ column \\ (SR_V \times 2 + 16) \times 16 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots Other\ MBs \end{cases} \tag{2-2}$$

For an example with QCIF image format and ±8 search range, the data bandwidth requirements are 33Kbytes and 99Kbytes per frame for full search with and without Level C data reuse, respectively. Although full search with data reuse scheme can greatly reduce the bandwidth requirements for ME, it still needs high bandwidth requirements in case of large search window, which could occur in large size video. Besides, the search range for full search in hardware implementation is usually larger than other algorithms to keep the quality since hardware regularity forces its search center at (0, 0) instead of motion vector predictor [34]. For an example of 480p frame size, the bandwidth will be 3849.19 Kbytes per frame if the search range of ±64 is used.

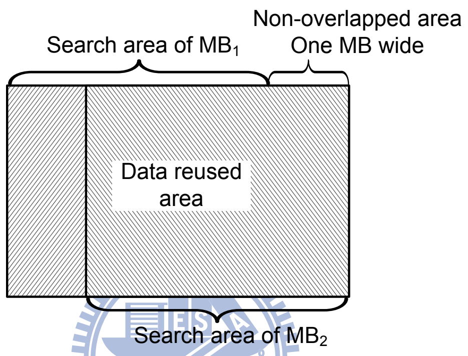


Fig. 2-1 Illustration of data reuse scheme for ME

2.2.2. Nearest Neighbours Search Algorithm and Its Bandwidth Modeling

To gain the benefits both from data reuse scheme and search range size reduction with MVP, this paper adopts nearest neighbours (NN) ME [35] for its highly data reuse in consecutive search and small search range due to MVP.

Fig. 2-2 shows the concept of nearest neighbours algorithm. It first calculates the MVP for search process and the Sum of Absolute Differences (SAD) of five positions centered at MVP (labeled by 1). If the minimum SAD is located at center position, the search operation is finished and the coordinate of center position is set as motion vector of current coding MB. Otherwise, the position with minimum SAD is set as the search center and another three positions labeled by 2 are checked. This operation is repeated until the position with minimum SAD is located at center or the search boundary is reached.

With above operations, Fig. 2-3 shows the data overlapping cases for different nearest neighbours search results. In this figure, the lighter pixels indicate the overlapped area and the dark pixel row or column is referred to the additional pixels to be loaded. From this figure, we can observe that there is a large portion of data overlapping between any two adjacent search positions, which leads to highly regular data reuse for hardware implementation. Therefore, by adopting nearest neighbours search pattern, only one extra column or row of pixels have to be loaded for search process.

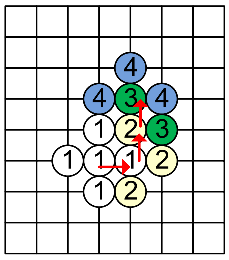


Fig. 2-2 Example of nearest neighbours search algorithm

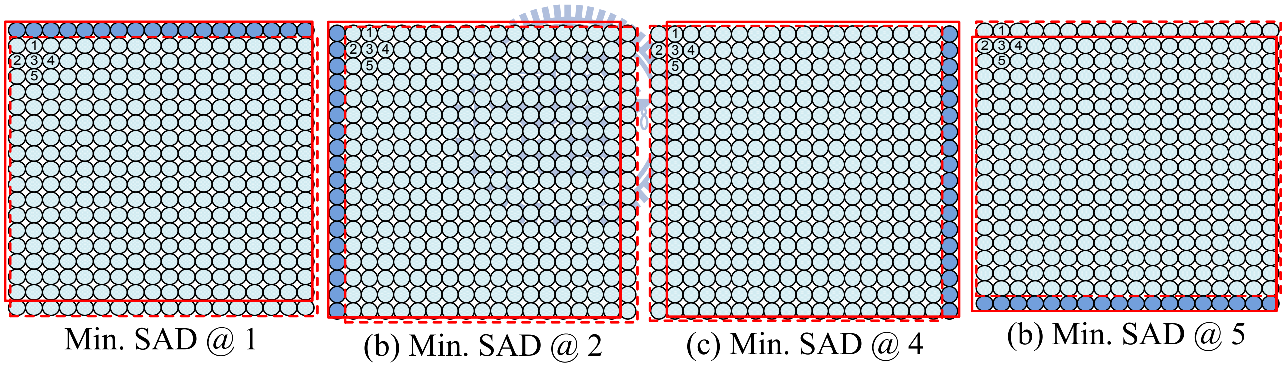


Fig. 2-3 Data overlapping for different nearest neighbours search results

The bandwidth requirements of nearest neighbours pattern are analyzed as follows. In the first step, 18×18 reference pixels for first five search points should be loaded from external memory to evaluate the SADs. If the minimum SAD is located at the center position, no more pixels are needed from external memory. Otherwise, 18 additional pixels are needed to be loaded from external memory for evaluation, if the position with minimum SAD is located at any one of four corner positions. Therefore, the data bandwidth per MB of nearest neighbours search pattern (DB_{NN}) can be formulated as follows.

$$DB_{NN} = (18 \times 18) + n \times 18 \tag{2-3}$$

where *18×18* indicates the required pixels for computing the SADs for first five positions and the *n* is the remaining steps to search the best result. With this, we can model the data requirements of nearest neighbours search pattern with step *n*. With step *n*, the data requirement for ME can be adjusted freely for different quality.

2.3. Proposed Framework

2.3.1. RD Bandwidth Modeling for Video Contents and Search Steps

The SAD is commonly used as similarity measurement in ME process due to its computational simplicity, but failed to consider the coding rate brought by motion vector encoding. Therefore, the commonly used Rate-Distortion cost (*RDCost)* is adopted in this paper and can be calculated as follows.

$$RDCost(MV, \lambda_{MOTION}) = SAD\big(s, c(MV)\big) + \lambda_{MOTION} R(MV - MVP) \tag{2-4}$$

where λ_{Motion} indicates the Lagrange multiplier and the term *R(MV-MVP)* represents the number of bits for coding the motion vector difference between the motion vector (*MV)* and the predicted motion vector. Through the adoption of *RDCost*, the best rate-distortion performance can be achieved.

Fig. 2-4 shows the relationship between the rate distortion performance improvement and search steps of the nearest neighbours search. In which the vertical axis is the percentage of *RDCost* improvement and the horizontal axis is the steps of *n* in nearest neighbours pattern ME. This simulation uses JM11 reference software [36], quantization parameter (QP) with 28, ±16 search range and only 16x16 block size for simplicity. The mechanism of variable block size is not included for simplicity due to the SADs of other small block size can be obtained from the data of 16x16 block size in SAD tree based hardware realization, and it will not affect the data bandwidth. Thus, the *RDCost* improvement can be derived as follows.

$$\Delta RDCost_n = \left(\frac{RDCost_n - RDCost_0}{RDCost_0}\right) \times 100$$

(2-5)

where $RDCost_0$ and $RDCost_n$ indicate the *RDCosts* after the first and $n+1$ steps search of nearest neighbours search, respectively. From these figures we can observe that the Δ*RDCost* is increased significantly in the first few steps. However, the Δ*RDCost* is increased slightly after more steps. For example, for the *Akiyo* sequence in Fig. 2-4(a), the Δ*RDCost* would be stable after three steps. For *Football* sequence in Fig. 2-4(b), sixteen steps are required for the Δ*RDCost* stabilization. Therefore, it is unnecessary to search too many steps since the improvement would be negligible after checking certain number of steps. Meanwhile, this also helps to save the data bandwidth requirement if the required steps can be properly predicted.

From Fig. 2-4, we can obtain three properties. First, the convergence speed of the sequence is content dependent. For example, the high motion sequence such as *Football* has slower convergence speed than the slow motion sequence such as *Akiyo*. This property mainly comes from that the high motion sequence needs more search steps to find out the best result. The second property is that the magnitude of Δ*RDCost* is also content dependent. For instance, the Δ*RDCost* of *Akiyo* sequence is smaller than that of the *Football* sequence since most MVs in low motion sequence have their best MV highly around the MVP. The last property is that the magnitude of *RDCost* significant influences the Δ*RDCost* convergence speed. That is, the sequences with smaller *RDCost* like *Akiyo* would have faster Δ*RDCost* convergence speed than the sequences with larger *RDCost* like *Football*.

By combining three above properties, we can use the *RDCost* and Δ*RDCost* of the first few steps to predict the data bandwidth requirement of ME process while maintain best rate distortion performance. Therefore, the convergence speed influenced by Δ*RDCost* can be modeled by the following equation.

$$\Delta RDCost_{improved} = \alpha \times exp\ [(\Delta RDCost_2 - \Delta RDCost_1) \times \beta]$$

(2-6)

where α is *RDCost* controlling factor which is related to initial *RDCost* and defined by Eq.(2-7) and the β stands for the bandwidth adjustment factor to achieve trade-off between rate distortion performance and data bandwidth requirements. That is, the larger the β is, the less the data bandwidth is necessary. Oppositely, the larger β also results in more rate distortion performance degradation. The β is set to 0.1 in this paper empirically to achieve best trade-off.

Furthermore, since different *RDCost* magnitude might result in different convergence speed as mentioned in the last property, the $RDCost_0$ is used to derive the factor of α by following equation.

$$\alpha = RDCost_0 \times \gamma$$

(2-7)

The γ is set to 0.001 empirically. Therefore, through Eq.(2-6), the relationship between *RDCost* improvement and search steps can be described and its fitting results are shown as the curve labeled with "*Model*" in Fig. 2-4. Finally, the steps needed for executing nearest neighbours search while keeping acceptable rate distortion performance degradation can be decided by the following equation by considering the maximum allowed search range size.

$$n = \Delta RDCost_{improved} \times SR_{Max}$$

(2-8)

where SR_{Max} indicates the maximum size of search range. After the step number n has been estimated by Eq.(2-8), the allocated data bandwidth for current MB ($DB_{Allocated\_MB\_i}$) can be obtained as follows.

$$DB_{Allocated\_MB\_i} = n \times 18$$

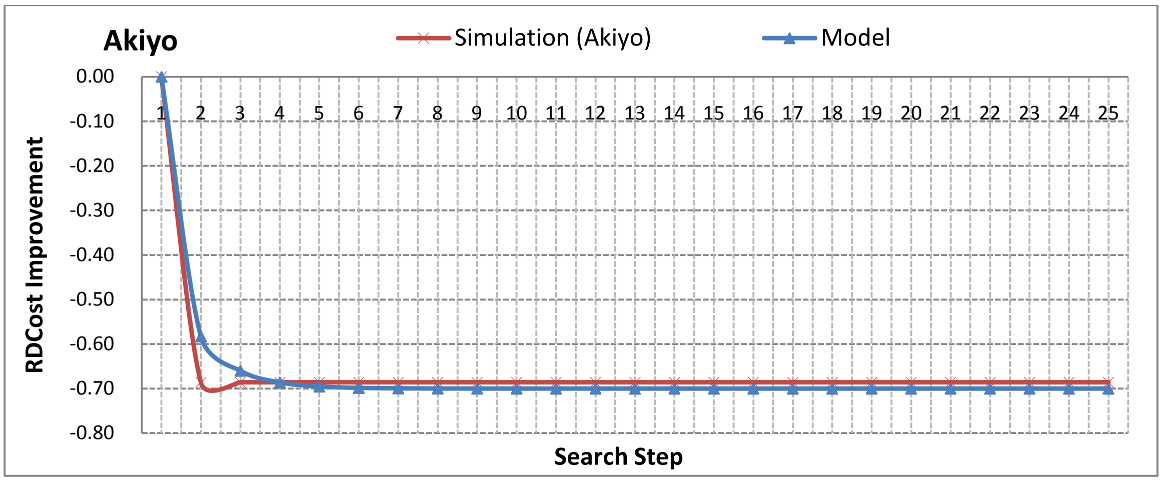
(2-9)

In summary, we can model the relationship of rate distortion performance and bandwidth by the search steps, initial *RDCost* and subsequent *RDCost* improvement according to (2-6). These two *RDCost* terms faithfully reflect the characteristics of video content. By this modeling, we can further determine the optimal steps to maximize the rate distortion performance under bandwidth constraints.

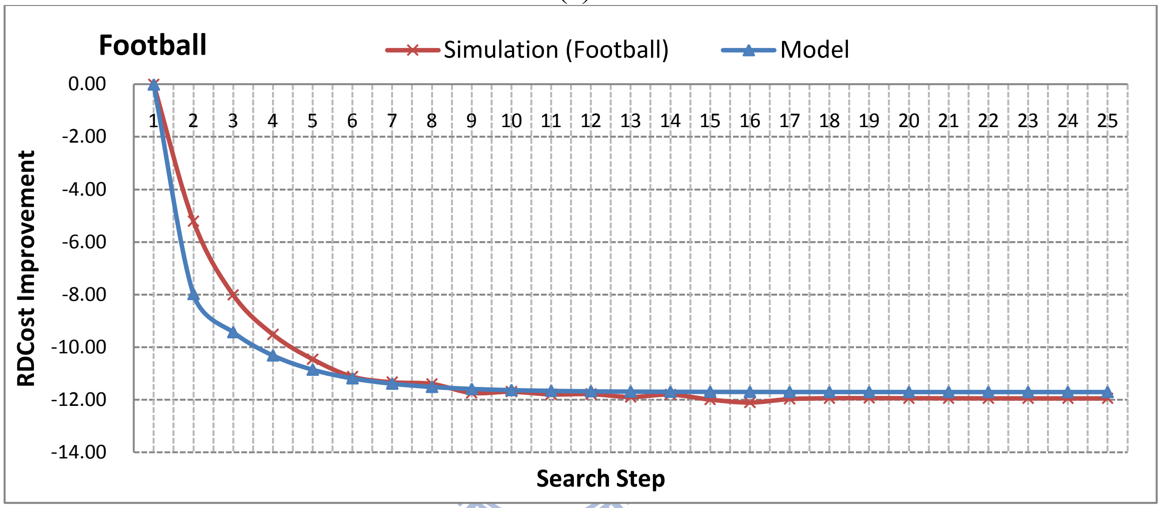
However, it is worth to mention that any other fast algorithm can also be used in the optimization framework of this paper if it satisfies the property of highly data reuse in its consecutive search, such as Full search, Logarithmic search [37], and One-at-a-time search [38]. To fit different kinds of search algorithms, Eq.(2-3) and Eq.(2-6)-(2-9) should be really remodeled. For example, the one-at-a-time search algorithm checks three candidates in the first step so that 16×18 pixels are loaded from external memory. For each search candidate, 16 pixels in average are demanded to be loaded for cost evaluation. Therefore, Eq.(2-3) can be rewritten as $(16\times18)+n\times16$ and Eq.(2-6)-(2-9) should be remodeled. The modeling of Eq.(2-6)-(2-9) can be done by a curve fitting tool once the relationship between rate distortion cost and search step is derived.

2.3.2. Proposed Bandwidth Aware ME Algorithm

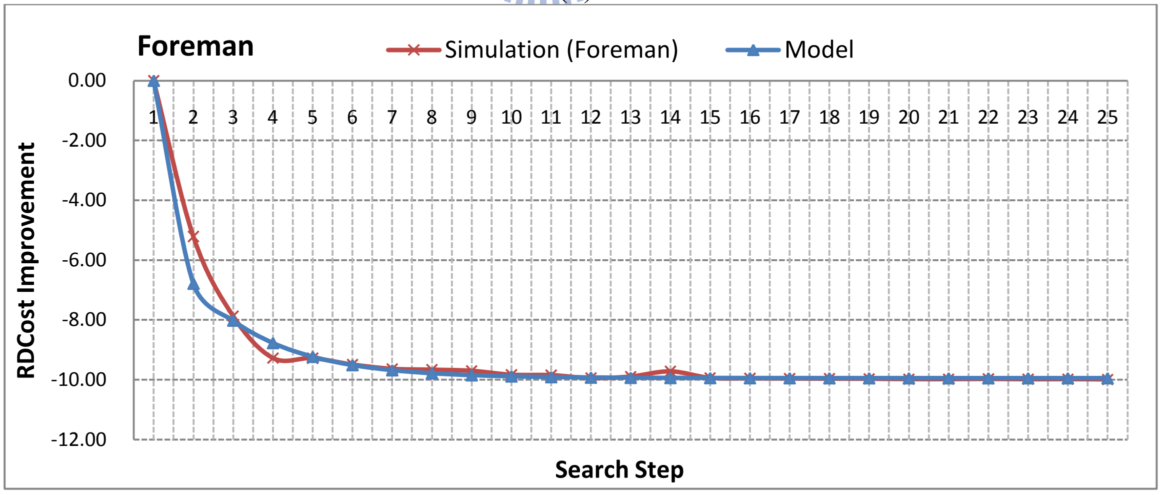
Fig. 2-5 shows the flow chart of proposed bandwidth aware ME algorithm. It operates as follows. First we initialize the available bandwidth for current frame. Then, we execute nearest neighbours search for the *step0, step1,* and *step2* to obtain $\Delta RDCost_1$ and $\Delta RDCost_2$. If the minimum *RDCost* is located at center position, the search process is finished. Otherwise, the $\Delta RDCost_1$ and $\Delta RDCost_2$ from previous steps are used to calculate the corresponding data bandwidth requirement of current MB. Afterwards, more steps are applied to search the best result according to the allocated data bandwidth. After finding the best result, the available data bandwidth pool is updated for further usage. The details of proposed algorithm are described as follows.



(a)

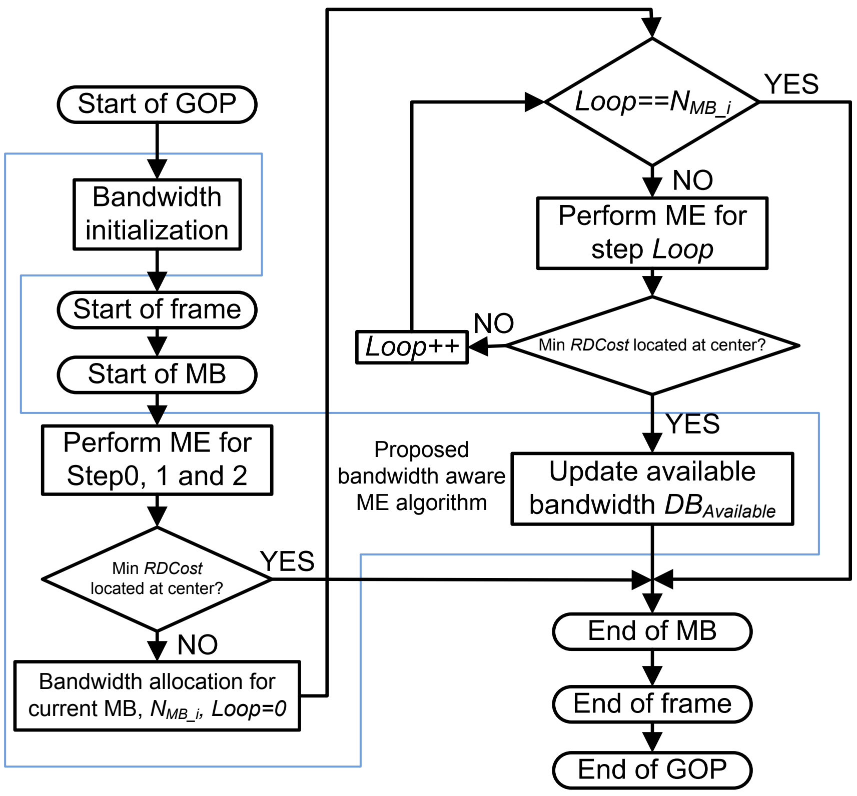


(b)



(c)

Fig. 2-4 The relationship between the steps *n* of nearest neighbours pattern and *RDCost* and modeled results for sequences of (a) Akiyo, (b) Football, and (c) Foreman



Start of GOP
Bandwidth initialization
Start of frame
Start of MB
Perform ME for Step0, 1 and 2
Min RDCost located at center?
YES
NO
Bandwidth allocation for current MB, N_MB_i, Loop=0
Loop==N_MB_i
YES
NO
Perform ME for step Loop
Min RDCost located at center?
NO
Loop++
YES
Proposed bandwidth aware ME algorithm
Update available bandwidth DB_Available
End of MB
End of frame
End of GOP

Fig. 2-5 Flowchart of proposed bandwidth aware ME algorithm

2.3.2.1. Bandwidth Initialization

In the first step, the total data bandwidth (DB_{Total}) for a certain encoding period is calculated by considering the available system bandwidth as follows.

$$DB_{Total} = \frac{BW_{Bus}}{FR} \times GOP$$

(2-10)

$$DB_{Used} = 0$$

(2-11)

where BW_{Bus} is the data transmission rate (Bytes/Second) of bus, FR indicates the frame rate, and the DB_{Used} indicates the used data bandwidth. In this paper the certain encoding period is defined as the group of pictures (GOP) and GOP refers to the number of frames per coding group. This total data bandwidth, DB_{Total}, stands for the available bandwidth per GOP. However, the available bandwidth for ME should subtract the bandwidth requirements for basic quality coding. Thus, the available data bandwidth ($DB_{Available}$) should be as follows.

$$DB_{Available} = DB_{Total} - DB_{Basic}$$

(2-12)

where DB_{Basic} is the necessary data bandwidth requirements for *step0, 1,* and *2* in nearest neighbours search algorithm and can be calculated by Eq. (2-3) so that $DB_{Basic} = (18\times18)+2\times18 = 360$ *bytes*.

2.3.2.2. <u>Bandwidth Allocation for Current MB</u>

After initialization step, the data bandwidth usage, DB_{MB_i}, for current MB i is allocated as follows.

$$DB_{MB\_i} = \begin{cases} DB_{Allocated\_MB\_i} \cdots\cdots\cdots\cdots\cdots\cdots\cdots if\ \ DB_{Allocated\_MB\_i} \leq \dfrac{DB_{Available} - DB_{Used}}{MB_{Remained}} \\ \dfrac{DB_{Available} - DB_{Used}}{MB_{Remained}} \cdots\cdots\cdots\cdots\cdots\cdots. Otherwise \end{cases}$$

(2-13)

where $DB_{Allocated\_MB\_i}$ stands for the allocated data bandwidth for current i-th MB by Eq. (2-6~2-9) and the $MB_{Remained}$ is the number of un-encoded MBs. In this allocation, the allocated bandwidth by Eq. (2-9) will be adopted only if the allocated bandwidth is smaller than average remaining data bandwidth per MB. Otherwise, the data bandwidth usage is restricted to average remaining data bandwidth per MB. By this restriction, the problem of over allocation can be avoided. After bandwidth allocation for current MB, the allocated bandwidth should be converted to the corresponding steps in nearest neighbours ME algorithm. The allocated steps can be calculated as follows.

$$N_{MB\_i} = \frac{DB_{MB\_i}}{18}$$

(2-14)

For the following step, the nearest neighbours search method is applied according to the allocated steps.

2.3.2.3. <u>Data Bandwidth Update</u>

The allocated data bandwidth might not be fully reused due to early termination condition of the search algorithm. Thus, the unused bandwidth can be recycled for further use. Therefore, an additional data bandwidth update stage should be added into the whole system. The data bandwidth update operations are as follows.

$$DB_{Used} = \sum_{j=0}^{i-1} DB_{Used\_MB\_j}$$

(2-15)

$$DB_{Remained\_MB\_i} = DB_{MB\_i} - DB_{Used\_MB\_i}$$

(2-16)

$$DB_{Available} = DB_{Available} + DB_{Remained\_MB\_i}$$

(2-17)

where $DB_{Remained\_MB\_i}$ refers to the remained data bandwidth for current MB i and $DB_{Used\_MB\_i}$ is used data bandwidth after executing nearest neighbours search.

2.4. Simulation Results

Table 2-I and Table 2-II show the simulation environment settings and test sequences. This simulation uses two scenarios to demonstrate the efficiency of our proposed algorithm. One scenario is to show the data bandwidth savings without data bandwidth constraint. In this scenario, we use the search range to represent BW_{Bus} for simplicity and compare with the full search with Level C data reuse scheme since it can achieve the highest data reuse [27],[39],[40]. Another scenario is to demonstrate the rate distortion performance under the data bandwidth constraint. This scenario compares three algorithms including full search (FS), nearest neighbours (NN) [35], and block-based gradient descent search algorithm (BBGS) [41] under the data bandwidth constraints of 20Kbytes and 38Kbytes for CIF sequences and 385Kbytes and 765Kbytes for 1080p sequences.

Table 2-I Environment settings for simulation

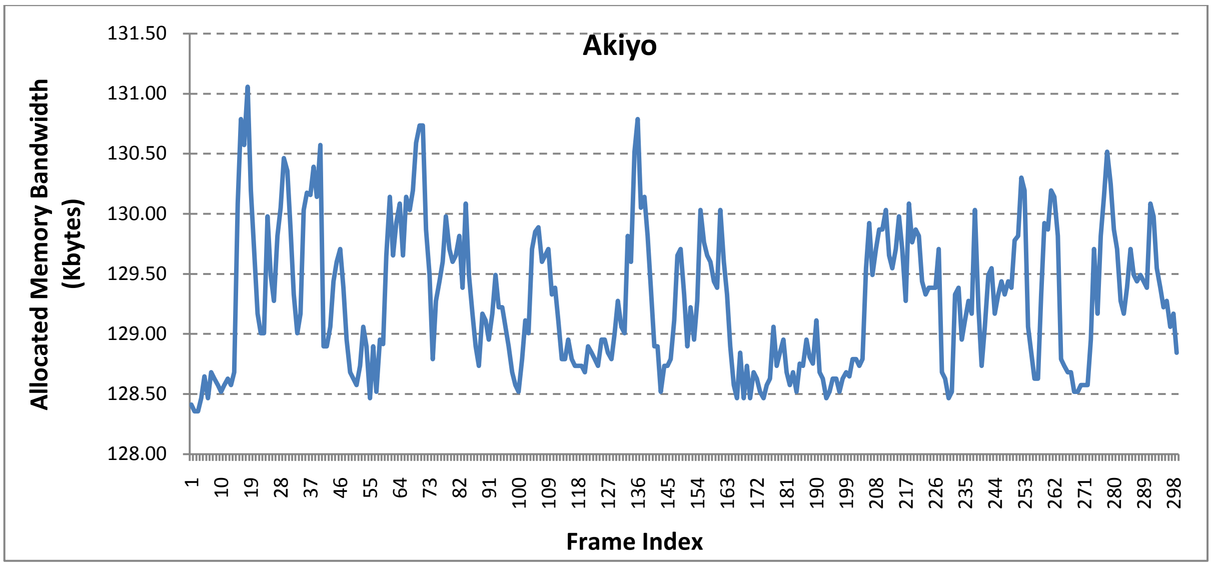
Platform	JM11
Frame structure	IPPPP….
Frames to be encoded	300 for CIF and 4CIF, 100 for 1080p
Frame rate	30fps
Frame resolution	CIF, 4CIF, and 1080p
Quantization Parameter (QP)	8, 18, 28, 32, 38
Group of Picture (GOP)	16
Entropy coding	CABAC
Rate Distortion Optimization (RDO)	Simple

Table 2-II Brief content description for test video sequences

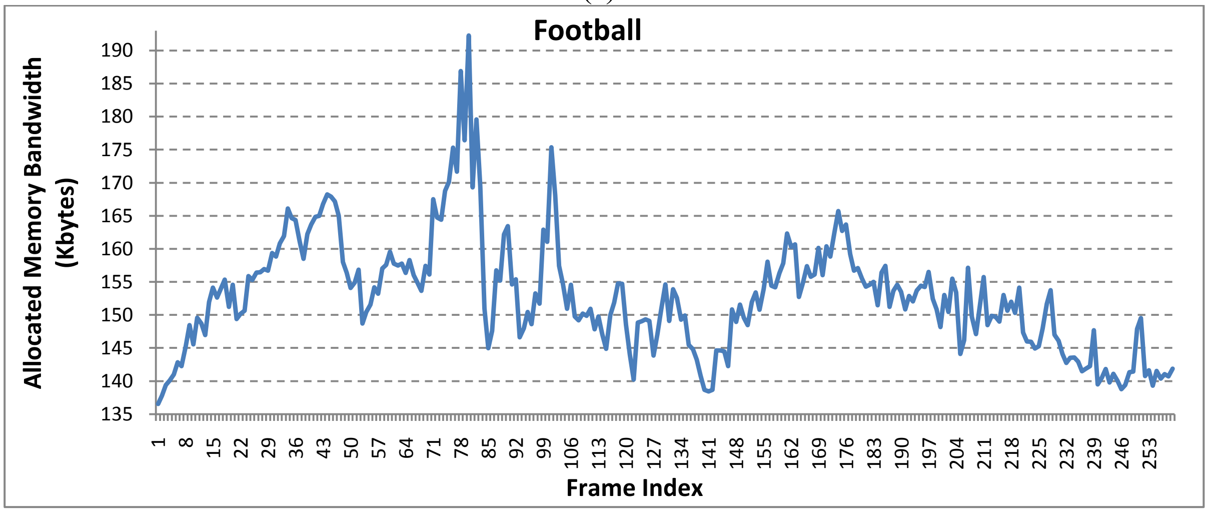
Motion behavior	Size	Sequence	Property
Slow motion	1080p	Sunflower	Complex texture in background and slow moving in object
		Rush hour	Quiescent motion in background and slow motion in moving objects
	CIF & 4CIF	Mother Daughter (MD), Hall, City, Akiyo, News	
Median motion	1080p	Tractor	Median motion in background and high motion in moving object
		Station2	Zoom out motion in all scene
		Riverbed	Monotonic texture in all scene
	CIF	Table tennis	Median motion in contents with scene changes
	CIF & 4CIF	Mobile	Complex texture and different motion direction
		Foreman	Zoom in and zoom out motion
		Coastguard	Median motion in background and moving objects
High motion	1080p	Blue sky	High motion in all scene
		Pedestrian,	High motion in moving objects
	CIF & 4CIF	Soccer, Stefan Football,	

Fig. 2-6 demonstrates the efficiency of bandwidth allocation for our method. For low motion *Akiyo* sequence in Fig. 2-6(a), the variation of allocated bandwidth is small, only 0.5Kbytes, for each frame. However, the variation of allocated bandwidth becomes larger for medium and high motion sequences. For example in *Table tennis* sequence, more bandwidth has been allocated to the scene change frame (frame index 131) to satisfy the demands of motion search.

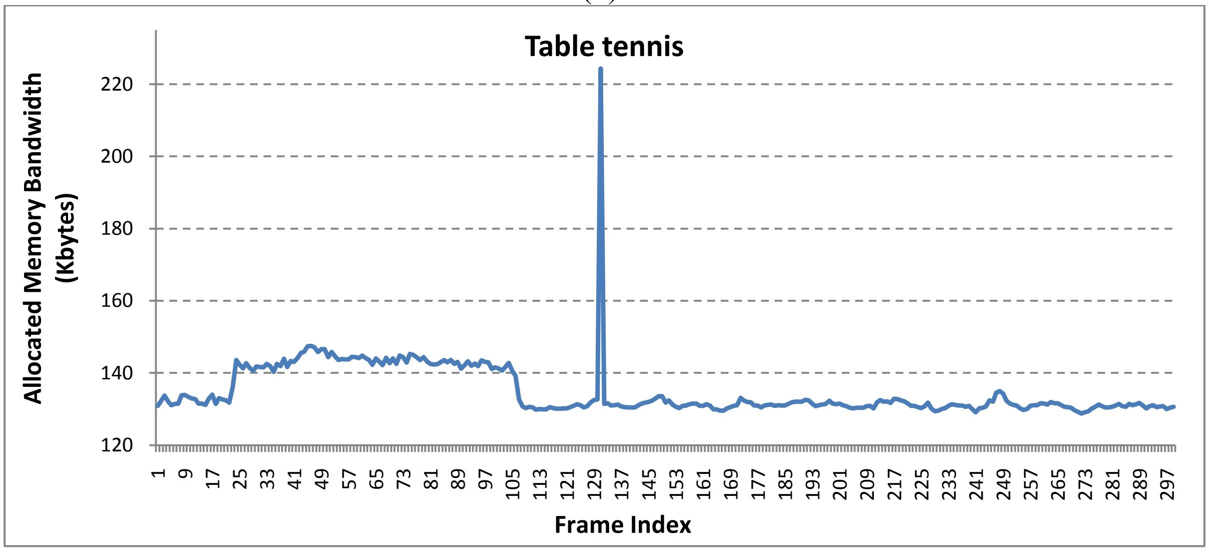
Table 2-III to Table 2-V show the comparisons of Peak Signal-to-Noise Ratio (PSNR), bitrate and consumed data bandwidth with search range ±16 and ±32 for CIF resolution, ±64 for 4CIF resolution and ±128 for 1080p resolution. The MVP is adopted as the prediction center for simulations of all algorithms. The data bandwidth here only considers the data amount accessed from external memory. The effect of DRAM latency will be discussed in the next Section. The result shows that the proposed algorithm consumes 18.00% less data bandwidth for the search range ±16 case. The saving becomes larger, 61.58% and 78.82%, for larger search range (search range ±32 for CIF) and larger frame size (search range ±64 for 4CIF) respectively. The bandwidth saving is also content dependent: more saving in low motion sequences and less saving in high motion sequences due to different allocated search steps. Above results show that the proper data bandwidth requirement can be dynamically allocated for different video content through our proposed algorithm. With less consumed data bandwidth, the PSNR degradation of our proposed algorithm is only 0.12dB for all search range size in maximum, with only 0.37% bitrate increase on average. As a result, the proposed algorithm can efficiently predict the suitable data bandwidth requirements for ME and thus result in less rate distortion performance degradation.



(a)



(b)



(c)

Fig. 2-6 Allocated bandwidth for difference video sequences (a) Akiyo, (b) Football, and (c) Table tennis

Table 2-III Comparison of average PSNR without bandwidth constraint

Sequences	CIF						Sequences	4CIF		
	SR: ±16			SR: ±32				SR: ±64		
	FS	Pro.	Δ (dB)	FS	Pro.	Δ (dB)		FS	Pro.	Δ (dB)
Akiyo	41.44	41.42	-0.02	41.44	41.42	-0.02	Akiyo	44.02	43.99	-0.03
Coastguard	37.54	37.54	-0.00	37.54	37.54	-0.00	City	38.20	38.19	-0.01
Football	38.75	38.78	+0.03	38.74	38.76	+0.03	Coastguard	40.04	40.03	-0.01
Foreman	38.76	38.74	-0.02	38.76	38.74	-0.02	Football	41.33	41.38	+0.05
Mobile	38.31	38.19	-0.12	38.19	38.19	-0.00	Foreman	41.43	41.41	-0.02
MD	42.01	41.99	-0.02	42.00	41.99	-0.01	Hall	42.39	42.38	-0.02
News	41.21	41.20	-0.01	41.21	41.20	-0.01	Mobile	38.90	38.89	-0.01
Table tennis	38.33	38.32	-0.01	38.34	38.32	-0.01	News	42.51	42.54	+0.03

Table 2-IV Comparison of average bitrate without bandwidth constraint

Sequences	CIF						Sequences	4CIF		
	SR: ±16			SR: ±32				SR: ±64		
	FS	Pro.	Δ (%)	FS	Pro.	Δ (%)		FS	Pro.	Δ (%)
Akiyo	550.67	550.53	-0.03	550.67	550.53	-0.03	Akiyo	1380.74	1379.57	-0.08
Coastguard	2988.49	2985.74	-0.09	2987.58	2985.75	-0.06	City	10840.42	10741.38	-0.91
Football	3349.39	3410.22	+1.82	3330.66	3389.97	+1.78	Coastguard	6915.81	6919.85	+0.06
Foreman	2311.85	2324.39	+0.54	2311.89	2323.88	+0.52	Football	7106.68	7284.16	+2.50
Mobile	5091.61	5072.55	-0.37	5099.20	5073.05	-0.51	Foreman	5380.74	5459.90	+1.47
MD	1355.34	1356.12	+0.06	1355.10	1355.71	+0.04	Hall	5460.56	5453.84	-0.12
News	1125.50	1126.60	+0.10	1125.55	1126.65	+0.10	Mobile	9431.67	9424.50	-0.08
Table tennis	2006.60	2025.80	+0.96	2007.39	2026.84	+0.97	News	2319.75	2326.26	+0.28
Average	**2347.43**	**2356.49**	**+0.37**	**2346.01**	**2354.05**	**+0.35**	**Average**	**6104.55**	**6123.68**	**+0.39**

Table 2-V Comparison of data bandwidth saving

Sequences	CIF		Sequences	4CIF
	SR: ±16	SR: ±32		SR: ±64
Akiyo	22.19%	63.55%	Akiyo	80.24%
Coastguard	17.69%	61.44%	City	79.05%
Football	10.18%	57.94%	Coastguard	78.91%
Foreman	15.92%	60.55%	Football	75.48%
Mobile	16.26%	60.77%	Foreman	77.99%
MD	20.61%	62.81%	Hall	79.79%
News	21.39%	63.16%	Mobile	79.17%
Table tennis	19.78%	62.41%	News	79.96%
Average	**18.00%**	**61.58%**	**Average**	**78.82%**

Table 2-VI to Table 2-VIII show the BD-PSNR and BD-Bitrate comparisons for low, median and high motion sequences under different data bandwidth constraint. All these results are relative to the results of the FS algorithm. In our simulation, the total data bandwidth is evenly distributed to each MB in FS, NN, and BBGS motion estimation algorithms and is dynamically allocated to each MB in our proposed algorithm. For low motion sequences, the rate distortion performance of our proposed algorithm is near the same as FS since the best MV can be easily found near MVP from a limited data bandwidth support. For median motion sequences, the BD-PSNRs of our proposed algorithm become better than FS algorithm and the BD-Bitrates of some sequences are decreased. In Table 2-VII, we can observe that the BD-Bitrate saving of *Station2* sequence is much higher than other sequences since this sequence has zoom out motion over entire frame. As a result, the data bandwidth requirement of MBs is different from the others due to the MBs in the outer have higher motion than the MBs in the center. Our proposed algorithm can detect such variation and allocate suitable amounts of data bandwidth to each MB. For high motion sequences in Table 2-VIII, the BD-PSNR and BD-Bitrate of our proposed algorithm are much better than FS algorithm due to better detection of data bandwidth requirement of each MB. The BD-Bitrate saving of our proposed algorithm can achieve up to 7.143% for *Stefan* sequence in maximum and 2.44% in average for all high motion sequences. In summary, our proposed algorithm has higher performance in high motion sequences which are also the hardest sequences to deal with.

Fig. 2-7 to Fig. 2-9 show the frame by frame bitrate usage of different algorithms. This simulation condition is the same as previous one but with unevenly distributed data bandwidth to each MB. This condition is to simulate the traditional ME design with fixed and pre-allocated bandwidth. In that condition, the ME design has fixed search range and expects the search range data arrival in time for computation. However, if the real allocated bandwidth is lower than expected due to bandwidth competition from other devices, the operations of motion estimation will be skipped and the intra mode will be selected as best prediction mode when the allocated data bandwidth has run out. For the bandwidth usage, the simulation shows that both of FS and our proposed algorithm would use up all the available bandwidth but the BBGS and NN algorithm wouldn't. The PSNR results are similar to the previous one but with quite different bit rate. The result shows that the bitrate usage of the proposed algorithm is much less than those in other algorithms due to fewer intra block coding. This proves that our algorithm can properly allocate bandwidth to MBs according to their content and use up the available bandwidth but never exceed the budget to support ME operations.

Table 2-VI Comparison of BD-PSNR and BD-Bitrate for low motion sequences

Image size	DB_{Total}	Sequence	BD-PSNR (dB)			BD-Bitrate (%)		
			NN	BBGS	Proposed	NN	BBGS	Proposed
1080p	385KB	Sunflower	-0.032	-0.025	+0.097	+0.550	+0.386	-0.832
		Rush hour	-0.049	-0.033	+0.034	+0.916	+0.615	-1.467
	765KB	Sunflower	-0.001	-0.001	+0.016	+0.020	+0.026	-0.294
		Rush hour	-0.006	-0.006	+0.021	+0.046	+0.046	-0.669
CIF	20KB	Akiyo	-0.013	-0.044	-0.002	+0.276	+1.046	+0.021
		News	-0.031	-0.051	-0.014	+0.825	+1.421	+0.743
		MD	-0.030	-0.051	-0.027	+0.626	+1.030	+0.287
	38KB	Akiyo	-0.011	-0.035	-0.001	+0.298	+0.877	+0.022
		News	-0.023	-0.056	+0.003	+0.452	+1.129	-0.057
		MD	-0.040	-0.063	-0.010	+1.109	+1.739	+0.259

Table 2-VII Comparison of BD-PSNR and BD-Bitrate for median motion sequences

Image size	DB_{Total}	Sequence	BD-PSNR (dB)			BD-Bitrate (%)		
			NN	BBGS	Proposed	NN	BBGS	Proposed
1080p	385KB	Tractor	-0.029	-0.016	+0.007	+0.439	+0.244	-0.144
		Station2	-0.243	-0.145	+0.069	+4.586	+2.902	-1.467
		Riverbed	-0.002	-0.003	-0.002	+0.009	+0.020	+0.021
	765KB	Tractor	-0.001	-0.001	+0.004	+0.006	+0.006	-0.068
		Station2	-0.031	-0.031	+0.129	+0.454	+0.454	-2.048
		Riverbed	-0.000	-0.000	+0.001	+0.005	+0.005	-0.017
CIF	20KB	Table tennis	-0.128	-0.164	-0.100	+2.680	+3.337	+2.053
		Mobile	-0.000	-0.039	+0.019	+0.024	+0.605	-0.248
		Foreman	-0.182	-0.106	+0.043	+3.670	+2.222	+0.839
		Coastguard	-0.005	-0.038	+0.010	-0.027	+0.631	-0.156
	35KB	Table tennis	-0.113	-0.139	-0.051	+2.287	+2.817	+1.046
		Mobile	+0.019	-0.018	+0.000	-0.247	+0.318	+0.009
		Foreman	-0.293	-0.159	+0.021	+6.304	+3.268	-0.514
		Coastguard	+0.004	-0.029	+0.024	-0.052	+0.595	-0.494

Table 2-VIII Comparison of BD-PSNR and BD-Bitrate for high motion sequences

Image size	DB_{Total}	Sequence	BD-PSNR (dB)			BD-Bitrate (%)		
			NN	BBGS	Proposed	NN	BBGS	Proposed
1080p	385KB	Blue sky	-0.449	-0.151	+0.035	+6.669	+2.235	-0.439
		Pedestrian	-0.017	-0.016	+0.146	+0.316	+0.273	-2.707
	765KB	Blue sky	-0.027	-0.027	+0.079	+0.361	+0.361	-1.509
		Pedestrian	-0.004	-0.004	+0.033	+0.055	+0.055	-0.645
CIF	20KB	Football	-0.203	-0.126	+0.038	+3.011	+1.853	-0.631
		Soccer	-0.215	-0.150	+0.105	+3.929	+2.689	-1.880
		Stefan	-0.152	-0.153	+0.544	+2.161	+2.143	-7.143
	38KB	Football	-0.149	-0.061	+0.085	+2.220	+0.927	-1.295
		Soccer	-0.209	-0.112	+0.086	+3.760	+2.090	-1.517
		Stefan	-0.085	-0.067	+0.516	+1.158	+0.876	-6.599

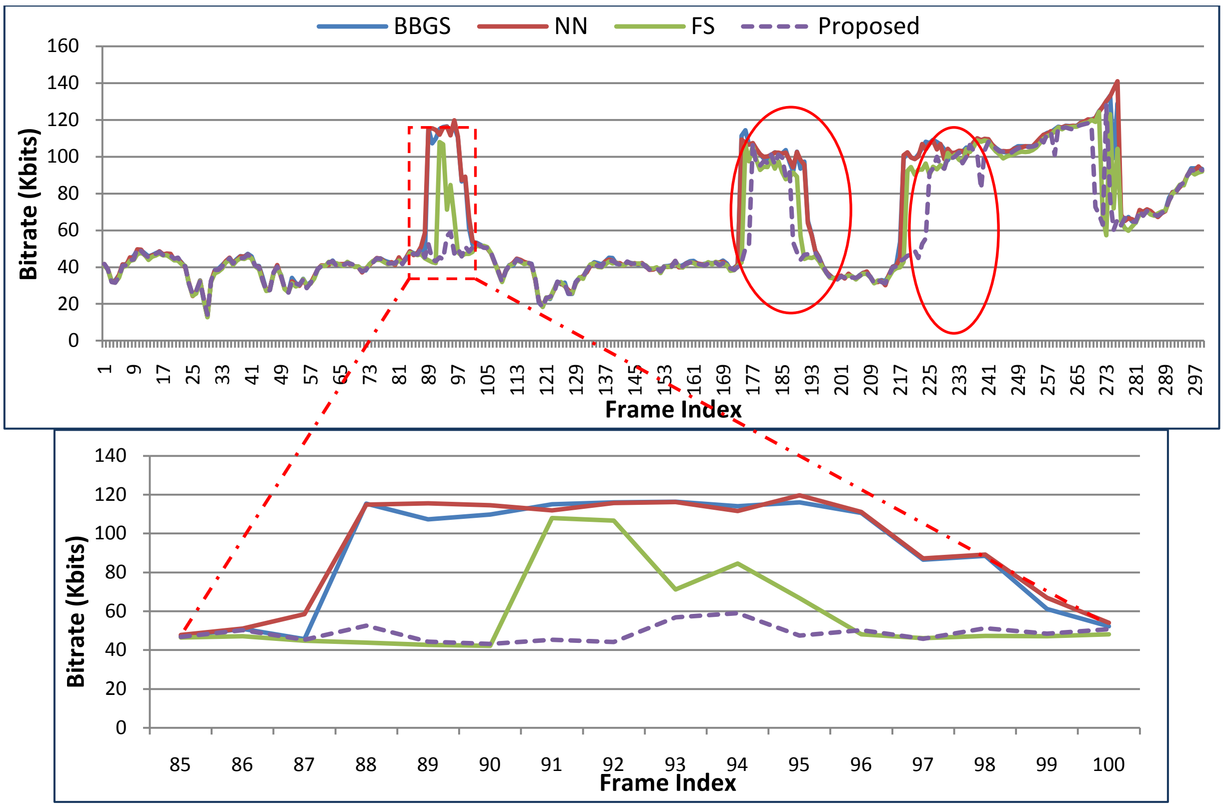


Fig. 2-7　Comparison of bitrate for *Stefan* sequence under 38KB data bandwidth constraint

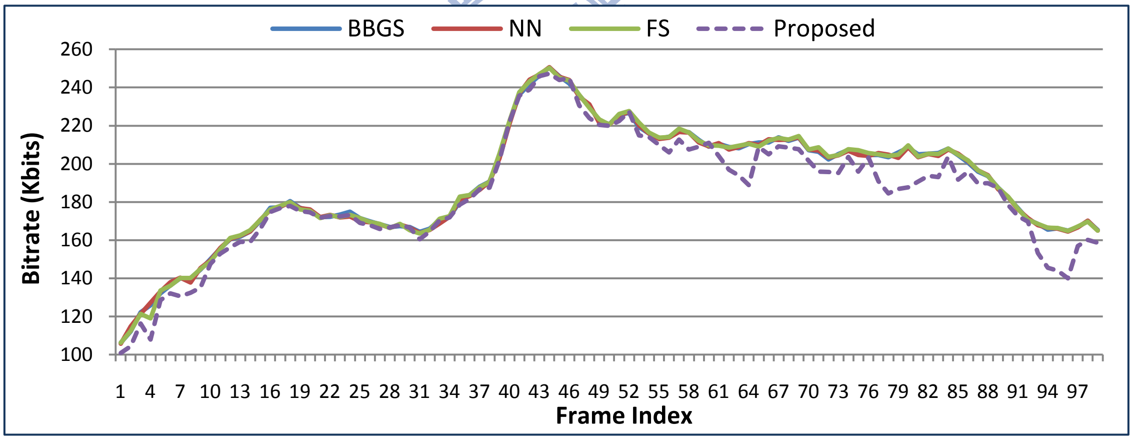


Fig. 2-8　Comparison of bitrate for *Pedestrian* sequence under 385KB data bandwidth constraint

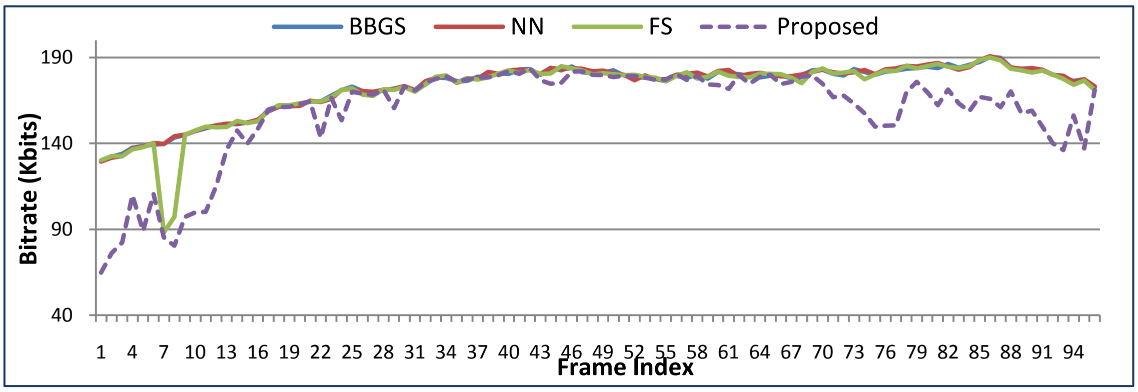


Fig. 2-9　Comparison of bitrate for *Station2* sequence under 385KB data bandwidth constraint

2.5. Proposed Architecture

Fig. 2-10 shows the proposed architecture and its operation is described as follows. First, the current pixels and 22×22 reference pixels will be respectively loaded into *Current buffer* and *Reference buffer* through the help of *Memory controller*. Here, the scheduling policy of *Memory controller* is first-in-first-serve so that the requests are processed in the request order. Afterwards, the pixels stored in *Current buffer* and *Reference buffer* will be used to calculate SADs through the *SAD calculation module* which will combine the motion vector costs generated from *MV cost generator* to obtain rate distortion costs of $RDCost_0$, $RDCost_1$, and $RDCost_2$. In our design, the *SAD calculation module* is composed of 16 *PE_4x4s*(Fig. 2-11), and each *PE_4x4*(Fig. 2-12(a)) computes the SAD value of a 4x4 block so that the SAD of one candidate position can be generated per cycle. The rate distortion costs generated from *SAD calculation module* are stored in *RDCost buffer* for the following usage. Once the $RDCost_0$, $RDCost_1$, and $RDCost_2$ have been calculated, the modeled information of search steps will be computed as Eq.(5) to Eq.(9) by the *Bandwidth modeling module* and sent to *Bandwidth aware ME controller*. Afterward, the *Bandwidth aware ME controller* will execute the computations as Eq.(10) to Eq.(17) with the calculated search steps, and generate the memory address to *Memory Controller* to load the required reference pixels for the following search steps. Once the allocated bandwidth has been run out or the termination criterion of nearest neighbours search has been met, the motion estimation search will be finished.

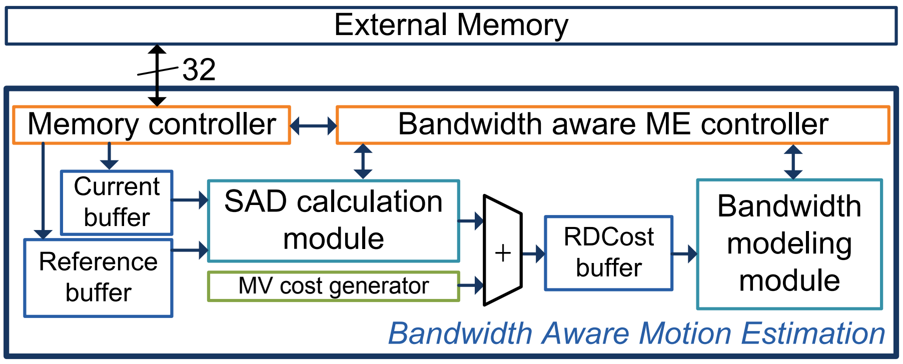


Fig. 2-10 Architecture of proposed bandwidth aware ME

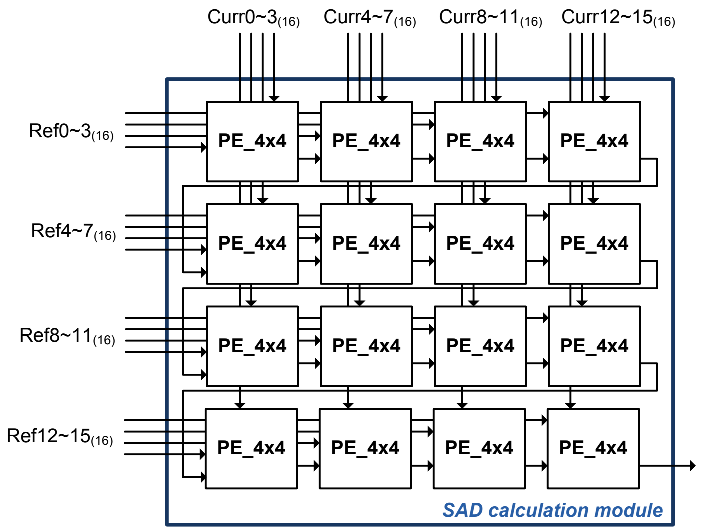
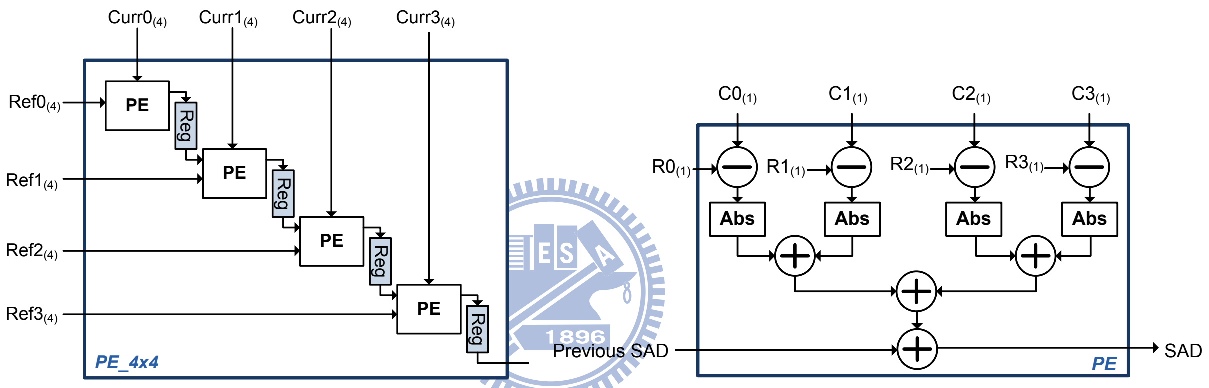


Fig. 2-11 Architecture of SAD Calculation Module



(a) (b)

Fig. 2-12 (a) PE_4x4 and (b) PE

2.5.1. Bandwidth Modeling Module Design

The required search steps of our bandwidth aware ME is mainly computed by Eq.(2-5) to Eq.(2-9). However, these equations need five multipliers, three dividers, three subtractions and a look-up table to generate the corresponding search steps by a direct implementation approach, which is too costly. Therefore, some approximations are applied to reduce the hardware cost. First, the operation of $\Delta RDCost_2$ - $\Delta RDCost_1$ is computed as in Eq.(2-18) so that the subtractions can be reduced from three to two with an additional left-shift operation.

$$\Delta RDCost = \Delta RDCost_2 - \Delta RDCost_1 = \left(\frac{RDCost_2 - RDCost_1 - 2RDCost_0}{RDCost_0}\right) \times 100 \tag{2-18}$$

In addition, since the β is set to 0.1, the $\Delta RDCost_{improved}$ can be rewritten as follows.

$$\Delta RDCost_{improved} = \alpha \times exp\ (\Delta RDCost \times 0.1)$$

(2-19)

From Eq.(2-7), we observed that γ is set to 0.001 empirically. However, we can rewrite Eq.(2-7) as follows.

$$\alpha = RDCost_0 \times \gamma \mid \gamma = 0.001 = \frac{1}{1000} \rightarrow \alpha = \frac{RDCost_0}{1000}$$

(2-20)

To reduce the hardware cost of division operation, the division operation of 1/1000 is simply approximated by 1/1024 in our design. The benefit of this approximation is that experimental results show that there is no rate distortion performance difference between the operations of 1/1000 and 1/1024 but the division operation of 1/1000 can be simply replaced by a right-shit operator to reduce the hardware cost. (The results in Section IV already include this approximation into simulation.) Finally, the $\Delta RDCost_{improved}$ can be rewritten as follows.

$$\Delta RDCost_{improved} = \frac{RDCost_0}{1024} \times exp\left[\left(\frac{RDCost_2 - RDCost_1 - 2RDCost_0}{RDCost_0}\right) \times 10\right]$$

(2-21)

Fig. 2-13 shows the architecture of our proposed *Bandwidth modeling module*. In which the *Exp. LUT* module is a look-up table to implement the exponential operation, and it is organized by 46 entries with 12 bits per each. With above approximation, the hardware cost can be reduced significantly due to our proposed architecture only needs two multipliers, two subtractions, one divider, three shifters, and a look-up table when compared to direct implementation approach.

2.5.2. Design of the On-demand Reference Buffer

Fig. 2-14 shows the design of the *Reference buffer*. In our reference buffer design, the *rc_sel* signal is used to determine where a row of 22 pixels or a column of 22 pixels should be replaced in *22×22 Pixels buffer*. In addition, the *addr_x* and *addr_y* signals play the role of determining which 256 pixels should be selected to output for SAD calculation according to ME search algorithm. Note that the size of *Reference buffer* is only 22×22 pixels which can cover the first three steps and help the data reuse in the remaining steps. In addition, another benefit of 22×22 buffer size is that the reference data loading request can be issued simultaneously to memory controller to load future necessary reference data once the search direction has been decided and SAD calculation module starts to compute SAD. As a result, the stalled cycles can be eliminated.

Fig. 2-15 shows the mechanism of reference data updating for our *Reference buffer*. In our *Reference buffer* design, we adopt the circular addressing mode so that the new loaded reference data will be stored in most-top-row (Fig. 2-15(a)), most-bottom-row (Fig. 2-15(b)), most-left-column (Fig. 2-15(c)), or most-right-column (Fig. 2-15(d)) registers depending on the required search direction. Through this updating mechanism, only one row or column reference data should be updated instead of overall reference data and consequently reducing the power consumption for updating *Reference buffer* contents.

With above scheme, the required buffer size is independent of required search range. This is quite different from other designs for fast algorithms [18],[19],[20],[32],[33] that load all search range data into on-chip memory. However, a common issue for above scheme is the DRAM access latency if the accessed data pattern is a column access as Fig. 2-3(b) and (c). If these access data patterns are required, the DRAM access latencies will be increased due to the required column of pixels are not stored continuously in DRAM. Therefore, the required cycle counts to load the data will be analyzed in the following subsection with the consideration of DRAM access latency.

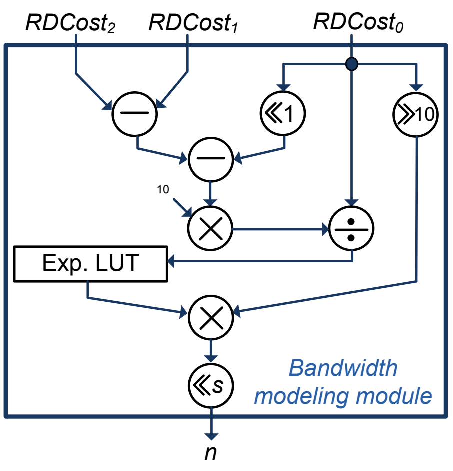


Fig. 2-13 Architecture of proposed bandwidth modeling module

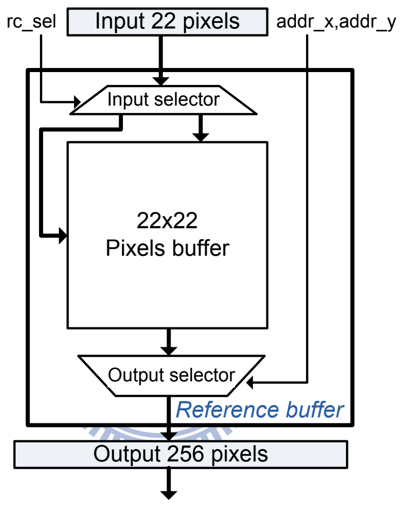
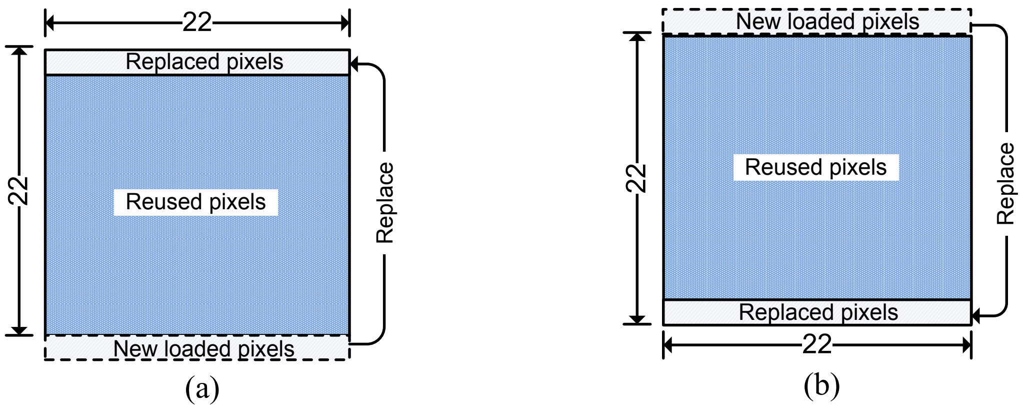


Fig. 2-14 Architecture of reference buffer



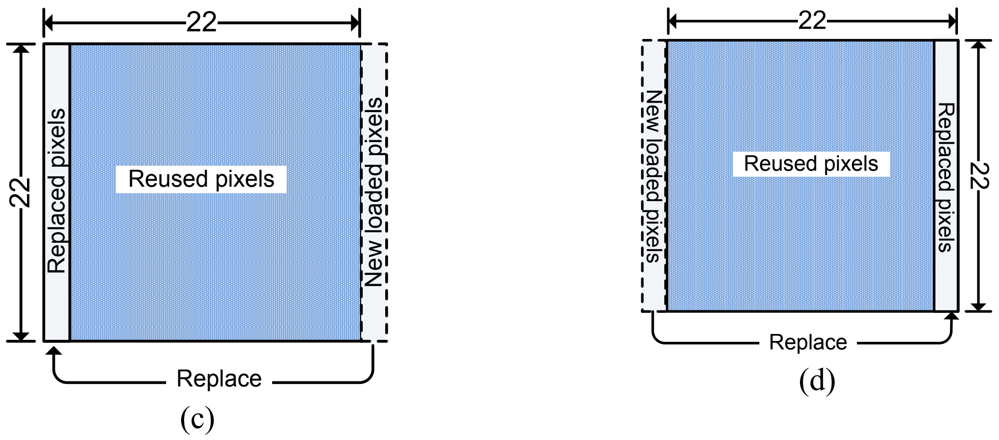


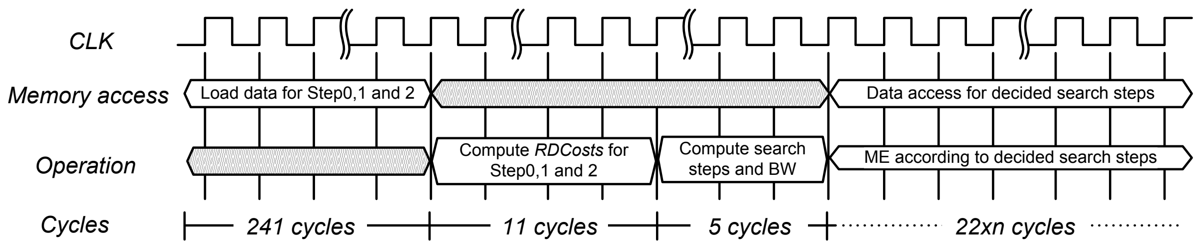
Fig. 2-15 The mechanism of *Reference buffer* updating in which reference data are loaded for (a) Down-toward, (b)Up-toward, (c)Right-toward, and (d)Left-toward search process

2.5.3. Timing Analysis with DRAM Access Latency

Fig. 2-16 shows the timing diagram of the proposed design. This timing diagram accurately includes the necessary memory latency caused by DRAM access into the simulation framework by adopting the Micron SDRAM model [42] in our design. In general, the DRAM latency is mainly from the row miss and page miss due to the cross page or cross row access in a data access. Such misses in our algorithm could occur in the same step access or the different step access if the required data are located in different pages or rows. Therefore, the video data mapping in DRAM uses the nearly optimal data mapping proposed in [43] to gain the benefit of high spatial locality of neighboring MBs to reduce the access penalty by row miss or page miss. Fig. 2-17 shows the data mapping of our design. In which the luminance, chrominance, and motion vector components of four adjacent MBs are grouped and stored at the same memory row. With above data mapping, the external DRAM is interconnected to the proposed ME module via a 32bits data bus so that four pixels can be loaded from external memory per cycle. Thus, initial 22×22 reference data loading needs 241 cycles in average according to the simulations. With these data, 11 cycles are needed to calculate the *RDCosts* for five candidates at the first step and three candidates at the second and third step. Once the *RDCosts* of *step0*, *1*, and *2* have been calculated, 5 cycles are used to derive necessary search steps and bandwidth for the following ME search. Finally, the nearest neighbours search is executed to obtain the best results according to the decided search steps *n*. It is worth to mention that the required cycles for each step are only the external memory access cycles, 22 cycles in maximum under our operating frequency, since the SAD computation cycles are less than external memory access cycles and thus these operations can be parallel executed.

Table 2-IX shows the memory bandwidth saving comparison per MB for our proposed algorithm and Level C data reuse full search ME scheme. Both of memory access latencies and data amounts are considered to derive the real memory bandwidth saving in this simulation results. From this table, we observed that the memory bandwidth savings for search range ±16, ±32, and ±64 are 13.51%, 59.47%, and 77.43%, respectively. In contrast to memory bandwidth saving listed in Table 2-V, we can observe 3% data bandwidth saving difference on average caused by DRAM latency. For the high motion sequence, the memory bandwidth saving of *Football* sequence is very different from the data bandwidth saving analyzed in Table 2-V The reason is that the most of data accesses in the *Football* sequence are tended to load a column of pixels for SAD calculation from external DRAM since it has higher motion behavior in objects and moving in horizontal direction so that many search steps are allocated in this direction. Unfortunately, the memory access cycles for a column of pixels are much higher than a row of pixels in case of the same data access amounts, when constrained by the external DRAM data mapping mechanism. Therefore, the memory bandwidth saving of *Football* sequence is lower than other sequences and different from the data bandwidth saving listed in Table 2-V However, it still has 2.08% memory bandwidth saving. For the median and low motion sequences, the difference between these two tables is not noticeable. As a result, the impact on the bandwidth usage depends on the motion of the sequences instead of the frame resolution, said the higher impact on higher motion sequences.

In summary, the simulation results demonstrate that our proposed algorithm can still work well and achieve efficient bandwidth usage even considering the DRAM latency.



CLK
Memory access
Load data for Step0,1 and 2
Data access for decided search steps
Operation
Compute RDCosts for Step0,1 and 2
Compute search steps and BW
ME according to decided search steps
Cycles
241 cycles
11 cycles
5 cycles
22xn cycles

Fig. 2-16 Timing diagram of proposed bandwidth aware ME

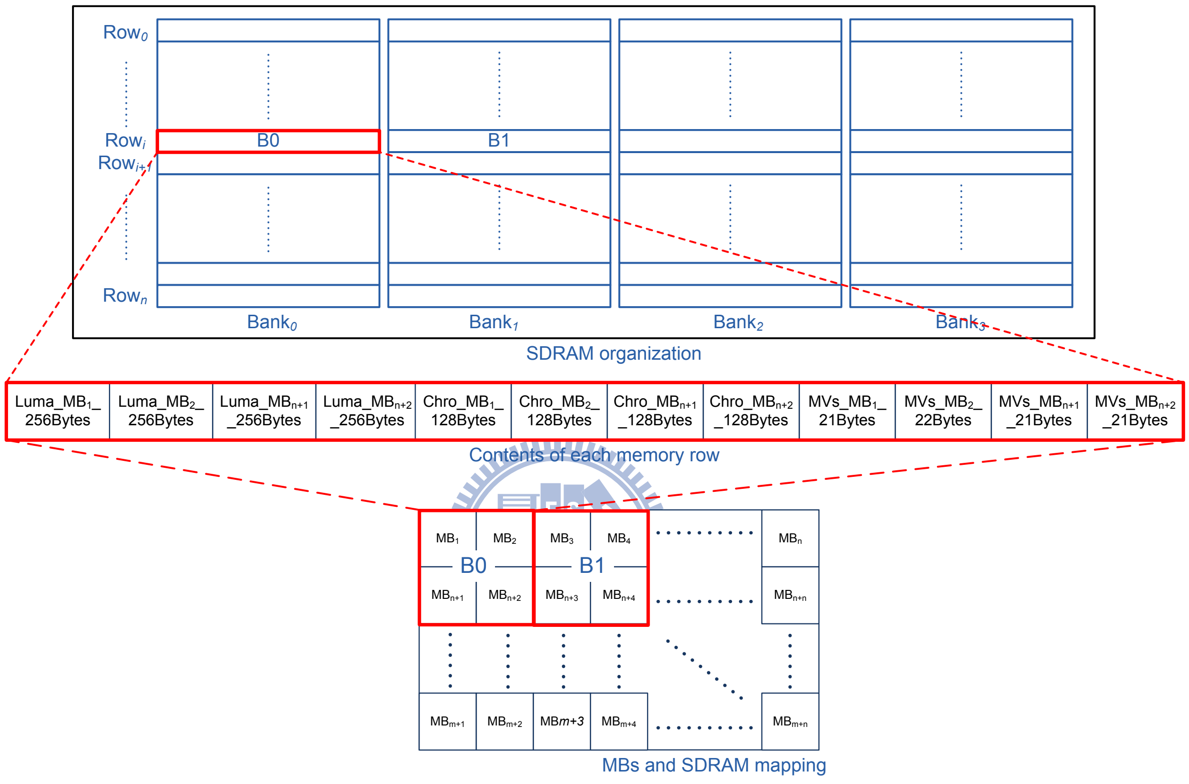


Fig. 2-17 SDRAM data mapping of our proposal

2.5.4. Implementation Results

The proposed design is implemented and synthesized by the UMC 90nm technology. Table 2-X shows the design comparison with other architectures. As shown in the table, the proposed design can process CIF and 4CIF sized video at 30fps when running at 5.6MHz and 23MHz operating frequency, respectively. It is worth to mention that the on-chip SRAM size is usually increased with the growth of search range size for most of ME hardware due to noticeable reference pixels have to be stored temporally in on-chip SRAM for SAD computation process. However, the main difference and contribution of our proposed architecture is that the demanded on-chip SRAM size is irrelevant to the search range size due to the reference pixels are loaded from external memory on real demand. When compared to other designs, our proposal can support large search range with very slight hardware overhead, said 75.27K gate counts.

Table 2-IX Comparison of memory bandwidth saving per MB with DRAM latency

Sequences	**CIF**		**Sequences**	**4CIF**
	SR: ±16	**SR: ±32**		**SR: ±64**
Akiyo	17.59%	61.40%	Akiyo	79.15%
Coastguard	14.52%	59.96%	City	78.29%
Football	2.08%	54.13%	Coastguard	78.07%
Foreman	11.72%	58.52%	Football	71.73%
Mobile	13.60%	59.53%	Foreman	76.38%
MD	16.39%	60.83%	Hall	78.71%
News	16.79%	61.00%	Mobile	78.37%
Table tennis	15.41%	60.35%	News	78.77%
Average	**13.51%**	**59.47%**	**Average**	**77.43%**

Table 2-X Comparison of hardware implementation results

Algorithm	**Supporting Resolution**	**Max. Search Range**	**Gate Counts (K)**	**Average PSNR loss**	**On-chip SRAM (Bytes)**	**Frequency (MHz)**	**Process**
[18]	D1@25fps	16X16	165.9	0.00	1.66K	180	0.25um
[19]	CIF@30fps	16X16	89.39	0.08	3.01K	27.8	0.35um
[20]	CIF@30fps	16X16	250	0.10	5K	13.5	0.13um
[32]	CIF@30fps	16X16	68.5	0.19	1.23K	1.67	0.18um
[33]	CIF@30fps	16X16	62.6	0.23	1.08K	1.67	0.18um
Proposed	CIF @30fps	32X32	75.27	0.003	0	5.6	90nm
	4CIF@30fps	64X64		0.005		23	90nm

2.6. Summary

In this paper, an on-demand data bandwidth efficient ME with rate distortion efficient algorithm is proposed. Three properties are revealed in our proposal. First, the on-demand reference data acquiring mechanism leads our proposal only accessing reference data on-demand and can avoid unnecessary reference data loading. Second, through modeling the relationship between the data bandwidth and rate distortion performance, the data bandwidth can be efficiently allocated to the ME process according to the characteristics of video content under the bandwidth constraint. Finally, via the efficient prediction of proposed method, the low data bandwidth requirement ME can also be achieved. Simulation results show that our proposed method can save 79.15% of data bandwidth requirements in maximum with only 0.03dB PSNR drop and 2.50% of bitrate increase when compared with the full search method for search range ±64 in 4CIF resolution. Furthermore, our proposed algorithm can achieve 2.43%, 0.08%, and 0.20% BD-Bitrate saving with 0.17dB, 0.01dB, and 0.01dB BD-PSNR increase on average for high, median, and low motion sequences under the available data bandwidth constraint when compared to the full search motion estimation algorithm. The resulted hardware implementation with simplified bandwidth control scheme reveals that our proposed ME design can process 30 frames in 4CIF resolution per second when running at 23MHz operating frequency with only 75.27K gate counts and search range independent buffer.

Chapter 3

Data and Computation Efficient Inter Predictor Design for H.264/AVC Scalable Extension

3.1. Introduction

To achieve frame resolution adaptation, the SVC [11] adopts layered coding structure with pyramid relationship between success images as shown in Fig. 3.1 to aim at spatial scalability. However, since the content between consecutive frame layers is almost equal to each other except the resolution changing, it strongly exists data similarities between two image layers and thus leading to the possibility of reusing the information for prediction. To fully exploit the data similarities between spatial layers, the SVC additional includes Inter-layer prediction modes to help the exploiting of further rate distortion performance improvement. In Inter-layer prediction mode, the information of spatial base layers is used as the reference for the prediction of spatial enhancement layers. However, the inclusion of additional Inter-layer prediction modes significantly increases the computational complexity as well as data bandwidth requirements and thus results in the hardware design and implementation difficulties. To solve high computation complexity problem of Inter-layer prediction, many literatures [44]-[47] have been proposed. However, these literatures didn't address the issue of data access bandwidth which is the most critical part in video coding system hardware implementations. Works [48]-[49] proposed some low data bandwidth approaches to lighten the problem of heavy data access of Inter-layer prediction mode in SVC. However, the data access bandwidth still remains high in SVC especially for HDTV applications. Therefore, the main target of this chapter is to propose several data efficient algorithms to reduce the data access bandwidth of Inter and Inter-layer prediction modes. In the following subsections, we will introduce all supported Inter-layer prediction modes adopted in SVC in detail and point out what the design and implementation problems they brought to. Afterwards, several data efficient algorithms are proposed and introduced in detail. In addition, two commonly adopted low data bandwidth motion estimation algorithms are reviewed since they will be adopted in our proposed low data bandwidth Inter predictor design.

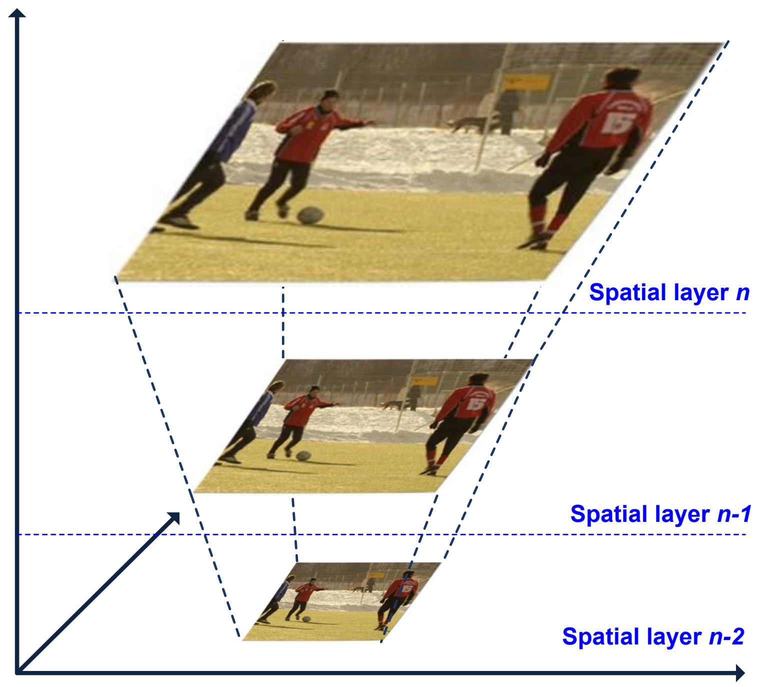


Fig. 3.1. Layered coding structure of SVC spatial scalability

3.1.1. Introduction to Inter-Layer Prediction Modes in H.264/AVC Scalable Extension

In addition to the inherent prediction modes supported in H.264/AVC (Inter16×16, Inter16×8, Inter8×16, Inter8×8, Inter8×4, Inter4×8, Inter4×4, Intra16×16, Intra8×8, and Intra4×4), four more macroblock prediction modes called Inter-layer motion, Inter-layer residual, InterBL, and Inter-layer intra prediction, which can be arbitrary combined together to form a specific prediction mode as shown in Fig. 3.2, are additionally supported to encode the macroblocks of enhancement layers in SVC. In these Inter-layer prediction modes, the base layer information is used as reference for the prediction purpose in spatial enhancement layers to further increase the coding performance. In the following subsections, the four Inter-layer prediction modes adopted in SVC are briefly described as follows.

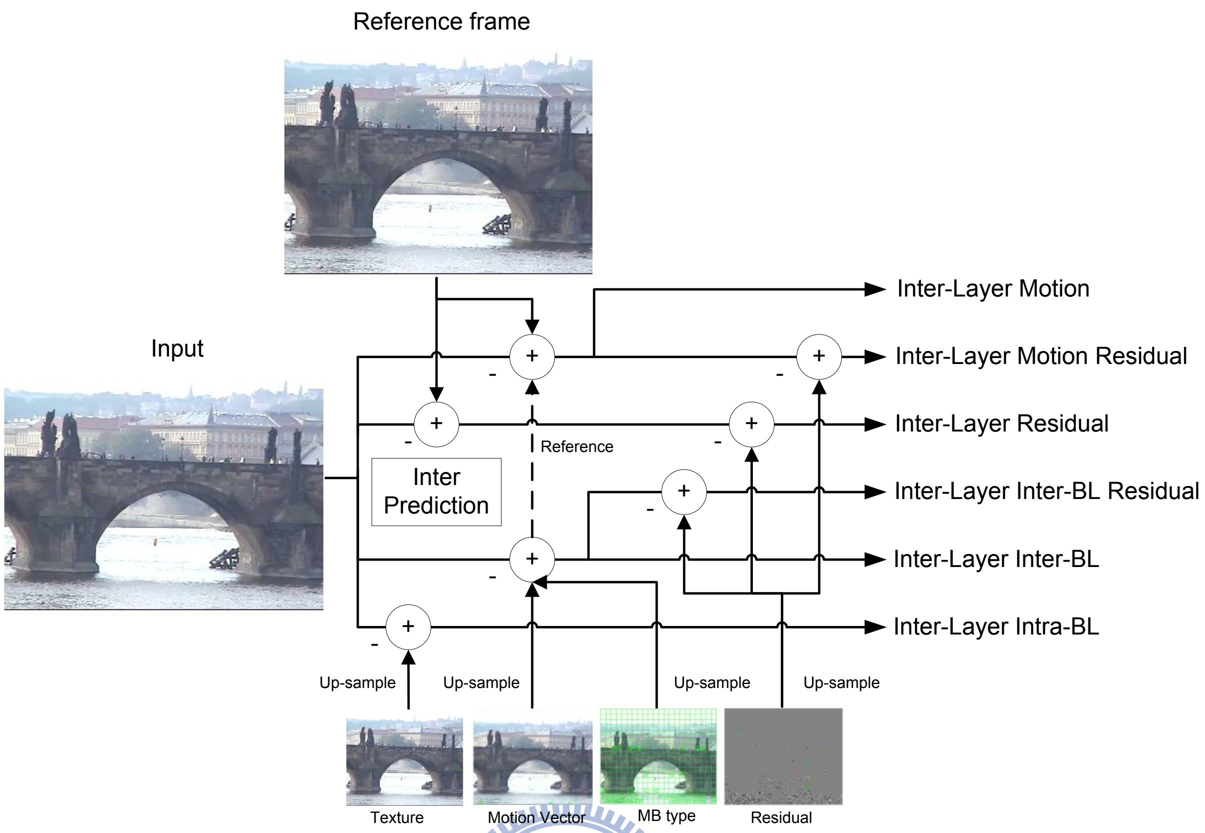


Fig. 3.2. Supported prediction mode combination of Inter-layer prediction

3.1.1.1. Inter-layer Motion Prediction

In this prediction mode, when the enhancement layer as well as the base layer is Inter prediction mode, the motion information of base layer can be used as reference for prediction in enhancement layer as shown in Fig. 3.3. In this prediction mode, the macroblock partition of the enhancement layer is acquired from the corresponding 8×8 block of the base layer associated with a scaling operation. For example, if the prediction mode of corresponding 8×8 block size is 4×8, the block size of 4×8 is scaled to 8×16 block size when the frame resolution ratio is two between base layer and enhancement layer. In addition to the block size, the motion vectors of the enhancement layer are obtained by multiplying the motion vectors of corresponding 8×8 block size in base layer by two. Furthermore, the up-sampled motion information is used to refine the search results.

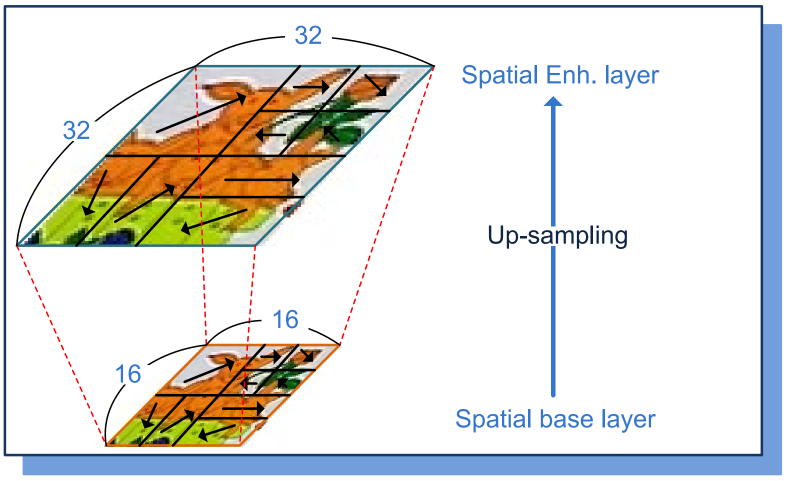


Fig. 3.3. Illustration of Inter-layer motion prediction

3.1.1.2. Inter-layer Residual Prediction

Fig. 3.4 shows the concept of Inter-layer residual prediction mode. When Inter-layer residual prediction is applied, the residual data is up-sampled from corresponding 8×8 block of the base layer by bilinear interpolation. Afterwards, the up-sampled residuals are used for predicting the residuals of the current macroblock in the enhancement layer.

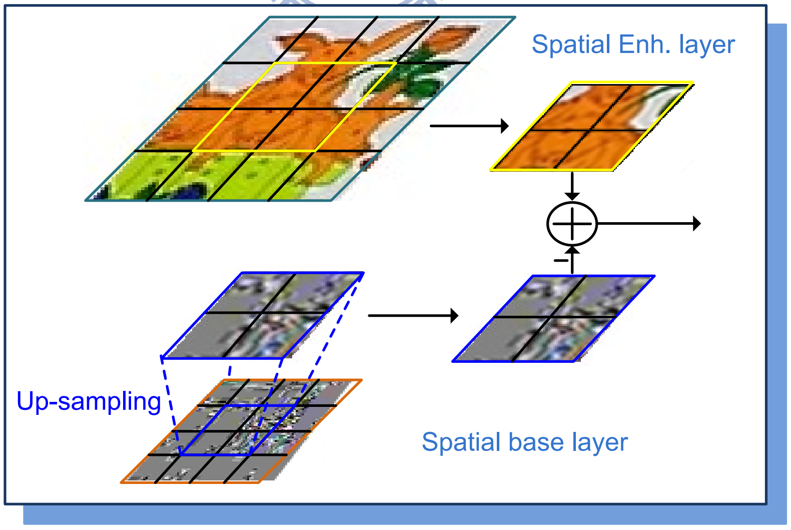


Fig. 3.4. Illustration of Inter-layer residual prediction

3.1.1.3. Inter-layer Intra Prediction

Inter-layer intra prediction can be employed for the macroblock in the enhancement layer if the corresponding block in base layer is intra mode block. That is, the enhancement layer macroblock can be predicted by up-sampling the reconstructed macroblock of the base layer and the concept of Inter-layer intra prediction is shown in Fig. 3.5. For up-sampling the reconstructed macroblock in base layer, one-dimensional four-tape and bilinear filer are used for up-sampling the luminance and chrominance components, respectively.

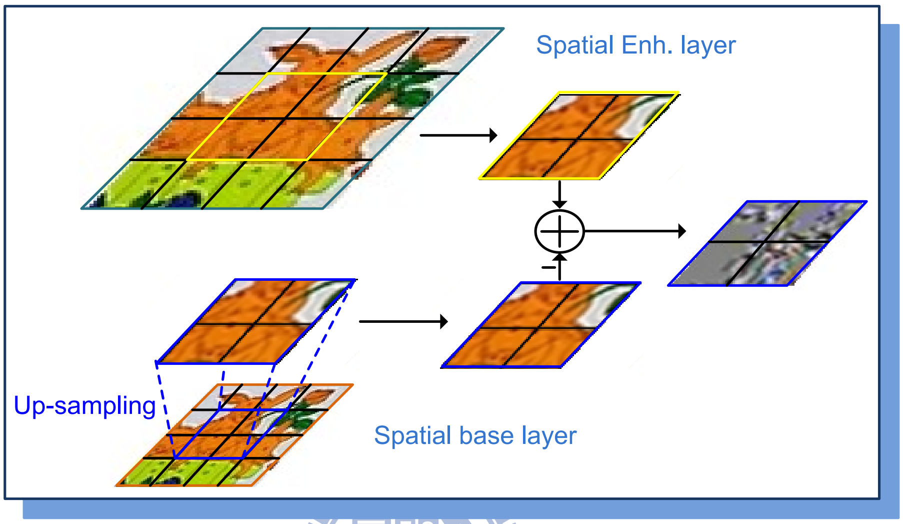


Fig. 3.5. Illustration of Inter-layer intra prediction

3.1.2. Problem Description

As mentioned in previous section that the Inter prediction dominates the overall video coding system performance due to external memory access. Consequently, this situation is more significant in SVC due to the additional adoption of Inter-layer prediction modes. From the operation of Inter-layer motion prediction, it can be found that the search procedure of Inter-layer motion prediction is almost the same as the Inter prediction except that the motion vector predictors are different. In other words, the different motion vector predictors imply that different reference data should be loaded separately for the search process and thus increase the data access requirements. In addition, the Inter-layer residual prediction also significant increases the data access requirements due to it is more reasonable to store the residual information in external memory instead of internal memory when implementing the

Inter prediction hardware. Fig. 3.6 shows the external memory bandwidth requirement comparison for H.264 and SVC video coding standards in terms of Inter prediction. This figure is derived by using QCIF image format and ±8 search range. From this figure, it can be seen that the external memory bandwidth requirements of Inter prediction in SVC is increased up to 186.4% larger than that of the external memory bandwidth requirements of H.264. It has to be pointed out that only two spatial layers with equal frame resolution have been used to conduct the results of Fig. 3.6. However, if more spatial layers have been supported in SVC, the external memory bandwidth requirements will be increased significantly and thus seriously affect the overall performance of video coding system. Therefore, investigating the possibility of reducing the external memory bandwidth of Inter prediction in SVC to improve the coding performance gains the highly demands in the research field of SVC. As a result, the primitively goal of this chapter is to propose several low data bandwidth requirement prediction techniques for Inter-layer prediction in SVC to lighten the problems mentioned above.

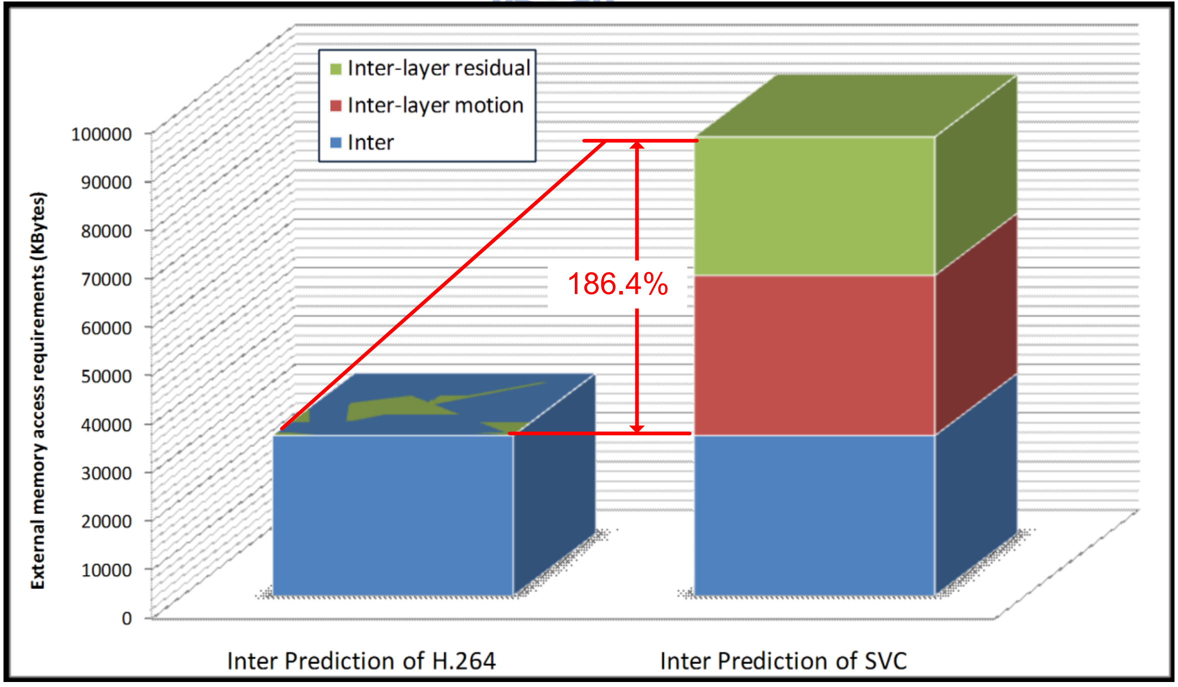


Fig. 3.6. External memory bandwidth requirement comparison for H.264 and SVC

3.1.3. Introduction to Bandwidth Efficient Motion Estimation Algorithms

To alleviate the heavy data bandwidth problem of motion estimation, many literatures have been proposed recently. However, the most suitable and commonly adopted approach in implementing motion estimation hardware is the Level C data reuse scheme [16]. For short, as mentioned in previous chapter, the main concept of Level C data reuse scheme is tried to avoid the loading of overlapped reference data. Although the Level C data reuse scheme can achieve better data bandwidth requirement reduction, the needs of data bandwidth still high especially in high resolution video applications. As a result, an alternative way is to adopt the multi-level [50] concept to realize the motion estimation hardware. Fig. 3.7 illustrates the concept of three-level motion estimation design with different resolutions and different search range sizes. The Level 0 is the finest level which has no pixel sub-sampling and use search range of -8 to +7 to search the best result surrounding the motion vector predictor. The Level 1 adopts quarter-pixel sub-sampling technique to reduce the reference data access requirements and it uses the search range of -64 to +62 to find out the best result surrounding the origin position (0,0). The concept of Level 2 is very similar to Level 1 approach except that the sixteenth-pixel sub-sampling technique and search range of -128 to +124 have been adopted for the search procedure. The benefits of multi-level motion estimation can be listed as follows. For the slow and medium motion sequences, the motion estimation in Level 0 is able to catch best result due to the helpfulness of motion vector predictor. However, for the high and extreme high motion sequences, the widest search range size supported in Level 1 and Level 2 motion estimation can help to catch the best results which can't been caught by Level 0 motion estimation. In addition, since the search origin of Level 1 and Level 2 motion estimation is surrounding the (0,0) position, the Level C data reuse scheme can be further applied in Level 1 and Level 2 to reduce the reference data access requirements. Furthermore, the sub-sampling technique adopted in Level 1 and Level 2 motion estimation is also helpful for the reference data access requirements reduction.

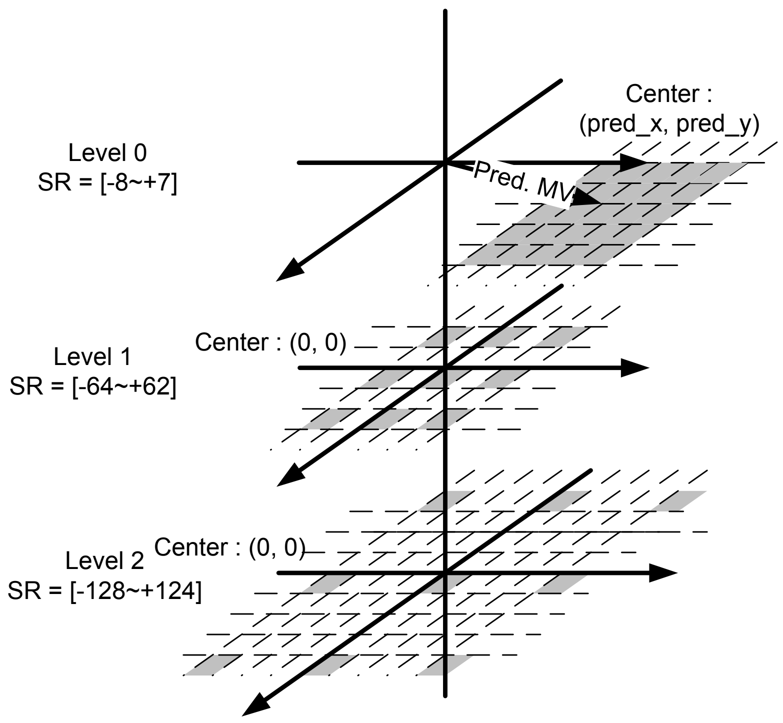


Fig. 3.7. Illustration of multi-level motion estimation

3.1.4. Organization of this Chapter

The rest of this chapter is organized as follows. In subsection 3.2, we described our proposed data reuse algorithm for Inter and Inter-layer residual prediction in detail. The proposed low data bandwidth Inter and Inter-layer motion prediction algorithm is introduced in subsection 3.3. Subsection 3.4 explains our proposed data efficient InterBL prediction algorithm in detail. Furthermore, all proposed algorithms are combined together to form a complete Inter prediction algorithm for SVC and the combined algorithm will be presented in subsection 3.5. Finally, the summary is given in subsection 3.6.

3.2. Inter and Inter-Layer Residual Prediction Data Reuse Method

In the Inter-layer prediction, the information including motion information, texture, and residuals are up-sampled from the base layers and used for the prediction in enhancement layers. From our observation, we found that some common parts of data and computation can be reused during the Inter-layer prediction process. Therefore, in this subsection, we first analyze the matching criteria of different prediction modes to find out the common parts and distinct parts. Afterwards, based on our analysis, we propose our data reuse method for Inter and Inter-layer prediction.

3.2.1. Matching Criteria of Different Prediction Modes

To evaluate which prediction mode should be selected as best mode or which position should be chosen as the best matching position, the rate distortion cost is widely adopted in most of video coding standards to achieve best tradeoff between coding rate and distortion. The rate distortion cost (*RDCost*) can be expressed as follows.

$$RDCost = D + \lambda \cdot R \tag{3-1}$$

where λ is the Lagrange multiplier, R refers to the rate for coding the motion information, and D is the distortion which can be defined as follows.

$$D(x, y) = \sum_{i=0}^{block\_x} \sum_{j=0}^{block\_y} |C(i, j) - F(x + i, y + j)| \tag{3-2}$$

where C and F are the current and reference pixels, respectively.

However, for different prediction modes, the calculation of distortion term D has slight difference. The distortion term of Inter prediction D_{Inter} can be calculated as follows.

$$D_{Inter}(x, y) = \sum_{i=0}^{block\_x} \sum_{j=0}^{block\_y} |C(i, j) - F(x + i, y + j)| \tag{3-3}$$

For Inter-layer residual prediction, the distortion term D_{ILR} can be calculated as follows.

$$D_{ILR}(x, y) = \sum_{i=0}^{block\_x} \sum_{j=0}^{block\_y} |C(i, j) - B(i, j) - F(x + i, y + j)| \tag{3-4}$$

where B is the residuals up-sampled from base layer.

For Inter-layer motion prediction, the distortion term D_{ILM} can be calculated as follows.

$$D_{ILM}(x', y') = \sum_{i=0}^{block\_x} \sum_{j=0}^{block\_y} |C(i, j) - F(x' + i, y' + j)| \tag{3-5}$$

For Inter-layer motion + residual prediction, the distortion term D_{ILMR} can be calculated as follows.

$$D_{ILMR}(x',y') = \sum_{i=0}^{block\_x} \sum_{j=0}^{block\_y} |C(i,j) - B(i,j) - F(x'+i, y'+j)| \quad (3\text{-}6)$$

From above distortion terms calculation, we can observe some properties. First, for the Inter and Inter-layer motion prediction mode, the only difference between D_{Inter} and D_{ILM} calculation is only on that the motion vector predictor. However, although only slight difference has been observed from the equations of D_{Inter} and D_{ILM}, the reference data is quite different actually between these two prediction modes since different motion vector predictors will acquire different reference data for the search process. Therefore, there is nothing except current data can be reused when computing the D_{Inter} and D_{ILM}. Second, for the Inter and Inter-layer residual prediction modes, we can observe that the only difference between these two prediction modes is that the residual information should be subtracted from the current data ahead as Fig. 3.8 shown. In other words, the current and reference data can be reused for both of Inter and Inter-layer residual prediction modes. Similarly, the same situation can also be seen from the Inter-layer motion and Inter-layer motion + residual prediction modes as Fig. 3.9 shown. As a result, through the reusing of current and reference data for different prediction modes, the data bandwidth requirements and computational components can be reduced significantly.

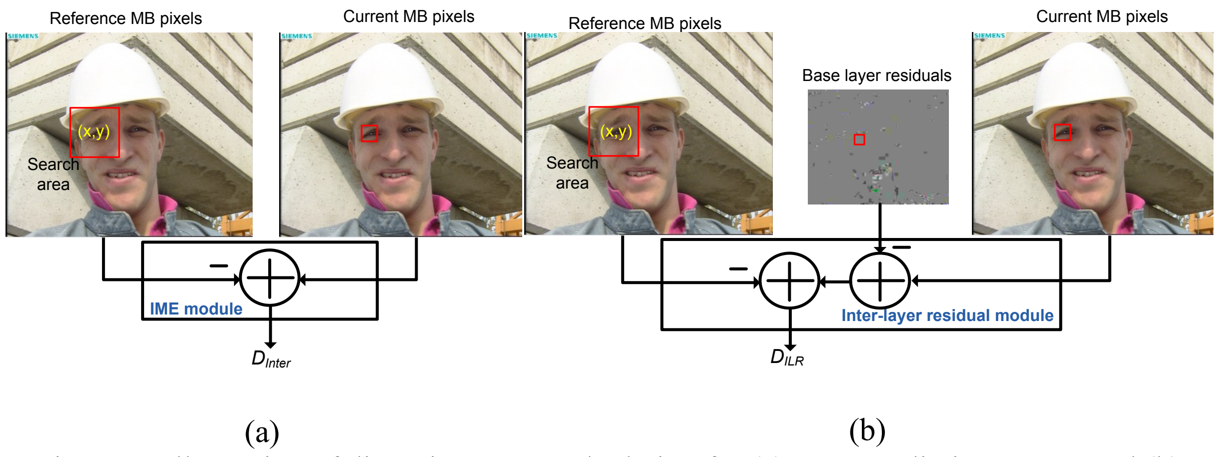
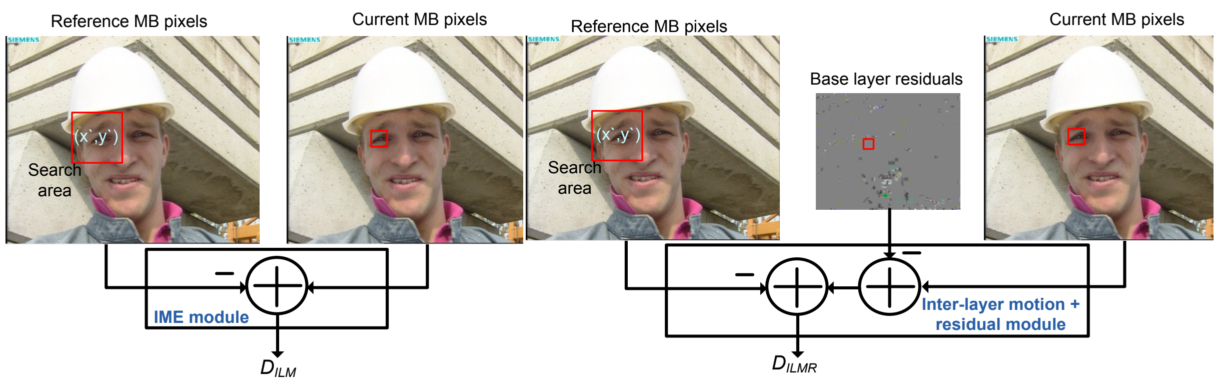


Fig. 3.8. Illustration of distortion terms calculation for (a) Inter prediction : D_{Inter} and (b) Inter-layer residual prediction : D_{ILR}



(a) (b)

Fig. 3.9. Illustration of distortion terms calculation for (a) Inter-layer motion prediction : D_{ILM} and (b) Inter-layer motion + residual prediction : D_{ILMR}

3.2.2. Proposed Data Reuse Method

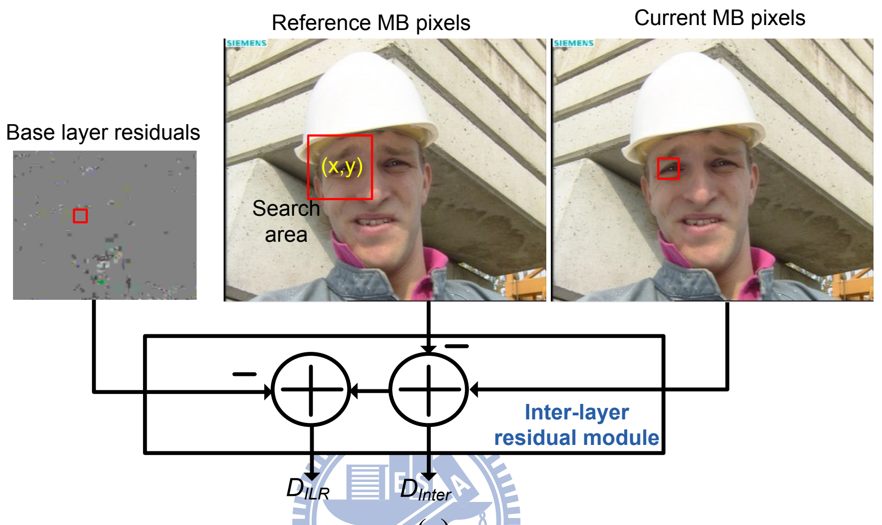
Fig. 3.10 exhibits our proposed data reuse scheme for different prediction modes. Compared to the original form of distortion terms calculation, we only slightly change the subtraction order of distortion terms calculation in our proposal. The benefits of our proposal are described as follows. First, the computational modules can be shared in hardware implementation. In the original form, two hardware modules may be implemented to generate the distortion terms of D_{ILR} and D_{Inter} or D_{ILMR} and D_{ILM} intuitively. However, through our proposed data reuse scheme, the distortion term of one prediction mode can be generated first and the other one will be generated right after so that the unnecessary repeated computations can be avoided. For example, through our proposal, the distortion term of D_{ILR} can be calculated following the calculation of D_{Inter}. Hence, the repeated subtractions for current and reference data can be saved. Second, the reference data can be reused for different prediction modes. More precisely, the reference data for Inter mode prediction can be reused for the Inter-layer residual prediction. Similarly, the reference data for Inter-layer motion prediction can be reused for the Inter-layer motion + residual prediction as well. As a result, the distortion terms calculation can be redefined as follows when applying our proposed data reuse scheme.

For Inter-layer residual prediction, the distortion term D_{ILR} can be calculated as follows.

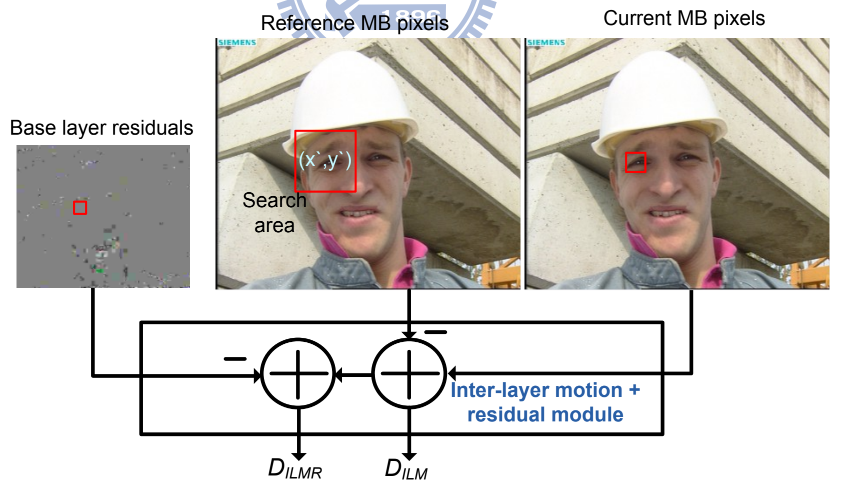
$$D_{ILR}(x,y)=\sum_{i=0}^{block\_x}\sum_{j=0}^{block\_y}|C(i,j)-F(x+i,y+j)-B(i,j)| \quad (3\text{-}7)$$

For Inter-layer motion + residual prediction, the distortion term D_{ILMR} can be calculated as follows.

$$D_{ILMR}(x',y')=\sum_{i=0}^{block\_x}\sum_{j=0}^{block\_y}|C(i,j)-F(x'+i,y'+j)-B(i,j)| \quad (3\text{-}8)$$



(a)



(b)

Fig. 3.10. Proposed data reuse scheme for (a) Inter and Inter-layer residual prediction and (b) Inter-layer motion and Inter-layer motion + residual prediction

3.2.3. Results

Since our proposed data reuse scheme is lossless approach, there is no rate distortion performance difference between our proposal and original form. However, for the data bandwidth savings, our proposal can achieve 44.45% data bandwidth savings per macroblock when search range of ±8 is applied.

3.3. Low Bandwidth Inter and Inter-Layer Motion Prediction Algorithm

In the supported prediction modes in SVC, the Inter and Inter-layer motion prediction have almost the same operations except that the different motion vector predictors have been individually applied as the search center. However, since the difference between successive spatial layers is only the variation of frame resolution, it can be expected that the motion vector predictors of Inter and Inter-layer motion prediction mode may be very closer to each other. In this situation, there might has large portion of overlapped area between two search windows as Fig. 3.11 shown. Motivated from this phenomenon, we are tried to analyze the relationship between motion vector predictor of Inter (MVP_{Inter}) and motion vector predictor of Inter-layer motion (MVP_{ILM}) prediction modes, and seeking out the possibility that can be used to reduce the data bandwidth requirements.

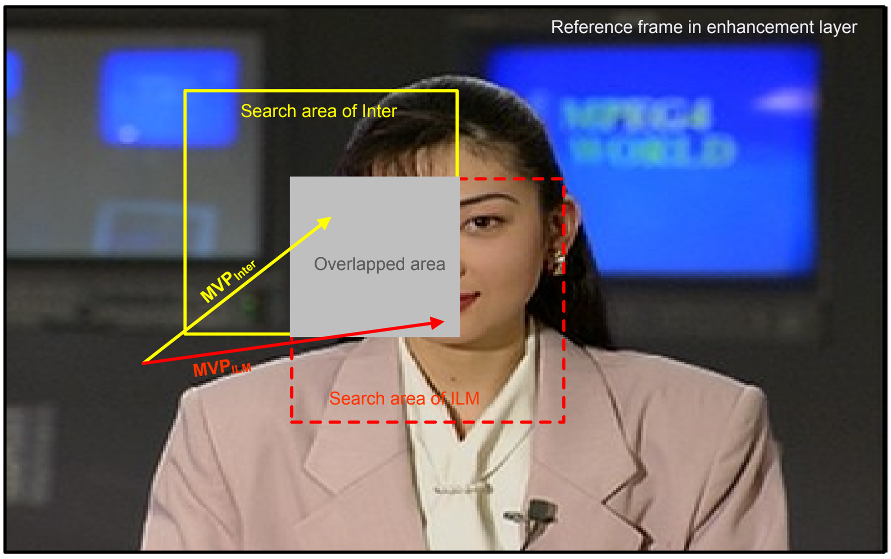


Fig. 3.11. Illustration of search window overlapping between the search area of Inter and Inter-layer motion prediction

3.3.1. Analysis for Motion Vector Predictors

In this subsection, we first analyze the relationship between motion vector predictor of Inter and Inter-layer motion prediction. To measure how close that two motion vector predictors is, we define the term of motion vector predictor difference as follows.

$$Diff_{MVP}(i) = |MVP_{Inter}(i) - MVP_{ILM}(i)|, i \in x, y \tag{3-9}$$

where $MVP_{Inter}(i)$ stands for the motion vector predictor of Inter prediction along i direction and $MVP_{ILM}(i)$ is the motion vector predictor of Inter-layer motion prediction along i direction.

Fig. 3.12 reveals the accumulated probability of $Diff_{MVP}$ for different test sequences. In this figure, the vertical axis is the accumulated probability and the horizontal axis is the motion vector predictor difference $Diff_{MVP}$. From this figure, we can observe that if the motion vector predictor difference between Inter and Inter-layer motion predictions is less than eight, the accumulated probability can be reached up to 90% for all sequences both in vertical (Fig. 3.12 (a)) and horizontal (Fig. 3.12 (b)) directions. Therefore, it is highly possible to find out the best result for either prediction mode from the reference data of another prediction mode due to the MVPs' differences between two prediction modes are very small. Consequently, the search data of Inter and Inter-layer motion can be reused for the prediction usage if the $Diff_{MVP}$ is less than a predefined threshold.

From above analyses, it is very clear that the motion vector predictor differences are very small between Inter and Inter-layer motion prediction modes. However, reusing the reference data of Inter prediction or Inter-layer motion prediction for the search process is one issue. In our proposal, the reference data of Inter prediction mode is reused for the prediction of Inter-layer motion prediction since the motion estimation for Inter prediction mode is ahead of Inter-layer motion prediction.

Although the reference data of Inter prediction mode can be reused for the Inter-layer motion prediction, the issue is therefore became how large the search range is sufficient to cover the final motion vector difference of Inter-layer motion prediction (*MVD_{ILM}*). If the search range is too small, the proposed data reuse scheme might still result in the ill coding performance. However, if the search range is too large, it wastes the data bandwidth. Therefore, in the following analyses, we discuss the relationship between *$Diff_{MVP}$* and *MVD_{ILM}* by using the conditional probability as defined as follows.

$$P(SR|S) = P(MVD_{ILM}(i) \leq SR | Diff_{MVP} = S) | SR \in \{1,4,8,12,16\}, S \in [1 \sim 20] \quad (3\text{-}10)$$

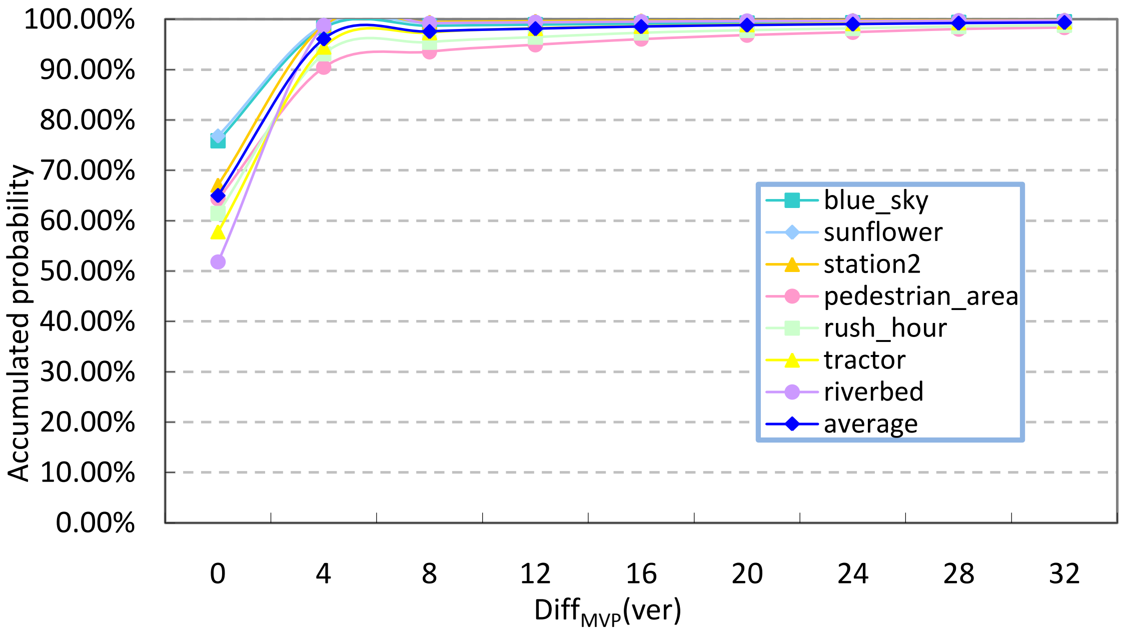
$$MVD_{ILM}(i) = |MV_{ILM}(i) - MVP_{Inter}(i)|, i \in x, y \quad (3\text{-}11)$$

where *SR* is the search range and *$MV_{ILM}(i)$* stands for the final motion vector of Inter-layer motion prediction mode along the *i* direction. It should be mentioned that since the reference data of Inter prediction mode is reused for Inter-layer motion prediction mode, the *MVD_{ILM}* is thus calculated by using the motion vector predictor of Inter prediction mode *MVP_{Inter}*. Table 3-I tabulates the statistical results of Eq.10 and Fig. 3.13 shows the graphical version. In Fig. 3.13, the horizontal axis is the index of *S* and vertical axis refers to the probability. In addition, each curve represents a case of different *SR* size. From these statistical results, we can observe some properties. First, the probability is decreased with the growing of *S*. However, it is very easy to explain this situation. From the definition of *S*, it can be seen that the *S* is used to represent *$Diff_{MVP}$*. For larger *S*, it stands for that the motion vector predictor difference between Inter and Inter-layer motion prediction modes is much larger and thus harder to find out the best results from the reference data of Inter prediction mode. Second, the probability is dropped off slowly for the larger *SR*. For example, when the search range is fixed to 16, the probability can reach 80% even the *S* is up to 16. This property can be explained that the larger *SR* is much easier to cover the final results intuitively.

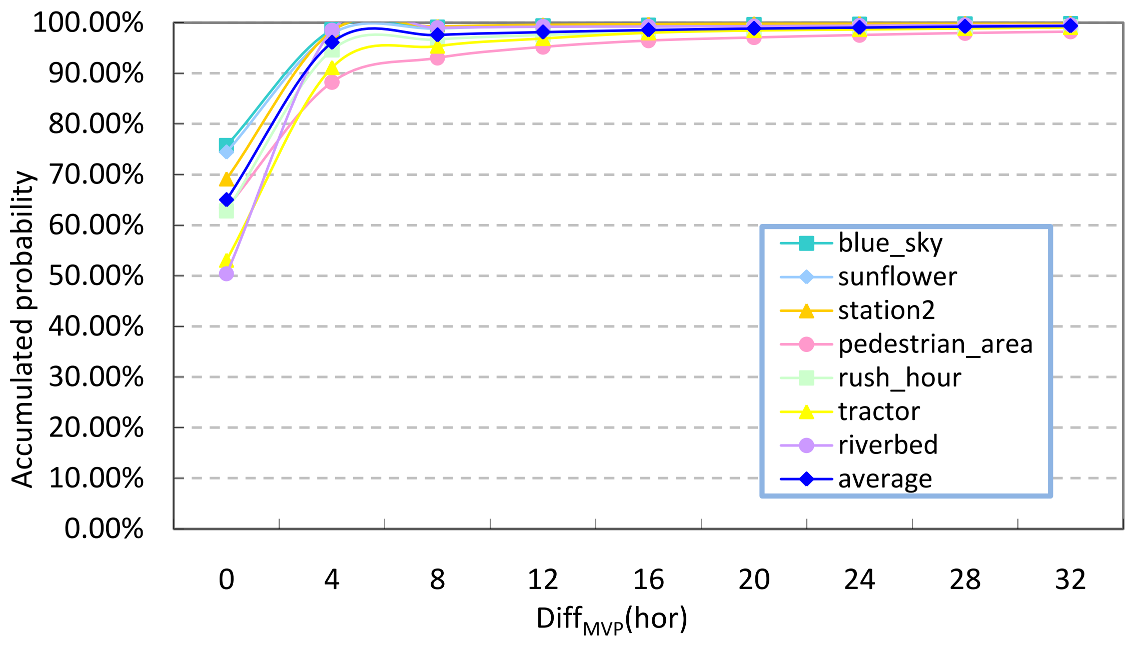
From the above analyses, we derive two conclusions. The first conclusion is that up to 90% of *$Diff_{MVP}$* is less than ±8 for the search range *SR* larger than that of eight. Second, when the search range is fixed to ±8, the probability can reach up to 80% while the *$Diff_{MVP}$* is smaller than eight. As a result, both of the threshold and search range are set to eight in our proposal.

Table 3-I The analytical results obtained by Eq.3-10

	Vertical direction					Horizontal direction				
S	**SR:1**	**SR:4**	**SR:8**	**SR:12**	**SR:16**	**SR:1**	**SR:4**	**SR:8**	**SR:12**	**SR:16**
1	0.889	0.955	0.972	0.979	0.985	0.875	0.954	0.981	0.988	0.993
2	0.413	0.910	0.950	0.964	0.974	0.418	0.919	0.962	0.975	0.984
3	0.326	0.832	0.906	0.932	0.949	0.332	0.877	0.947	0.966	0.976
4	0.298	0.795	0.903	0.932	0.946	0.344	0.813	0.938	0.963	0.975
5	0.323	0.556	0.884	0.916	0.936	0.311	0.510	0.924	0.955	0.970
6	0.324	0.490	0.852	0.904	0.931	0.396	0.541	0.921	0.953	0.972
7	0.282	0.415	0.783	0.851	0.895	0.376	0.495	0.900	0.946	0.963
8	0.372	0.506	0.812	0.897	0.926	0.354	0.463	0.845	0.929	0.957
9	0.315	0.471	0.627	0.869	0.905	0.297	0.408	0.567	0.898	0.940
10	0.250	0.390	0.493	0.786	0.858	0.317	0.455	0.541	0.902	0.941
11	0.305	0.429	0.527	0.794	0.886	0.295	0.414	0.499	0.856	0.933
12	0.261	0.362	0.471	0.747	0.836	0.272	0.386	0.451	0.816	0.926
13	0.276	0.391	0.484	0.575	0.858	0.347	0.443	0.522	0.657	0.923
14	0.248	0.354	0.448	0.551	0.826	0.330	0.426	0.494	0.575	0.896
15	0.290	0.433	0.517	0.582	0.845	0.354	0.448	0.496	0.552	0.921
16	0.313	0.420	0.509	0.590	0.818	0.357	0.466	0.545	0.598	0.840
17	0.255	0.372	0.448	0.536	0.669	0.293	0.381	0.466	0.518	0.652
18	0.296	0.389	0.458	0.517	0.591	0.378	0.486	0.531	0.596	0.665
19	0.287	0.412	0.481	0.551	0.617	0.293	0.403	0.451	0.517	0.584
20	0.236	0.329	0.401	0.487	0.545	0.307	0.411	0.463	0.564	0.636

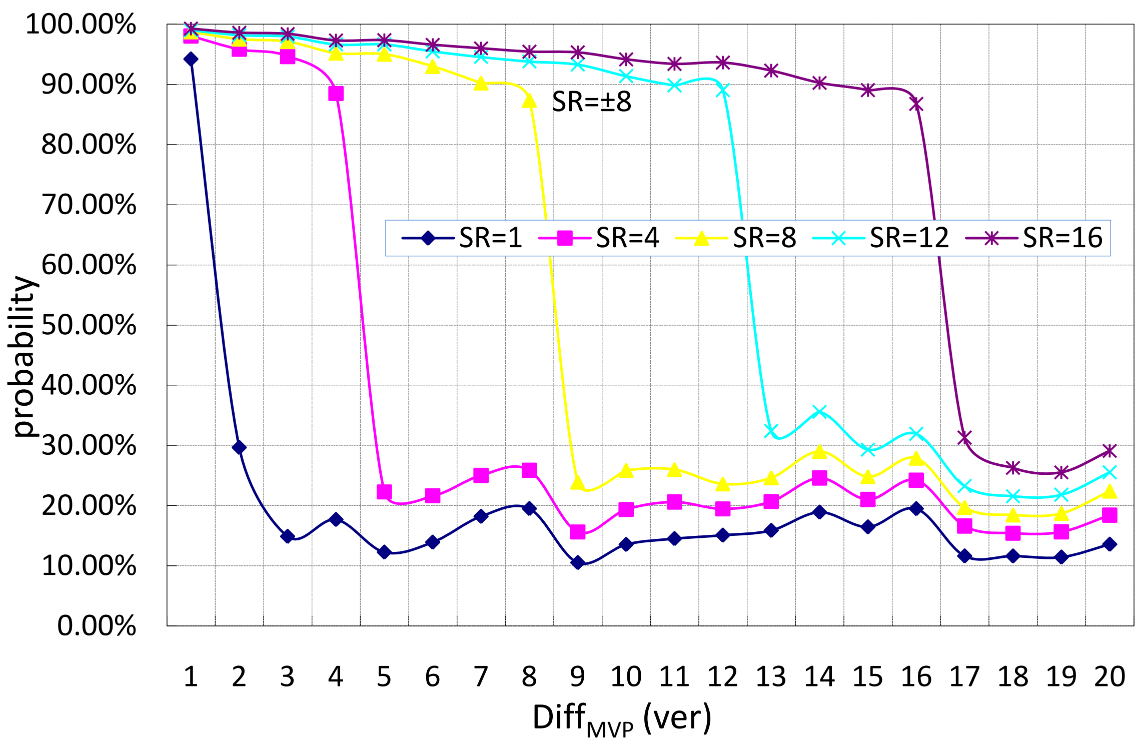


(a)

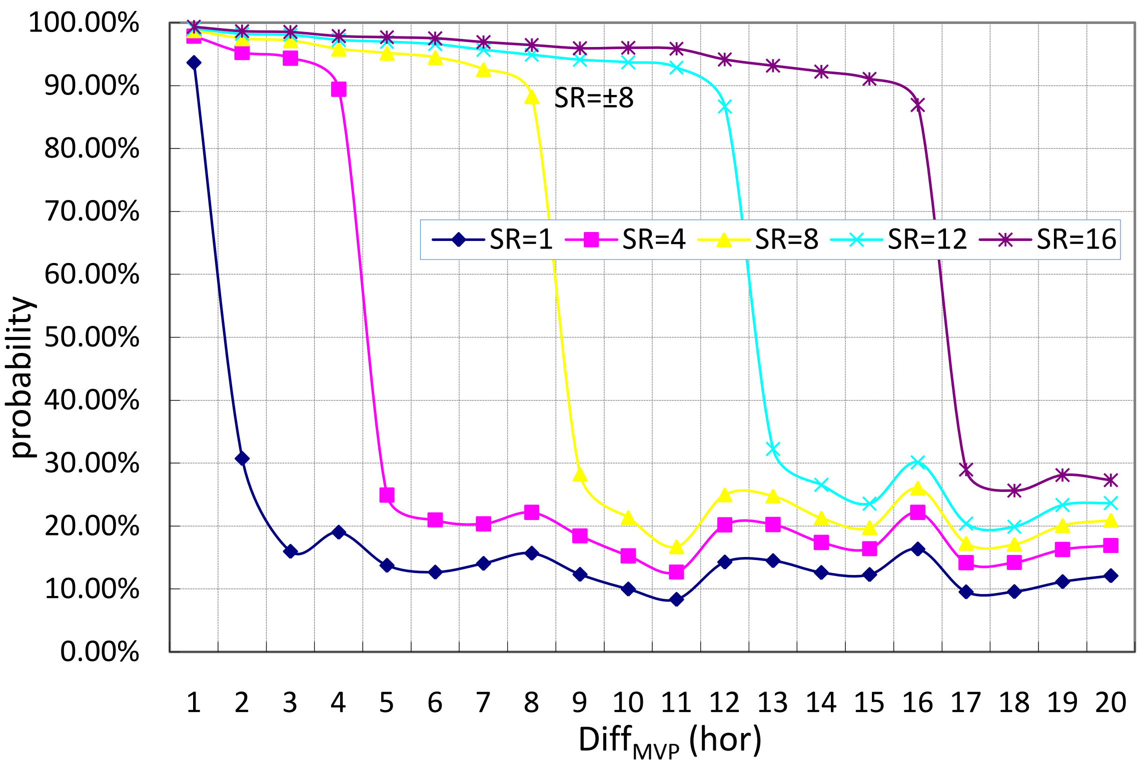


(b)

Fig. 3.12. The accumulated probability of MVP difference between Inter and Inter-layer motion predictions (a) vertical and (b) horizontal axis



(a)



(b)

Fig. 3.13. The statistical results of Eq.10 in (a) vertical and (b) horizontal direction

3.3.2. Proposed Low Bandwidth Data Reuse Method for Inter-Layer Motion Prediction

Based on the observations and analytical results shown in previous subsection, the data reuse scheme for Inter and Inter-layer motion prediction mode is proposed in this subsection. Fig. 3.14 exhibits the flowchart of our proposed algorithm. At the beginning, both of motion vector predictors of Inter and Inter-layer motion prediction are derived individually. For Inter prediction mode, the regular motion estimation operation is executed with its reference data to find out the best result of Inter prediction mode. However, for the Inter-layer motion prediction mode, the motion vector predictor difference $Diff_{MVP}$ is calculated first. Afterwards, the $Diff_{MVP}$ is compared with a predefined threshold th_1 (the th_1 is set to 8 in this dissertation) to determine whether the $Diff_{MVP}$ is less than the th_1 or not. If $Diff_{MVP}$ is larger than th_1, the rate distortion cost of Inter-layer motion prediction mode $RDCost_{ILM}$ is set to infinite to skip the Inter-layer motion prediction mode. Otherwise, the regular motion estimation operations are executed for the Inter-layer motion prediction mode by using the reference data of Inter prediction. Once both of Inter and Inter-layer motion prediction modes have finished their own prediction processes. The best result will be selected in the last step and the best result will be transmitted to the following coding usage.

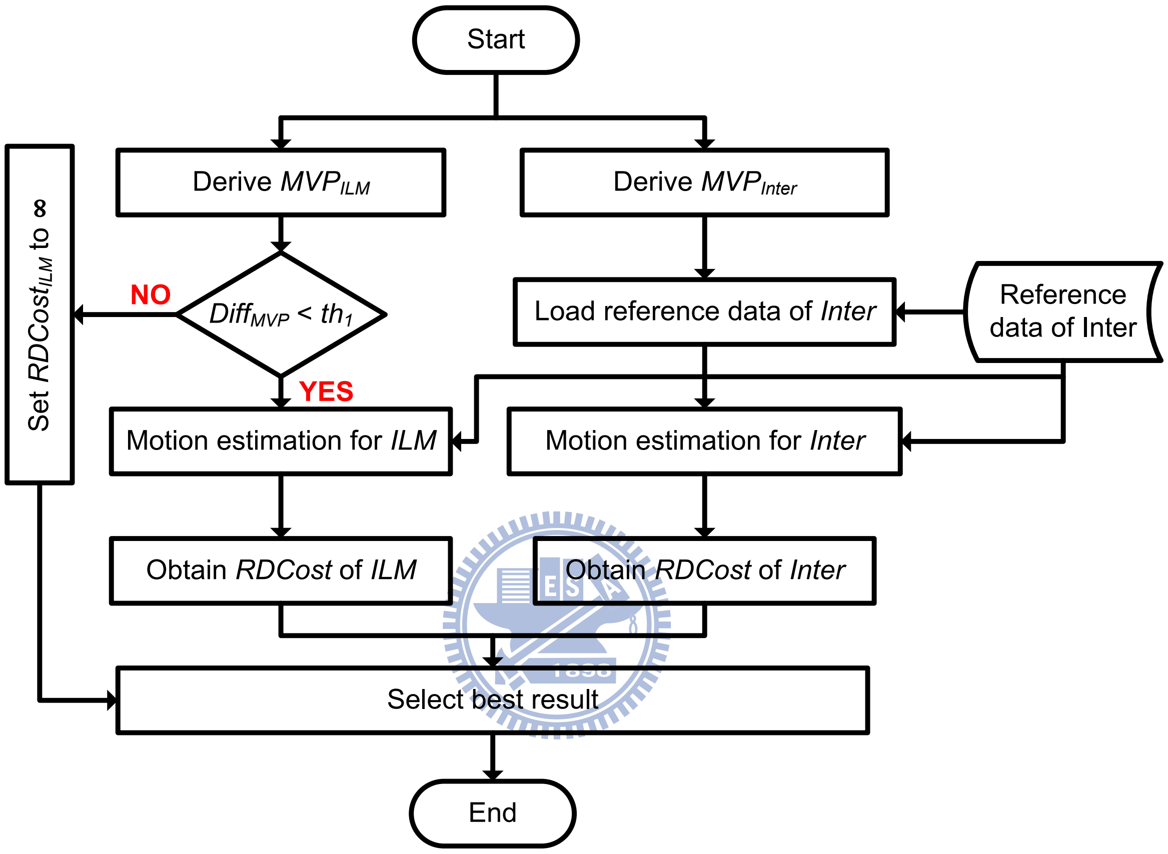


Fig. 3.14. Flowchart of proposed data reuse scheme for Inter and Inter-layer motion prediction mode

In our proposed flowchart, we skip the Inter-layer motion prediction mode if the $Diff_{MVP}$ is smaller than th_1. To exam how the skipped Inter-layer motion prediction mode affects the rate distortion performance, we conduct some simulations to verify and the simulation results are shown in Table 3-II. In this our simulation, two scenarios are proposed to test. The first scenario, we called Case1, is that the Inter-layer motion prediction is executed with reference data of Inter prediction mode. In other words, the Inter-layer motion prediction is forced to search the best result from the reference data of Inter prediction mode. The second scenario, we called Case 2" is that the Inter-layer motion prediction mode is thus skipped. From Table 3-II, we can observe that the rate distortion performance of both cases are almost the same. Therefore, we adopt the Case 2 in our proposed algorithm due to skipping the Inter-layer motion prediction mode can further save the computational complexity when compared to adopting Case 1.

Table 3-II The simulation results of two cases for 480p sequences (Blue_sky, Tractor, Station2, Pedestrain_area, Rush_hour, Riverbed)

	Case 1		Case 2	
QP	ΔPSNR	Δ Bit-rate (%)	ΔPSNR	Δ Bit-rate (%)
28	-0.102	0.590	-0.103	0.575
32	-0.103	0.535	-0.103	0.540

3.4. Data Efficient InterBL Prediction Algorithm

In SVC, a specific prediction mode call InterBL is also supported to achieve better coding performance. In InterBL prediction mode, all motion information including motion vectors, partition sizes, and reference index are up-sampled from base layer for the prediction in enhancement layer. Similar to the motion compensation, the up-sampled motion vectors in InterBL prediction mode will be used to indicate where to derive prediction pixels from the reference frame for current macroblock as Fig. 3.15 shown. Therefore, inspired by the high content similarity between successive spatial layers, we are tried to find out the possibility that whether the reference data of Inter prediction mode can be reused to derive the prediction pixels of InterBL prediction mode. In the following subsections, we first analyze the motion vector relationship between Inter and InterBL prediction mode and propose our data reuse method for Inter and InterBL prediction mode.

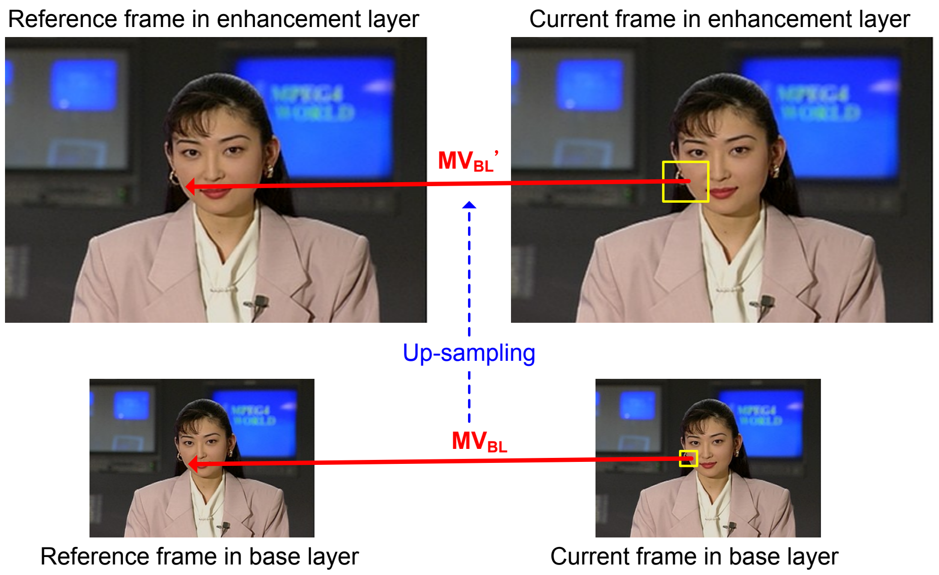

Reference frame in enhancement layer
Current frame in enhancement layer
MV_BL'
Up-sampling
MV_BL
Reference frame in base layer
Current frame in base layer

Fig. 3.15. Illustration of InterBL prediction mode

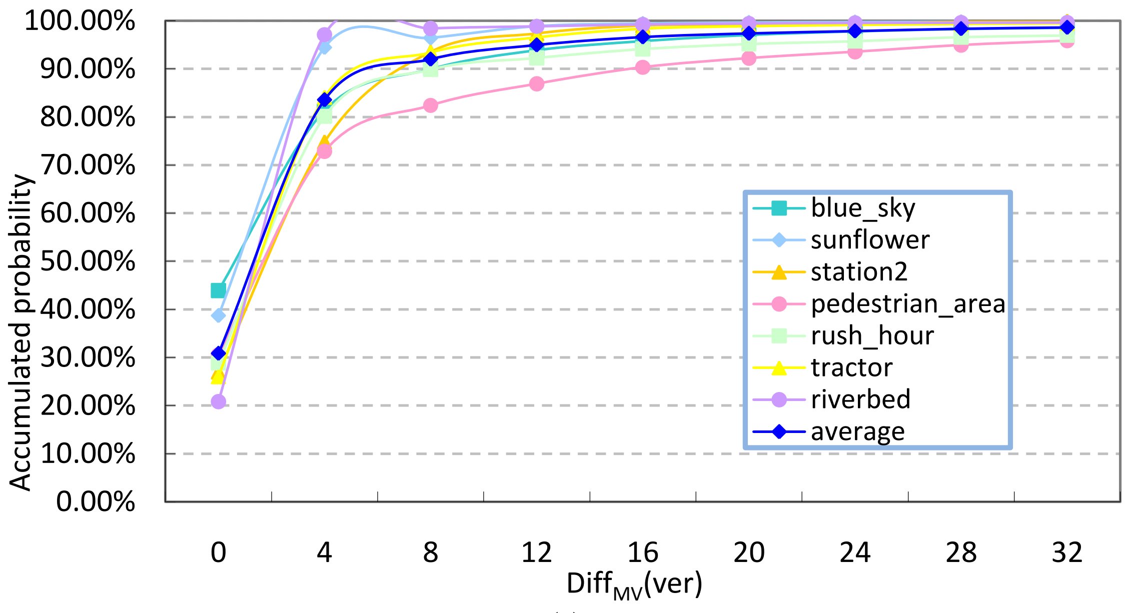
3.4.1. Analysis for Motion Vector Predictor and Motion Vector

Unlike Inter-layer motion prediction mode, the motion vectors in the base layer in InterBL prediction mode are used to derive reference data directly without any motion vector refinement. Therefore, instead of analyzing the motion vector predictor difference just like Inter-layer motion prediction mode does, we analyze the motion vector relationship between motion vector of InterBL prediction mode and motion vector predictor of Inter prediction mode. The motion vector difference for measuring the relationship between motion vector of InterBL prediction mode and motion vector predictor of Inter prediction mode is defined as follows.

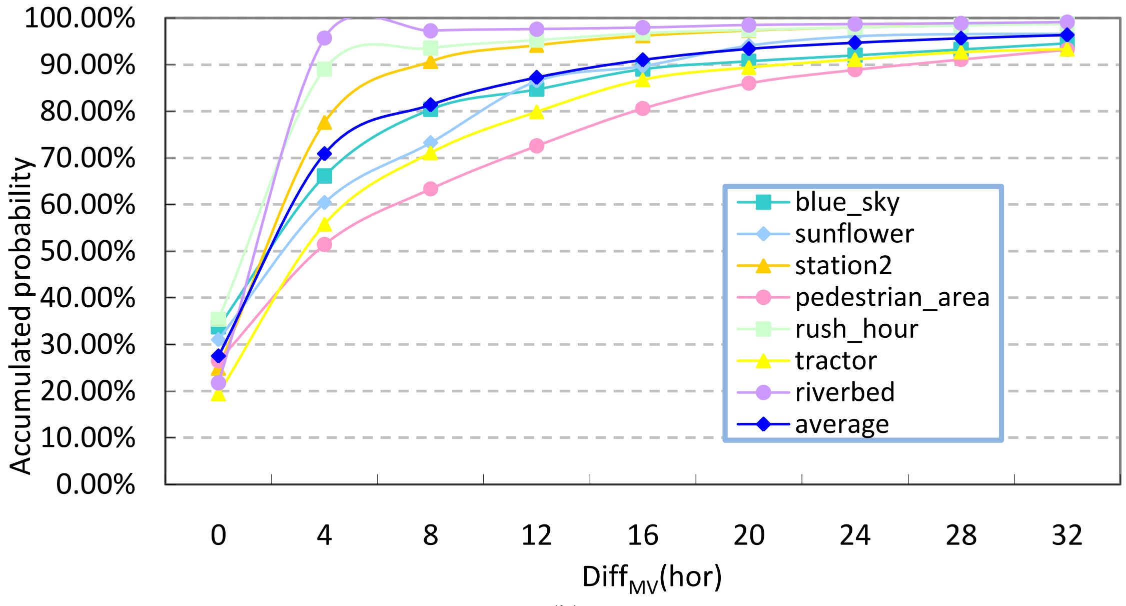
$$Diff_{MV}(i) = |MVP_{Inter}(i) - MV_{InterBL}(i)|, i \in x, y \tag{3-12}$$

where $MV_{InterBL}$ is the motion vector of InterBL prediction mode.

Fig. 3.16 shows the statistical result. From this figure, we can observe that when the motion vector difference between MVP_{Inter} and $MV_{InterBL}$ is smaller than eight pixels, the probability can reach 90% and 80% for vertical and horizontal direction in average, respectively. Since the InterBL mode has high probability to be selected as best mode in enhancement layer, to avoid great performance degradation, it is unreasonable to abandon this prediction mode as data efficient Inter-layer motion prediction algorithm done in case of the reference data outside the search range. Therefore, based on such high correlation between MVP_{Inter} and $MV_{InterBL}$, we further propose an efficient InterBL prediction algorithm by adding an additional checking mechanism which plays the role of determining whether the required reference data of InterBL prediction mode should be loaded from external memory or not.



(a)



(b)

Fig. 3.16. The accumulated probability of MV difference between Inter and InterBL prediction modes (a) vertical and (b) horizontal axis

3.4.2. Proposed Data Efficient InterBL Data Reuse Prediction Algorithm

Fig. 3.17 exhibits the flowchart of proposed data efficient InterBL data reuse prediction algorithm. At the beginning, both of MVP_{Inter} and the number of partitions in the base layer of InterBL prediction mode are derived first. For Inter prediction mode, the regular motion estimation algorithm is applied to search the best result with the help of its reference data. However, for InterBL prediction modes, an iterative operation is executed depending on the number of partitions. For each partition of InterBL prediction mode, the corresponding motion vector is derived in order to obtain the $Diff_{MV}$. Once the $Diff_{MV}$ has been calculated successively, the $Diff_{MV}$ will be compared to a predefined threshold th_2 (the th_2 is set to 8 in this dissertation) to determine whether the reference data of Inter prediction mode should be reused or not. If $Diff_{MV}$ is less than th_2, it implies that the reference data for forming the prediction pixels of InterBL prediction mode can be obtained from the Inter prediction mode reference data memory. As a result, the reference data inside the Inter prediction mode reference data memory will be used to compute the *RDCost* of partition *n* in InterBL prediction mode. Otherwise, if $Diff_{MV}$ is larger than or equal to th_2, it stands for that the reference data for deriving prediction pixels of InterBL prediction mode can't be found in the Inter prediction mode reference data memory. Consequently, the reference data of InterBL prediction mode should be loaded from the external memory individually for the purpose of computing the *RDCost*. Once the *RDCost* of partition *n* has been successfully calculated, the *RDCost* of partition *n* will be accumulated to compose the actual *RDCost* of an macroblock since the adoption of variable block size motion estimation technique leads to the situation that an macroblock could be organized by several portions. Afterwards, the next unprocessed partition *n+1* will be treated as previous operation stated. After all partitions have been processed, the final *RDCost* will be obtained and will be used in the final mode decision process.

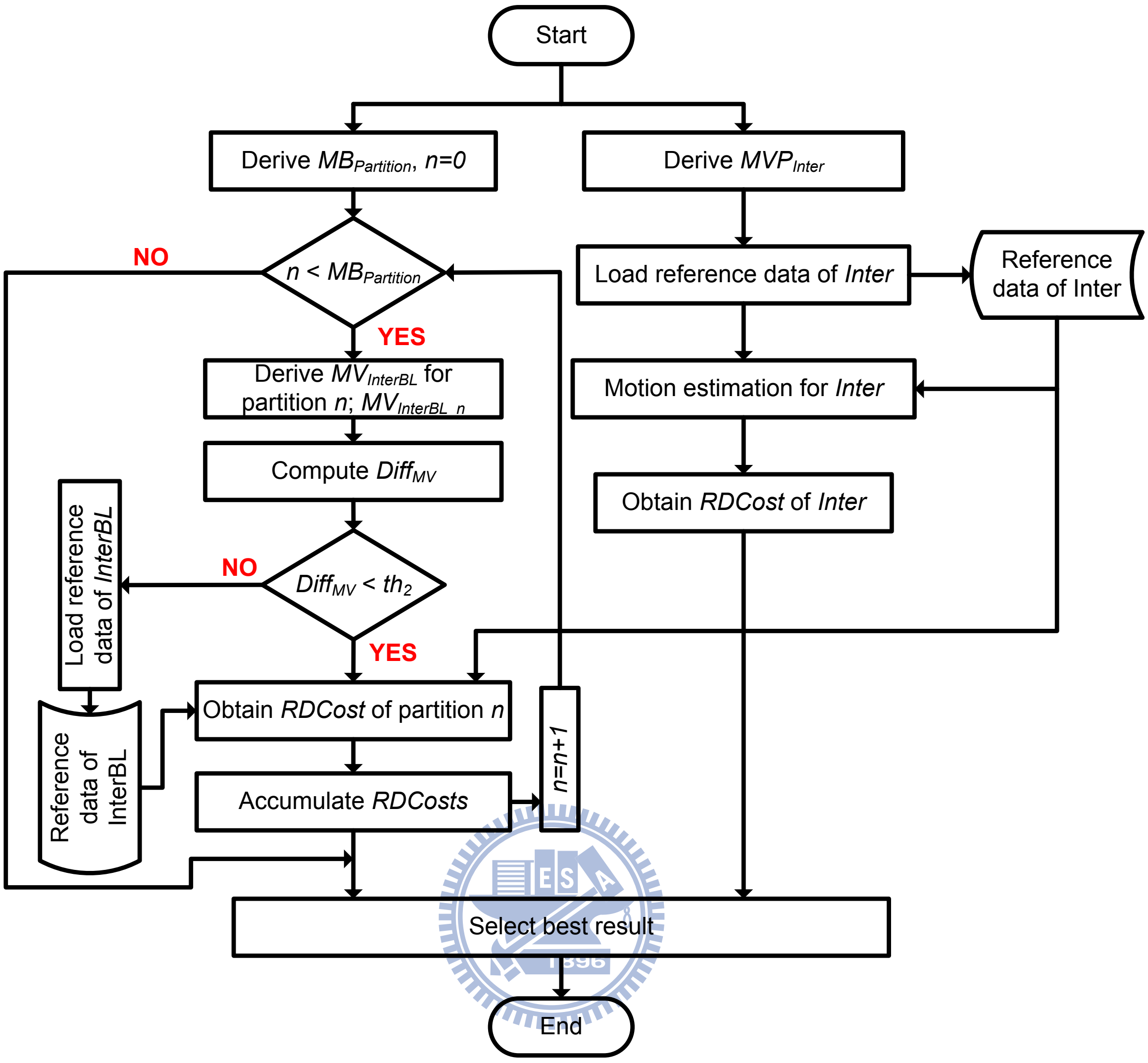

Start
Derive $MB_{Partition}$, $n=0$
Derive MVP_{Inter}
$n < MB_{Partition}$
NO
YES
Load reference data of Inter
Reference data of Inter
Derive $MV_{InterBL}$ for partition n; $MV_{InterBL\_n}$
Motion estimation for Inter
Compute $Diff_{MV}$
Obtain RDCost of Inter
Load reference data of InterBL
$Diff_{MV} < th_2$
NO
YES
Obtain RDCost of partition n
$n=n+1$
Reference data of InterBL
Accumulate RDCosts
Select best result
End

Fig. 3.17. Flowchart of proposed data reuse scheme for Inter and InterBL prediction mode

3.5. Combination of All Low Bandwidth Inter-Layer Prediction Algorithms

In previous subsections, we proposed several data reuse schemes for Inter-layer prediction modes. However, although our proposal can reduce the data access bandwidth significantly, the Inter prediction mode also consumes large portion of data access bandwidth and all our proposed data efficient Inter-layer prediction algorithms have reused the reference data of Inter prediction mode. Therefore, we further discuss the approach for data bandwidth reduction in Inter prediction mode. Afterwards, we propose a frame size adaptive motion estimation switching algorithm to further achieve better data bandwidth reduction. Finally, all proposed algorithm will be combined together to form a complete low data bandwidth Inter predictor design.

3.5.1. Discussion of Data Bandwidth Reduction for Inter Prediction Mode

As mentioned in previous subsection that the most efficient and commonly adopted low data bandwidth motion estimation algorithms are the Level C data reuse full search motion estimation and multi-level motion estimation algorithm. For Level C data reuse full search motion estimation algorithm, although it can achieve noticeable data bandwidth reduction, the data bandwidth requirements are still high especially for high resolution video sequence. Oppositely, the multi-level motion estimation algorithm can reduce large amount of data bandwidth when high resolution video sequences have been applied but the amount of data bandwidth reduction is insignificant in smaller resolution video sequences. As a result, in this subsection, the goal is to find out how large of image resolution is better for Level C data reuse full search motion algorithm and how large of image resolution is better for multi-level motion estimation algorithm. Table 3-III tabulates the data access bandwidth requirements of different frame resolutions and different data efficient motion estimation algorithms. The corresponding search range size in different search level of different frame resolution for multi-level motion estimation algorithm is listed in Table 3-IV. In Table 3-III, the term of Inter layer data reuse stands for whether our proposed low data bandwidth Inter-layer prediction algorithms have been applied or not. From this table, we can observer a very interesting phenomenon that the Level C data reuse full search motion estimation algorithm is much suitable for small resolution sequences such as QCIF and CIF. However, for larger resolution video sequences, such as 408P, 4CIF, 720P, and 1080P, the multi-level motion estimation algorithm has better data bandwidth savings when compared to Level C data reuse full search motion estimation algorithm. Therefore, from the observation, we propose a frame size adaptive motion estimation switching algorithm for better data bandwidth savings. That is, if the video sequence resolution is larger than or equal to 480P, the multi-level motion estimation algorithm is applied. Otherwise, the Level C data reuse scheme is applied.

Table 3-III Data access requirement in different frame resolutions (Kbyte/frame)

	Inter layer data reuse	QCIF	CIF	480P	4CIF	720P	1080P
Level C	N (Method 1)	173.77	1300.16	10626.66	12472.60	83592.05	636766.44
	Y (Method 2)	**78.63**	**425.39**	2201.31	2586.85	9975.65	41894.34
Multi-level	N (Method 3)	215.62	1435.77	10525.77	12351.20	80243.69	617101.49
	Y (Method 4)	115.67	541.74	**2034.76**	**2388.41**	**6452.18**	**21835.40**
[1]Data access savings (%)		**54.74**	**67.28**	**80.85**	**80.85**	**92.28**	**96.57**

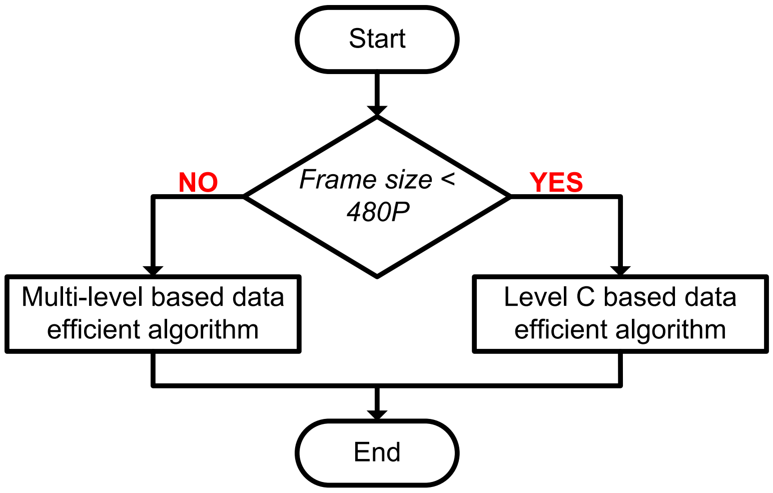
[1] Compared to Method 1

Table 3-IV Corresponding search range size of different frame resolution and search level

	QCIF	CIF	480P	4CIF	720P	1080P
Level 0	±8	±8	±8	±8	±8	±8
Level 1	0	±16	±32	±32	±64	±128
Level 2	0	0	0	0	0	±128

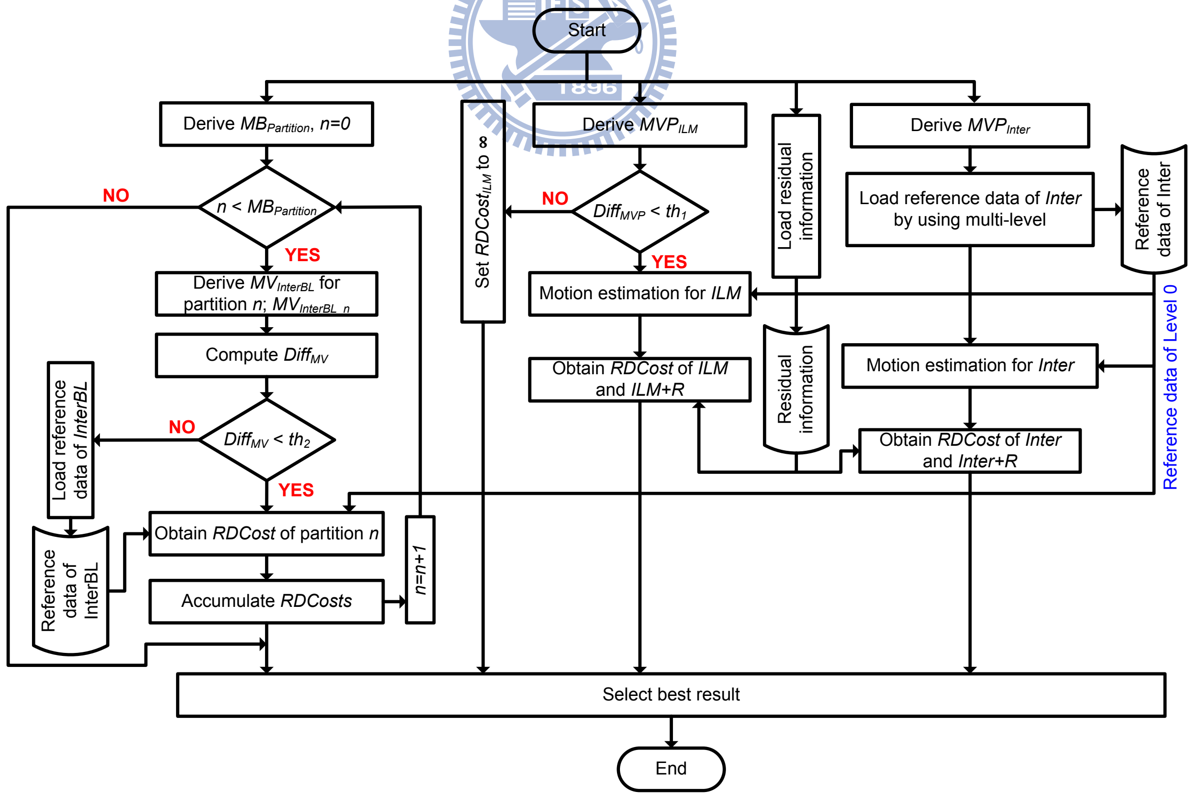
3.5.2. Algorithms Combination

Fig. 3.18 exhibits the overall flowchart of our proposed data efficient algorithm. At the beginning, the frame size is determined to decide which data efficient algorithm should be applied. If the frame size is less than 480P, the Level C based data efficient algorithm is applied. Otherwise, the multi-level based data efficient algorithm is applied. Fig. 3.19 and Fig. 3.20 show the detail flowchart of our proposed multi-level based and Level C based data efficient algorithm. From the figures, the main difference between two algorithms is only that the reference data reusing. In Level C data reuse scheme, the overall reference data can be reused for other Inter-layer prediction modes. However, for the multi-level data efficient algorithm, since the reference data memory of Inter prediction mode includes the reference data of Level 0, Level 1, and Level 2, only the reference data of Level 0 have be reused for other Inter-layer prediction modes due to the reference data of Level 1 and Level 2 are the sub-sampled version and they are unsuitable to be used for Inter-layer prediction. In addition, the residual information is also loaded to compute the corresponding *RDCosts* of Inter-layer residual prediction.



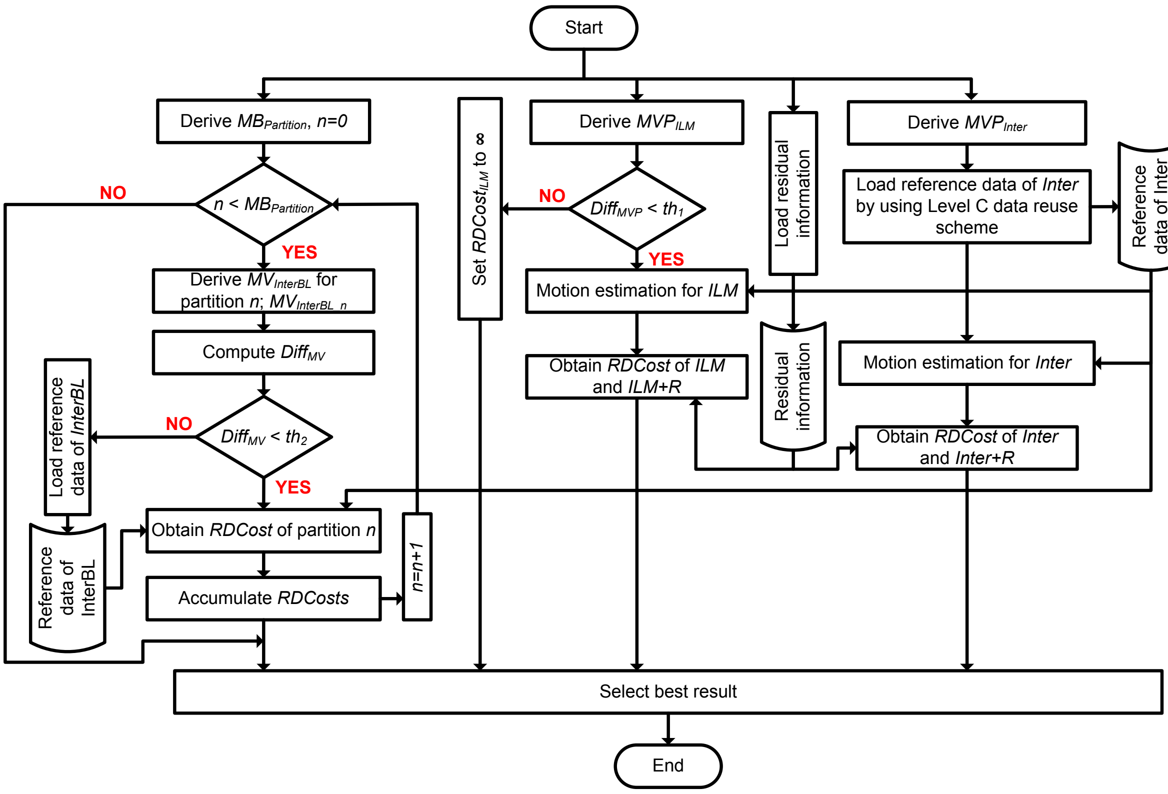
Start
Frame size < 480P
NO
YES
Multi-level based data efficient algorithm
Level C based data efficient algorithm
End

Fig. 3.18. Flowchart of proposed data efficient algorithm



Start
Derive $MB_{Partition}$, $n=0$
$n < MB_{Partition}$
NO
YES
Derive $MV_{InterBL}$ for partition n; $MV_{InterBL\_n}$
Compute $Diff_{MV}$
$Diff_{MV} < th_2$
NO
YES
Load reference data of *InterBL*
Reference data of InterBL
Obtain *RDCost* of partition *n*
Accumulate *RDCosts*
$n=n+1$
Set $RDCost_{ILM}$ to ∞
Derive MVP_{ILM}
$Diff_{MVP} < th_1$
NO
YES
Motion estimation for *ILM*
Obtain *RDCost* of *ILM* and *ILM+R*
Load residual information
Residual information
Derive MVP_{Inter}
Load reference data of *Inter* by using multi-level
Reference data of Inter
Reference data of Level 0
Motion estimation for *Inter*
Obtain *RDCost* of *Inter* and *Inter+R*
Select best result
End

Fig. 3.19. Flowchart of proposed multi-level based data efficient algorithm



Start
Derive $MB_{Partition}$, $n=0$
$n < MB_{Partition}$
NO
YES
Derive $MV_{InterBL}$ for partition n; $MV_{InterBL\_n}$
Compute $Diff_{MV}$
$Diff_{MV} < th_2$
NO
YES
Load reference data of $InterBL$
Reference data of InterBL
Obtain $RDCost$ of partition n
Accumulate $RDCosts$
$n=n+1$
Set $RDCost_{ILM}$ to ∞
Derive MVP_{ILM}
$Diff_{MVP} < th_1$
NO
YES
Motion estimation for ILM
Obtain $RDCost$ of ILM and $ILM+R$
Load residual information
Residual information
Derive MVP_{Inter}
Load reference data of $Inter$ by using Level C data reuse scheme
Reference data of Inter
Motion estimation for $Inter$
Obtain $RDCost$ of $Inter$ and $Inter+R$
Select best result
End

Fig. 3.20. Flowchart of proposed Level C based data efficient algorithm

3.6. Simulation Results

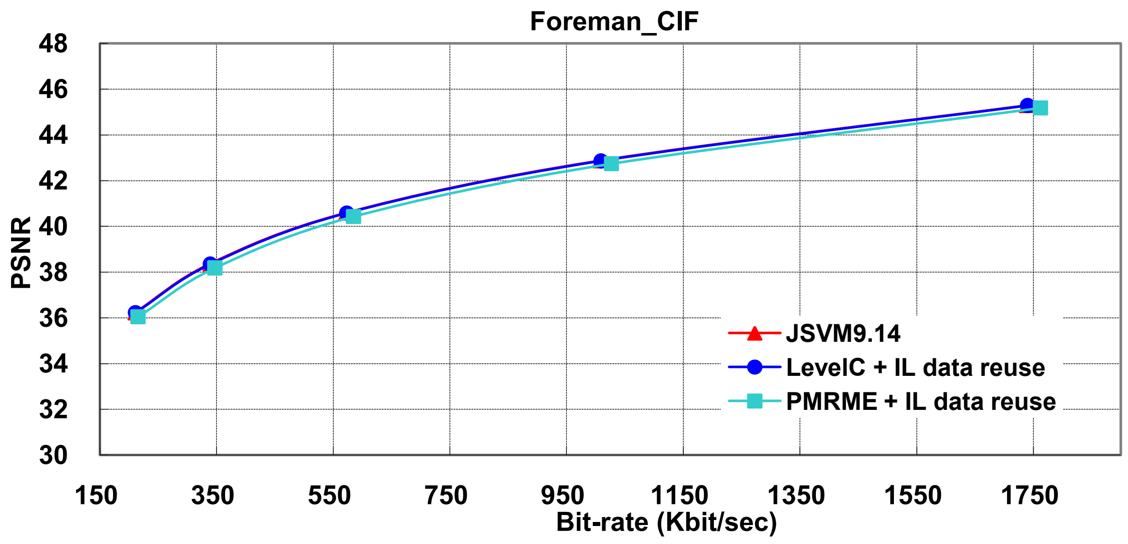
In this subsection, several simulation results are conducted to demonstrate the efficiency of our proposed data efficient Inter-layer prediction algorithms. The simulation settings are shown in Table 3-V. Fig. 3.21 and Fig. 3.22 show the rate distortion curve comparisons for different frame resolution. In Fig. 3.21 the frame size of spatial base and enhancement layer is QCIF and CIF, separately. For non-dyadic spatial resolution relationship between spatial layers, Fig. 3.22 shows the case that the frame size of spatial base layer is QCIF and the frame size of spatial enhancement layer is 480P. In the simulation results, the multi-level based and Level C based motion estimation algorithm with our proposed Inter-layer data efficient algorithm are individually compared to the JSVM reference software 9.14 [52]. However, although there are many test sequences have been used to conduct our simulation result, only few rate distortion curves of some test sequences have been listed in our dissertation. Nevertheless, all listed rate distortion curves cover all motion behavior ranged from slow to high motion. From the rate distortion curve comparisons, we can observe that both of multi-level and Level C based data efficient motion estimation algorithms result in near the same rate distortion performance when compared to JSVM reference software.

Table 3-VI and Table 3-VII show the detailed rate distortion comparisons for different spatial relationship configuration. In Table 3-VI, the frame size of base and enhancement layers is QCIF and CIF, respectively. In Table 3-VII, the frame size of QCIF and 480P is individually applied for the spatial base and enhancement layer. From these tables, we can observer some interesting phenomena from the rate distortion results. First, it can be seen that the rate distortion performance of multi-level based data efficient motion estimation algorithm is worth than that of the rate distortion performance of Level C based data efficient motion estimation algorithm in the small frame resolution coding configuration due to Level C data reuse scheme can easily find out the best result from the limited search area when frame size is small. Therefore, this is the reason that why the rate distortion performance of Level C based algorithm is much better than multi-level based algorithm in small frame resolution case. However, for the larger frame resolution case, the multi-level based data efficient algorithm can achieve much better rate distortion performance when compared to Level C based algorithm. Second, from the Table 3-VII, we can observe that the Level C based data efficient algorithm results in very significant rate distortion performance drop. Fortunately, thanks the help of our proposed adaptive motion estimation switching algorithm, the Level C based data efficient motion estimation algorithm will not been applied in our Inter prediction operation when frame size is larger than 480P. Therefore, the rate distortion performance drop of Level C data efficient algorithm will not presented in our proposed algorithm in larger frame resolution case.

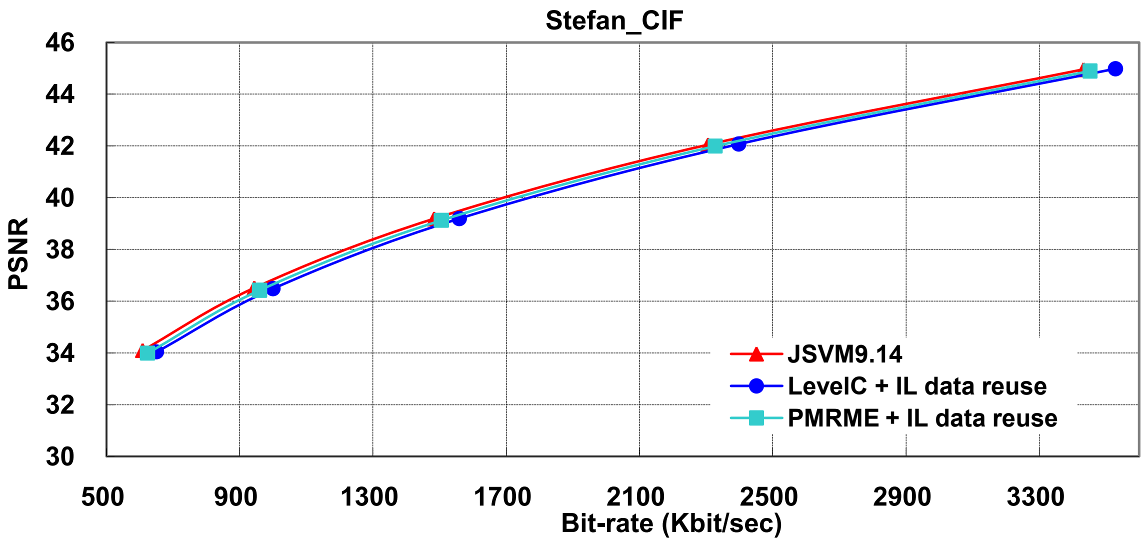
On average, the rate distortion performance drop of our proposed Level C based data efficient motion estimation algorithm is only 0.161% bitrate increasing and 0.002 dB PSNR decrease for CIF frame resolution. For the higher frame resolution sequence 480P, our proposed multi-level based data efficient motion estimation algorithm only results in 0.91% and 0.09 dB bitrate increasing and PSNR decrease, respectively.

Table 3-V Encoding setting configuration

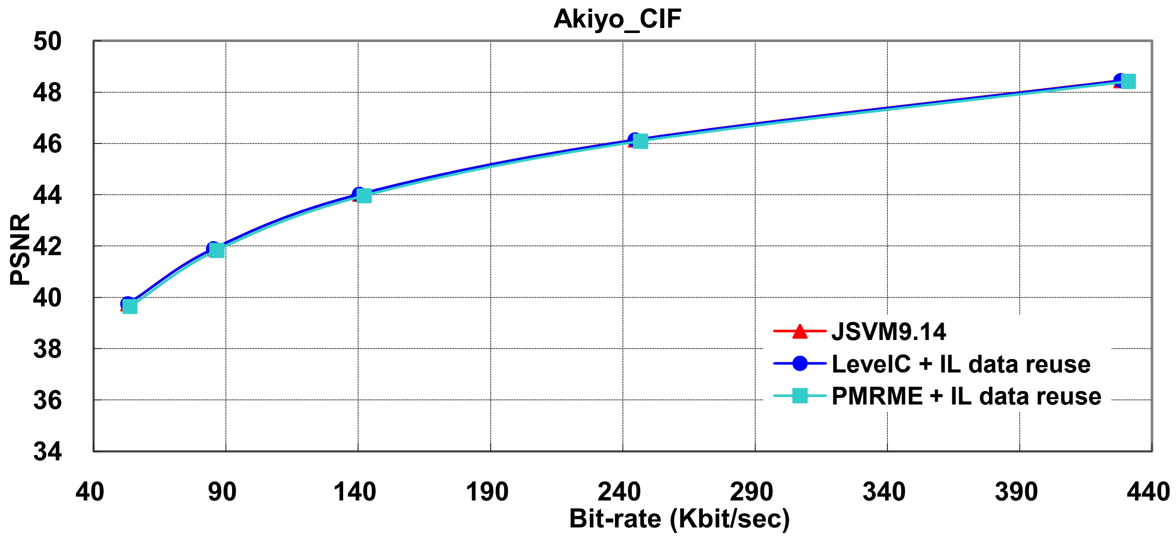
Codec	JSVM 9.14
Test sequences	Akiyo, Coastguard, Container, Foreman, Mobile, Silent, Blue_sky, Tractor, Station2, Pedestrain_area, Rush_hour, Riverbed
QP	16, 20, 24, 28, 32
Resolution	QCIF and CIF, CIF and 480p
Search range	±16 and ±32
Frame rate	15 Hz
GOP	8
Inter-layer prediction	Adaptive Inter-layer prediction
Frames to be encoded	100



(a)

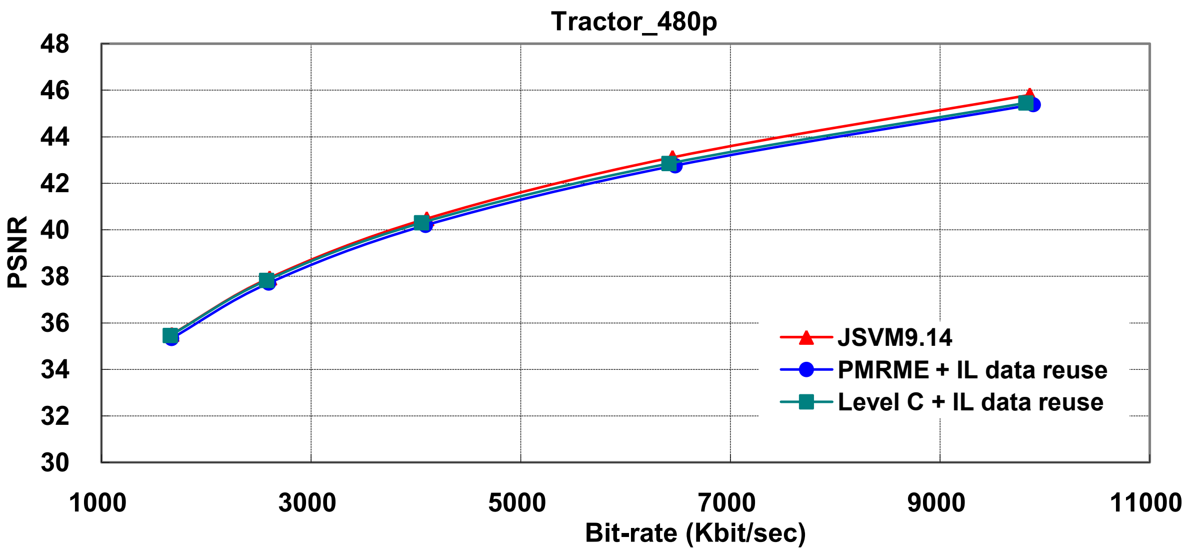


(b)

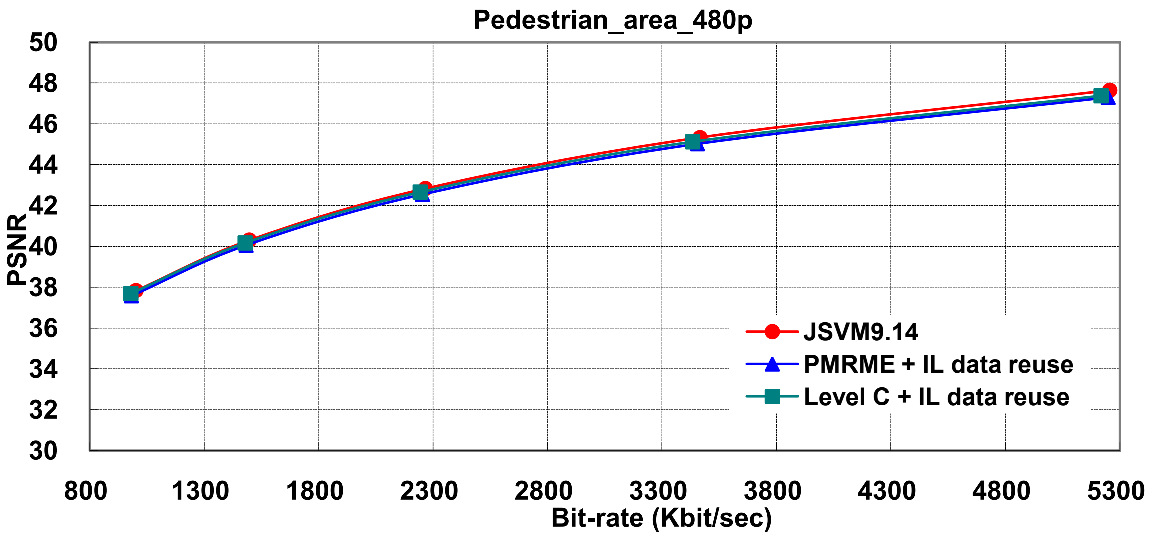


(c)

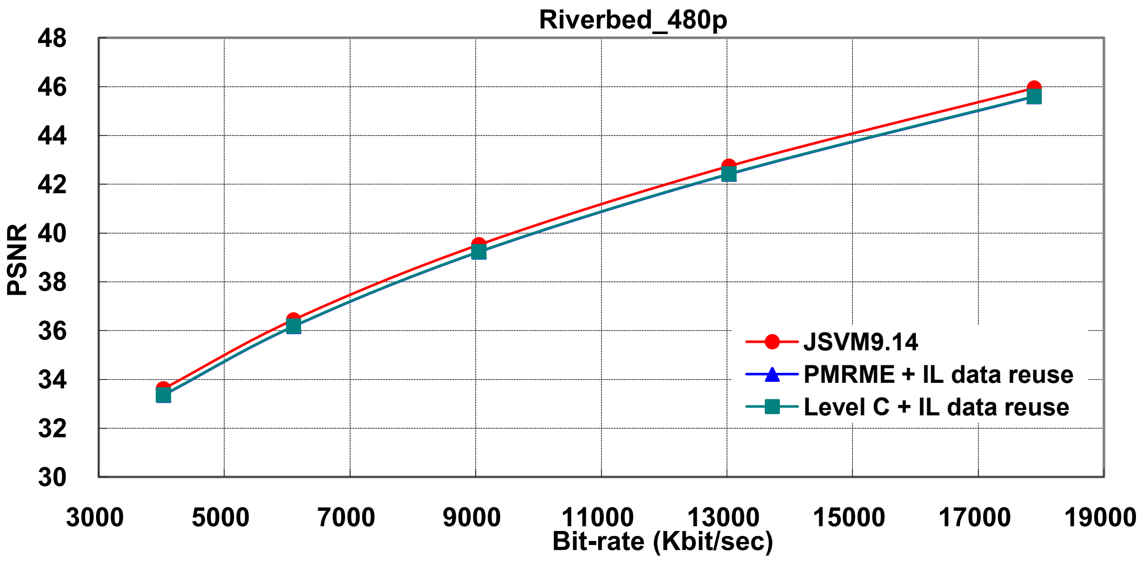
Fig. 3.21. RD curves comparisons (a) Foreman, (b) Stefan, and (c) Akiyo



(a)



(b)



(c)

Fig. 3.22.RD curves comparison (a) Tractor, (b) Pedestrian_area, and (c) Riverbed

Table 3-VI Detailed rate distortion comparisons for different encoding configuration settings (BL:QCIF, EL:CIF)

CIF sequences@15Hz			QP				
			16	20	24	28	32
Akiyo	Level C	ΔPSNR	-0.003	-0.002	-0.003	-0.001	-0.001
		ΔBitrate(%)	-0.007	-0.029	-0.166	0.180	0.116
	PMRME	ΔPSNR	-0.036	-0.052	-0.064	-0.063	-0.104
		ΔBitrate (%)	0.660	0.816	1.157	1.706	1.547
Coastguard	Level C	ΔPSNR	-0.001	-0.003	-0.004	-0.010	-0.015
		ΔBitrate (%)	0.385	0.506	0.614	1.045	1.293
	PMRME	ΔPSNR	-0.031	-0.037	-0.048	-0.054	-0.056
		ΔBitrate (%)	0.376	0.445	0.493	0.580	0.781
Container	Level C	ΔPSNR	0.001	-0.003	0.000	-0.002	-0.008
		ΔBitrate (%)	-0.010	0.031	-0.041	-0.026	-0.128
	PMRME	ΔPSNR	-0.027	-0.027	-0.039	-0.053	-0.049
		ΔBitrate (%)	0.231	0.695	0.797	0.927	1.313
Foreman	Level C	ΔPSNR	0.000	-0.006	-0.002	-0.016	0.000
		ΔBitrate (%)	0.112	0.175	0.259	0.158	0.335
	PMRME	ΔPSNR	-0.105	-0.130	-0.155	-0.180	-0.180
		ΔBitrate (%)	1.388	1.978	2.318	2.480	2.478
Mobile	Level C	ΔPSNR	0.001	0.002	-0.003	0.003	0.000
		ΔBitrate (%)	-0.001	0.014	-0.044	-0.026	-0.105
	PMRME	ΔPSNR	-0.051	-0.053	-0.070	-0.079	-0.084
		ΔBitrate (%)	0.547	0.667	0.852	1.197	1.553
Stefan	Level C	ΔPSNR	0.033	0.029	0.008	-0.028	-0.059
		ΔBitrate (%)	2.787	3.977	5.090	5.928	6.811
	PMRME	ΔPSNR	-0.053	-0.049	-0.058	-0.089	-0.108
		ΔBitrate (%)	0.569	0.925	1.476	1.660	2.376
Average	Level C	ΔPSNR	**0.000**	**-0.002**	**-0.002**	**-0.005**	**-0.004**
		ΔBitrate (%)	**0.092**	**0.120**	**0.089**	**0.228**	**0.276**
	PMRME	ΔPSNR	**-0.051**	**-0.061**	**-0.076**	**-0.085**	**-0.090**
		ΔBitrate (%)	**0.719**	**0.978**	**1.149**	**1.355**	**1.415**
	Level C average : ΔBitrate = 0.161%, ΔPSNR = -0.002 dB						
	PMRME average : ΔBitrate = 1.123%, ΔPSNR = -0.073 dB						

Table 3-VII Detailed rate distortion comparisons for different encoding configuration settings (BL:QCIF, EL:480P)

480p sequences@15Hz			QP				
			16	20	24	28	32
Blue_sky	Level C	ΔPSNR	0.178	0.135	0.084	0.048	0.027
		ΔBitrate(%)	2.849	5.021	7.713	10.851	13.791
	PMRME	ΔPSNR	-0.037	-0.043	-0.051	-0.059	-0.070
		ΔBitrate (%)	0.414	0.497	0.704	0.824	1.060
Pedestrian	Level C	ΔPSNR	0.263	0.209	0.172	0.139	0.147
		ΔBitrate (%)	0.725	0.927	0.987	1.269	2.281
	PMRME	ΔPSNR	-0.077	-0.100	-0.110	-0.116	-0.117
		ΔBitrate (%)	0.594	0.612	0.446	0.277	0.194
Riverbed	Level C	ΔPSNR	0.343	0.318	0.284	0.259	0.244
		ΔBitrate (%)	0.006	-0.006	0.007	0.001	0.002
	PMRME	ΔPSNR	-0.012	-0.010	-0.009	-0.010	-0.013
		ΔBitrate (%)	0.000	0.023	0.033	0.049	0.074
Rush_hour	Level C	ΔPSNR	0.226	0.188	0.143	0.127	0.105
		ΔBitrate (%)	0.208	0.385	0.396	0.885	1.213
	PMRME	ΔPSNR	-0.131	-0.135	-0.142	-0.149	-0.135
		ΔBitrate (%)	0.458	0.637	0.495	0.549	0.350
Station2	Level C	ΔPSNR	0.406	0.066	0.015	0.023	0.025
		ΔBitrate (%)	3.695	5.702	6.933	7.793	6.845
	PMRME	ΔPSNR	-0.131	-0.097	-0.129	-0.178	-0.227
		ΔBitrate (%)	1.268	1.550	1.902	2.044	1.928
Tractor	Level C	ΔPSNR	0.339	0.260	0.173	0.101	0.049
		ΔBitrate(%)	0.359	0.520	1.235	1.162	0.931
	PMRME	ΔPSNR	-0.077	-0.092	-0.109	-0.120	-0.122
		ΔBitrate (%)	0.713	0.920	0.937	0.776	0.724
Average	**Level C**	**ΔPSNR**	**0.276**	**0.186**	**0.141**	**0.122**	**0.123**
		ΔBitrate (%)	**2.703**	**3.825**	**4.835**	**6.501**	**8.061**
	PMRME	**ΔPSNR**	**-0.075**	**-0.081**	**-0.093**	**-0.103**	**-0.103**
		ΔBitrate (%)	**0.663**	**0.826**	**1.093**	**1.040**	**0.987**
	Level C average : ΔBitrate = 5.19%, ΔPSNR = 0.17dB						
	PMRME average : ΔBitrate = 0.91%, ΔPSNR = -0.09 dB						

Table 3-VIII exhibits the data access bandwidth saving comparison for different algorithms and frame sizes. From this table, 50.55, 64.78, 80.07, 80.06, 90.18, and 95.00 % data access bandwidth savings can be achieved by our proposed data efficient Inter-layer prediction algorithm for frame resolution of QCIF, CIF, 480P, 4CIF, 720P, and 1080P, respectively on average.

Table 3-VIII Data access bandwidth savings comparison (%)

Proposal	QCIF	CIF	480P	4CIF	720P	1080P
Level C based[1]	54.74	67.28	79.29	79.26	88.07	93.42
Multi-level based[2]	46.35	62.27	80.85	80.85	92.28	96.57
Average	**50.55**	**64.78**	**80.07**	**80.06**	**90.18**	**95.00**

[1] Compared to Level C data reuse scheme without our proposed data efficient IL algorithm
[2] Compared to multi-level scheme without our proposed data efficient IL algorithm

3.7. Summary

In this section, we proposed several data efficient Inter-layer prediction algorithms to lighten the data access bandwidth overheads. For Inter-layer residual prediction, we propose an efficient data reuse scheme for helping the derivation of *RDCosts* of Inter-layer residual prediction. Through our proposed algorithm, not only the reference data can be reused but the *RDCosts* of Inter/Inter-layer motion and Inter+residual/Inter-layer motion+residual can be derived by single motion estimation module. For Inter-layer motion and InterBL prediction algorithms, we propose an efficient data reuse scheme for saving the data bandwidth by reusing the reference data of Inter prediction mode. In addition, we also propose an adaptive motion estimation switching algorithm to dynamically select the motion estimation algorithm by determining the frame size. Several simulation results show that our proposed Level C based data efficient motion estimation algorithm only results in 0.161% bitrate increasing and 0.002 dB PSNR decrease for CIF frame resolution. Besides, our proposed multi-level based data efficient motion estimation algorithm only leads to 0.91% and 0.09 dB bitrate increasing and PSNR decrease, respectively. For data access bandwidth savings, 50.55, 64.78, 80.07, 80.06, 90.18, and 95.00 % data access bandwidth savings can be achieved by our proposed data efficient Inter-layer prediction algorithm for frame resolution of QCIF, CIF, 480P, 4CIF, 720P, and 1080P, respectively on average.

Chapter 4

An Efficient Mode Pre-Selection Algorithm for Fractional Motion Estimation in H.264/AVC Scalable Video Extension

4.1. Introduction

To efficiently exploit the temporal relationship between successive frames, the video coding system usually adopts the technique of Inter prediction to remove the temporal redundancies. In the older video coding systems, the Inter prediction usually stands for the integer pixel position motion estimation based on the assumption that the object moving is only occurred in the integer position. However, some literatures shown that taking sub-pixel position into account in the Inter prediction can further improve the coding performance since the object moving would not be always in the integer accuracy. Therefore, the concept of fractional accuracy motion estimation is thus getting popular and literature [53] shown that 4+ dB in PSNR improvement can be achieved by adopting fractional motion estimation (FME). As a result, the fractional motion estimation is thus been widely adopted in the existing video coding standards [7].

Fig. 4.1 illustrates the concept of FME and its operation is explained in detail as follows. The operation of FME is mainly composed by two steps called half pixel and quarter pixel position motion estimation. After the best integer position has been decided by integer pixel motion estimation (IME) as labeled by circle symbols, the half pixel position motion estimation will be executed around the best integer pixel position. The half pixel position motion estimation checks eight additional candidate positions, as labeled by square symbols, to find out the best search results. During the checking process of half pixel position motion estimation, the absent pixels on the half pixel position will be interpolated by a six-tape interpolator with the coefficients of [1, -5, 20, 20, -5, 1]. Once the best half pixel position has been decided, the quarter pixel position motion estimation will be executed around the best position decided by the half pixel position motion estimation. Similar to the half pixels position motion estimation, the quarter pixel position motion estimation also checks eight additional candidate positions, as labeled by triangle symbols, to derive the final results of

Inter prediction. However, different from the half pixel position motion estimation, the absent pixels during the quarter pixel position motion estimation will be generated by a bi-linear interpolator. Therefore, from the operation of FME, we can find that the interpolation operations should be involved in both of half-pel and quarter-pel search stages.

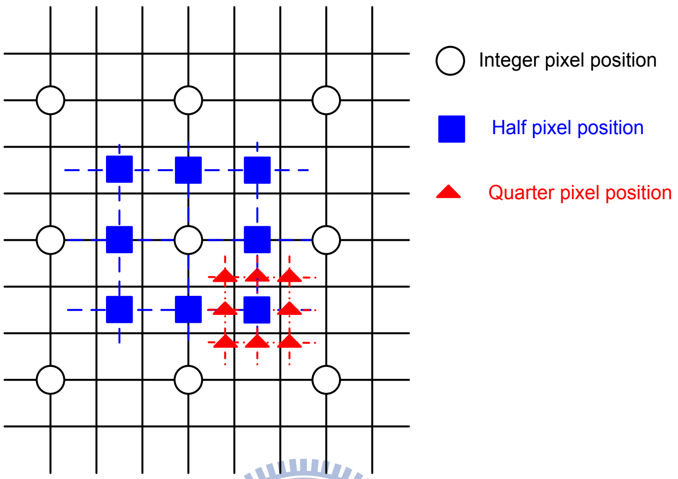


Fig. 4.1. Illustration of factional motion estimation

Although FME only checks several positions around the best motion vectors resulted by integer motion estimation, the computational complexity of IME and FME are almost equal to each other especially in the hardware realization due to the complex interpolation process and lots of prediction modes need to be checked by FME [53][54]. To measure the computational complexity of fractional motion estimation, Fig. 4.2 shows the percentage of CPU usage profile for Mobile sequence reported in [55]. From this figure, we observed that the CPU usage of sub-pixel motion estimation is much higher than that of integer pixel motion estimation. Totally, 76% of CPU usage has been occupied by the fractional motion estimation component. As a result, the operations of IME and FME are usually divided into two different pipeline stages in hardware design [56] to balance the computational complexity of pipeline stage and thus aim at higher coding performance.

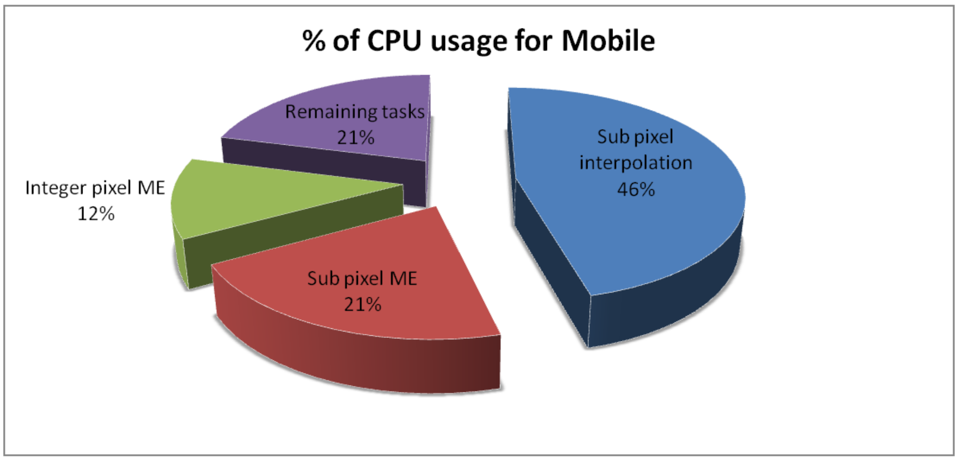


Fig. 4.2. Percentage of CPU usage profile of H.264 encoder. For Mobile sequence at CIF size, using CAVLC, ±16 search range, fast motion estimation, and no RD mode selection.

In H.264 video coding standard [7], variable block size motion estimation is supported in Inter prediction mode and each partition size has to be checked by IME and FME one by one to select the best prediction result as shown in Fig. 4.3. Thus, 41 blocks have to go through IME and FME operation. In addition to the inherent prediction modes in H.264, the mechanism of Inter-layer prediction modes adopted in SVC [11] including Inter-layer motion prediction (ILM), and Inter-layer motion residual prediction (ILM+R) also significantly increases the computational complexity of FME as shown in Fig. 4.4. As a result, 41×4=164 blocks have to be examined by FME in SVC.

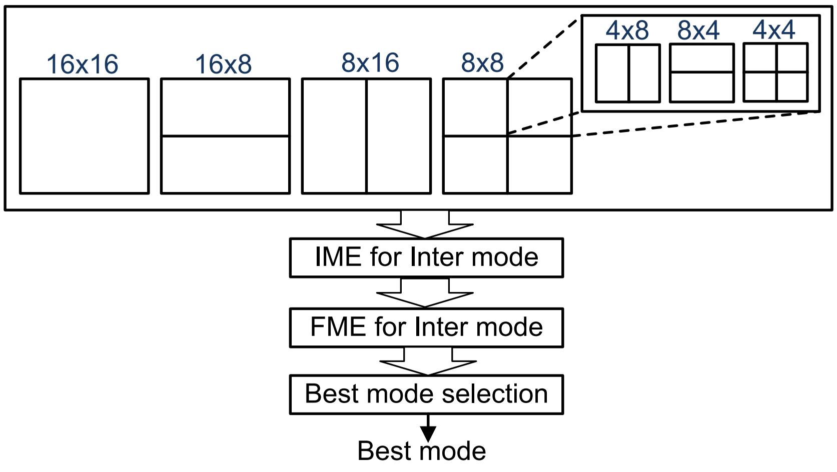


Fig. 4.3. Illustration of mode selection process of H.264

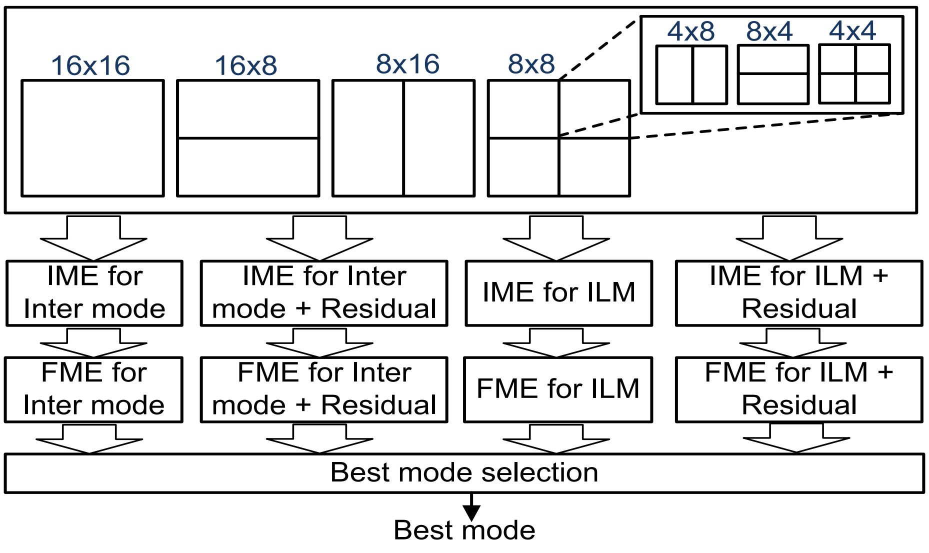


Fig. 4.4. Illustration of mode selection process of SVC

To simplify the design complexity in hardware design, the small blocks ranged from 8×8 to 4×4 are early decided in IME stage to derive a Submode and thus only partition sizes of 16×16, 16×8, 8×16, and Submode (9 blocks in minimum and 21 blocks in maximum) have to be examined by FME operation as Fig. 4.5(a) shown. Similarly, the idea of Submode early decision can be also applied to SVC for easing the overheads of hardware implementation. As a result, only 36 to 84 blocks have to be examined by FME in SVC as shown in Fig. 4.6. Although the early decision method for Submode can efficiently reduce the overheads of FME, the computational complexity of FME is still high. Several works [57]-[60] have been proposed to increase the coding speed of FME through the hardware implementation. In contrast to check all prediction modes, [61],[62] proposed a mode pre-selection method as shown in Fig. 4.5(b) to pre-selecting the potential skippable prediction modes before entering FME prediction process in H.264. However, none of above literature has addressed the issues of SVC. Thus, this chapter proposes an efficient mode pre-selection algorithm to lighten the computational complexity of FME for SVC by using the concept of mode pre-selection. In our proposed algorithms, the rate distortion cost relationship between different prediction modes are first analyzed and observed to find out the clues that we can use to pre-select the potentially ignorable prediction modes. Based on the analytical results, several mode pre-selection rules are proposed in this chapter.

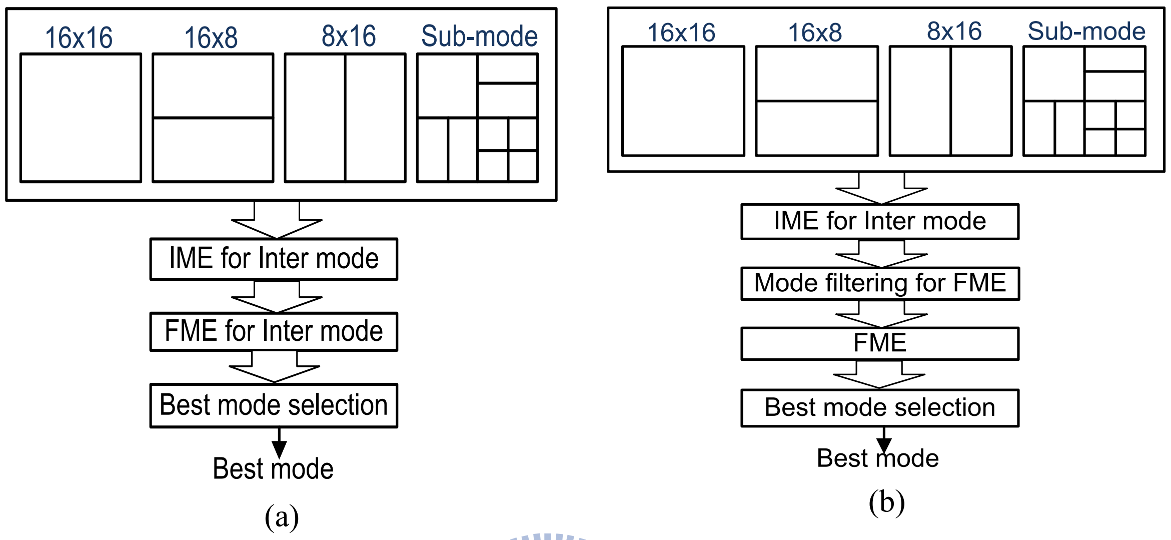


Fig. 4.5. Illustration of (a) mode selection process for H.264/AVC and (b) mode pre-selection concept for H.264/AVC

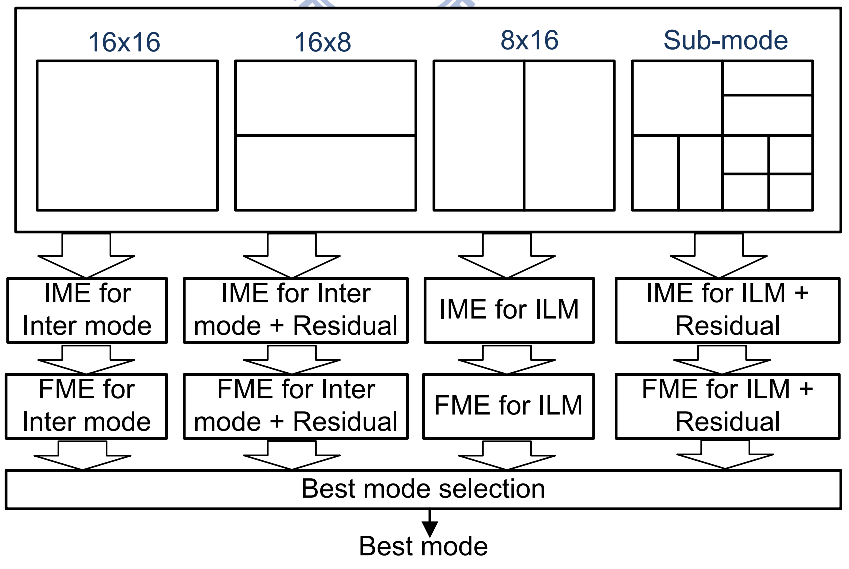


Fig. 4.6. Illustration of mode selection process for SVC

The rests of this chapter are organized as follows. In Section 4.2, some observations are introduced to indicate the rate distortion cost relationship between different prediction modes.

Afterwards, the mode pre-selection algorithms are proposed according to the observations in Section 4.3. Simulation results are shown in Section 4.4 to demonstrate the efficiency of our proposed algorithms. The hardware architecture design is presented in Section 4.5. Finally, the conclusions are made in Section 4.6.

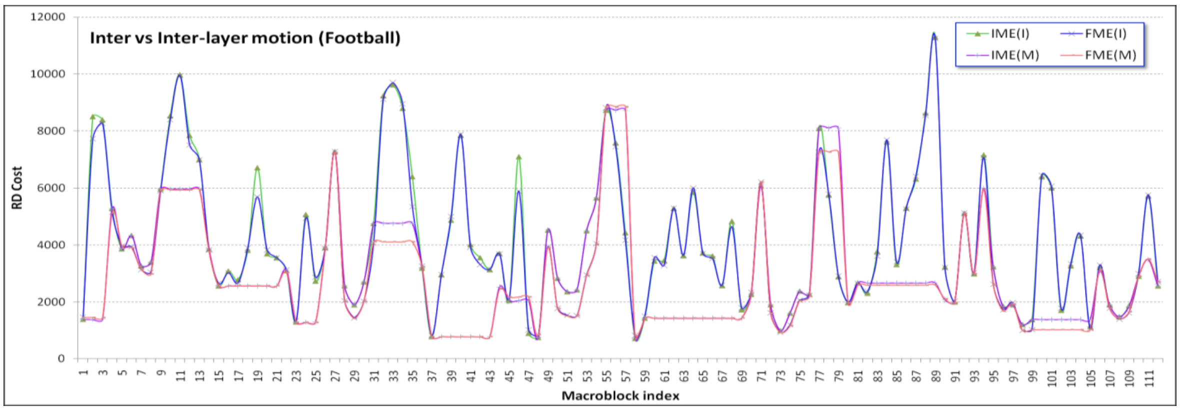
4.2. Analysis of Rate Distortion Cost between Prediction Modes

To find the valuable clues for helping the derivation of FME mode pre-selection algorithm, we conduct several analyses for the rate distortion cost of IME and FME between different prediction modes. Afterwards, we carry out several simulations to confirm the observed results. The reason why we choose the rate distortion cost to observe is that the best prediction mode is judged by the magnitude of rate distortion cost in mode selection process. Therefore, it is very intuitive to choose rate distortion cost for observation.

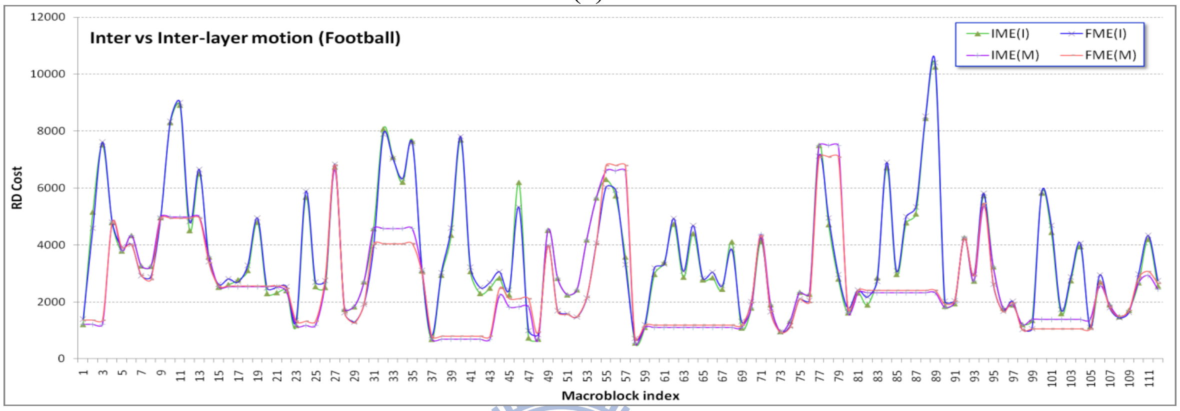
4.2.1. Observing the RDCost Relationship between Different Prediction Modes

In this subsection, we analyze the rate distortion cost (*RDCost*) relationship of IME and FME for four different prediction modes combination. These four combinations are "*Inter versus Inter-layer motion* (*Type1*)", "*Inter+residual versus Inter-layer motion+residual* (*Type2*)", "*Inter versus Inter+residual* (*Type3*)", and "*Inter-layer motion versus Inter-layer motion+residual* (*Type4*)". Fig. 4.7 to Fig. 4.9, Fig. 4.10 to Fig. 4.12, Fig. 4.13 to Fig. 4.15, and Fig. 4.16 to Fig. 4.18 individually show the rate distortion cost relationship of *IME* and *FME* for *Type1, Type2, Type3*, and *Type4* prediction modes combinations with different partition sizes. In these figures, the vertical axis indicates the *RDCosts* and the horizontal axis is the index of macroblocks. The terms of *I*, *M,* and *R* individually stand for the *Inter*, *Inter-Layer Motion (ILM),* and *Inter-layer Residual* prediction mode. From these figures, we can observe two properties called *spatial locality* and *no spatial locality* property. The *spatial* *locality* here is roughly defined as that the macroblocks which satisfy the condition that rate distortion costs of *IME* and *FME* are very close to each other. For example, the *RDCost* of *IME* is very close to the *RDCost* of *FME* for *Inter* mode and the same situation can be seen from *ILM* prediction mode in *Type1* prediction mode combination. That is, for example, if the *IME RDCost* of *Inter* mode is sufficiently larger than that of *IME RDCost* of *ILM* mode, it has high probability that the *FME RDCost* of *Inter* mode will be larger than *FME RDCost* of *ILM* mode and vice versa due to the *spatial locality* property. On the other hand, the *no spatial locality property* is defined as that the macroblocks don't have the *spatial locality* property. From these figures we can observe that the most of macroblocks have the *spatial locality* property for block size of 16×16, 16×8, and 8×16. However, for the block size of Submode, such *spatial locality* property can not been observed easily from the figures.

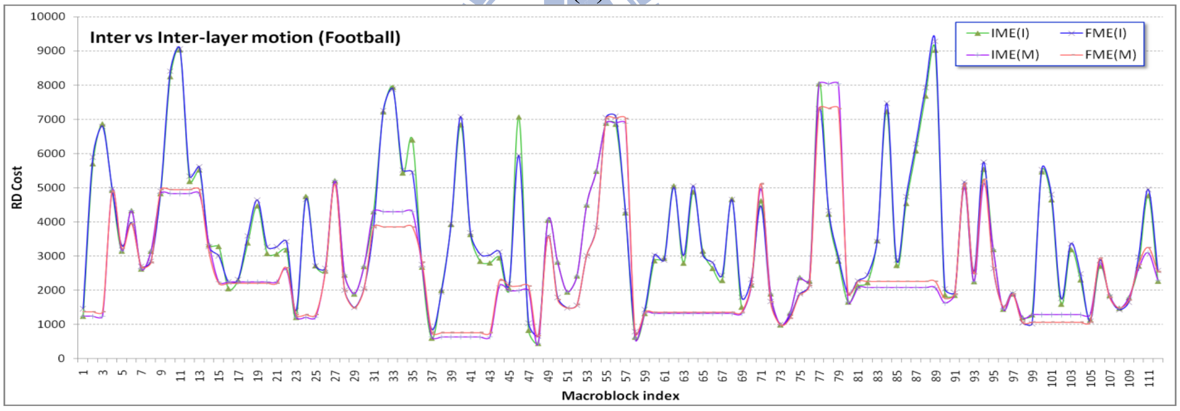
From the above observations, we can summarize the following observations. First, the mode pre-selection process should take the block prediction size into consideration. That is, the block sizes of 16×16, 16×8, and 8×16 are jointed into account for deriving the FME mode pre-selection algorithm because these block sizes have much similar behavior. For the Submode block size, it should be treated individually for better prediction performance due to it reveals different behavior when compared to block sizes of 16×16, 16×8, and 8×16. Second, the macroblocks should be dealt with separately depending on what property they belong to. In other words, the macroblocks which have the *spatial locality* property should be processed by one mode pre-selection algorithm which considers the *spatial locality* property, and the other macroblocks have to be processed by another mode pre-selection algorithm.



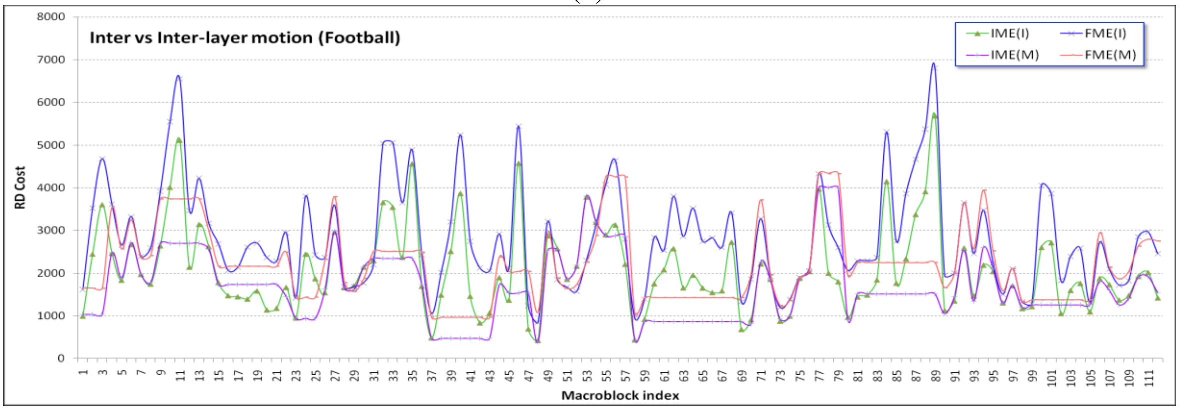
(a)



(b)

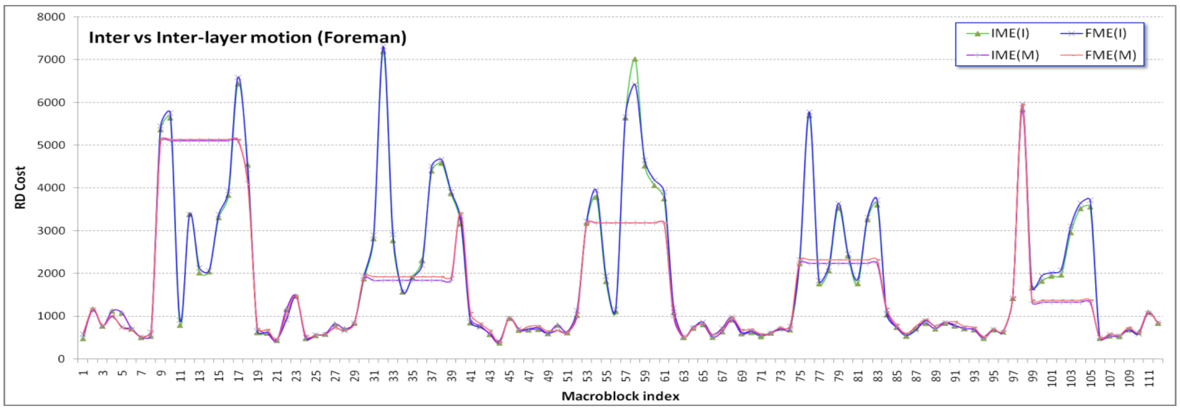


(c)

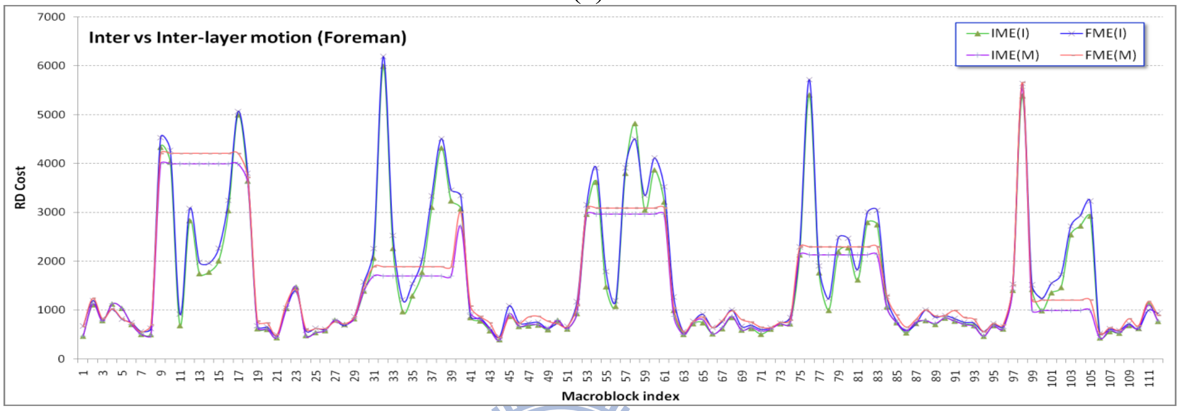


(d)

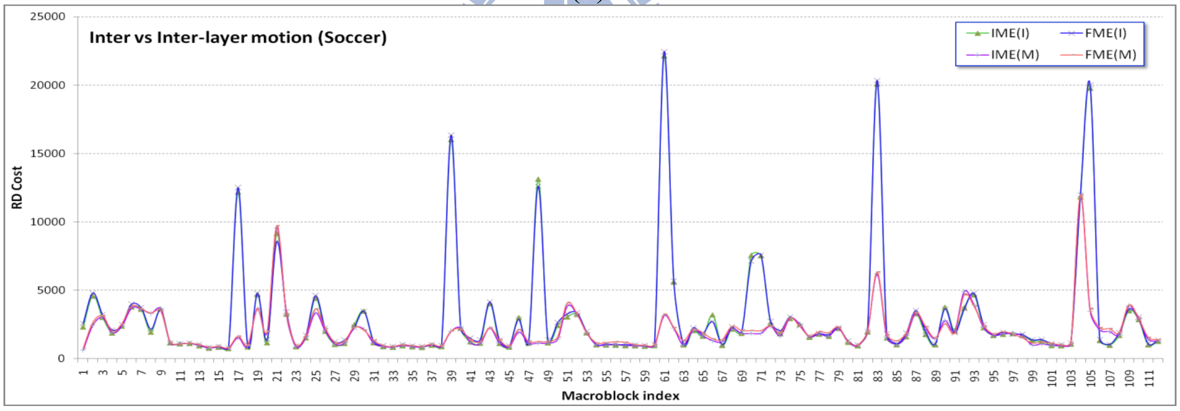
Fig. 4.7. Relationship between *RDCost*s of *IME* and *FME* of Football sequence for *Type1* prediction mode combination (a) 16×16, (b) 16×8 (c) 8×16, and (d) Submode



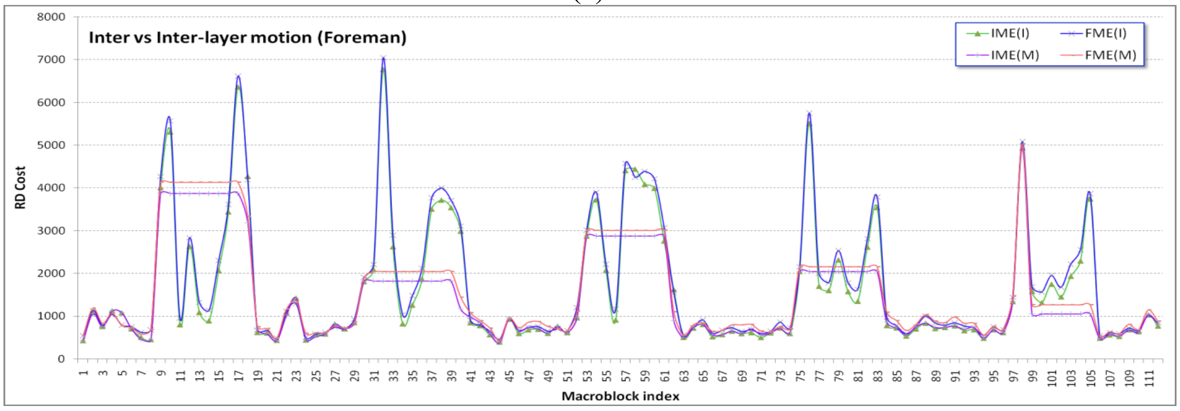
(a)



(b)

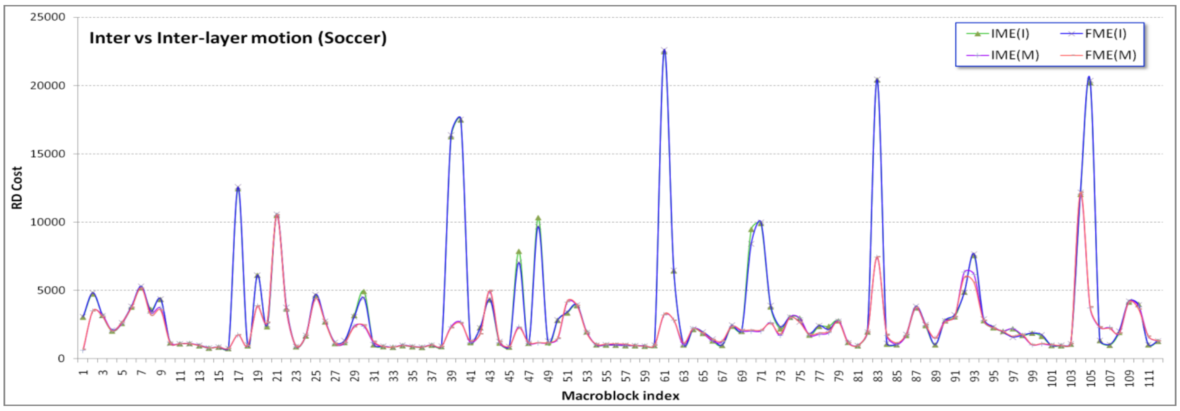


(c)

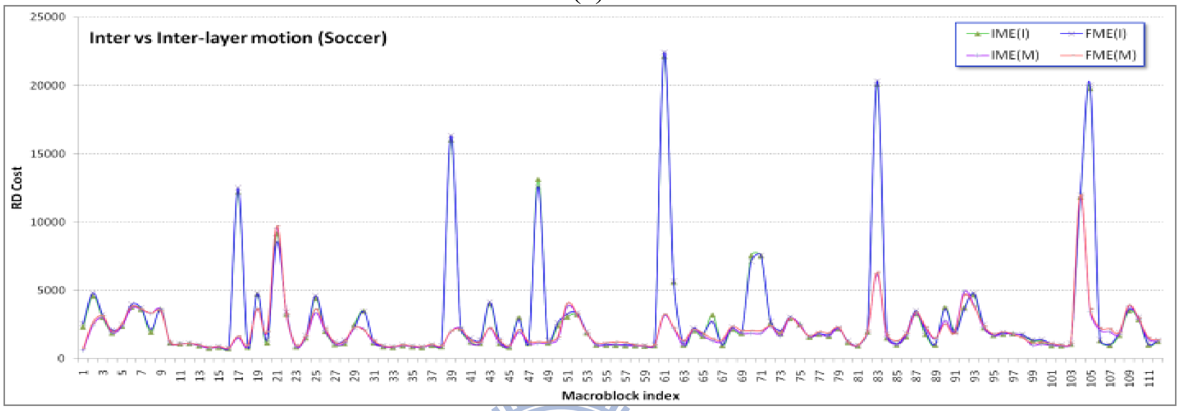


(d)

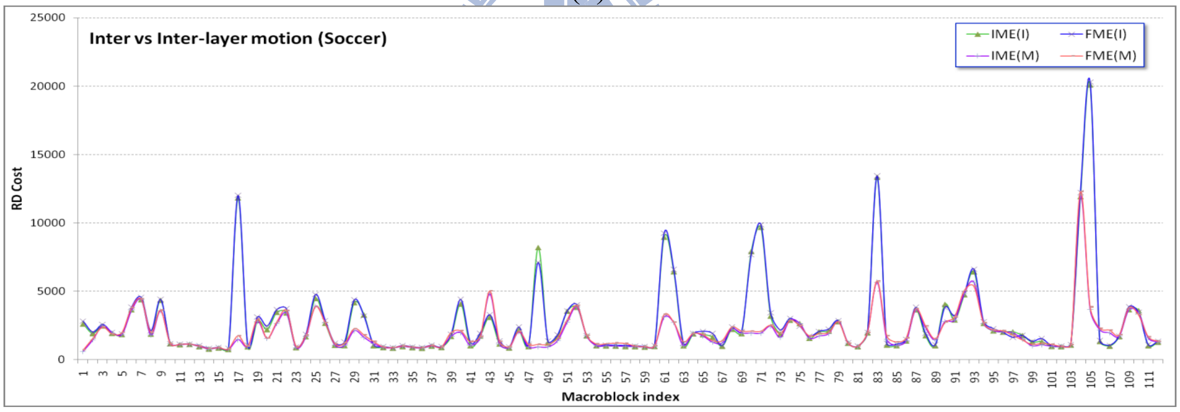
Fig. 4.8. Relationship between *RDCost*s of *IME* and *FME* of Foreman sequence for *Type1* prediction mode combination (a) 16×16, (b) 16×8 (c) 8×16, and (d) Submode



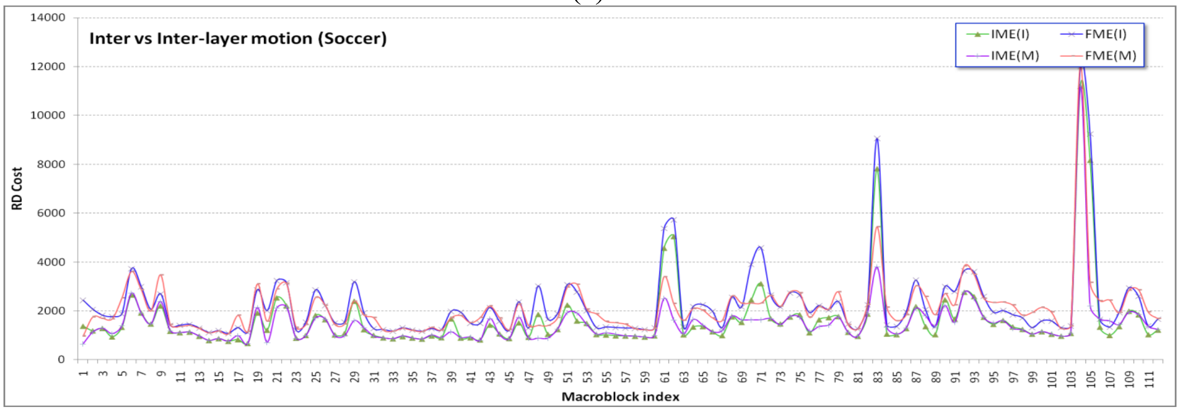
(a)



(b)

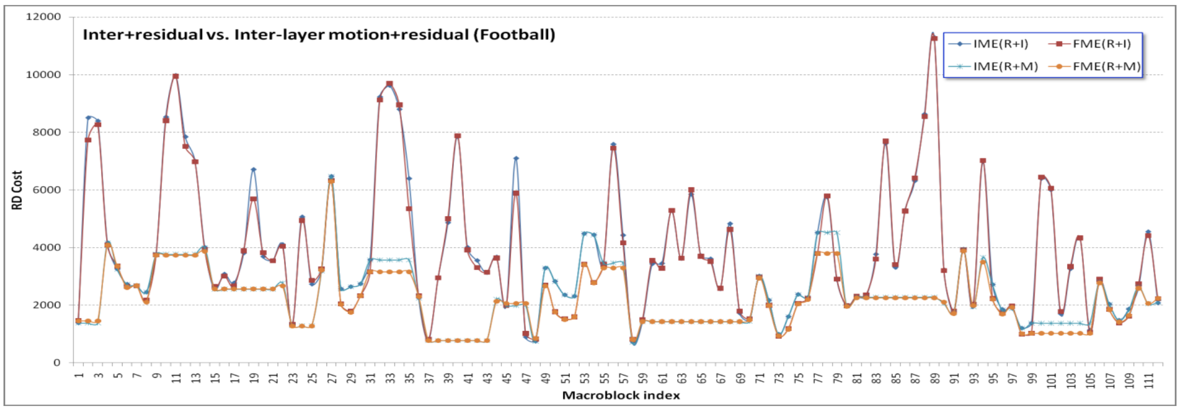


(c)

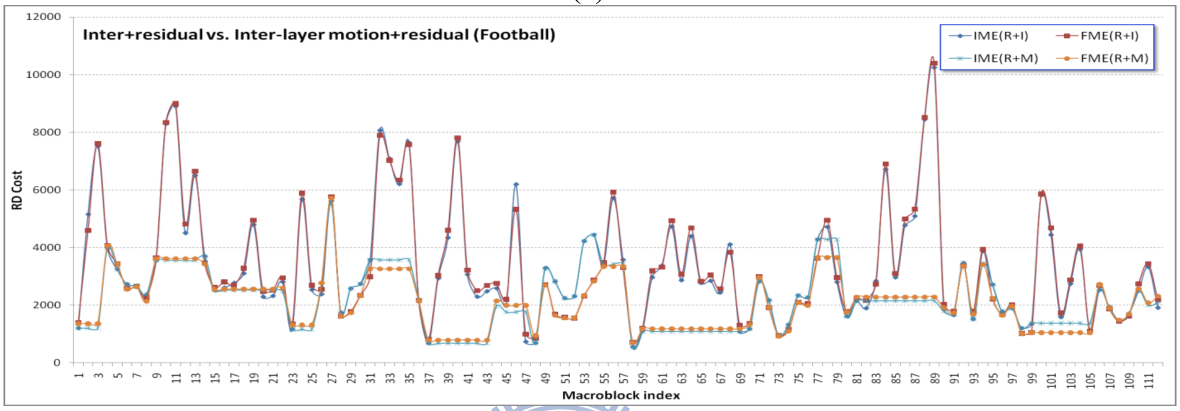


(d)

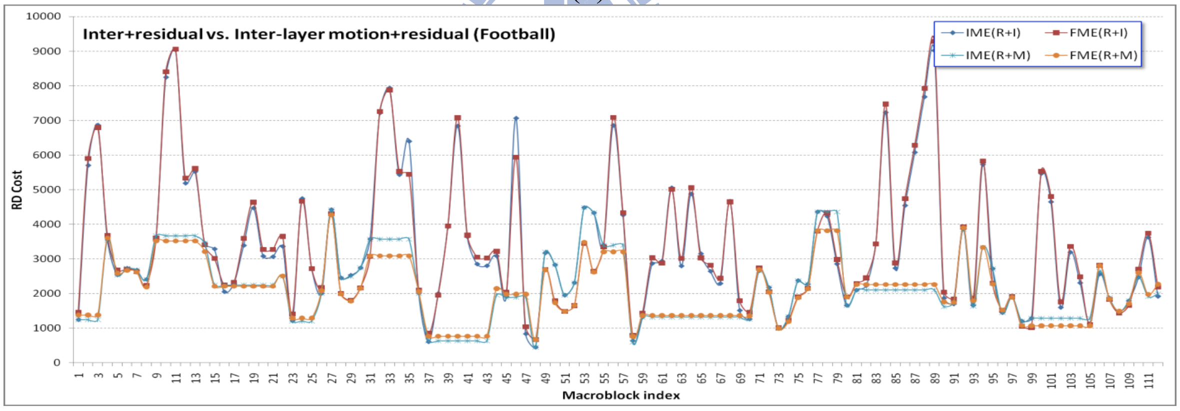
Fig. 4.9. Relationship between *RDCost*s of *IME* and *FME* of Soccer sequence for *Type1* prediction mode combination (a) 16×16, (b) 16×8 (c) 8×16, and (d) Submode



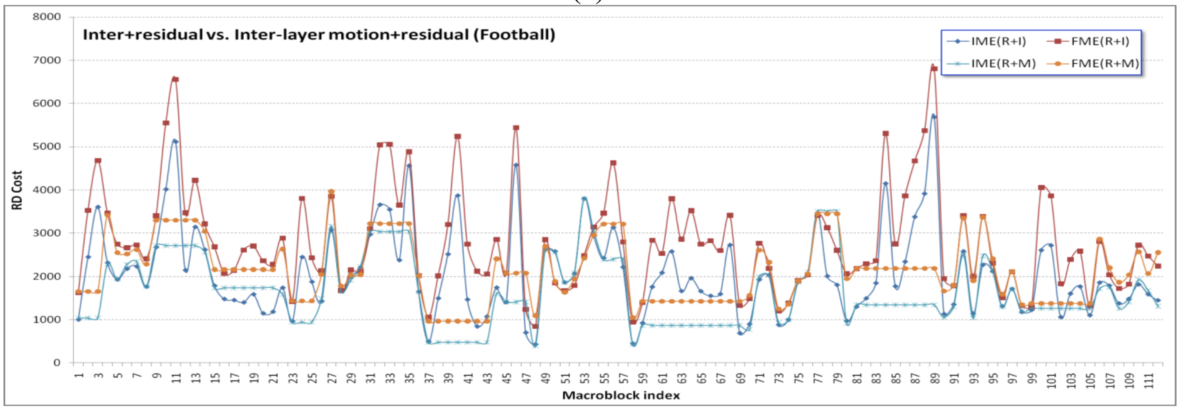
(a)



(b)

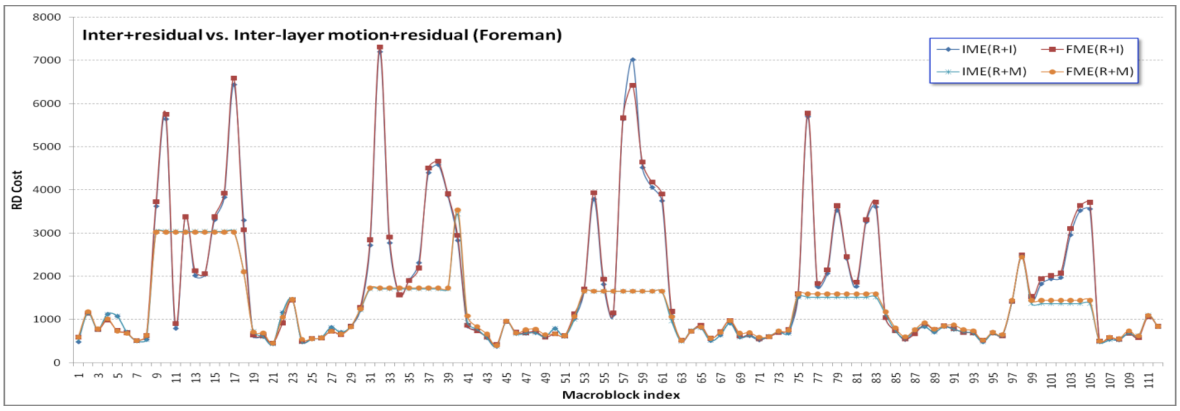


(c)

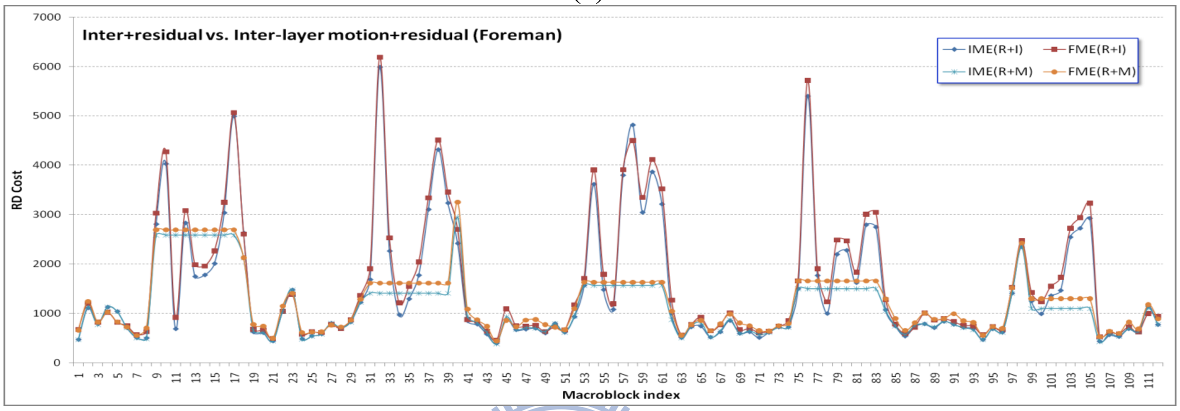


(d)

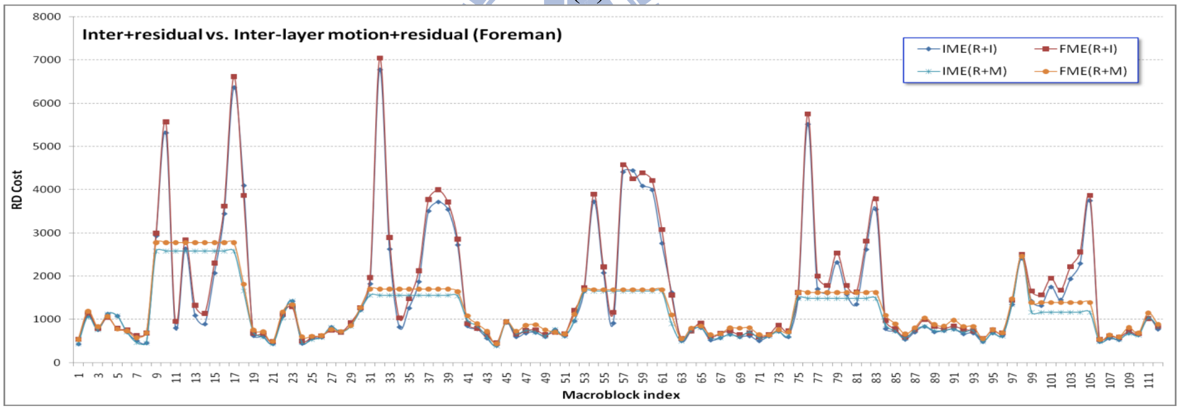
Fig. 4.10. Relationship between *RDCost*s of *IME* and *FME* of Football sequence for *Type2* prediction mode combination (a) 16×16, (b) 16×8 (c) 8×16, and (d) Submode



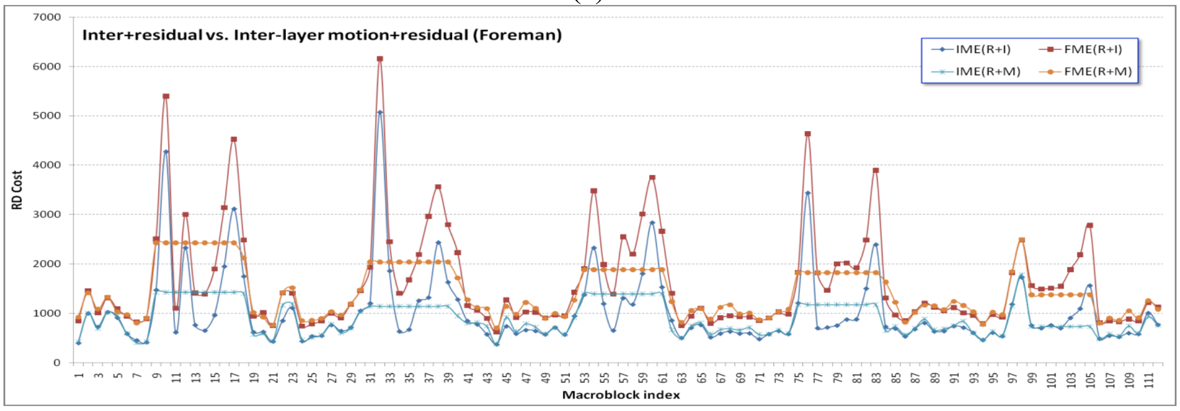
(a)



(b)

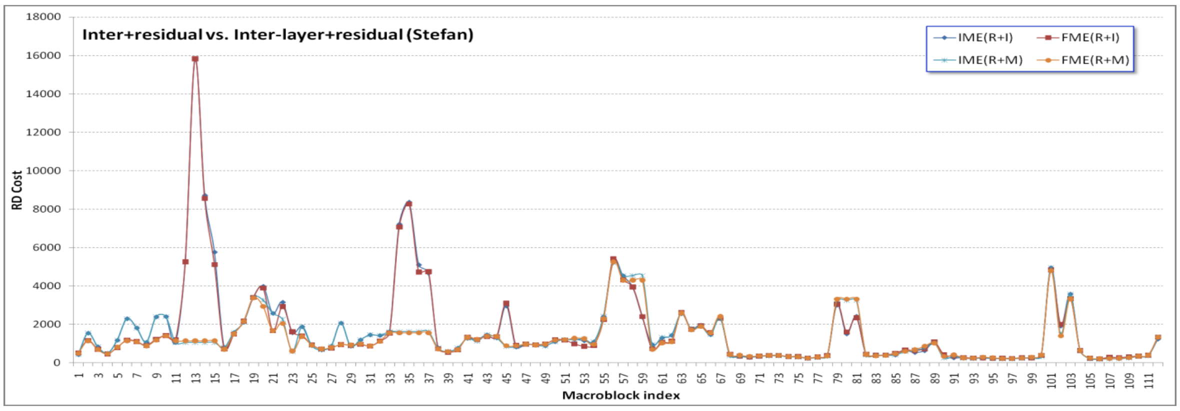


(c)

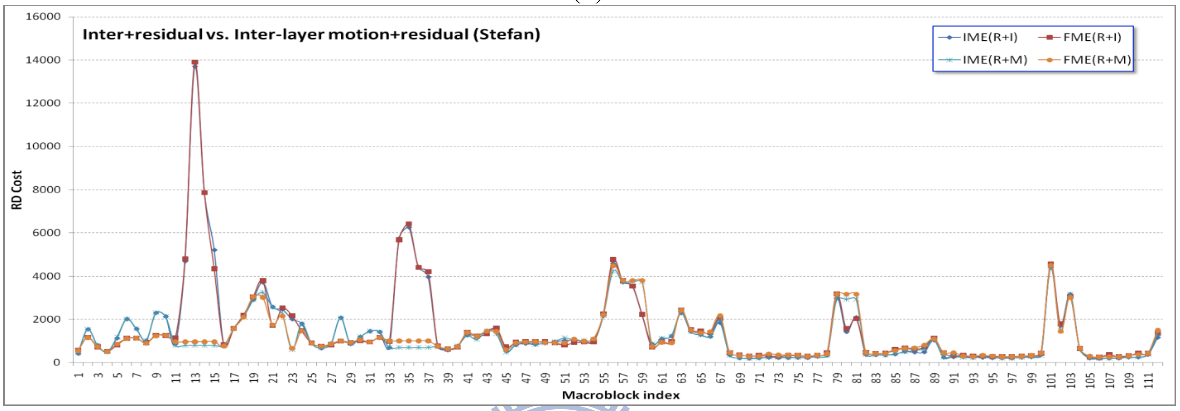


(d)

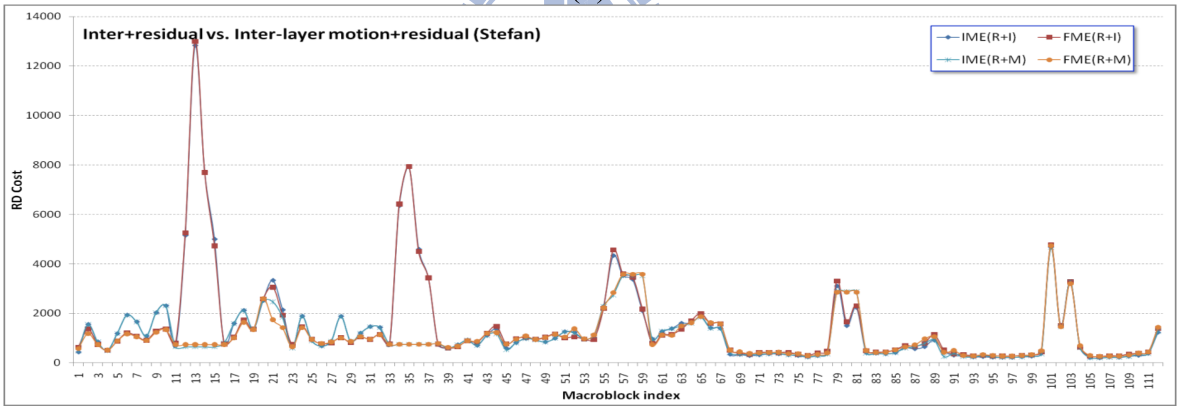
Fig. 4.11. Relationship between *RDCost*s of *IME* and *FME* of Foreman sequence for *Type2* prediction mode combination (a) 16×16, (b) 16×8 (c) 8×16, and (d) Submode



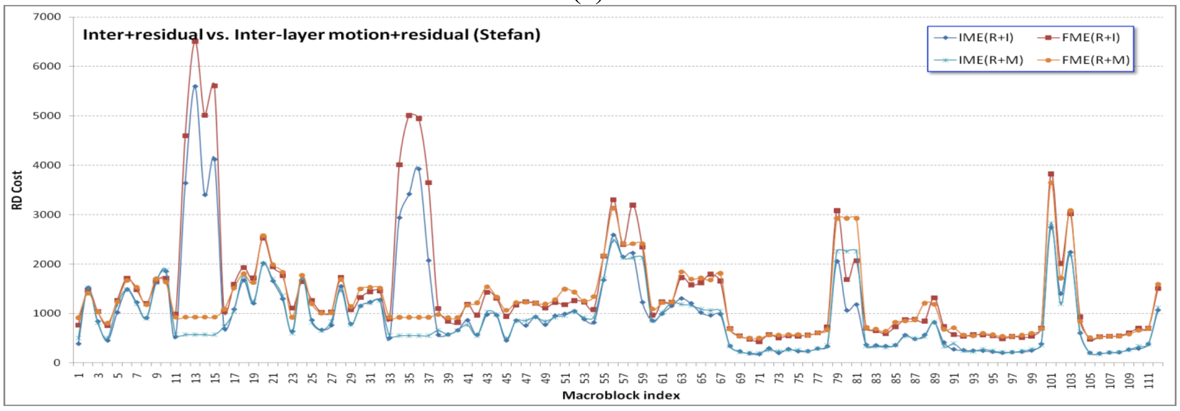
(a)



(b)

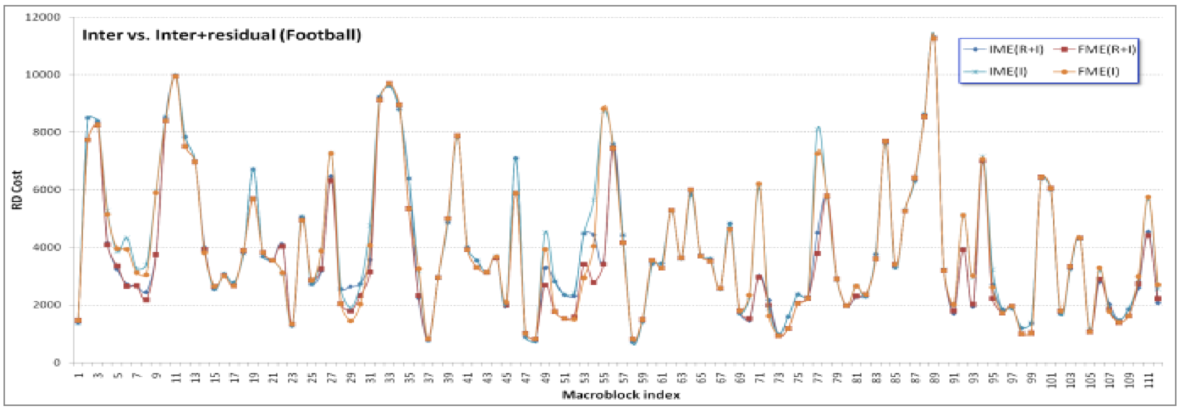


(c)

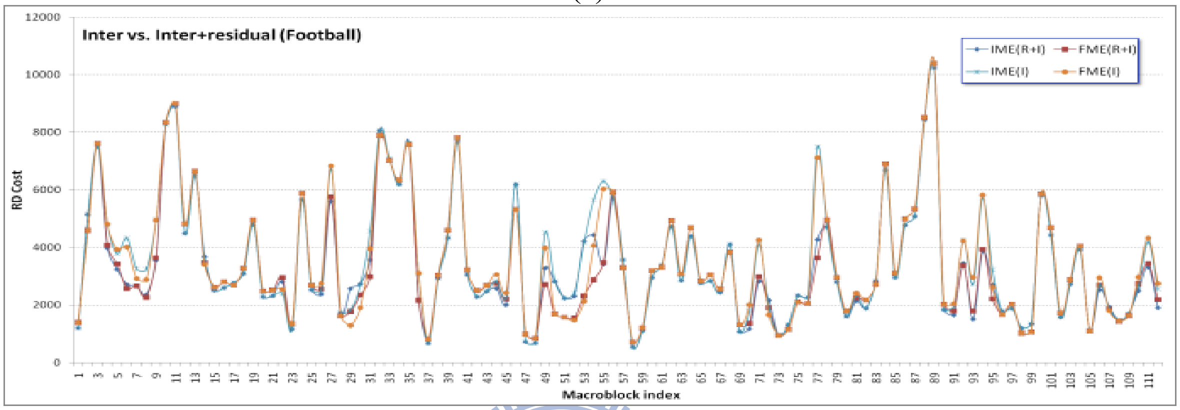


(d)

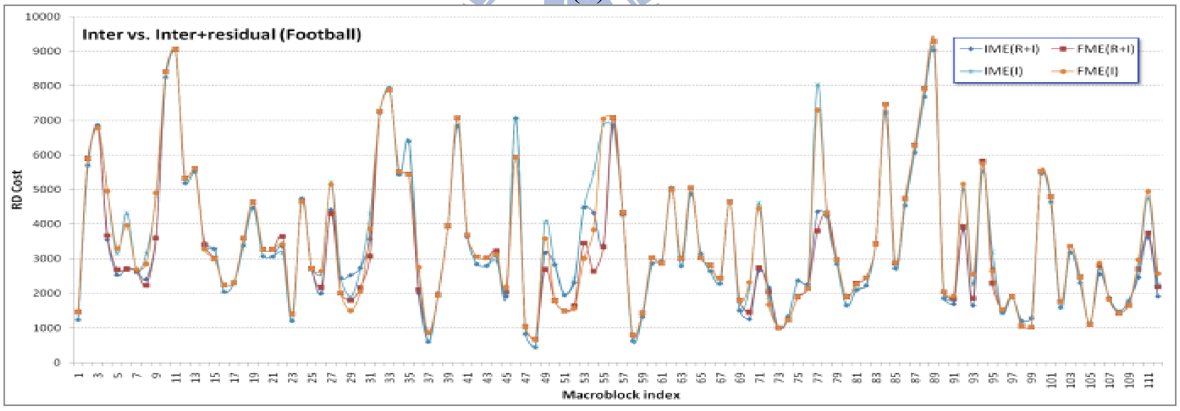
Fig. 4.12. Relationship between *RDCost*s of *IME* and *FME* of Soccer sequence for *Type2* prediction mode combination (a) 16×16, (b) 16×8 (c) 8×16, and (d) Submode



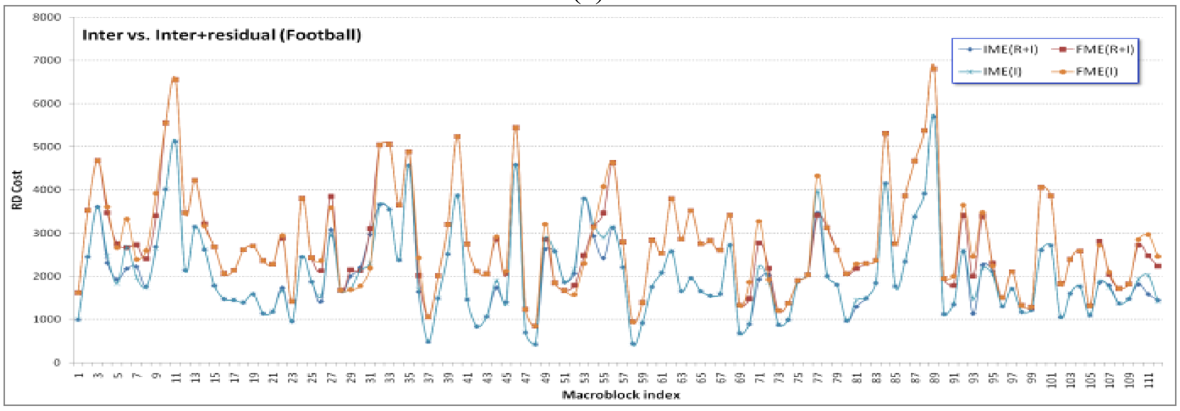
(a)



(b)

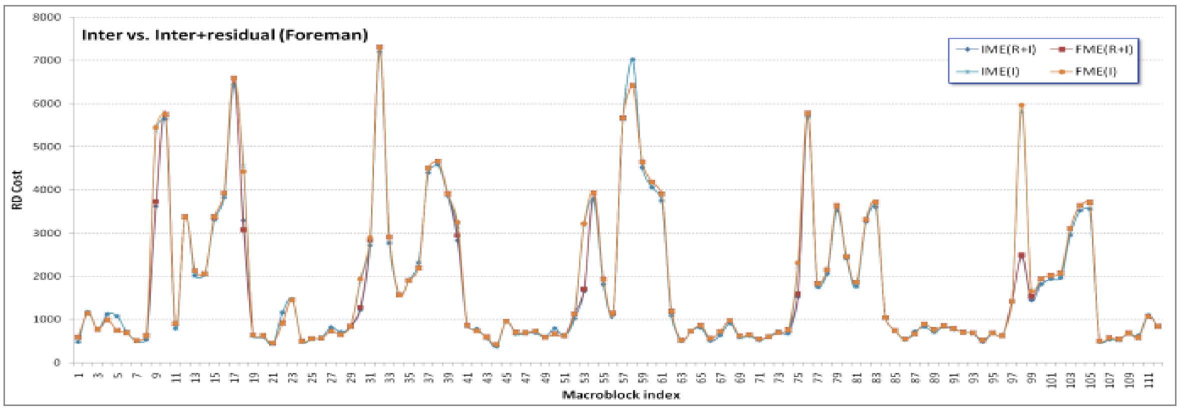


(c)

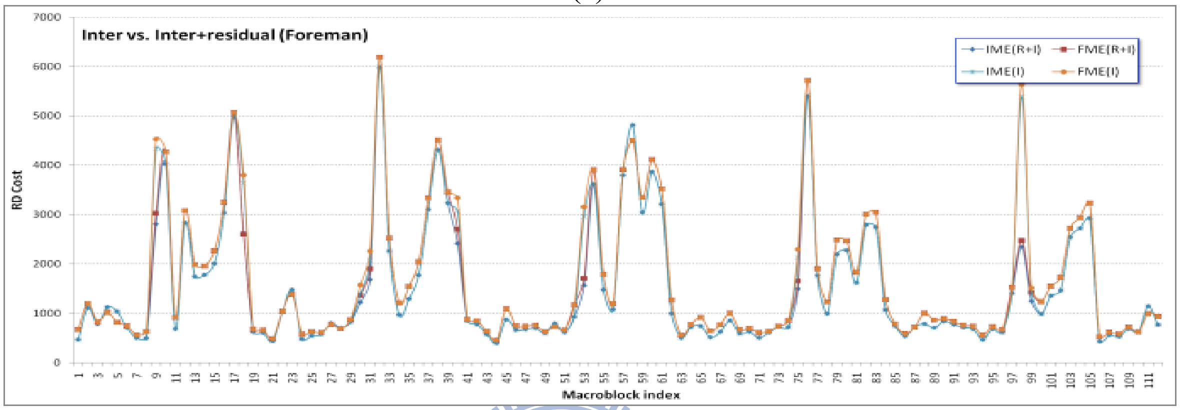


(d)

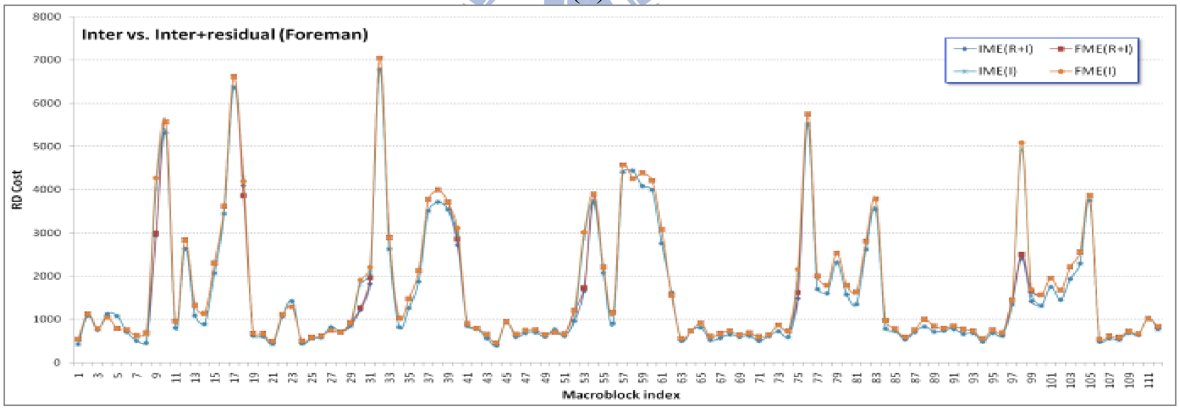
Fig. 4.13. Relationship between *RDCost*s of *IME* and *FME* of Football sequence for *Type3* prediction mode combination (a) 16×16, (b) 16×8 (c) 8×16, and (d) Submode



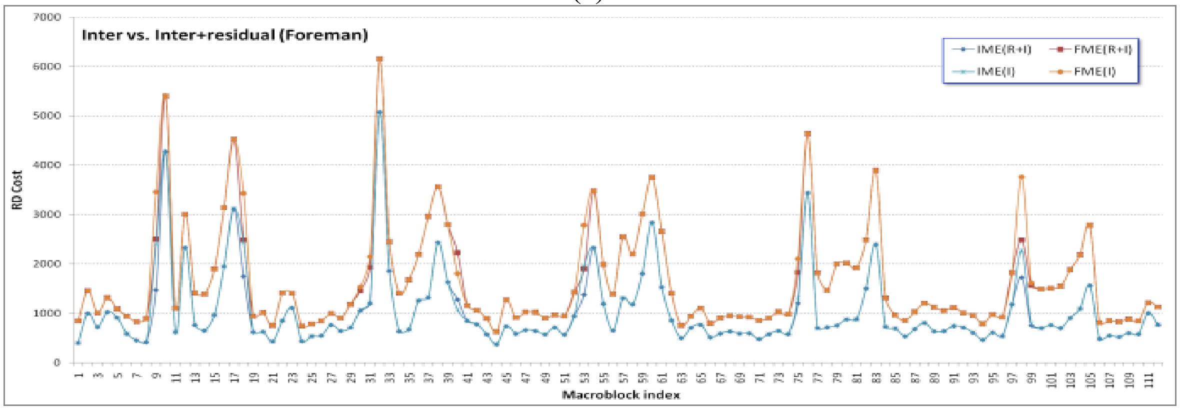
(a)



(b)

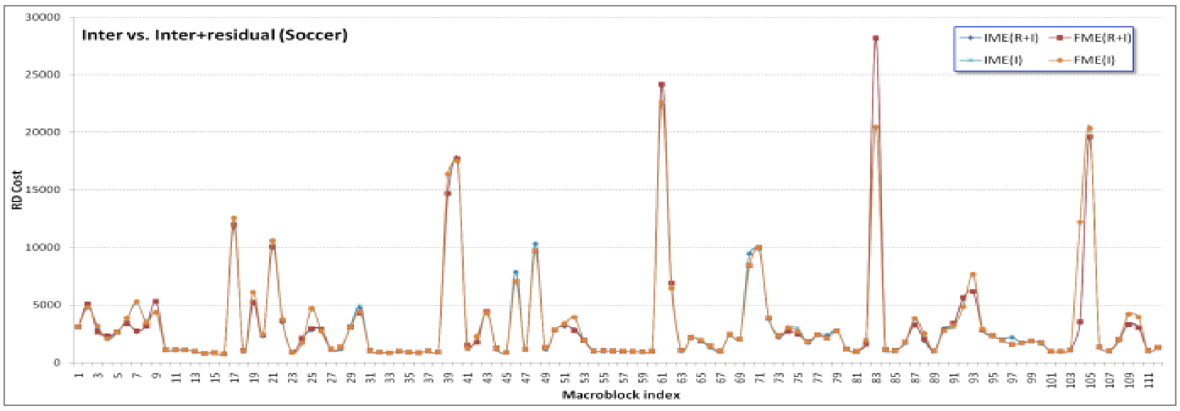


(c)

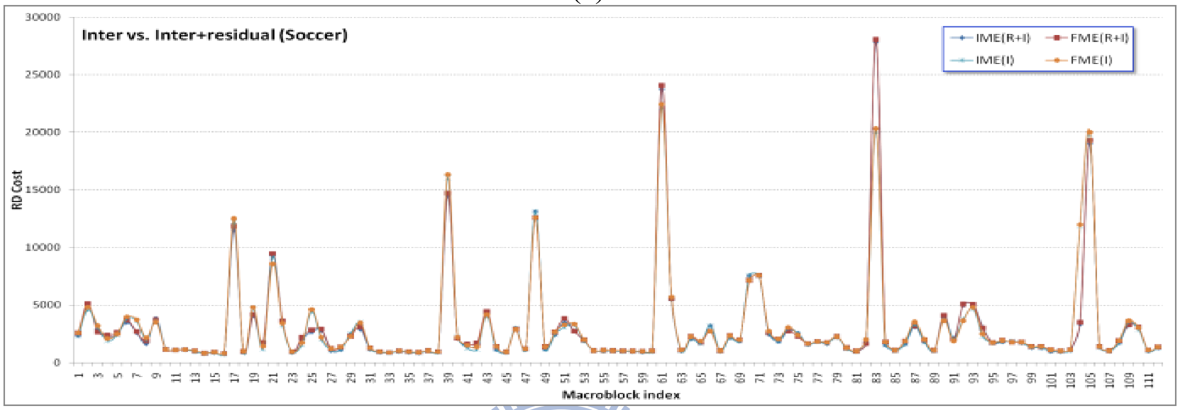


(d)

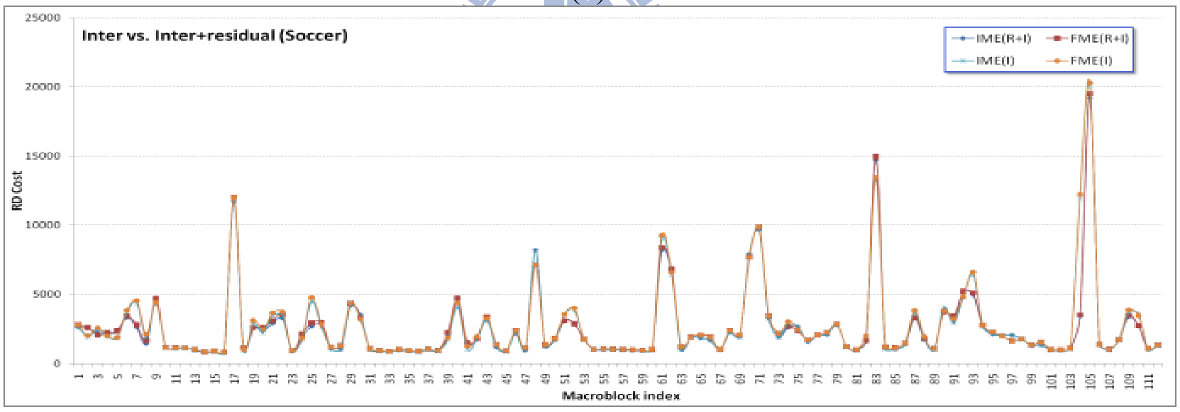
Fig. 4.14. Relationship between *RDCost*s of *IME* and *FME* of Foreman sequence for *Type3* prediction mode combination (a) 16×16, (b) 16×8 (c) 8×16, and (d) Submode



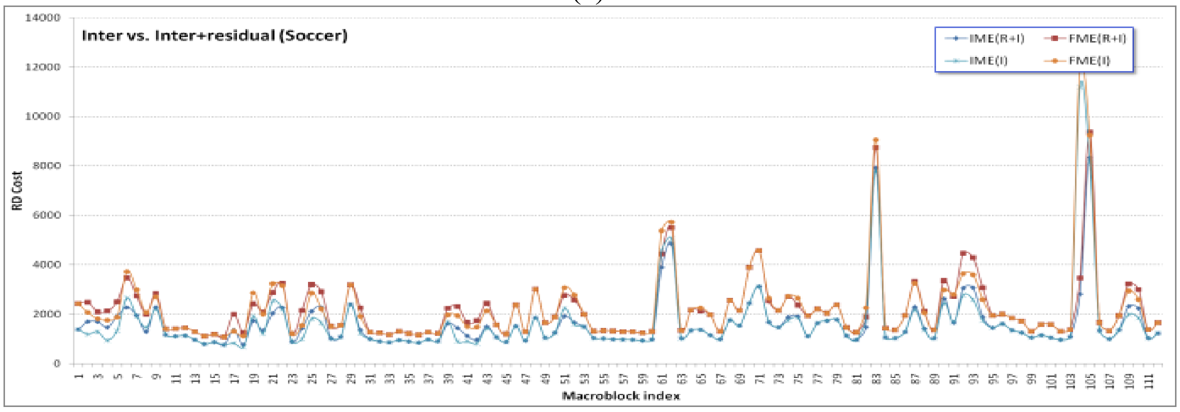
(a)



(b)

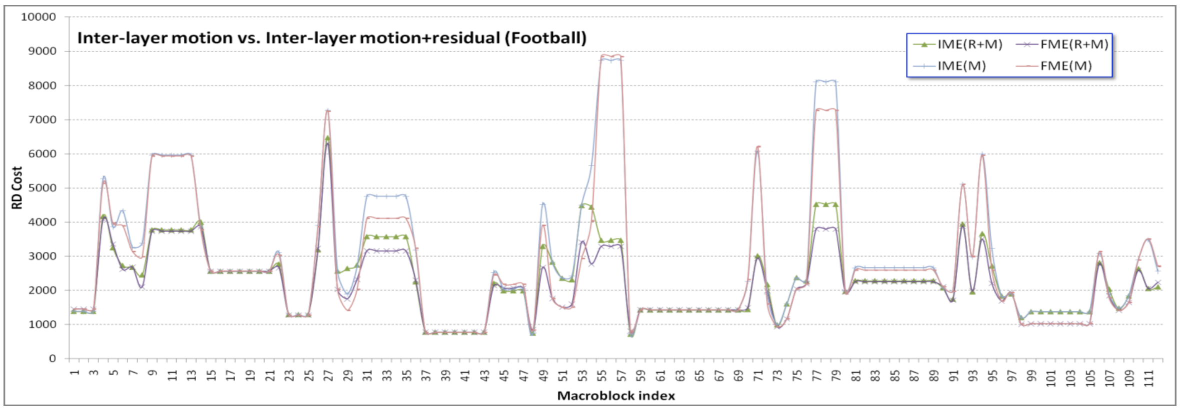


(c)

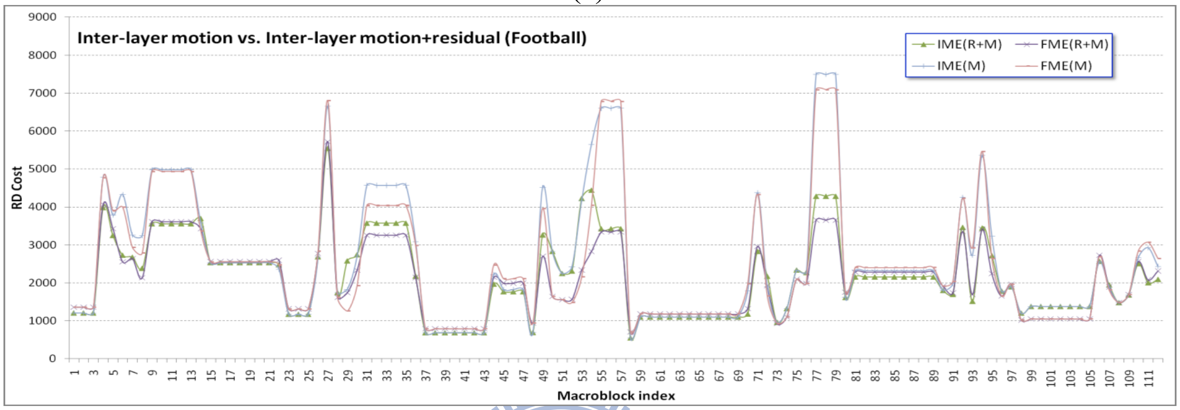


(d)

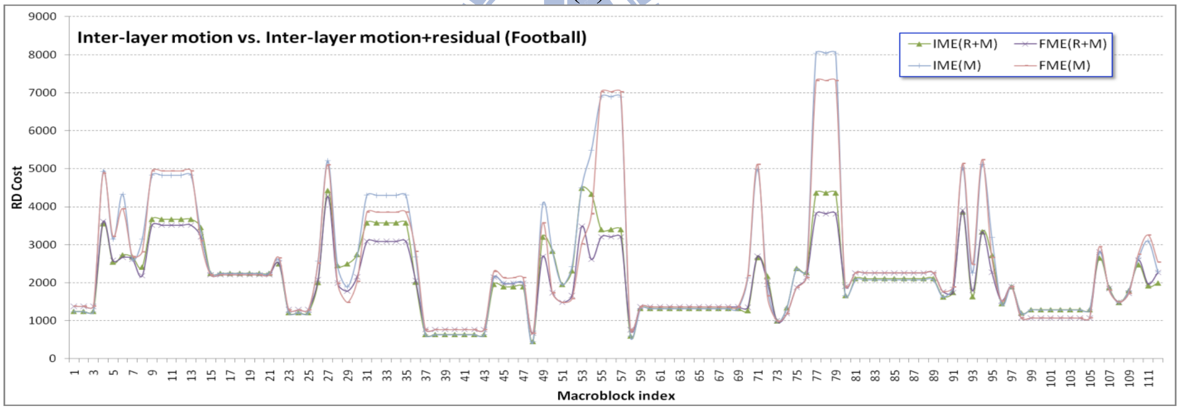
Fig. 4.15. Relationship between *RDCost*s of *IME* and *FME* of Soccer sequence for *Type3* prediction mode combination (a) 16×16, (b) 16×8 (c) 8×16, and (d) Submode



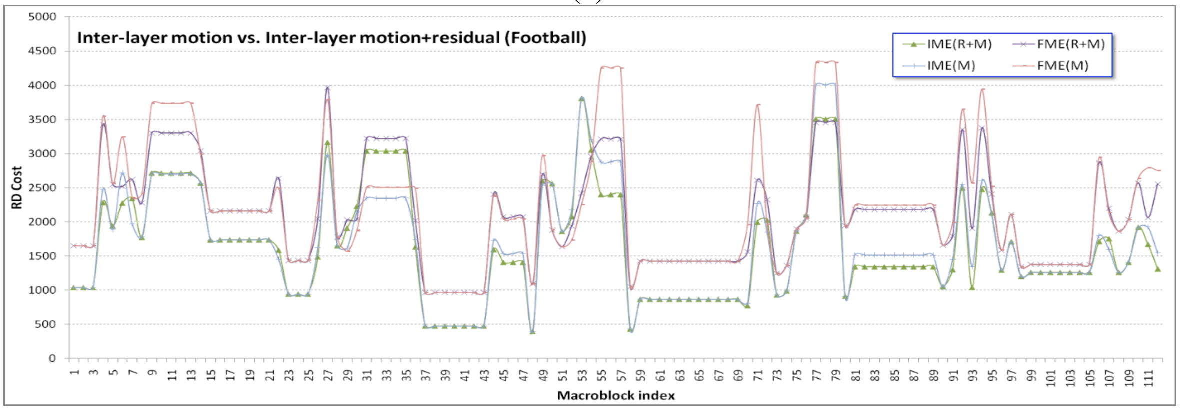
(a)



(b)

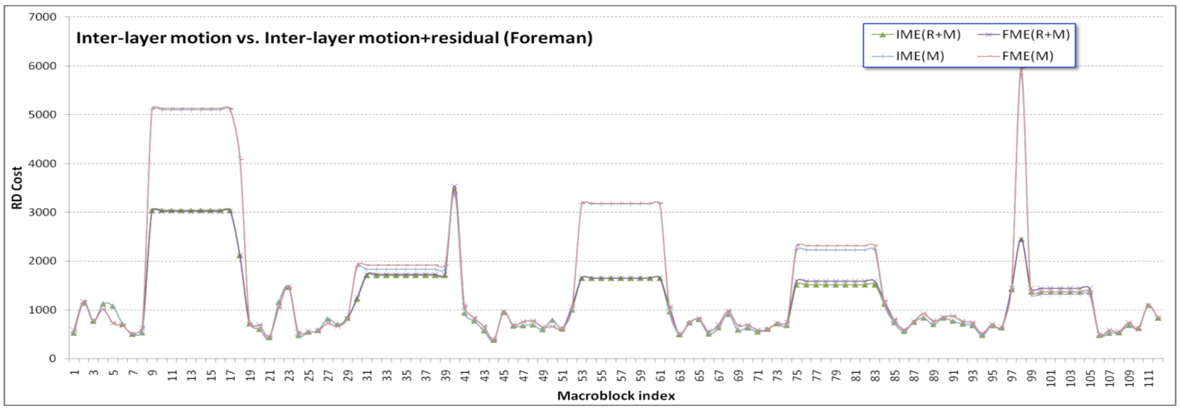


(c)

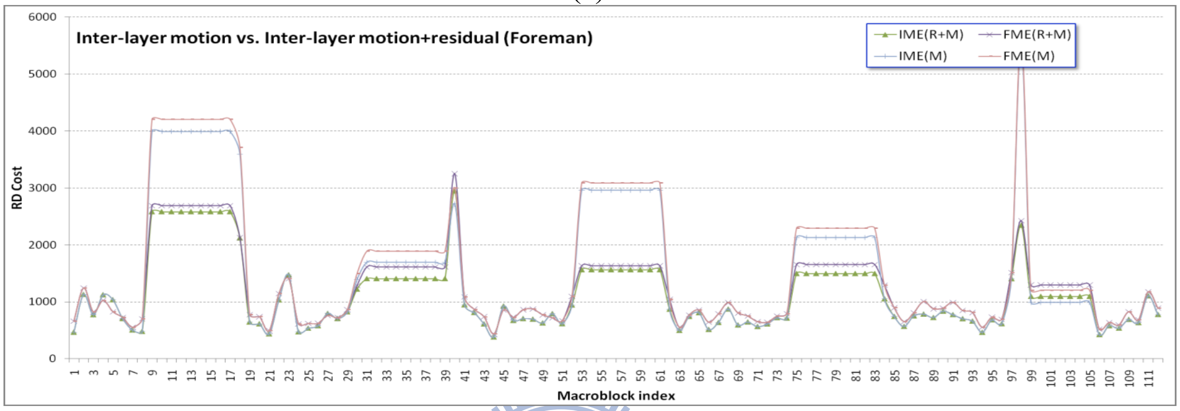


(d)

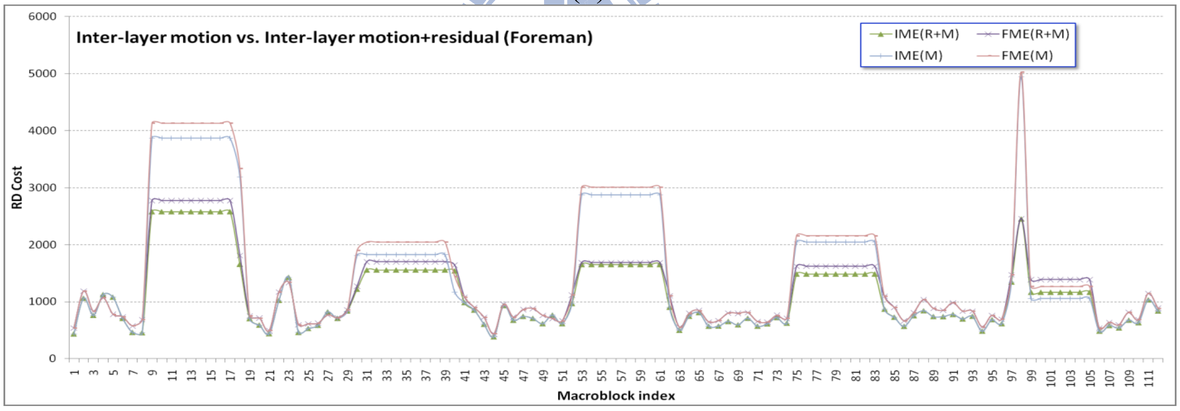
Fig. 4.16. Relationship between *RDCost*s of *IME* and *FME* of Football sequence for *Type4* prediction mode combination (a) 16×16, (b) 16×8 (c) 8×16, and (d) Submode



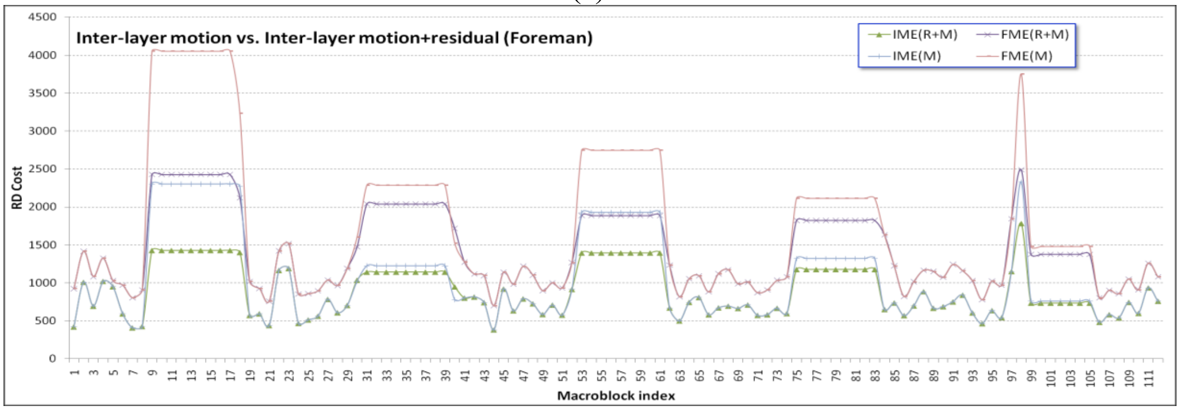
(a)



(b)

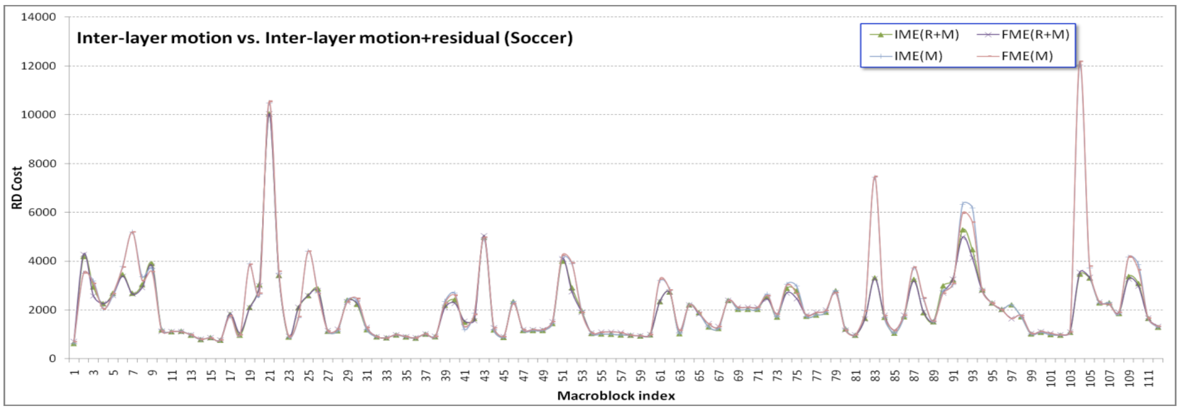


(c)

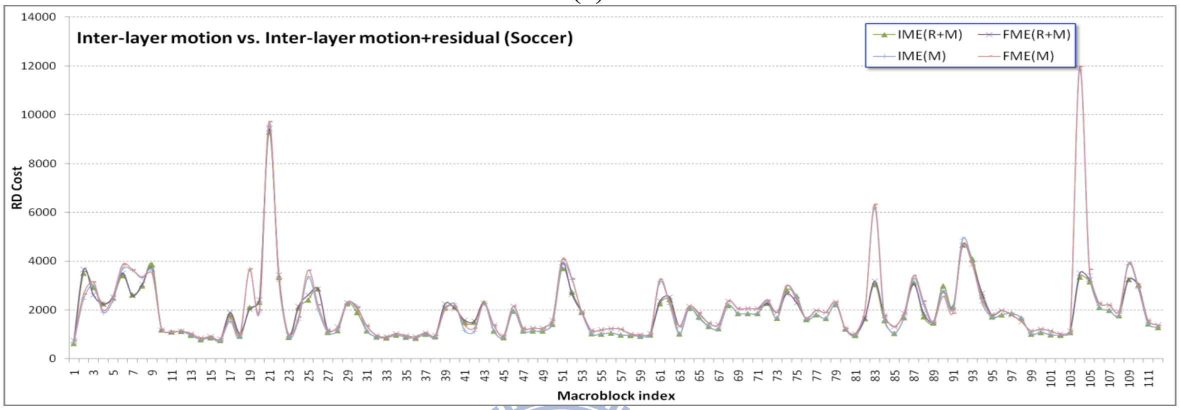


(d)

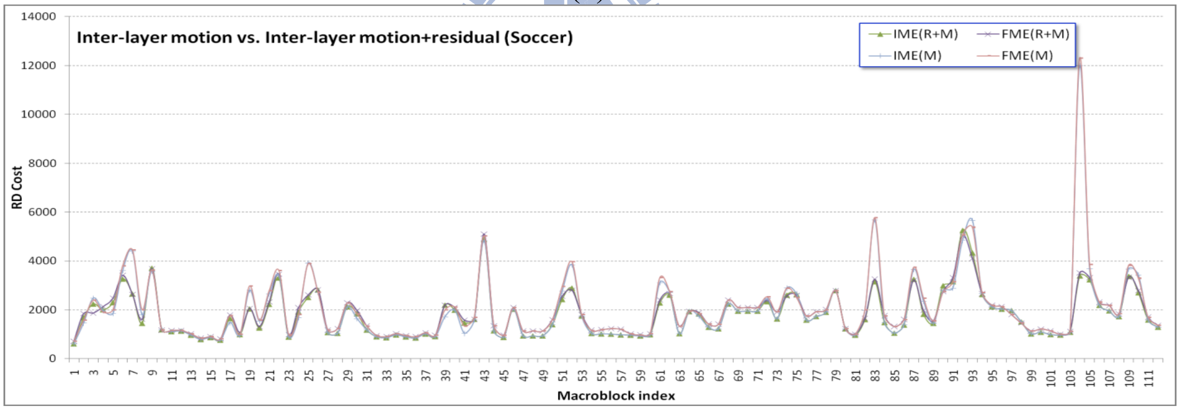
Fig. 4.17. Relationship between *RDCost*s of *IME* and *FME* of Foreman sequence for *Type4* prediction mode combination (a) 16×16, (b) 16×8 (c) 8×16, and (d) Submode



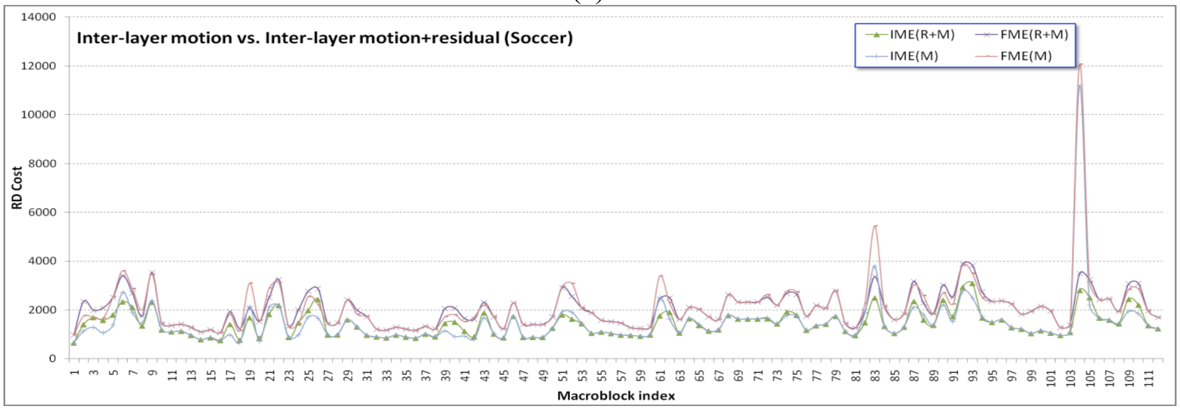
(a)



(b)



(c)



(d)

Fig. 4.18. Relationship between *RDCost*s of *IME* and *FME* of Soccer sequence for *Type4* prediction mode combination (a) 16×16, (b) 16×8 (c) 8×16, and (d) Submode

As mentioned above that if the macroblocks with the *spatial locality* property can be detected, the unnecessary FME mode checking can be skipped and thus achieve the computational complexity reduction. However, quantitatively defining the threshold of "sufficiently large" is not an easy task since different video sequences have different rate distortion cost behavior. There are two directions that can be used to define the term of "sufficiently large" threshold. The first one is to run a large number of simulations and find a properly good value to be the term of "sufficiently large" threshold quantitatively. However, this approach lacks of flexibility to sense the content variation of target images and thus result in the ill rate distortion performance. An alternative approach is to dynamically adjust the threshold according to the image content. To do so, we utilize the relationship between prediction modes to derive the thresholds for different FME mode pre-selection algorithms. The threshold deriving principle for different prediction modes is listed in Table 4-I.

Table 4-I Information used for deriving thresholds for different prediction modes combination

Mode	**Block size**	***Type1***	***Type2***	***Type3***	***Type4***
Inter	**16×16**	YES	NO	NO	YES
	16×8	YES	NO	NO	YES
	8×16	YES	NO	NO	YES
	Submode	NO	NO	NO	NO
Inter+R	**16×16**	NO	NO	YES	YES
	16×8	NO	NO	YES	YES
	8×16	NO	NO	YES	YES
	Submode	NO	NO	NO	NO
ILM	**16×16**	YES	YES	NO	NO
	16×8	YES	YES	NO	NO
	8×16	YES	YES	NO	NO
	Submode	NO	NO	NO	NO
ILM+R	**16×16**	NO	YES	YES	NO
	16×8	NO	YES	YES	NO
	8×16	NO	YES	YES	NO
	Submode	NO	NO	NO	NO

To confirm the validity of our observation, several simulations are conducted by using the conditional probability of *P(A|E)* listed below.

$$\mathrm{P(A|E)} = \frac{P(\mathrm{A \cap E})}{P(\mathrm{E})} \tag{4-1}$$

For different prediction mode combinations, the probability of each event is individually defined as follows.

For Type1

$$\mathrm{P(E)} = \mathrm{P}(IME(\mathrm{I})_{\mathrm{Mode}} + \omega \leq IME(\mathrm{M})_{\mathrm{Mode}} \tag{4-2}$$

$$\mathrm{P}(A) = \mathrm{P}(FME(\mathrm{I})_{\mathrm{Mode}} \leq FME(\mathrm{M})_{\mathrm{Mode}} \tag{4-3}$$

or

$$\mathrm{P(E)} = \mathrm{P}(IME(M)_{\mathrm{Mode}} + \omega \leq IME(I)_{\mathrm{Mode}} \tag{4-4}$$

$$\mathrm{P}(A) = \mathrm{P}(FME(M)_{\mathrm{Mode}} \leq FME(I)_{\mathrm{Mode}} \tag{4-5}$$

$$\omega = \begin{cases} \omega_1 = \dfrac{1}{3} \displaystyle\sum_{m \in \{16\times16, 16\times8, 8\times16\}} |IME(I)_m - IME(M)_m| & _{Mode \in \{16\times16, 16\times8, 8\times16\}} \\ \omega_2 = 0, \quad Mode \in \{Submode\} & \end{cases} \tag{4-6}$$

For Type2

$$\mathrm{P(E)} = \mathrm{P}(IME(R + \mathrm{I})_{\mathrm{Mode}} + \omega \leq IME(R + \mathrm{M})_{\mathrm{Mode}} \tag{4-7}$$

$$\mathrm{P}(A) = \mathrm{P}(FME(R + \mathrm{I})_{\mathrm{Mode}} \leq FME(R + \mathrm{M})_{\mathrm{Mode}} \tag{4-8}$$

or

$$\mathrm{P(E)} = \mathrm{P}(IME(R + M)_{\mathrm{Mode}} + \omega \leq IME(R + I)_{\mathrm{Mode}} \tag{4-9}$$

$$\mathrm{P}(A) = \mathrm{P}(FME(R + M)_{\mathrm{Mode}} \leq FME(R + I)_{\mathrm{Mode}} \tag{4-10}$$

$$\omega = \begin{cases} \omega_1 = \dfrac{1}{3} \displaystyle\sum_{m \in \{16\times16, 16\times8, 8\times16\}} |IME(R + I)_m - IME(R + M)_m| & _{Mode \in \{16\times16, 16\times8, 8\times16\}} \\ \omega_2 = 0, \quad Mode \in \{Submode\} & \end{cases} \tag{4-11}$$

For Type3

$$\mathrm{P(E)} = \mathrm{P}(IME(\mathrm{I})_{\mathrm{Mode}} + \omega \leq IME(R+I)_{\mathrm{Mode}} \tag{4-12}$$

$$\mathrm{P}(A) = \mathrm{P}(FME(\mathrm{I})_{\mathrm{Mode}} \leq FME(R+I)_{\mathrm{Mode}} \tag{4-13}$$

or

$$\mathrm{P(E)} = \mathrm{P}(IME(R+I)_{\mathrm{Mode}} + \omega \leq IME(I)_{\mathrm{Mode}} \tag{4-14}$$

$$\mathrm{P}(A) = \mathrm{P}(FME(R+I)_{\mathrm{Mode}} \leq FME(I)_{\mathrm{Mode}} \tag{4-15}$$

$$\omega = \begin{cases} \omega_1 = \frac{1}{3} \sum_{m \in \{16\times16, 16\times8, 8\times16\}} |IME(I)_m - IME(R+I)_m| & _{Mode \in \{16\times16, 16\times8, 8\times16\}} \\ \omega_2 = 0, \quad Mode \in \{Submode\} \end{cases} \tag{4-16}$$

For Type4

$$\mathrm{P(E)} = \mathrm{P}(IME(M)_{\mathrm{Mode}} + \omega \leq IME(R+M)_{\mathrm{Mode}} \tag{4-17}$$

$$\mathrm{P}(A) = \mathrm{P}(FME(M)_{\mathrm{Mode}} \leq FME(R+M)_{\mathrm{Mode}} \tag{4-18}$$

or

$$\mathrm{P(E)} = \mathrm{P}(IME(R+M)_{\mathrm{Mode}} + \omega \leq IME(M)_{\mathrm{Mode}} \tag{4-19}$$

$$\mathrm{P}(A) = \mathrm{P}(FME(R+M)_{\mathrm{Mode}} \leq FME(M)_{\mathrm{Mode}} \tag{4-20}$$

$$\omega = \begin{cases} \omega_1 = \frac{1}{3} \sum_{m \in \{16\times16, 16\times8, 8\times16\}} |IME(M)_m - IME(R+M)_m| & _{Mode \in \{16\times16, 16\times8, 8\times16\}} \\ \omega_2 = 0, \quad Mode \in \{Submode\} \end{cases} \tag{4-21}$$

4.2.2. Statistical Results

The statistical results are shown in Table 4-II to Table 4-V. For *Type1, Type2, Type3*, and *Type4* prediction mode combinations, the conditional probability can achieve 82.39%, 72.27%, 96.21%, and 95.93% on average, respectively. Therefore, from these tables, we can make sure that our observations work well and the conditions listed above can be used to derive our FME mode pre-selection algorithm to skip the potentially ignorable prediction modes and thus achieve computational complexity savings.

Table 4-II Statistical results of *Type1*

Sequences	**16×16**	**16×8**	**8×16**	**Submode**
Akiyo	96.75	98.22	96.39	96.02
Table	76.48	79.59	75.63	82.10
News	95.45	97.93	95.10	94.45
Tempete	78.53	82.68	76.43	74.46
Football	71.02	85.31	70.18	72.31
Foreman	74.11	81.55	73.05	74.84
M&D	95.58	97.10	94.75	95.00
Soccer	69.27	76.77	69.10	75.72
Stefan	71.25	80.43	71.31	70.96
Average	**80.94**	**86.62**	**80.22**	**81.76**

Table 4-III Statistical results of *Type2*

Sequences	**16×16**	**16×8**	**8×16**	**Submode**
Akiyo	83.31	99.31	86.99	94.78
Table	62.24	81.27	64.80	58.01
News	70.84	93.03	66.29	64.67
Tempete	75.56	79.32	73.23	58.73
Football	63.35	86.66	66.12	61.26
Foreman	64.77	72.21	73.41	72.59
M&D	76.66	81.71	71.01	64.02
Soccer	56.74	76.30	61.73	60.40
Stefan	67.41	79.23	65.71	67.94
Average	**68.99**	**83.23**	**69.92**	**66.93**

Table 4-IV Statistical results of *Type3*

Sequences	**16×16**	**16×8**	**8×16**	**Submode**
Akiyo	93.55	94.46	91.60	88.37
Table	99.47	98.97	98.72	96.50
News	99.77	99.88	99.96	98.24
Tempete	95.20	93.96	92.73	87.57
Football	99.63	99.25	98.94	92.41
Foreman	99.25	99.35	99.59	96.93
M&D	99.18	98.33	97.30	74.39
Soccer	99.63	98.97	99.04	96.91
Stefan	99.12	96.88	96.88	92.56
Average	**98.31**	**97.78**	**97.20**	**91.54**

Table 4-V Statistical results of *Type4*

Sequences	**16×16**	**16×8**	**8×16**	**Submode**
Akiyo	89.35	100.00	93.62	88.07
Table	99.08	99.10	99.14	95.59
News	99.83	99.94	99.77	98.14
Tempete	94.63	93.74	92.69	89.39
Football	99.21	97.81	97.46	90.05
Foreman	99.70	99.11	99.17	93.22
M&D	95.79	99.88	97.21	78.01
Soccer	99.50	99.31	99.35	96.52
Stefan	97.51	95.29	95.94	91.46
Average	**97.18**	**98.24**	**97.15**	**91.16**

4.3. Proposed FME Mode Pre-selection Algorithm

Fig. 4.19 shows the proposed FME mode pre-selection concept for SVC. In our proposed FME mode pre-selection algorithm, the IME is executed for the prediction modes of *Inter*, *Inter+residual*, *Inter-layer motion*, and *Inter-layer motion+residual* first. Afterwards, the proposed FME mode pre-selection algorithm is applied for all prediction modes coming from the results of IME to skip the potentially ignorable prediction modes before entering the FME operation. Once the candidate prediction modes have been decided by the proposed FME mode pre-selection algorithm, the selected candidate modes will be fed into the FME module to choose the best prediction mode.

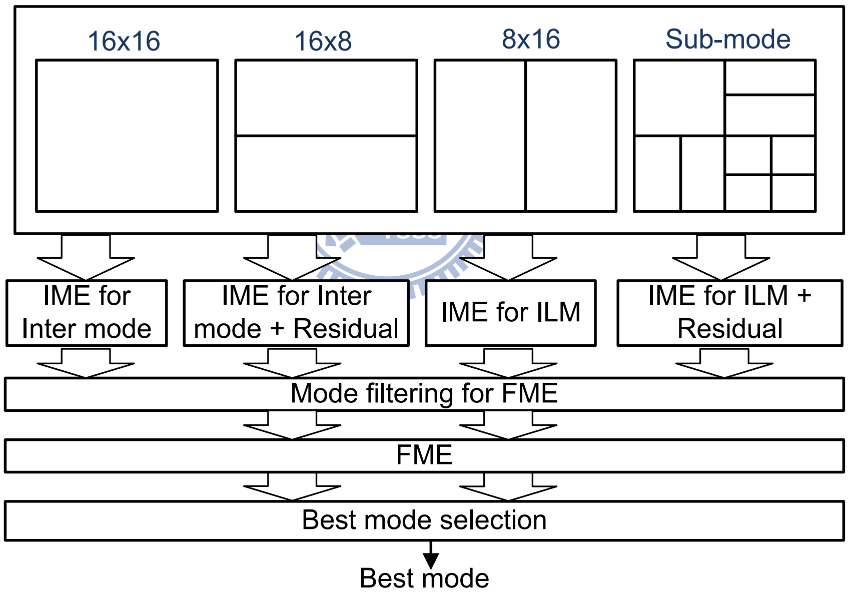


Fig. 4.19. Illustration of mode FME mode pre-selection for SVC

Fig. 4.20 shows the flowchart of our proposed mode pre-selection algorithm which includes four types of mode pre-selection algorithm. In this flowchart, the candidate set of prediction modes Φ is defined as follows.

$$\Phi = \{\Phi_{\mathrm{ij}} | \mathrm{i} \in \{Inter, ILM, InterR, ILMR\}, \mathrm{j} \in \{16 \times 16, 16 \times 8, 8 \times 16, Submode\}\} \tag{4-22}$$

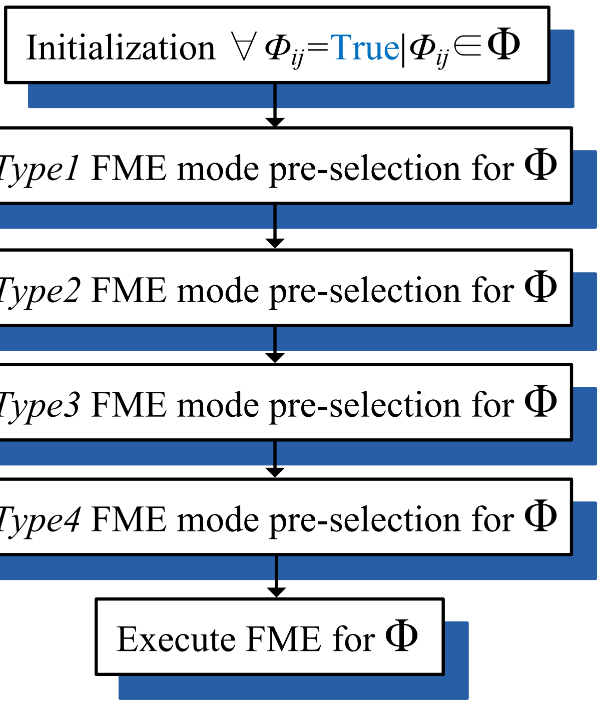
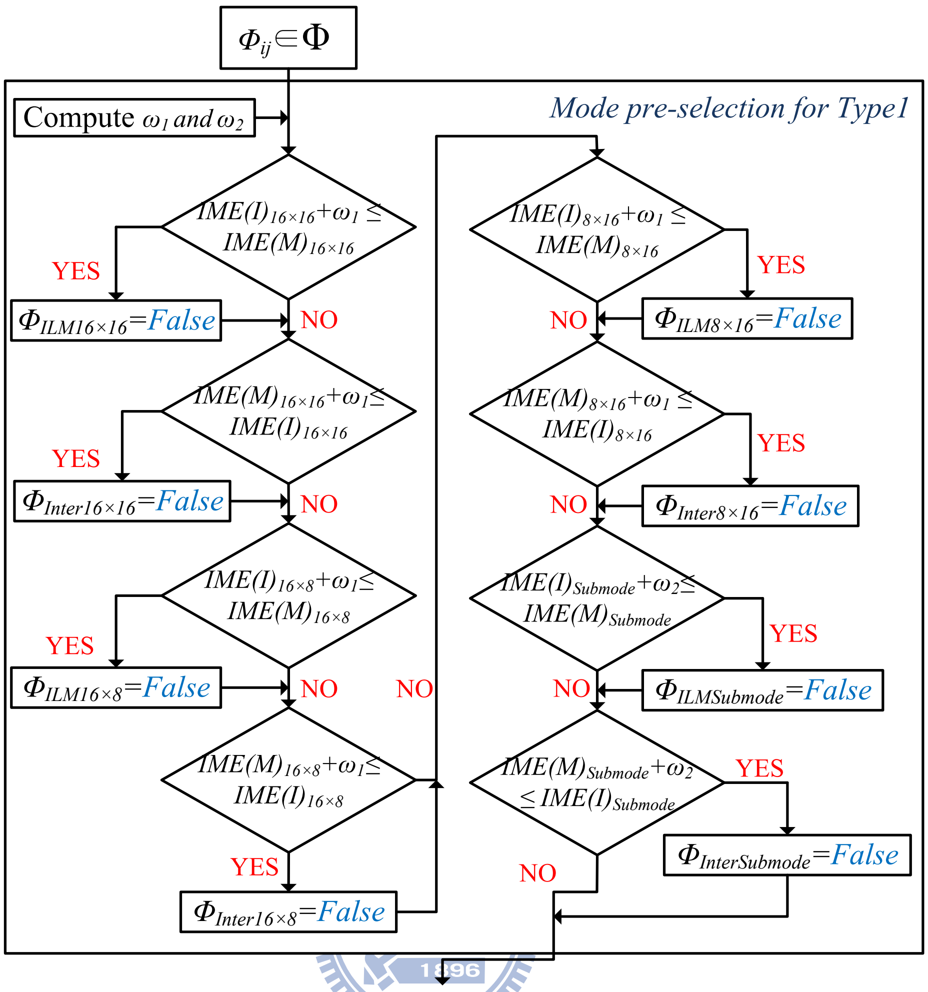

Initialization $\forall\, \Phi_{ij}=\text{True}|\Phi_{ij}\in\Phi$
Type1 FME mode pre-selection for Φ
Type2 FME mode pre-selection for Φ
Type3 FME mode pre-selection for Φ
Type4 FME mode pre-selection for Φ
Execute FME for Φ

Fig. 4.20. Flowchart of proposed FME mode pre-selection algorithm for SVC

Fig. 4.21 shows the detailed flowchart of proposed FME mode pre-selection algorithm for *Type1*. In this flowchart, the weightings of ω_1 and ω_2 are computed first. Afterwards, the IME rate distortion cost relationship between Inter and Inter-layer motion is compared for all block size one by one to skip the potentially ignorable prediction modes. Once the *Type1* mode pre-selection algorithm has been done, the candidate set of Φ will be fed into the next mode pre-selection algorithm to further pre-select possible modes. Fig. 4.22 exhibits the detailed flowchart of proposed FME mode pre-selection algorithm for *Type2*. The flowchart of *Type2* mode pre-selection algorithm is very similar to the mode pre-selection algorithm of *Type1* except that the rate distortion cost relationship is compared between *Inter+residual* and *Inter-layer motion+residual*.



$\Phi_{ij} \in \Phi$
Compute ω_1 and ω_2
Mode pre-selection for Type1
$IME(I)_{16\times16}+\omega_1 \leq IME(M)_{16\times16}$
YES
$\Phi_{ILM16\times16}$=False
NO
$IME(M)_{16\times16}+\omega_1 \leq IME(I)_{16\times16}$
YES
$\Phi_{Inter16\times16}$=False
NO
$IME(I)_{16\times8}+\omega_1 \leq IME(M)_{16\times8}$
YES
$\Phi_{ILM16\times8}$=False
NO
NO
$IME(M)_{16\times8}+\omega_1 \leq IME(I)_{16\times8}$
YES
$\Phi_{Inter16\times8}$=False
$IME(I)_{8\times16}+\omega_1 \leq IME(M)_{8\times16}$
YES
$\Phi_{ILM8\times16}$=False
NO
$IME(M)_{8\times16}+\omega_1 \leq IME(I)_{8\times16}$
YES
$\Phi_{Inter8\times16}$=False
NO
$IME(I)_{Submode}+\omega_2 \leq IME(M)_{Submode}$
YES
$\Phi_{ILMSubmode}$=False
NO
$IME(M)_{Submode}+\omega_2 \leq IME(I)_{Submode}$
YES
$\Phi_{InterSubmode}$=False
NO

Fig. 4.21. Detailed flowchart of proposed *Type1* mode pre-selection algorithm

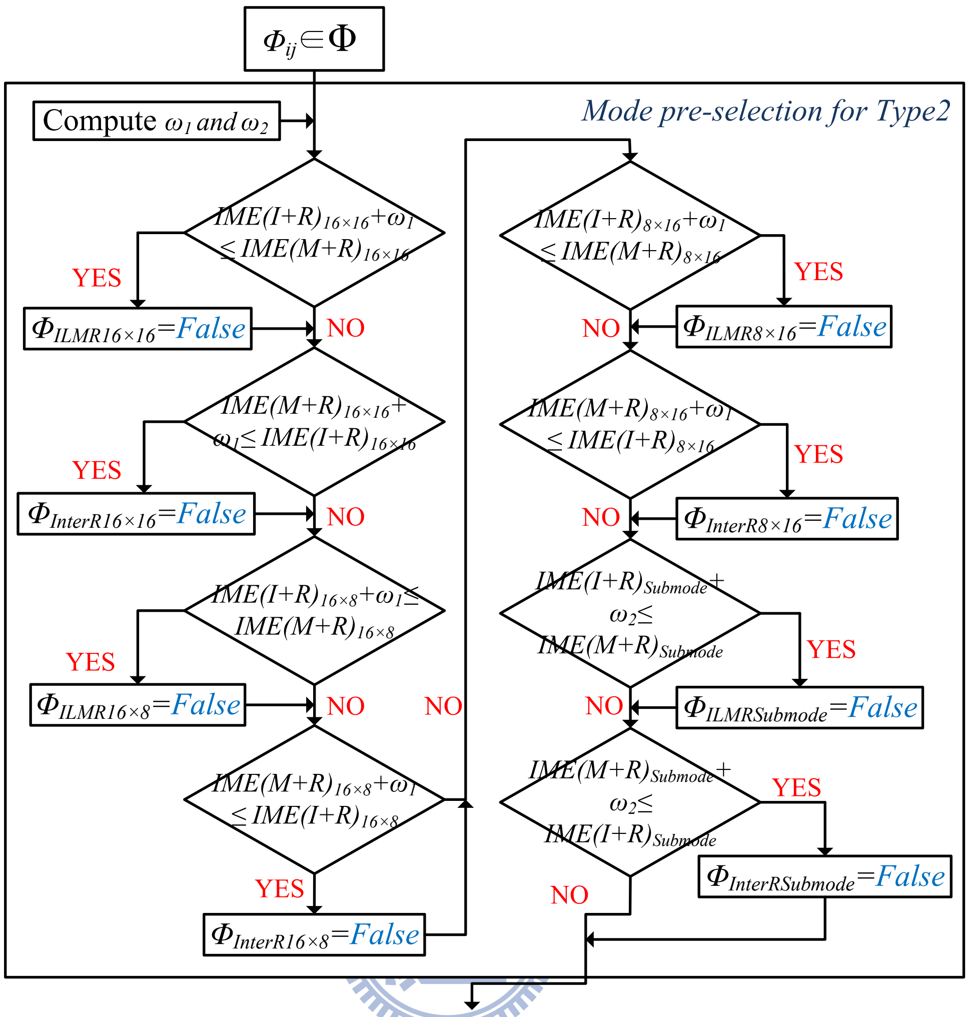
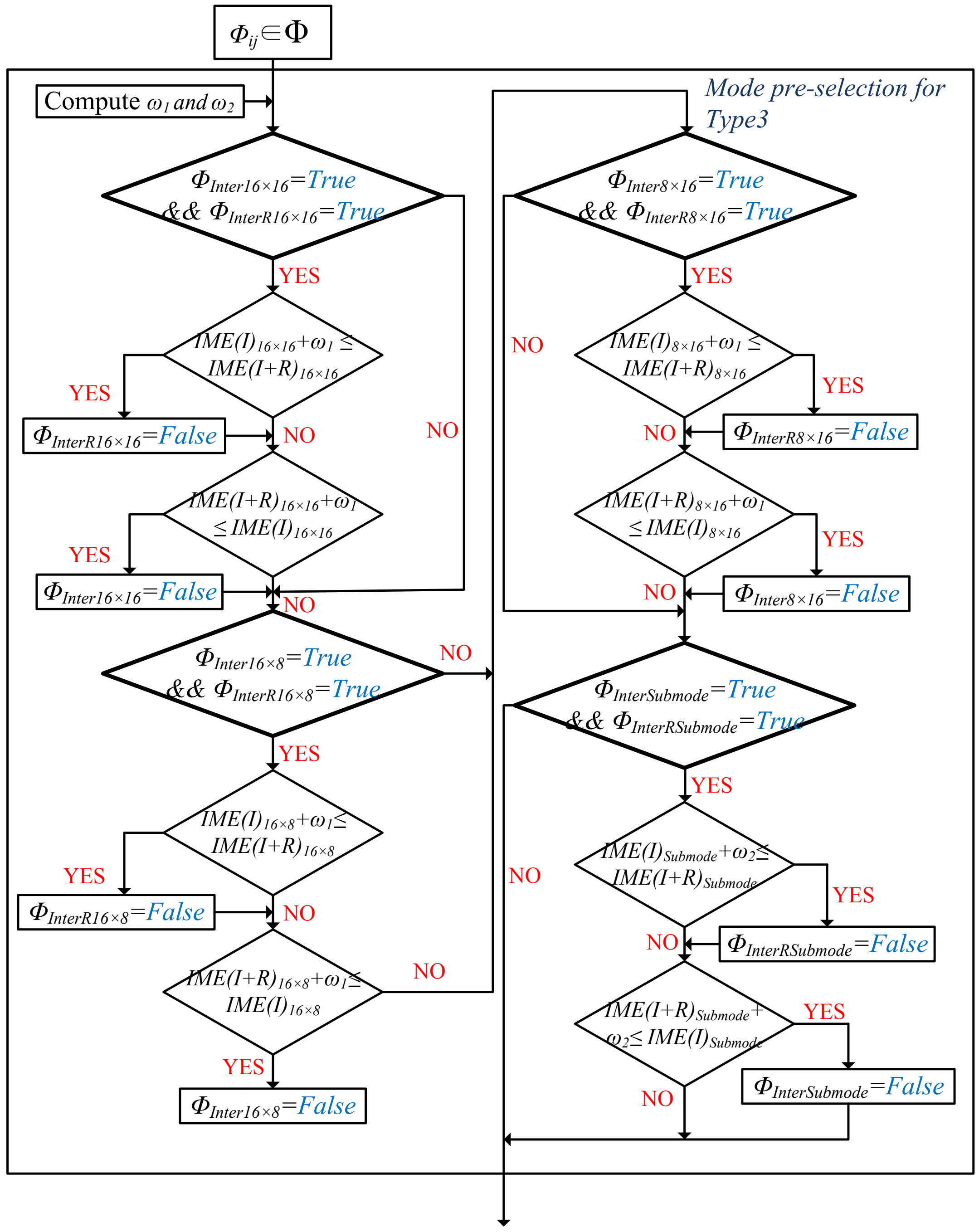

$\Phi_{ij} \in \Phi$
Compute ω_1 and ω_2
Mode pre-selection for Type2
$IME(I+R)_{16\times16}+\omega_1 \leq IME(M+R)_{16\times16}$
YES
$\Phi_{ILMR16\times16}$=False
NO
$IME(M+R)_{16\times16}+\omega_1 \leq IME(I+R)_{16\times16}$
YES
$\Phi_{InterR16\times16}$=False
NO
$IME(I+R)_{16\times8}+\omega_1 \leq IME(M+R)_{16\times8}$
YES
$\Phi_{ILMR16\times8}$=False
NO
NO
$IME(M+R)_{16\times8}+\omega_1 \leq IME(I+R)_{16\times8}$
YES
$\Phi_{InterR16\times8}$=False
$IME(I+R)_{8\times16}+\omega_1 \leq IME(M+R)_{8\times16}$
YES
$\Phi_{ILMR8\times16}$=False
NO
$IME(M+R)_{8\times16}+\omega_1 \leq IME(I+R)_{8\times16}$
YES
$\Phi_{InterR8\times16}$=False
NO
$IME(I+R)_{Submode}+\omega_2 \leq IME(M+R)_{Submode}$
YES
$\Phi_{ILMRSubmode}$=False
NO
$IME(M+R)_{Submode}+\omega_2 \leq IME(I+R)_{Submode}$
YES
$\Phi_{InterRSubmode}$=False
NO

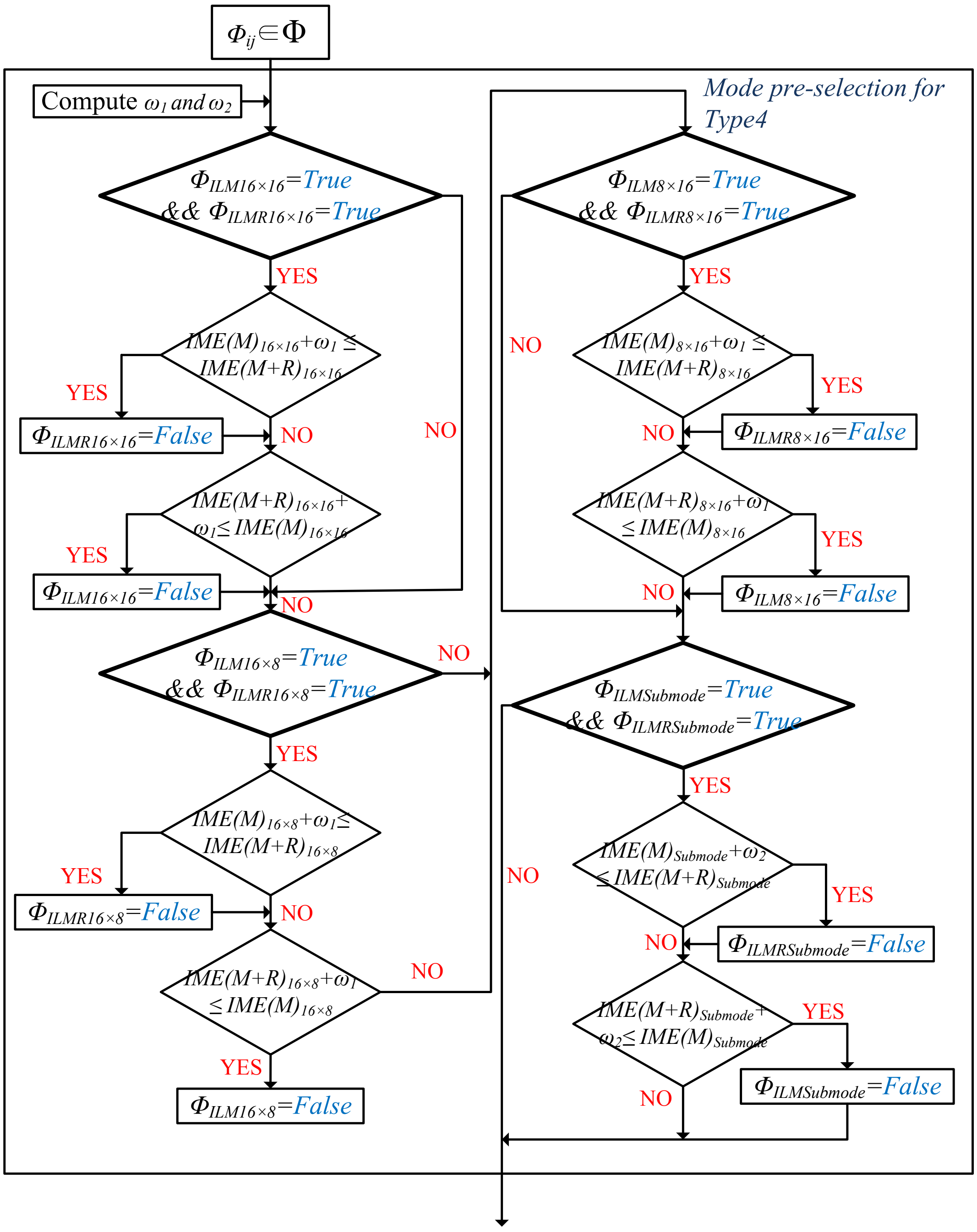
Fig. 4.22. Detailed flowchart of proposed *Type2* mode pre-selection algorithm

Fig. 4.23 and Fig. 4.24 show the detailed flowchart of proposed *Type3* and *Type4* mode pre-selection algorithm, respectively. From these figures, the most determination processes are similar to the determination processes of *Type1* and *Type2* mode pre-selection algorithms. However, there are some addition determination processes labeled by bold line have been further included. The additional included determination processes here are in charge of avoiding unnecessary determination operations since some candidate prediction modes might have been already disabled in the previous *Type1* and *Type2* mode pre-selection algorithms. Therefore, by adding additional determination processes into the mode pre-selection flowchart, the unnecessary operations can be avoided and thus achieve computational complexity saving.



$\Phi_{ij} \in \Phi$
Compute ω_1 and ω_2
Mode pre-selection for Type3
$\Phi_{Inter16\times16}$=True && $\Phi_{InterR16\times16}$=True
YES
$IME(I)_{16\times16}+\omega_1 \leq IME(I+R)_{16\times16}$
YES
$\Phi_{InterR16\times16}$=False
NO
NO
$IME(I+R)_{16\times16}+\omega_1 \leq IME(I)_{16\times16}$
YES
$\Phi_{Inter16\times16}$=False
NO
$\Phi_{Inter16\times8}$=True && $\Phi_{InterR16\times8}$=True
NO
YES
$IME(I)_{16\times8}+\omega_1 \leq IME(I+R)_{16\times8}$
YES
$\Phi_{InterR16\times8}$=False
NO
$IME(I+R)_{16\times8}+\omega_1 \leq IME(I)_{16\times8}$
NO
YES
$\Phi_{Inter16\times8}$=False
$\Phi_{Inter8\times16}$=True && $\Phi_{InterR8\times16}$=True
YES
NO
$IME(I)_{8\times16}+\omega_1 \leq IME(I+R)_{8\times16}$
YES
$\Phi_{InterR8\times16}$=False
NO
$IME(I+R)_{8\times16}+\omega_1 \leq IME(I)_{8\times16}$
YES
$\Phi_{Inter8\times16}$=False
NO
$\Phi_{InterSubmode}$=True && $\Phi_{InterRSubmode}$=True
YES
NO
$IME(I)_{Submode}+\omega_2 \leq IME(I+R)_{Submode}$
YES
$\Phi_{InterRSubmode}$=False
NO
$IME(I+R)_{Submode}+\omega_2 \leq IME(I)_{Submode}$
YES
$\Phi_{InterSubmode}$=False
NO

Fig. 4.23. Detailed flowchart of proposed *Type3* mode pre-selection algorithm



Φij∈Φ
Compute ω1 and ω2
Mode pre-selection for Type4
ΦILM16×16=True && ΦILMR16×16=True
YES
IME(M)16×16+ω1 ≤ IME(M+R)16×16
YES
ΦILMR16×16=False
NO
NO
IME(M+R)16×16+ω1≤ IME(M)16×16
YES
ΦILM16×16=False
NO
ΦILM16×8=True && ΦILMR16×8=True
NO
YES
IME(M)16×8+ω1≤ IME(M+R)16×8
YES
ΦILMR16×8=False
NO
IME(M+R)16×8+ω1 ≤ IME(M)16×8
NO
YES
ΦILM16×8=False
ΦILM8×16=True && ΦILMR8×16=True
YES
NO
IME(M)8×16+ω1 ≤ IME(M+R)8×16
YES
NO
ΦILMR8×16=False
IME(M+R)8×16+ω1 ≤ IME(M)8×16
YES
NO
ΦILM8×16=False
ΦILMSubmode=True && ΦILMRSubmode=True
YES
NO
IME(M)Submode+ω2 ≤ IME(M+R)Submode
YES
NO
ΦILMRSubmode=False
IME(M+R)Submode+ ω2≤ IME(M)Submode
YES
NO
ΦILMSubmode=False

Fig. 4.24. Detailed flowchart of proposed *Type4* mode pre-selection algorithm

4.4. Simulation Results

In this section, several simulation results are shown to demonstrate the performance of our proposed FME mode pre-selection algorithm. The simulation settings are summarized in Table 4-VI.

Table 4-VI Simulation settings

Reference software		JSVM9.17 [63]
QP for spatial base layer		18, 28, 33, 38
QP for Spatial enhancement layer		12, 22, 27, 32
Frame size in spatial base layer		QCIF and 540P
Frame size in spatial enhancement layer		CIF and 1080P
Frames to be encoded		300 and 150
Frame rate		30 and 15
Adaptive inter-layer prediction		ON
Search range size		±8
GOP		8
Test sequences	**QCIF and CIF**	Akiyo, Dancer, Coastguard, Table tennis, Tempete, Football, Foreman, MD, Mobile, News, Soccer, Stefan
	540P and 1080P	Blue_sky, Pedestrian, Riverbed, Station2, Sunflower, Tractor

Table 4-VII shows the BD-PSNR and BD-Rate comparisons for our proposed FME mode pre-selection algorithm subject to full mode FME. In this table, the quantization parameter values of 12, 22, 27, and 32 are adopted to derive the results. It should be mentioned that since the only highest quality (including spatial, temporal, SNR) layers would be decoded from the SVC bitstream, we only show the QP values for highest spatial layers. For QCIF and CIF case, the average BD-PSNR degradation and BD-Rate increasing is 0.034 and 0.0777%, respectively. In addition, for the 540P and 1080P case, 0.037 and 0.914% BD-PSNR degradation and BD-Rate increase can be achieved. From this table, it is obvious that our proposed FME mode pre-selection algorithm results in ignorable rate distortion performance loss when compared to full mode FME. Table 4-VIII further shows the detailed PSNR degradation and bitrate increasing for different QP values. On average, our proposed algorithm only results in 0.005dB PSNR degradation and 0.89% bitrate increasing.

Table 4-VII BD-PSNR and BD-Rate comparisons for proposed algorithm subject to full mode FME

Resolution	Sequences	BD-PSNR	BD-Rate (%)
BL: QCIF EL: CIF	Akiyo	-0.025	+0.774
	Coastguard	-0.030	+0.469
	Dancer	-0.056	+0.754
	Football	-0.041	+0.733
	Foreman	-0.052	+1.619
	Mobile	-0.037	+0.636
	MD	-0.031	+1.141
	News	+0.056	-0.783
	Table tennis	-0.045	+0.920
	Tempete	-0.032	+0.586
	Stefan	-0.051	+1.112
	Soccer	-0.069	+1.364
Average		**-0.034**	**+0.777**
BL: 540P EL: 1080P	Blue_sky	-0.051	+1.216
	Pedestrian	-0.017	+0.038
	Riverbed	-0.006	+0.006
	Station2	-0.043	+1.520
	Sunflower	-0.067	+2.547
	Tractor	-0.035	+0.156
Average		**-0.037**	**+0.914**

Table 4-VIII Rate distortion performance comparisons for proposed algorithm

	PSNR degradation (dB)				Bitrate increase (%)			
	QP=12	QP=22	QP=27	QP=32	QP=12	QP=22	QP=27	QP=32
Akiyo	0.00	0.00	0.00	0.01	0.28	0.94	1.48	1.26
Dancer	0.01	0.00	0.00	-0.01	0.47	0.68	1.11	1.34
Coastguard	0.00	-0.01	-0.01	-0.01	0.22	0.37	0.56	0.66
Table	0.00	0.00	-0.01	0.00	0.41	0.92	1.23	1.72
Tempete	0.00	0.00	-0.01	-0.01	0.31	0.55	0.71	1.07
Football	0.00	0.00	0.00	0.00	0.34	0.45	0.88	1.58
Foreman	0.00	-0.01	0.00	-0.02	0.51	1.17	1.42	1.90
MD	0.00	0.00	0.00	0.02	0.42	1.25	1.77	2.22
Mobile	0.00	-0.01	-0.01	0.00	0.33	0.40	0.56	0.86
News	-0.01	-0.01	0.00	-0.02	0.23	0.65	0.54	1.06
Soccer	-0.01	0.00	-0.02	-0.01	0.33	1.67	1.59	0.91
Stefan	0.00	-0.01	0.00	-0.01	0.35	1.14	0.75	0.59
Blue_sky	0.01	-0.01	-0.02	-0.03	0.79	0.67	0.96	1.27
Pedestrian	0.01	-0.01	-0.01	-0.01	0.28	0.49	0.85	0.90
Riverbed	0.01	0.01	0.01	0.01	0.05	0.17	0.43	0.79
Station2	0.02	-0.01	-0.02	-0.02	0.43	1.04	1.51	1.92
Sunflower	0.02	-0.02	-0.01	0.00	0.56	1.16	1.96	2.72
Tractor	0.01	-0.02	-0.04	-0.03	0.50	0.30	-0.84	3.17
Average	**0.003**	**-0.006**	**-0.008**	**-0.008**	**0.38**	**0.78**	**0.97**	**1.44**

Table 4-IX tabulates the mode reductions of our proposed algorithm. On average, our proposed algorithm can achieve 65.97% mode reduction whatever what quantization parameters have been used.

Table 4-IX Mode reductions for proposed algorithm subject to full mode FME

(%)		QP=12	QP=22	QP=27	QP=32
BL:QCIF EL:CIF	Akiyo	-75.00	-75.00	-75.00	-75.00
	Dancer	-75.00	-75.00	-75.00	-75.00
	Coastguard	-62.50	-62.50	-62.50	-62.50
	Table	-62.50	-62.50	-62.50	-62.50
	Tempete	-75.00	-75.00	-75.00	-75.00
	Football	-62.50	-62.50	-62.50	-62.50
	Foreman	-62.50	-62.50	-62.50	-62.50
	MD	-75.00	-75.00	-75.00	-75.00
	Mobile	-62.50	-62.50	-62.50	-62.50
	News	-75.00	-75.00	-75.00	-75.00
	Soccer	-62.50	-62.50	-62.50	-62.50
	Stefan	-62.50	-62.50	-62.50	-62.50
BL:540P EL:1080P	Blue_sky	-62.50	-62.50	-62.50	-62.50
	Pedestrian	-62.50	-62.50	-62.50	-62.50
	Riverbed	-62.50	-62.50	-62.50	-62.50
	Station2	-62.50	-62.50	-62.50	-62.50
	Sunflower	-62.50	-62.50	-62.50	-62.50
	Tractor	-62.50	-62.50	-62.50	-62.50
Average		**-65.97**	**-65.97**	**-65.97**	**-65.97**

4.5. Hardware Architecture Design

Fig. 4.25 reveals the hardware architecture design of our proposed FME more pre-selection algorithm. When designing a modern video encoder system, the IME and FME are usually separated into two individual pipeline stages as Fig. 4.25(a) shown due to the reason of balancing the computational loads for each stage. Therefore, the proposed FME mode pre-selection algorithm should be placed in the middle of two pipeline stages. However, there are two places where we can use for arranging our proposed FME mode pre-selection algorithm. The first place is at the same pipeline stage of IME and the alternative place is at the same pipeline stage of FME. However, placing our proposed FME mode pre-selection algorithm in different places would result in different effects on hardware implementation costs. Intuitively, since our proposed algorithm is used to pre-select potentially ignorable modes for FME, the proposed FME mode pre-selection algorithm is placed at the same pipeline stage of FME naturally. However, although the pre-selected modes can be passed to the FME module for the forthcoming processing directly, there are significant amounts of mode prediction information should be passed from the IME pipeline stage. Hence, in this case, not only the module hardware costs but the pipeline registers would be increased noticeably for storing all information of mode prediction. As a result, it is unreasonable to arrange our proposed FME mode pre-selection algorithm at the same stage of FME. Therefore, our proposed FME mode pre-selection algorithm is thus arranged at the same stage of IME.

Fig. 4.25(b) shows the overall hardware architecture of our proposed FME mode pre-selection algorithm which is mainly composed by four major modules called *Type1, Type2, Type3*, and *Type4* FME mode pre-selection module. The four major modules corresponding to four major mode pre-selection algorithms mentioned in Section III. For *Type1* and *Type2* FME mode pre-selection modules, the mode pre-selection rules are applied to valid the possible modes for the following process. Once the *Type1* and *Type2* modules have finished their mode pre-selection tasks, some mode valid signals will be generated and passed to the other modules. Similarly, the *Type3* and *Type4* modules have the same operations just like *Type1* and *Type2* modules do, but they further take the mode valid signals received from *Type1* and *Type2* modules into consideration. Once all modules have finished their mode pre-selection operations, all generated mode valid signals will send to a multiplexer to select the mode prediction information for output.

The detailed hardware architecture design of proposed *Type1/Type2* and *Type3/Type4* FME mode pre-selection algorithm is shown in Fig. 4.26 and Fig. 4.27, respectively.

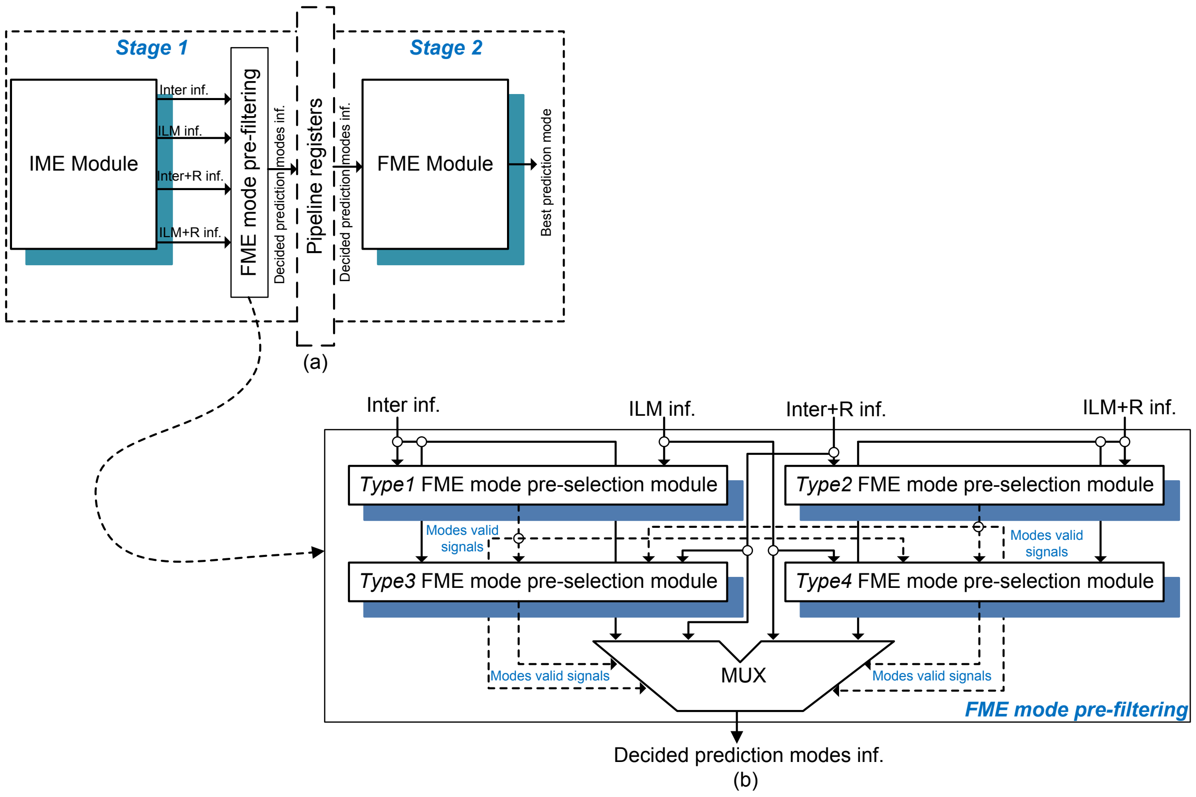
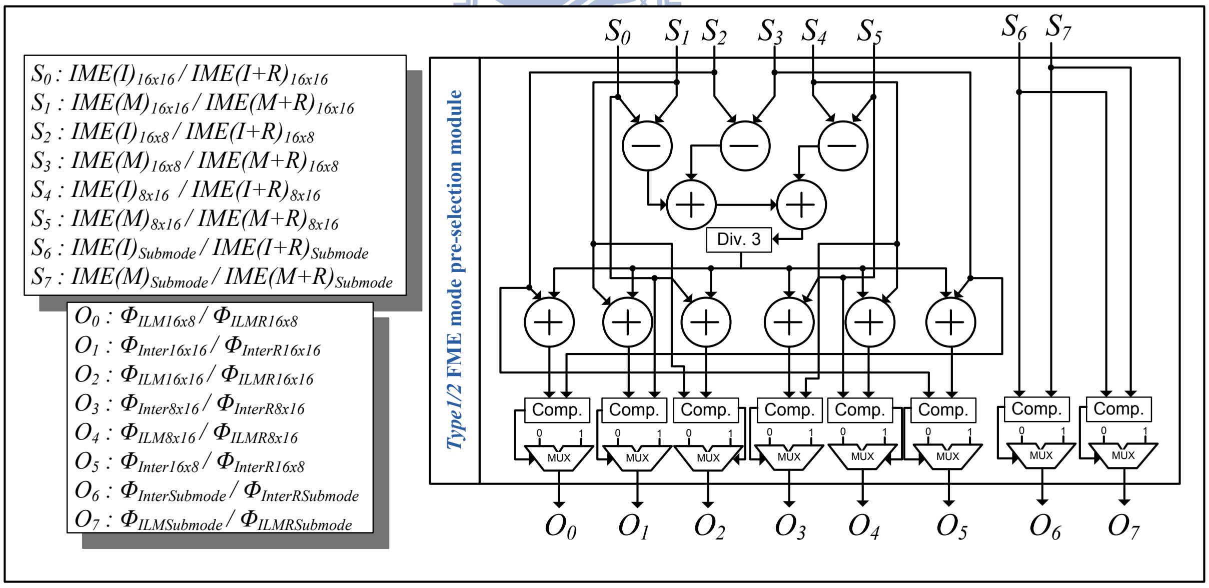
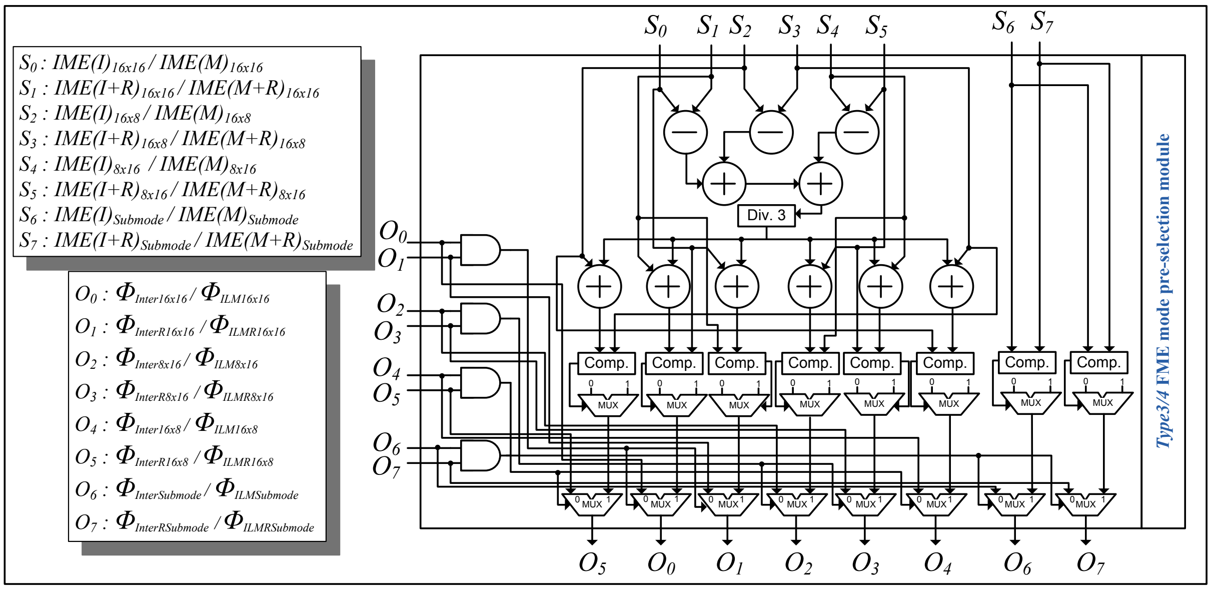


Fig. 4.25. Hardware architecture of our proposed FME mode pre-selection algorithm (a) combination with IME module, (b) detailed architecture of FME mode pre-selection



S_0 : $IME(I)_{16x16}$ / $IME(I+R)_{16x16}$
S_1 : $IME(M)_{16x16}$ / $IME(M+R)_{16x16}$
S_2 : $IME(I)_{16x8}$ / $IME(I+R)_{16x8}$
S_3 : $IME(M)_{16x8}$ / $IME(M+R)_{16x8}$
S_4 : $IME(I)_{8x16}$ / $IME(I+R)_{8x16}$
S_5 : $IME(M)_{8x16}$ / $IME(M+R)_{8x16}$
S_6 : $IME(I)_{Submode}$ / $IME(I+R)_{Submode}$
S_7 : $IME(M)_{Submode}$ / $IME(M+R)_{Submode}$
O_0 : $\Phi_{ILM16x8}$ / $\Phi_{ILMR16x8}$
O_1 : $\Phi_{Inter16x16}$ / $\Phi_{InterR16x16}$
O_2 : $\Phi_{ILM16x16}$ / $\Phi_{ILMR16x16}$
O_3 : $\Phi_{Inter8x16}$ / $\Phi_{InterR8x16}$
O_4 : $\Phi_{ILM8x16}$ / $\Phi_{ILMR8x16}$
O_5 : $\Phi_{Inter16x8}$ / $\Phi_{InterR16x8}$
O_6 : $\Phi_{InterSubmode}$ / $\Phi_{InterRSubmode}$
O_7 : $\Phi_{ILMSubmode}$ / $\Phi_{ILMRSubmode}$
Type1/2 FME mode pre-selection module
S_0 S_1 S_2 S_3 S_4 S_5 S_6 S_7
Div. 3
Comp.
MUX
0
1
O_0 O_1 O_2 O_3 O_4 O_5 O_6 O_7

Fig. 4.26. Detailed hardware architecture of *Type1* and *Type2* FME mode pre-selection algorithm



S_0 : $IME(I)_{16x16}$ / $IME(M)_{16x16}$
S_1 : $IME(I+R)_{16x16}$ / $IME(M+R)_{16x16}$
S_2 : $IME(I)_{16x8}$ / $IME(M)_{16x8}$
S_3 : $IME(I+R)_{16x8}$ / $IME(M+R)_{16x8}$
S_4 : $IME(I)_{8x16}$ / $IME(M)_{8x16}$
S_5 : $IME(I+R)_{8x16}$ / $IME(M+R)_{8x16}$
S_6 : $IME(I)_{Submode}$ / $IME(M)_{Submode}$
S_7 : $IME(I+R)_{Submode}$ / $IME(M+R)_{Submode}$
O_0 : $\Phi_{Inter16x16}$ / $\Phi_{ILM16x16}$
O_1 : $\Phi_{InterR16x16}$ / $\Phi_{ILMR16x16}$
O_2 : $\Phi_{Inter8x16}$ / $\Phi_{ILM8x16}$
O_3 : $\Phi_{InterR8x16}$ / $\Phi_{ILMR8x16}$
O_4 : $\Phi_{Inter16x8}$ / $\Phi_{ILM16x8}$
O_5 : $\Phi_{InterR16x8}$ / $\Phi_{ILMR16x8}$
O_6 : $\Phi_{InterSubmode}$ / $\Phi_{ILMSubmode}$
O_7 : $\Phi_{InterRSubmode}$ / $\Phi_{ILMRSubmode}$
Div. 3
Comp.
MUX
Type3/4 FME mode pre-selection module

Fig. 4.27. Detailed hardware architecture of *Type3* and *Type4* FME mode pre-selection algorithm

From the detailed hardware architecture figures shown in Fig. 4.26 and Fig. 4.27, we observed that our design needs a division to calculate the weighting of ω_1. In the hardware design perspective, the operations of multiplication and division are usually avoided to reduce the hardware implementation costs. Therefore, we use a mathematical way to approximate the result of division. Mathematically, our division operations can be expressed as follows

$$x = \frac{s}{3} = \sum_{i=1}^{n} \frac{s}{2^{2i}} = \frac{s}{2^2} + \frac{s}{2^4} + \frac{s}{2^6} + \cdots + \frac{s}{2^{2n}}$$

(4-23)

where x and s stand for the computed result and the dividend, respectively. From this equation, it can be seen that although the division operation still existed in the equation, the division operations can be easily implemented by the right sift operations since the divisors are all the square of two. As a result, the hardware cost of division can be saved. However, we can further observe that a variable n is existed in the Eq. (4-23). This variable n is used to define the precision of the computed result. In other words, the larger the variable n is, the higher the computed precision can be achieved. However, the larger n also results in the increasing of the computations and hardware costs. Therefore, the *n* is set to four in our design since *n=10* is sufficient.

4.6. Summary

In this chapter, we propose an FME more pre-selection algorithm to skip potentially ignorable prediction modes before FME to lighten the computation complexity. The rate distortion cost relationship between different prediction modes is analyzed first to exploit the possibility of ignoring some prediction modes. Afterwards, several statistical results are conducted to confirm our observation and the results prove the high possibility that some prediction modes can be skipped before FME by using observed properties. Based on the observations and statistical results, we propose several FME mode pre-selection algorithms to pre-select modes before entering FME operation. Simulations results show that our proposed FME mode pre-selection algorithm only results in 0.036 and 0.496% BD-PSNR degradation and BD-Rate increase, respectively. On average, our proposed algorithm can achieve 65.97% mode reduction.

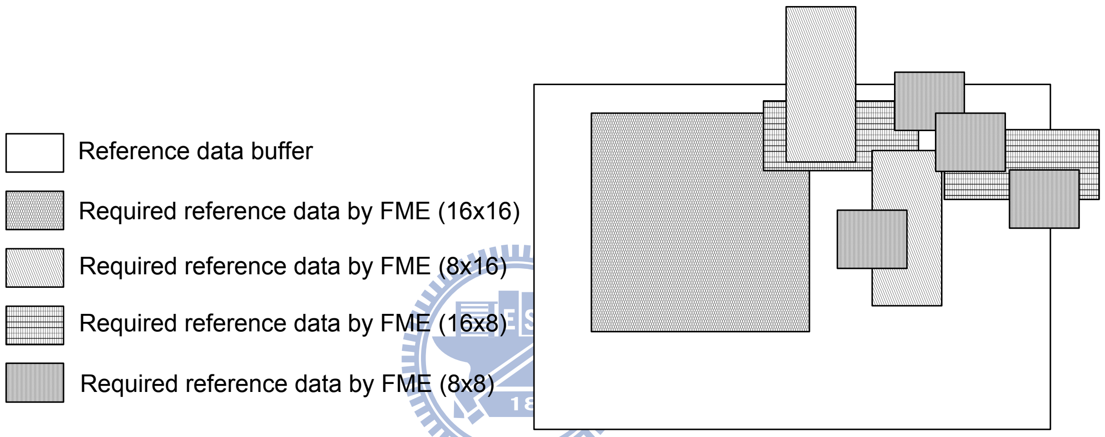
Chapter 5

Rate Distortion Oriented Search Range Adjustment for Motion Estimation Hardware Design

5.1.Introduction

To achieve better rate distortion performance, the motion estimation usually consists of IME and FME. The IME first executes search process to find out the best position and the FME is followed to refine the final results. In H.264/AVC, seven block sizes (16×16, 16×8, 8×16, 8×8, 8×4, 4×8, and 4×4) are supported for better rate distortion performance consideration. However, the adoption of variable block size motion estimation also increases difficulty of hardware realization. As a result, many literatures [64]-[67] proposed many VLSI architecture designs to implement variable block size motion estimation. In these literatures, the SAD-tree architecture was commonly adopted for realizing the SAD calculation for different block sizes. The main concept of SAD-tree is that the SADs of smaller block size 4×4 are calculated first and the SADs of larger block size such as 4×8 to 16×16 are derived by summing up the corresponding SADs of 4×4 blocks. Through the SAD-tree architecture, the variable block size motion estimation can be easily realized. However, from the specification of H.264/AVC [8], it specifies that each block size has its own motion vector predictor due to the consideration of high rate distortion performance. To solve the problem of using individual motion vector predictor for each block size, the SAD-tree architecture usually adopts the concept that all block sizes use the motion vector predictor of block size 16×16 as their motion vector predictor. Once the motion vector predictors of other block sizes have been derived at the end of the MB processing, an addition compensation procedure will be applied for other block sizes to derive the correct motion vectors of each block size.

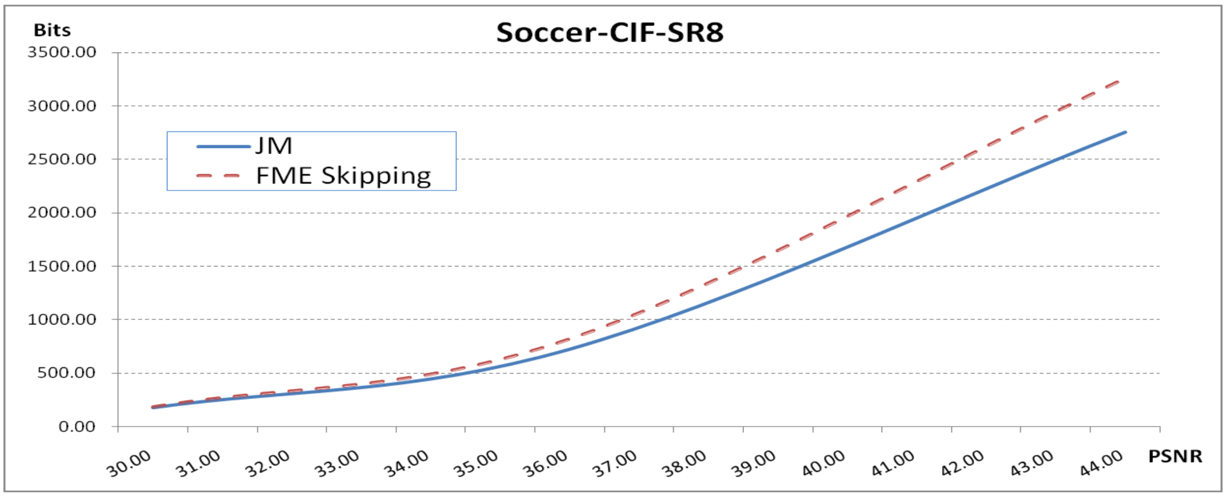
However, as mentioned before that the IME and FME are usually separated into two distinct pipeline stage [53][54]. The FME has to check every block size for deriving the final prediction results. In this coding flow, a problem is thus arisen in hardware realization. In the motion estimation prediction hardware design, a fixed size reference data buffer is intrinsically allocated for storing the pixels of search area. In the normal motion estimation flow, the search area data is initially loaded from the external memory before the process of IME began. From this point, the system has no insight about what reference data that FME required for finishing the operation since the IME has not been finished and the best position of each block size has not been derived as well. In this case, it will result in a situation that the reference data required by FME is absent in the reference data buffer as Fig. 5.1 shown due to the IME has found out the best position for each block size and the motion vector predictor of each block size has been compensated. As a result, the FME will face the reference data absence problem in the coding process.



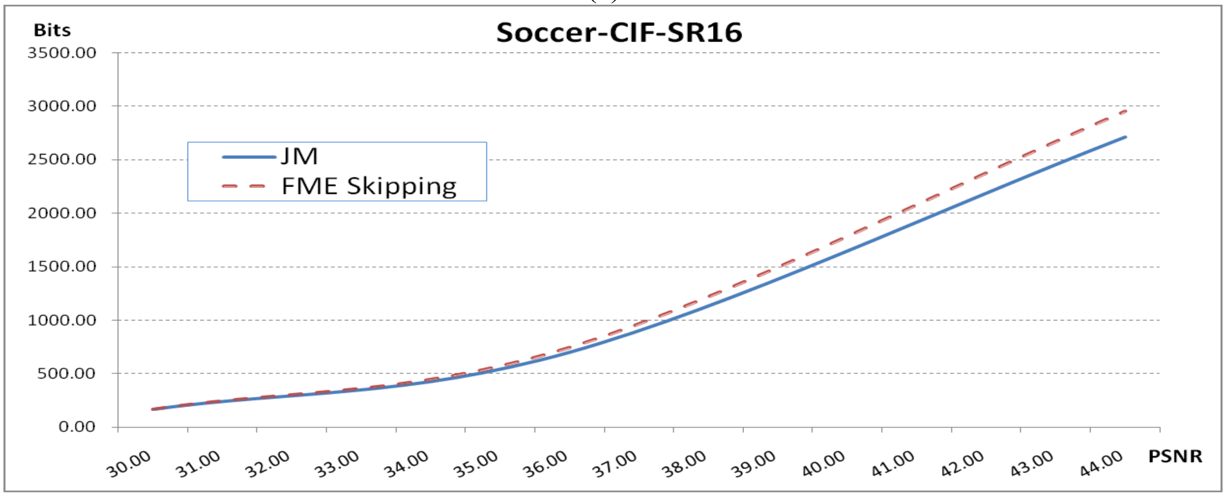
Reference data buffer
Required reference data by FME (16x16)
Required reference data by FME (8x16)
Required reference data by FME (16x8)
Required reference data by FME (8x8)

Fig. 5.1. Illustration of reference data absence from reference data buffer

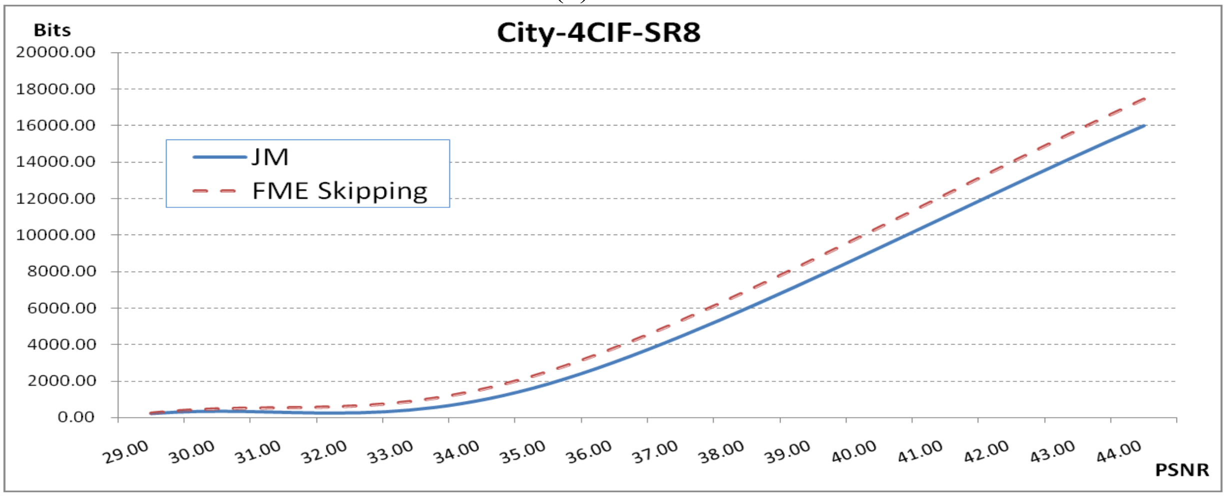
As mention early that if the reference data is absent in the search window buffer, the FME either re-fetching the absent reference data from external memory or skipping the checking process for the unavailable candidates. If the absent reference data is re-fetched from external memory, the FME module must be idled for waiting the reference data to be fetched and consequently result in the system performance degradation. In addition, the irregular memory accesses caused by re-fetching required reference data for each block size also deteriorate overall system performance. However, if FME module skips the checking for the unavailable candidates (we called this mechanism as *FME Skipping*), it will lead to the significant rate distortion performance decrease. Fig. 5.2 shows the rate distortion performance comparisons for JM and FME skipping approach for different sequences, different frame resolutions and different search range sizes. From these figures, we can observe very significant rate distortion performance degradation resulted by FME skipping mechanism when compared to JM approach.



(a)



(b)



(c)

Fig. 5.2. Rate distortion performance degradation of FME skipping for (a) Soccer with search range 8, (b) Soccer with search range 16, and (c) City with search range 8

As stated before that if the absent reference data inside the reference data buffer is re-fetched from external memory, the overall system performance will be degraded due to the irregular reference data accesses. However, if the FME checking for the unavailable pixels is skipped, significant rate distortion performance degradation will be received. Therefore, to solve the above problems, this chapter proposes a search range adjust algorithm which tries to re-decide the search range size before motion estimation such that the new decided search range area can cover the reference data required by FME as much as possible. To do so, we first analyze the relationship between motion vector predictor and non-overlapping area size. Based on the analytical results, a mathematical model is proposed to describe the relationship between motion vector predictor and non-overlapping area size. Afterwards, the modeled equations can be used to derive our proposed search range adjust algorithm and search range aspect ratio adjust algorithm.

The organization of this chapter is described as follows. In Section 5.2, we present the analysis for the motion vector predictor and non-overlapping area size. The proposed search range and search range aspect ratio adjust algorithm is introduced in Section 5.3 in detail. Section 5.4 exhibits several simulation results to demonstrate the efficiency of our proposed algorithms. Finally, the conclusion is given in Section 5.5

5.2.Analysis for MVP and Non-overlapping Area Size

In this subsection, we would like to analyze the relationship between the motion vector predictor and the non-overlapping area size. The reason why we choose the motion vector predictor as the reference to observer the effects of non-overlapping area size is explained as follows. First, the motion vector predictors are the information that we can obtain before the motion estimation process and therefore no additional computations are required. Furthermore, the motion vector predictor can efficiently reflect the motion behavior of the video content so that we can use it to roughly estimate the motion activity of the video sequence. In the following, we conduct several simulations to observe the relationship between motion vector predictor and non-overlapping area size. Before conducting our simulation, the term of non-overlapping area size should be exactly defined as follows.

$$NonOvp_{Mode\_x} = \begin{cases} \left(MVP_{Mode\_x} + Size_{Mode\_x}\right) - (pos_x + 16) & if\ \left(MVP_{Mode\_x} + Size_{Mode\_x}\right) > (pos_x + 16) \\ -pos_x - MVP_{Mode\_x} & if\ -pos_x > MVP_{Mode\_x} \\ 0 & Others \end{cases} \tag{5-1}$$

$$NonOvp_{Mode\_y} = \begin{cases} \left(MVP_{Mode\_y} + Size_{Mode\_y}\right) - \left(pos_y + 16\right) & if\ \left(MVP_{Mode\_y} + Size_{Mode\_y}\right) > \left(pos_y + 16\right) \\ -pos_y - MVP_{Mode\_y} & if\ -pos_y > MVP_{Mode\_y} \\ 0 & Others \end{cases} \tag{5-2}$$

$$where$$

$$pos_n = MVP_{16\times16\_n} + SR_n | n \in \{x, y\} \tag{5-3}$$

$$-pos_n = MVP_{16\times16\_n} - SR_n | n \in \{x, y\} \tag{5-4}$$

$$Mode \in \{16 \times 8, 8 \times 16, 8 \times 8, 8 \times 4, 4 \times 8, 4 \times 4\} \tag{5-5}$$

where MVP_{Mode} is the motion vector predictor of certain prediction mode, *Mode*. The *SR* stands for the search range. Fig. 5.3 illustrates the calculation of non-overlapping area size in *x* direction. However, for the *y* direction, the non-overlapping area size calculation is the same as the *x* direction as shown in Fig. 5.4.

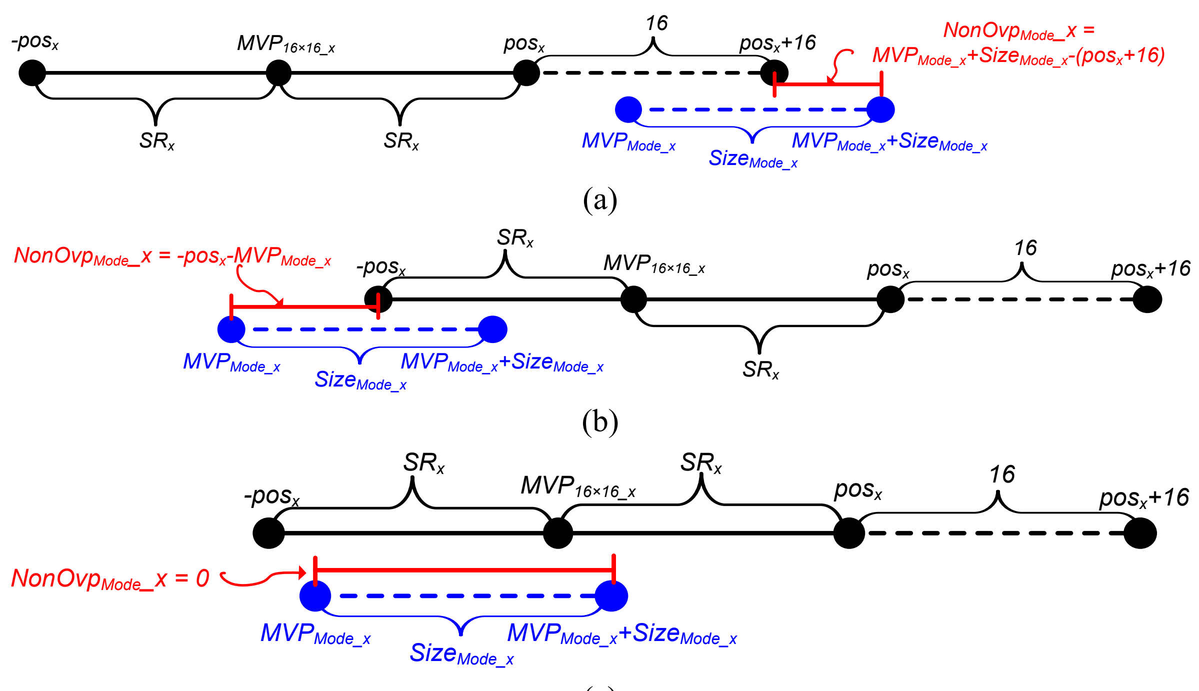


Fig. 5.3. Illustration of $NonOvp_{Mode\_x}$ calculation for (a) $MVP_{Mode\_x}+Size_{Mode\_x} > pos_x+16$, (b) $-pos_x > MVP_{Mode\_x}$, and (c) *Others*

Fig. 5.5 and Fig. 5.6 show the analytical results for the relationship between motion vector predictor and non-overlapping area size. In the figures, the x axis is the magnitude of motion vector predictor and the y axis is the average non-overlapping area size calculated by (5-1) and (5-2). The test sequences of *Soccer, Stefan, Silent, Foreman, Football, Flower*, and *Coastguard* with various motion activities are used to derive our simulation results. In addition, the video sequence resolution is in CIF format and two search range sizes are applied. From the figures, several very interesting phenomena can be observed. The first phenomenon is that the relationship curve of the motion vector predictor and non-overlapping area size is very similar to the Gaussion function curve [68]. Second, the non-overlapping area size is increased with the growth of the absolute magnitude of motion vector predictors. This phenomenon can be explained as follows. In the previous subsection, we stated that the motion vector predictor can be used to roughly estimate the motion behavior of the video content since the motion vector predictor is derived from the motion vectors of neighboring macroblocks. If the motion vector predictor is larger, it implies that the video sequence has higher motion behavior and therefore the larger motion vectors are required to represent the movement of the video objects. Unfortunately, the larger motion vector predictors also increase the possibility of non-overlapping area size growing. The last phenomenon is that the search range size affects the degree of sharpness of the curve. For example, the curve of SR8 is much sharper than the curve of SR16. This situation can be easily explained by that the smaller search range size speeds up the possibility of on-overlapping area size increasing.

In summary, the above observations inspire us that we can use a Gaussion-like equation to express the relationship between motion vector predictor and non-overlapping area size. In addition, the search range can be further considered to adjust the sharpness of the modeled curves.

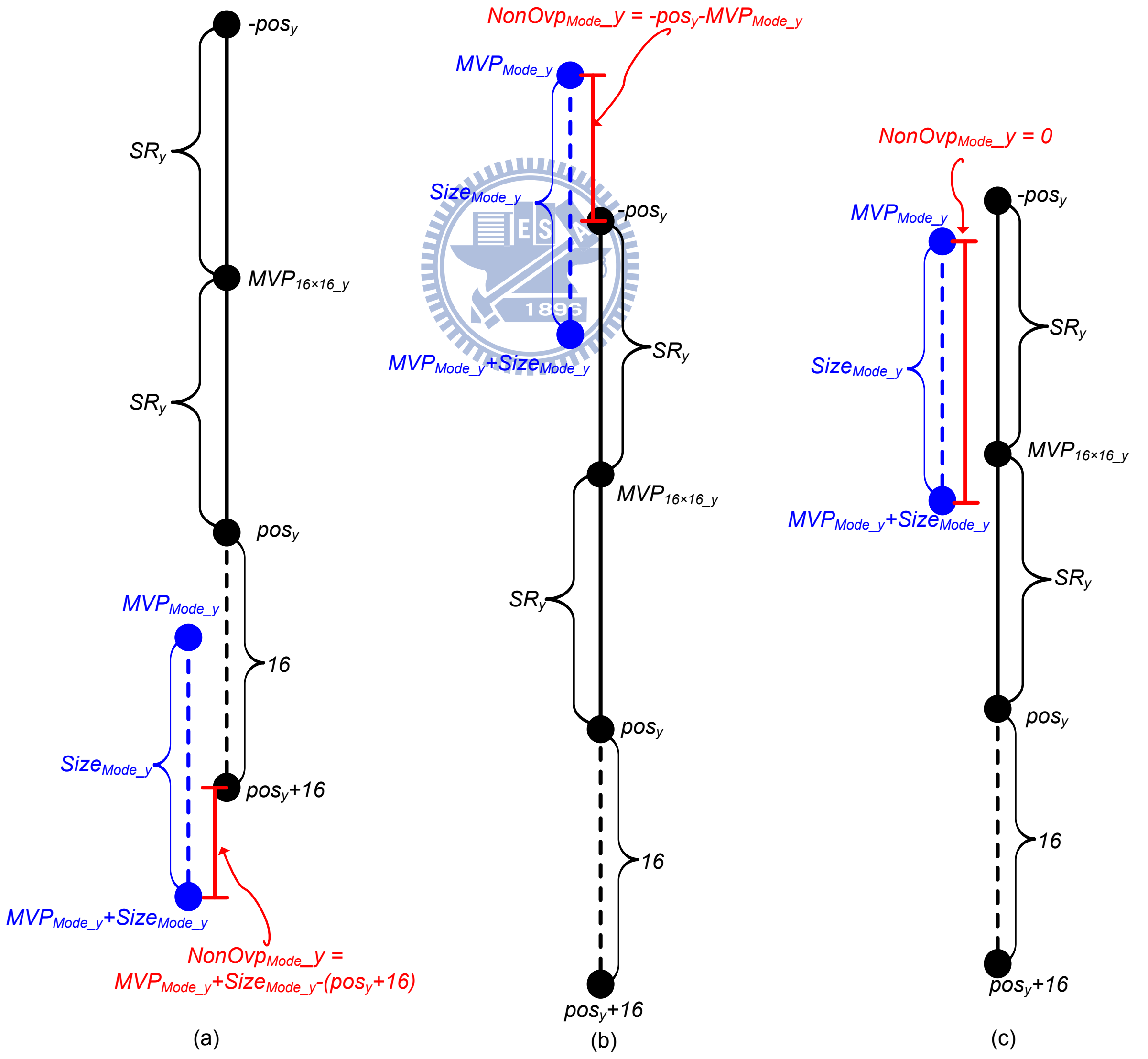
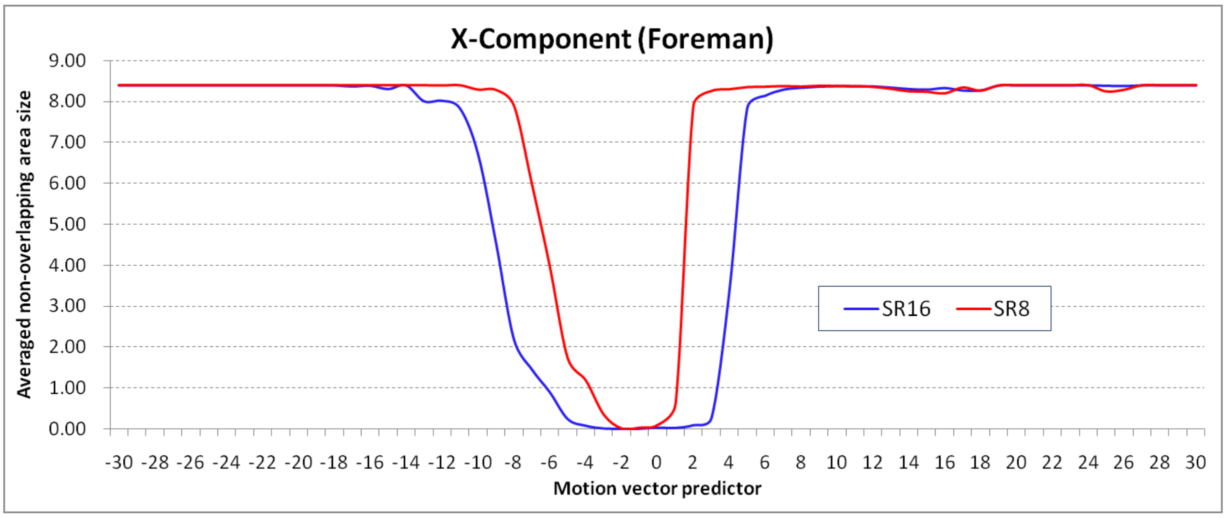
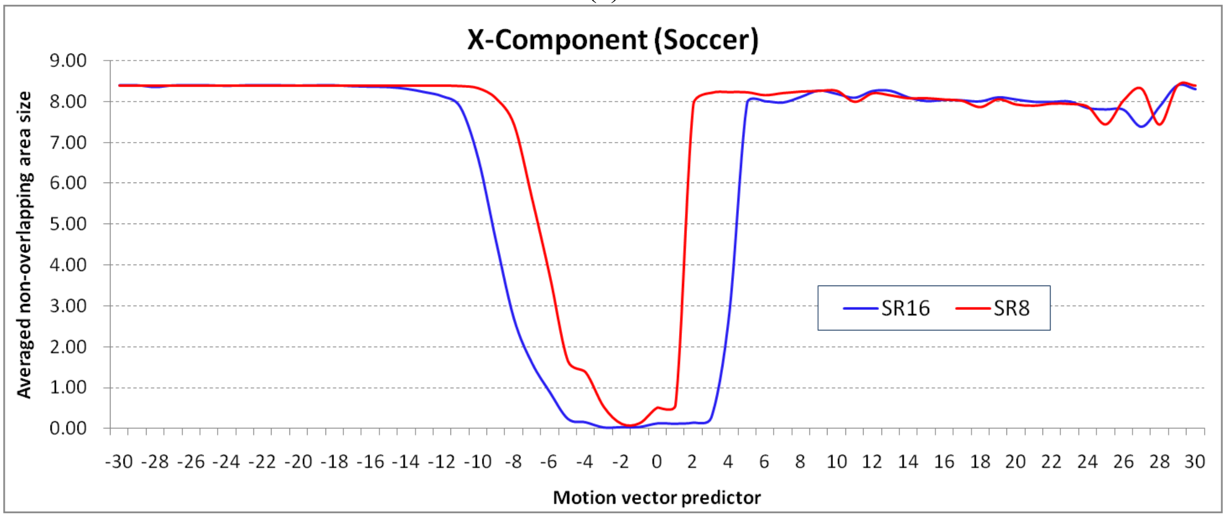


Fig. 5.4. Illustration of $NonOvp_{Mode\_y}$ calculation for (a) $MVP_{Mode\_y}+Size_{Mode\_y} > pos_y+16$, (b) $-pos_y > MVP_{Mode\_y}$, and (c) *Others*

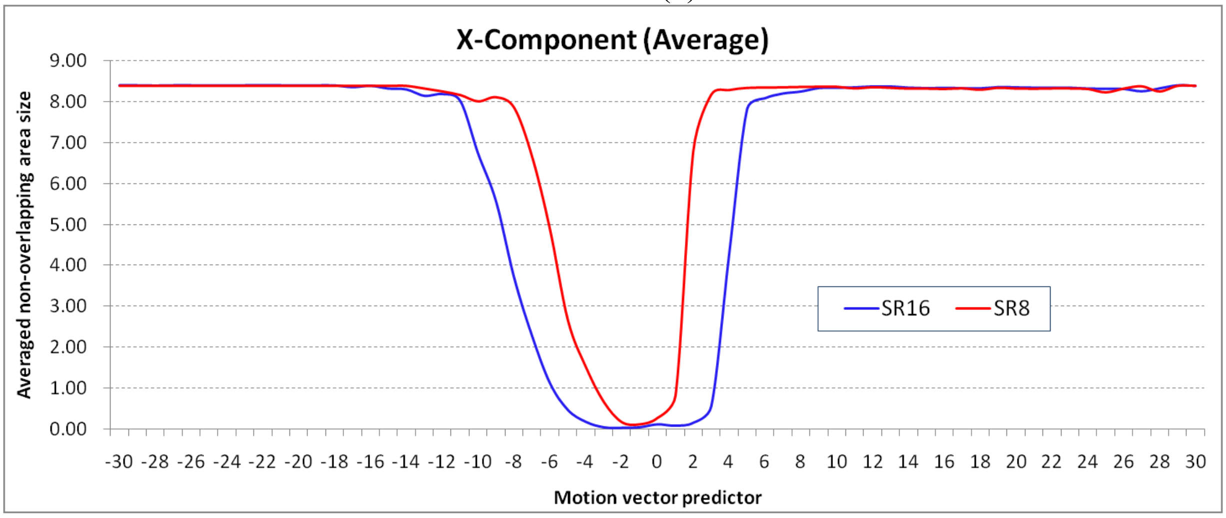


(a)

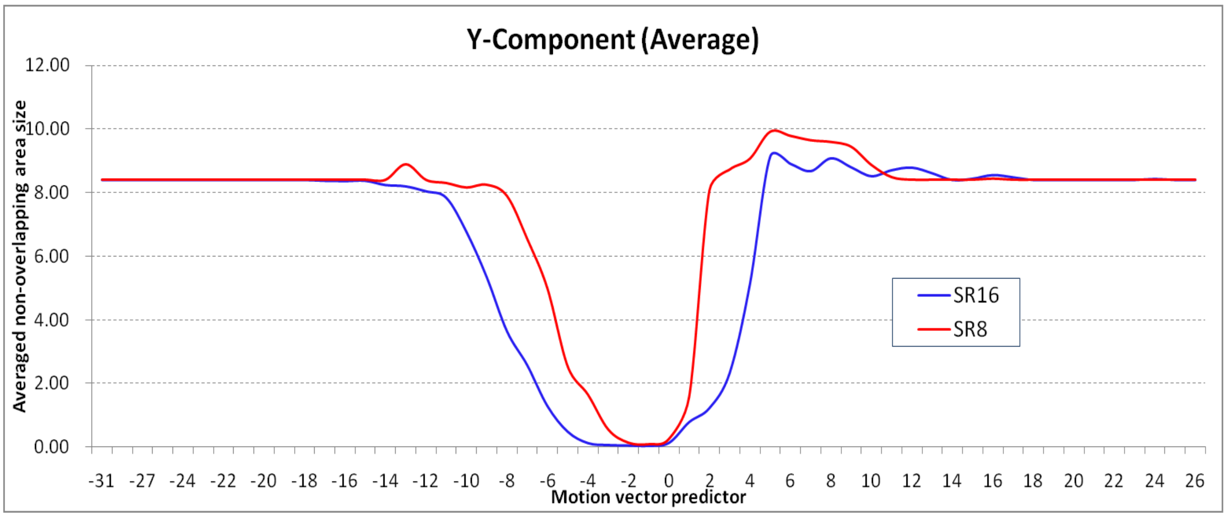


(b)

Fig. 5.5. Relationship between motion vector predictor and non-overlapped area for (a) Foreman and (b) Stefan



(a)



(b)

Fig. 5.6. Relationship between motion vector predictor and non-overlapped area in (a) X and (b) Y direction

5.3.Proposed Search Range and Aspect Ratio Adjust Algorithm

In previous section, we have analyzed the relationship between motion vector predictor and non-overlapping area size. Based on the analytical results, we proposed a search range adjust algorithm which recalculates the search range size for the upcoming motion estimation process. Instead of adjusting the search range size of motion estimation, the proposed algorithm mainly focuses on determining how many additional non-lapping area sizes are required so that the necessary reference data of FME can be covered as much as possible. In addition, we also proposed a search range aspect ratio adjust algorithm which determines the search range size aspect ratio so that the asymmetric search range size in horizontal and vertical direction can be derived. The proposed algorithms are described in detail as follows.

5.3.1. Search Range Adjust Algorithm

In this subsection, the proposed search range recalculation algorithm is described in detail. From the analysis section of 5.2, we find that the relationship between motion vector predictor and non-overlapping are size can be described by a Gaussion-like function. Therefore, we use the following equations to express the relationship between motion vector predictor and non-overlapping area size. The non-overlapping area size in the *x* direction can be derived as follows.

$$f(x) = \alpha_x exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_x}\right]^2\right\} - \alpha_x exp\left\{-\frac{1}{2}\left[\frac{(x-\mu_x)\beta_x}{\sigma_x}\right]^2\right\}$$

(5-6)

For the *y* direction, the non-overlapping area size can be obtained as follows.

$$f(y) = \alpha_y exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_y}\right]^2\right\} - \alpha_y exp\left\{-\frac{1}{2}\left[\frac{(y-\mu_y)\beta_y}{\sigma_y}\right]^2\right\}$$

(5-7)

where the parameters of α_x,α_y μ_x, μ_y, σ_x, and σ_y are derived experimentally and they have been set as $\alpha_x = \alpha_y = 18$, $\mu_x = \mu_y = 0$, and $\sigma_x = \sigma_y = 5.0$.

To sense the effect of different search range size on the calculated non-overlapping area size, the initial search range size *SR* is further taken into consideration as follows.

$$\beta_v | v \in \{x, y\} = \begin{cases} 0.4 \ldots\ldots\ldots\ldots (x-\mu) \le SR \gg 2 \\ 1.0 \ldots\ldots\ldots \ldots Otherwise \end{cases}$$

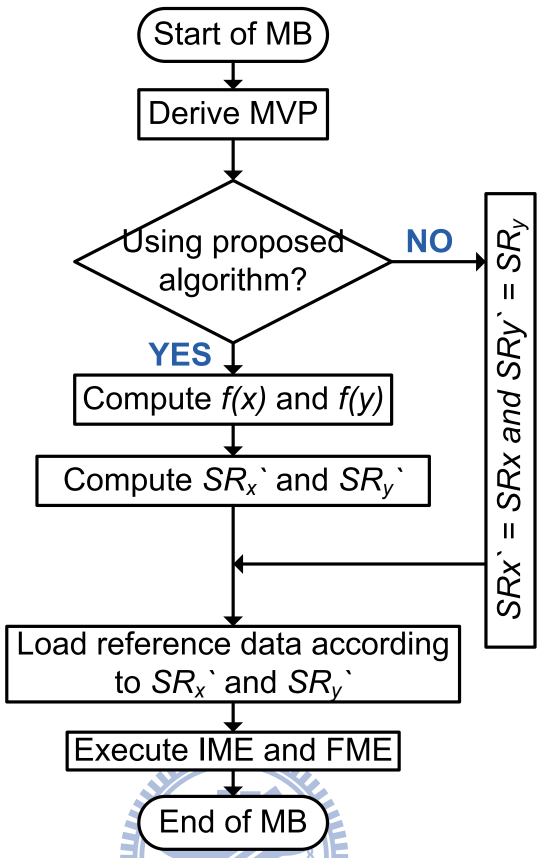
(5-8)

Finally, the adjusted search range *SR'* can be calculated as follows.

$$SR_v' = SR_v + f(v) | v \in \{x, y\}$$

(5-9)

Fig. 5.7 exhibits the flowchart of our proposed search range adjust algorithm. First, for the incoming macroblocks, the motion vector predictors are derived by the regular motion vector predictor derivation process specified in video coding standards. Afterwards, the corresponding non-overlapping area sizes *f(x)* and *f(y)* in *x* and *y* direction are computed by (5-6) and (5-7), respectively. Once the new search range size of SR_x' and SR_y' have been calculated, the reference data subject to the new adjusted search range size are loaded from external memory for the following IME and FME processes. The above operations are applied iteratively for all macroblocks.



Start of MB
Derive MVP
Using proposed algorithm?
NO
$SRx` = SRx$ and $SRy` = SR_y$
YES
Compute $f(x)$ and $f(y)$
Compute $SR_x`$ and $SR_y`$
Load reference data according to $SR_x`$ and $SR_y`$
Execute IME and FME
End of MB

Fig. 5.7. Flowchart of proposed search range adjust algorithm

5.3.2. Search Range Aspect Ratio Adjust Algorithm

Unlike most of dynamic search range adjust algorithms [69]-[73], the proposed search range aspect ratio adjust algorithm tries to adjust the search range aspect ratio both in horizontal and vertical directions under the constraint of the certain search range size so that the search range size in horizontal and vertical direction can be asymmetric and memory bandwidth variation can be consistent. Although some search range decision algorithms can achieve asymmetric search range size decision, they lack of considering the available memory bandwidth and reference data buffer size. As a result, they either result in the memory bandwidth access burden due to the large decided search range size or the memory bandwidth variation. In our proposed search range size aspect ratio adjust algorithm, we try to use the non-overlapping area size both in *x* and *y* directions to derive the ratio first. Afterwards, the ratio is adopted to compute the final aspect ratio of new adjusted search range size. The detailed derivation process of the search range aspect ratio is described as follows.

After successfully deriving the non-overlapping area size in *x* and *y* direction, we can compute the ratio between the non-overlapping area size of *x* and *y* as follows.

$$\frac{f(x)}{f(y)}=\frac{\alpha_x exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_x}\right]^2\right\}-\alpha_x exp\left\{-\frac{1}{2}\left[\frac{(x-\mu_x)\beta_x}{\sigma_x}\right]^2\right\}}{\alpha_y exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_y}\right]^2\right\}-\alpha_y exp\left\{-\frac{1}{2}\left[\frac{(y-\mu_y)\beta_y}{\sigma_y}\right]^2\right\}}=n$$

$$\Rightarrow \alpha_x exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_x}\right]^2\right\}-\alpha_x\, exp\left\{-\frac{1}{2}\left[\frac{(x-\mu_x)\beta_x}{\sigma_x}\right]^2\right\}$$

$$=n\left\{\alpha_y exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_y}\right]^2\right\}-\alpha_y exp\left\{-\frac{1}{2}\left[\frac{(y-\mu_y)\beta_y}{\sigma_y}\right]^2\right\}\right\}$$

(5-10)

where *n* is the ratio between two non-overlapping area sizes.

However, since our proposed algorithm needs to satisfy the constraint of certain search range size, the initial search range size is included here and hence results in a new equation shown as follows.

$$f(x)f(y)=\left\{\alpha_x exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_x}\right]^2\right\}-\alpha_x \exp\left\{-\frac{1}{2}\left[\frac{(x-\mu_x)\beta_x}{\sigma_x}\right]^2\right\}\right\}$$

$$\times\left\{\alpha_y exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_y}\right]^2\right\}-\alpha_y \exp\left\{-\frac{1}{2}\left[\frac{(x-\mu_y)\beta_y}{\sigma_y}\right]^2\right\}\right\}=2SR$$

$$\Rightarrow f(x)f(y)=\alpha_x\alpha_y exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_x}\right]^2\right\}exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_y}\right]^2\right\}$$

$$-\alpha_x\alpha_y exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_x}\right]^2\right\}\exp\left\{-\frac{1}{2}\left[\frac{(x-\mu_y)\beta_y}{\sigma_y}\right]^2\right\}$$

$$-\alpha_x\alpha_y exp\left\{-\frac{1}{2}\left[\frac{(x-\mu_x)\beta_x}{\sigma_x}\right]^2\right\}exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_y}\right]^2\right\}$$

$$+\alpha_x \alpha_y exp\left\{-\frac{1}{2}\left[\frac{(x-\mu_x)\beta_x}{\sigma_x}\right]^2\right\} exp\left\{-\frac{1}{2}\left[\frac{(y-\mu_y)\beta_y}{\sigma_y}\right]^2\right\} = 2SR$$

$$\Rightarrow \alpha_x exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_x}\right]^2\right\} - \alpha_x exp\left\{-\frac{1}{2}\left[\frac{(x-\mu_x)\beta_x}{\sigma_x}\right]^2\right\}$$

$$= \frac{2SR}{\alpha_y exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_y}\right]^2\right\} - \alpha_y exp\left\{-\frac{1}{2}\left[\frac{(y-\mu_y)\beta_y}{\sigma_y}\right]^2\right\}}$$

(5-11)

The above equation states that the product of new adjusted search range size *f(x)* and *f(y)* should satisfy the condition of twice search range size.

So far, by solving the operation of (5-10) – (5-11), we can obtain the following equation.

$$n\left\{\alpha_y exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_y}\right]^2\right\} - \alpha_y exp\left\{-\frac{1}{2}\left[\frac{(y-\mu_y)\beta_y}{\sigma_y}\right]^2\right\}\right\}$$

$$= \frac{2SR}{\alpha_y exp\left\{-\frac{1}{2}\left[\frac{0}{\sigma_y}\right]^2\right\} - \alpha_y exp\left\{-\frac{1}{2}\left[\frac{(y-\mu_y)\beta_y}{\sigma_y}\right]^2\right\}}$$

(5-12)

Afterwards, we can derive the new search range in *y* direction as follows with considering the different aspect ratio.

$$f(y)' = \sqrt{\frac{2SR}{n}}$$

(5-13)

By using *f(y)'* to substitute *f(y)* in (5-10), we can obtain new adjusted search range in *x* direction as follows.

$$f(x)' = n\sqrt{\frac{2SR}{n}}$$

(5-14)

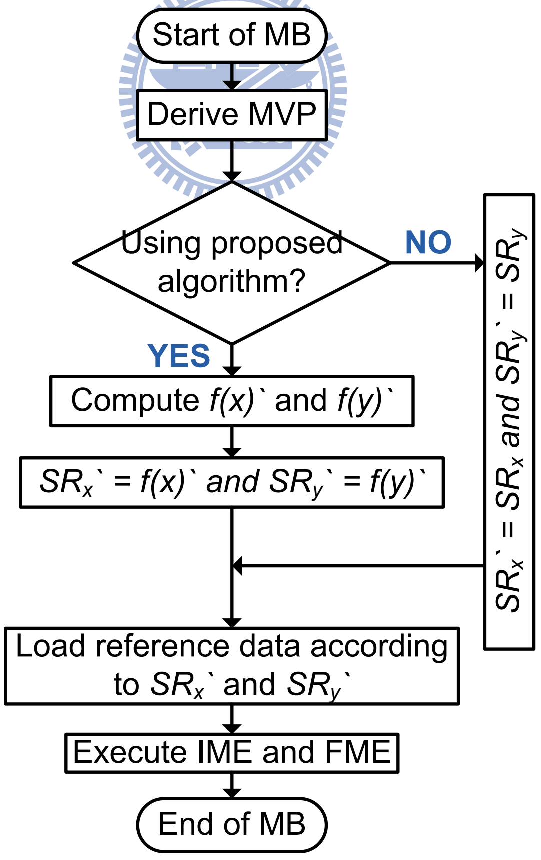
We can verify that the new adjusted search range satisfies the condition of follows.

$$f(x)'f(y)' = n\sqrt{\frac{2SR}{n}}\sqrt{\frac{2SR}{n}} = 2SR \text{ and } \frac{f(x)'}{f(y)'} = \frac{n\sqrt{\frac{2SR}{n}}}{\sqrt{\frac{2SR}{n}}} = n$$

(5-15)

Through the aid of equations (5-13) and (5-14), we can derive a new search range size in *x* and *y* direction.

Fig. 5.8 exhibits the flowchart of our proposed search range aspect ratio adjust algorithm. First, for the incoming macroblocks, the motion vector predictors are derived by the regular motion vector predictor derivation process specified in video coding standards. Afterwards, the corresponding new search range sizes *f(x)'* and *f(y)'* in *x* and *y* direction are computed by (5-13) and (5-14), respectively. Finally, the reference data subject to the new adjusted search range size are loaded from external memory for the following IME and FME processes.



Start of MB
Derive MVP
Using proposed algorithm?
NO
YES
Compute f(x)` and f(y)`
SRx` = f(x)` and SRy` = f(y)`
SRx` = SRx and SRy` = SRy
Load reference data according to SRx` and SRy`
Execute IME and FME
End of MB

Fig. 5.8. Flowchart of proposed search range aspect ratio adjust algorithm

5.4. Simulation Results

In this subsection, numerous simulations are conducted to verify the efficiency of our proposed algorithm. In addition, since we have proposed two algorithms for different applications, two simulation scenarios are created to individually show the performance of our proposals. The simulation settings are listed in Table 5-I.

Table 5-I Simulation settings

Reference software	JM12.2 [74]
Search range size	±8 and ±16
Quantization parameters	18, 28, and 38
Frame resolution	CIF and 4CIF
Frames to be encoded	300
Test sequences	City, Coastguard, Flower, Football, Foreman, Soccer, Stefan, and Mobile

5.4.1. Simulation Results of Search Range Adjust Algorithm

In this subsection, several simulation results of our proposed search range adjust algorithm are exhibited to show the rate distortion performance. Unlike most search range decision algorithms shown, we would like to show the rate distortion performance after fractional motion estimation with and without our proposed algorithm. Therefore, it can be expected that our proposed algorithm can result in the rate distortion performance improvement when compared to the results without our proposed algorithm.

Table 5-II shows the PSNR results of different approaches. In Table 5-II, the PSNR results are derived by exhaustive motion estimation without any FME skipping. For the case that the reference data required by FME is absent in reference data buffer, the FME operation will be skipped and the PSNR degradations are shown in Table 5-III. Table 5-IV exhibits the PSNR degradations of our proposed algorithm. From these tables, we can observe that our proposed algorithm and FME skipping approach respectively result in 0.01 and 0.08dB PSNR degradation on average for CIF resolution. For larger resolution sequence 4CIF, 0.007 and

0.07dB PSNR degradation will be resulted by our proposal and FME skipping approach, respectively. In other words, our proposed algorithm can reduce ten times PSNR degradation for 4CIF sequences when compared to FME skipping approach.

Table 5-II PSNR results without FME skipping

PSNR (dB)		SR8			SR16		
		18	28	38	18	28	38
CIF	Flower	44.96	35.07	25.46	44.96	35.07	25.47
	Football	45.01	37.25	30.29	44.97	37.21	30.25
	Foreman	44.45	36.76	30.24	44.45	36.75	30.23
	Soccer	44.58	36.75	30.77	44.56	36.71	30.72
	Stefan	45.01	36.71	28.18	45.00	36.71	28.18
Average		**44.802**	**36.508**	**28.988**	**44.788**	**36.49**	**28.97**
4CIF	City	44.61	36.11	29.11	44.61	36.11	29.15
	Coastguard	46.22	39.01	31.26	46.22	39.01	31.26
	Football	47.23	40.58	33.54	47.11	40.43	33.44
	Foreman	46.63	40.01	33.49	46.60	39.96	33.42
	Mobile	45.57	37.57	29.72	45.57	37.57	29.73
Average		**46.052**	**38.656**	**31.424**	**46.022**	**38.616**	**31.4**

Table 5-III PSNR degradation of FME skipping approach

ΔPSNR (dB)		SR8			SR16		
		18	28	38	18	28	38
CIF	Flower	-0.03	-0.03	0.00	0.00	0.00	0.00
	Football	-0.09	-0.11	-0.08	-0.09	-0.08	-0.06
	Foreman	-0.06	-0.08	-0.05	-0.04	-0.06	-0.04
	Soccer	-0.17	-0.13	-0.04	-0.14	-0.06	-0.03
	Stefan	-0.06	-0.20	-0.18	-0.07	-0.20	-0.19
Average		**-0.082**	**-0.11**	**-0.07**	**-0.068**	**-0.08**	**-0.064**
4CIF	City	-0.07	-0.20	-0.03	0.00	-0.02	-0.01
	Coastguard	-0.41	-0.36	-0.09	-0.05	-0.07	-0.01
	Football	-0.05	-0.07	-0.01	-0.05	-0.06	0.00
	Foreman	-0.13	-0.09	0.01	-0.10	-0.07	-0.01
	Mobile	-0.02	-0.02	-0.02	-0.01	0.00	-0.01
Average		**-0.136**	**-0.148**	**-0.028**	**-0.042**	**-0.044**	**-0.008**

0 show the bitrate values of different approaches, and the bitrate increasing comparisons of FME skipping approach and our proposed algorithm subject to no FME skipping approach are revealed in Table 5-VI and Table 5-VII, respectively. For the FME skipping approach, since it skips the FME operations when reference data is absent in the reference data buffer, the bitrate is increased significantly. On average, 9.53% bitrate will be increased by the FME skipping approach for CIF sequences. However, for the high motion sequence such as *Stefan*,

the FME skipping approach even more results in up to 42.79% bitrate increasing. In our proposed algorithm, the resulted bitrate increasing is only 0.90% on average for CIF sequences. For the high motion sequence *Stefan*, the maximum bitrate increasing is only 4.99%. However, it should be mentioned that bitrate increasing amount of 4CIF sequence is quite insignificant as CIF sequence shown due to the difference of the bitrate units. In other words, the bitrate of CIF sequences is much less than that of the bitrate of 4CIF sequences. Therefore, for the same increased bitrate amounts, the percentage of bitrate increasing of 4CIF sequences will be much less than that of the percentage of bitrate increasing of CIF sequences.

Table 5-IV PSNR degradation of proposed search range adjust algorithm

ΔPSNR (dB)		SR8			SR16		
		18	28	38	18	28	38
CIF	Flower	-0.07	-0.03	0.01	0.00	0.00	-0.01
	Football	-0.03	0.00	0.00	0.00	0.00	0.00
	Foreman	-0.02	-0.02	-0.01	-0.01	0.00	0.01
	Soccer	-0.04	-0.02	0.01	0.00	-0.01	0.00
	Stefan	-0.01	-0.04	-0.01	0.00	0.00	0.00
Average		**-0.034**	**-0.022**	**0.000**	**-0.002**	**-0.002**	**0.000**
4CIF	City	-0.02	-0.12	-0.03	0.00	-0.01	0.00
	Coastguard	-0.02	-0.04	0.00	0.00	0.00	0.00
	Football	0.01	0.01	0.02	0.01	0.01	0.02
	Foreman	-0.02	-0.02	0.01	0.00	-0.01	0.02
	Mobile	-0.01	-0.01	-0.01	-0.01	0.00	-0.01
Average		**-0.012**	**-0.036**	**-0.002**	**0.000**	**-0.002**	**0.006**

Table 5-VIII and Table 5-IX show the data access requirements of our proposed search range adjust algorithm subject to reference data reloading approach for 4CIF and CIF resolution sequences, respectively. Here, the reference data reloading approach stands for that the FME reloads the absent reference data from the external memory. From these tables, it can be seen that our proposed algorithm only occupies 18.42% and 46.99% data access requirements subject to reference data reloading FME for 4CIF and CIF sequences, respectively. In summary, our proposed search range adjust algorithm can significantly improve the rate distortion performance with slight data access requirements increasing.

Table 5-V Bitrate results without FME skipping

Bitrate (kbits)		SR8			SR16		
		18	28	38	18	28	38
CIF	Flower	6184.78	1989.36	284.80	6188.47	1988.15	285.00
	Football	4397.48	1484.51	401.93	4340.79	1444.79	383.92
	Foreman	2678.82	468.50	94.83	2672.59	464.55	91.09
	Soccer	2753.21	725.78	179.58	2716.86	701.13	166.82
	Stefan	5025.83	1511.05	310.74	4918.84	1439.49	268.76
Average		**4208.024**	**1235.84**	**254.376**	**4167.51**	**1207.622**	**239.118**
4CIF	City	16002.92	1843.98	215.62	16023.45	1838.61	214.34
	Coastguard	9052.03	2587.87	483.62	9054.16	2587.08	483.50
	Football	9482.93	3180.77	1009.43	9142.90	2964.10	938.65
	Foreman	6453.18	1138.58	283.06	6293.96	1048.34	249.63
	Mobile	12992.11	3650.71	585.99	12983.47	3651.39	585.51
Average		**10796.63**	**2480.382**	**515.544**	**10699.59**	**2417.904**	**494.326**

Table 5-VI Bitrate increasing of FME skipping approach

ΔBitrate (%)		SR8			SR16		
		18	28	38	18	28	38
CIF	Flower	2.99	3.78	3.64	0.07	0.15	0.05
	Football	7.63	7.28	4.70	4.93	4.35	2.55
	Foreman	8.74	13.30	3.26	4.60	5.97	1.10
	Soccer	18.89	14.81	6.90	9.31	7.75	2.76
	Stefan	13.12	31.82	42.79	8.40	20.82	29.33
Average		**10.274**	**14.198**	**12.258**	**5.462**	**7.808**	**7.158**
4CIF	City	9.16	38.81	21.02	0.55	2.09	1.89
	Coastguard	8.72	11.30	7.31	1.99	2.66	2.18
	Football	3.82	3.59	2.73	3.88	3.02	1.93
	Foreman	6.43	9.86	2.49	3.44	4.41	1.45
	Mobile	0.67	1.34	2.35	0.24	0.31	0.36
Average		**5.76**	**12.98**	**7.18**	**2.02**	**2.498**	**1.562**

Table 5-VII Bitrate increasing of proposed search range adjust algorithm

ΔBitrate (%)		SR8			SR16		
		18	28	38	18	28	38
CIF	Flower	1.14	1.79	2.18	0.02	0.11	-0.11
	Football	0.75	0.99	1.27	0.14	0.07	0.42
	Foreman	0.90	1.16	0.74	0.10	0.26	0.19
	Soccer	1.09	1.00	2.86	0.19	0.44	0.05
	Stefan	1.96	4.99	1.72	0.13	0.23	0.07
Average		**1.168**	**1.986**	**1.754**	**0.116**	**0.222**	**0.124**
4CIF	City	2.72	4.32	4.94	0.16	0.49	0.44
	Coastguard	0.77	0.77	1.54	0.05	0.14	0.27
	Football	0.63	0.89	1.06	0.41	0.30	0.70
	Foreman	1.22	1.98	1.64	0.18	0.16	1.02
	Mobile	0.31	0.69	1.15	0.09	0.12	0.11
Average		**1.13**	**1.73**	**2.066**	**0.178**	**0.242**	**0.508**

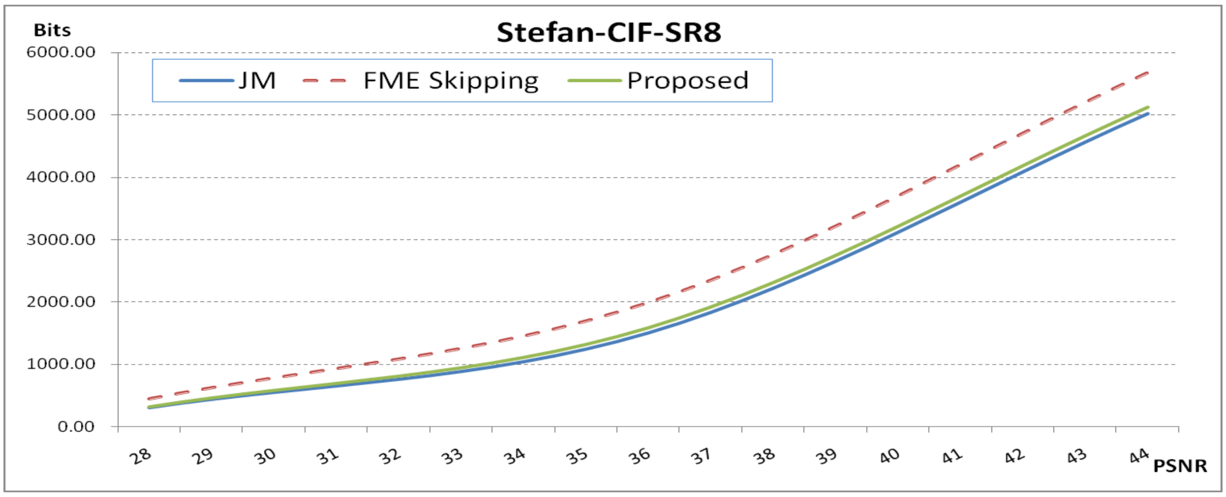
Table 5-VIII Data access requirements compared to reference data reloading for 4CIF sequences

4CIF	SR8			SR16		
	QP18	QP28	QP38	QP18	QP28	QP38
City	17.35	17.35	15.33	19.92	18.69	17.30
Football	20.77	17.14	48.53	17.04	14.98	14.98
Foreman	7.94	7.99	16.17	17.46	16.61	22.92
Mobile	21.01	18.66	15.06	23.49	20.01	15.37
Average	**16.77**	**15.29**	**23.77**	**19.48**	**17.57**	**17.64**

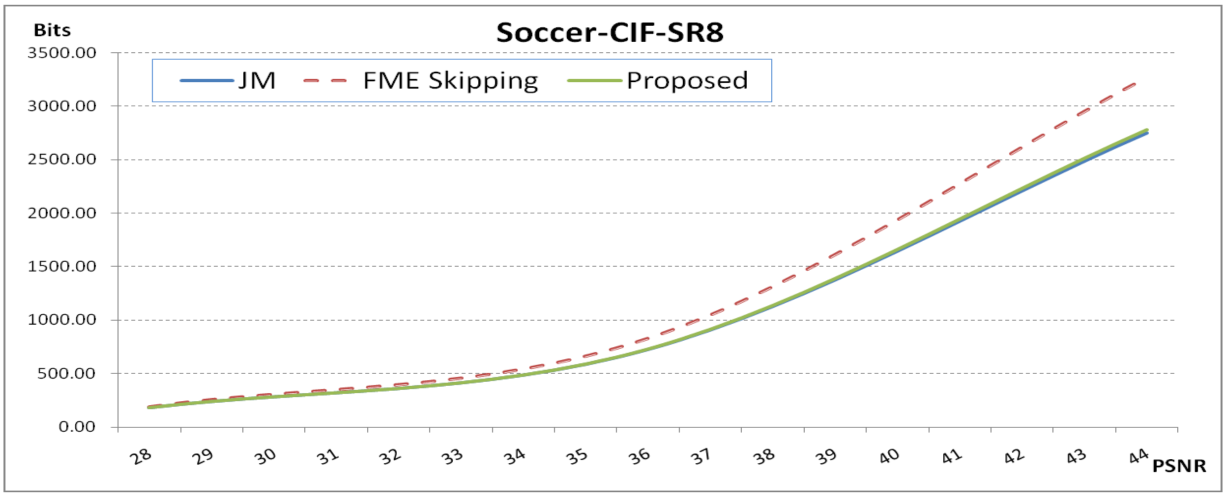
Table 5-IX Data access requirements compared to reference data reloading for CIF sequences

4CIF	SR8			SR16		
	QP18	QP28	QP38	QP18	QP28	QP38
City	33.16	33.24	33.78	40.94	38.71	44.19
Football	69.92	65.63	78.09	58.58	58.86	63.55
Foreman	53.73	46.06	42.81	41.57	38.85	51.58
Mobile	32.68	35.04	35.61	39.62	44.49	46.98
Average	**47.37**	**44.99**	**47.57**	**45.18**	**45.23**	**51.58**

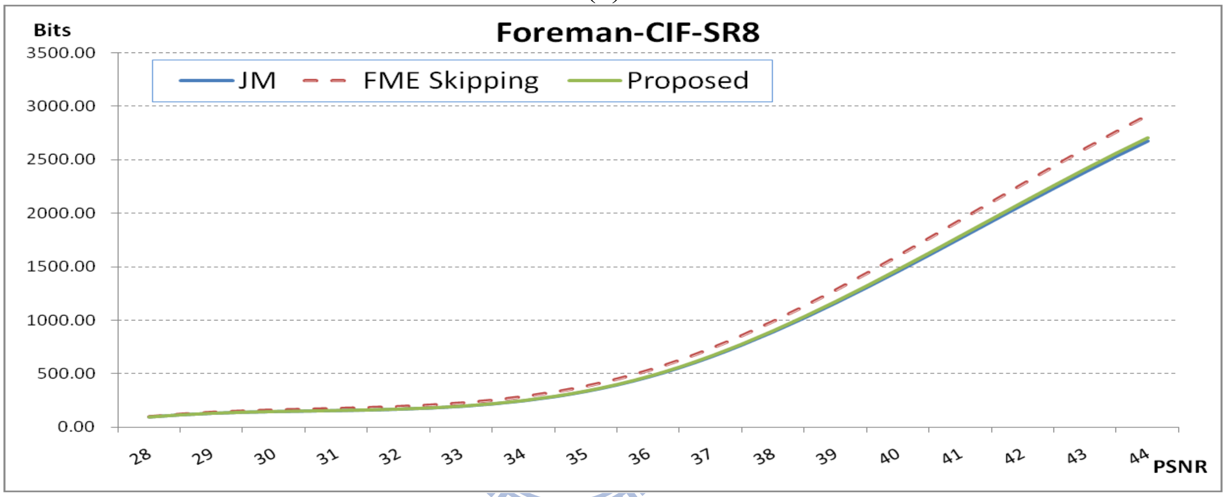
Fig. 5.9 shows the rate distortion curve comparison of different approaches for different test sequences. The term of JM stands for the approach without FME skipping. From these figures, it can be found that the FME skipping approach results in significant rate distortion performance degradation. Nevertheless, the proposed algorithm produces near the same rate distortion performance when compared to no FME skipping approach.



(a)



(b)



(c)

Fig. 5.9. Rate distortion curve comparisons of different approaches for (a)*Stefan*, (b)*Soccer*, and (c) *Foreman*

5.4.2. Simulation Results of Search Range Aspect Ratio Adjust Algorithm

In this subsection, the simulation results of our proposed search range aspect ratio adjust algorithm are exhibited to demonstrate the performance of our proposal. There is one thing that we should point out is that the main goal of our proposed search range aspect ratio adjust algorithm is tried to achieve better rate distortion performance under the constraint of the same search range area instead of minimizing the rate distortion performance degradation. In other words, we can expect that our proposed search range aspect ratio adjust algorithm can achieve ignorable PSNR degradation and bitrate decreasing under the same search range area constraint.

Table 5-X shows the PSNR results of JM. In addition, the PSNR difference between JM and our proposal is exhibited in Table 5-XI. From these tables, we can found that our proposed algorithm results in near no PSNR degradation. By calculation, only 0.006 and 0.02dB PSNR has been degraded on average by our proposed algorithm for CIF and 4CIF sequences, respectively.

Table 5-X PSNR results of JM

PSNR (dB)		SR8			SR16		
		18	**28**	**38**	**18**	**28**	**38**
CIF	**Flower**	44.96	35.07	25.46	44.96	35.07	25.47
	Football	45.01	37.25	30.29	44.97	37.21	30.25
	Foreman	44.45	36.76	30.24	44.45	36.75	30.23
	Soccer	44.58	36.75	30.77	44.56	36.71	30.72
	Stefan	45.01	36.71	28.18	45.00	36.71	28.18
Average		**44.802**	**36.508**	**28.988**	**44.788**	**36.49**	**28.97**
4CIF	**City**	44.61	36.11	29.11	44.61	36.11	29.15
	Coastguard	46.22	39.01	31.26	46.22	39.01	31.26
	Football	47.23	40.58	33.54	47.11	40.43	33.44
	Foreman	46.63	40.01	33.49	46.60	39.96	33.42
	Mobile	45.57	37.57	29.72	45.57	37.57	29.73
Average		**46.052**	**38.656**	**31.424**	**46.022**	**38.616**	**31.400**

Table 5-XI PSNR difference between proposal and JM

ΔPSNR (dB)		SR8			SR16		
		18	**28**	**38**	**18**	**28**	**38**
CIF	**Flower**	0.00	0.00	0.00	0.00	0.00	0.00
	Football	-0.03	-0.03	-0.03	-0.01	-0.01	-0.01
	Foreman	0.00	-0.01	0.01	0.00	0.00	0.01
	Soccer	-0.02	-0.03	-0.04	0.00	0.00	0.01
	Stefan	-0.01	0.00	0.01	-0.01	-0.01	0.02
Average		**-0.012**	**-0.014**	**-0.01**	**-0.004**	**-0.004**	**0.006**
4CIF	**City**	0.00	0.00	0.02	0.00	0.00	0.00
	Coastguard	0.00	-0.01	0.00	0.00	0.00	0.02
	Football	-0.10	-0.12	-0.06	-0.05	-0.07	-0.04
	Foreman	-0.03	-0.05	-0.05	-0.01	-0.02	-0.02
	Mobile	-0.01	0.01	0.01	-0.01	0.00	0.00
Average		**-0.028**	**-0.034**	**-0.016**	**-0.014**	**-0.018**	**-0.008**

For bitrate comparisons, Table 5-XII shows the bitrate results of JM. The bitrate difference comparisons is revealed in Table 5-XIII. From these tables, it can be seen that our proposed algorithm can achieve bitrate decreasing for almost all test sequences and quantization parameter settings. On average, our proposed algorithm can aim at 1.16% and 1.55% bitrate decreasing when compared to JM for CIF and 4CIF sequences. For the better case, up to 12.99% bitrate decreasing can be achieved by our proposal for the high motion sequence *Stefan*.

Table 5-XII Bitrate results of JM

Bitrate (kbits)		**SR8**			**SR16**		
		18	**28**	**38**	**18**	**28**	**38**
CIF	**Flower**	6184.78	1989.36	284.80	6188.47	1988.15	285.00
	Football	4397.48	1484.51	401.93	4340.79	1444.79	383.92
	Foreman	2678.82	468.50	94.83	2672.59	464.55	91.09
	Soccer	2753.21	725.78	179.58	2716.86	701.13	166.82
	Stefan	5025.83	1511.05	310.74	4918.84	1439.49	268.76
Average		**4208.024**	**1235.84**	**254.376**	**4167.51**	**1207.622**	**239.118**
4CIF	**City**	16002.92	1843.98	215.62	16023.45	1838.61	214.34
	Coastguard	9052.03	2587.87	483.62	9054.16	2587.08	483.50
	Football	9482.93	3180.77	1009.43	9142.90	2964.10	938.65
	Foreman	6453.18	1138.58	283.06	6293.96	1048.34	249.63
	Mobile	12992.11	3650.71	585.99	12983.47	3651.39	585.51
Average		**10796.63**	**2480.382**	**515.544**	**10699.59**	**2417.904**	**494.326**

Table 5-XIII Bitrate difference between proposal and JM

ΔBitrate (%)		**SR8**			**SR16**		
		18	**28**	**38**	**18**	**28**	**38**
CIF	**Flower**	0.06	-0.02	0.06	0.04	0.00	-0.13
	Football	-0.64	-1.33	-2.64	0.14	0.25	0.31
	Foreman	-0.01	-0.18	-2.57	0.09	0.31	0.66
	Soccer	-0.83	-2.04	-5.53	0.24	0.60	-0.83
	Stefan	-2.03	-4.42	-12.99	-0.14	-0.44	-0.74
Average		**-0.69**	**-1.598**	**-4.734**	**0.074**	**0.144**	**-0.146**
4CIF	**City**	-0.06	-0.20	-0.95	-0.01	0.11	0.02
	Coastguard	0.01	-0.03	0.15	0.03	0.00	0.10
	Football	-3.02	-5.53	-4.91	-1.11	-2.49	-3.01
	Foreman	-2.18	-6.91	-9.82	-0.17	-0.97	-5.13
	Mobile	-0.02	0.01	-0.35	0.04	-0.03	-0.07
Average		**-1.054**	**-2.532**	**-3.176**	**-0.244**	**-0.676**	**-1.618**

In summary, our proposed search range aspect ratio adjust algorithm can achieve the goal that the rate distortion performance could be better than JM under the same search range area constraint.

5.5. Integration of All Proposed Search Range Adjust Algorithms

Since our proposed search range adjust and aspect ratio adjust algorithm are distinct part, they can be easy combined together. Fig. 5.10 illustrates the flowchart of our proposed algorithms combination. First, for the incoming macroblocks, the motion vector predictors are derived by the regular motion vector predictor derivation process specified in video coding standards. Afterwards, the corresponding non-overlapping area size of $f(x)'$ and $f(y)'$ in x and y direction are computed by (5-6) and (5-7), respectively. After the calculation of non-overlapping area size has been done, our proposed aspect ratio adjust algorithm is applied to obtain the new search range SR_x' and SR_y' by (5-13) and (5-14). For the new decided search range size, our proposed search range adjust algorithm has been slightly modified as follows.

$$SR_v' = SR_v' + f(v) | v \in \{x, y\} \tag{5-16}$$

Finally, the reference data subject to the new adjusted search range size are loaded from external memory for the following IME and FME processes.

Table 5-XIV and Table 5-XV show the PSNR difference comparisons for our proposed search range adjust algorithm and combined algorithm for CIF and 4CIF sequences, respectively. From these tables, we can find that our combined algorithm can achieve 0.005dB PSNR increase on average when compared to our proposed search range adjust algorithm for CIF resolution.

Table 5-XVI and Table 5-XVII exhibit the bitrate comparisons for our combined and search range adjust algorithm in CIF and 4CIF resolution, respectively. For CIF size sequences, our combined algorithm can further reduce 1.83% bitrate requirements on average when compared to proposed search range adjust method. In addition, up to 2.25% bitrate can be reduced by our combined algorithm on average for the 4CIF size sequences.

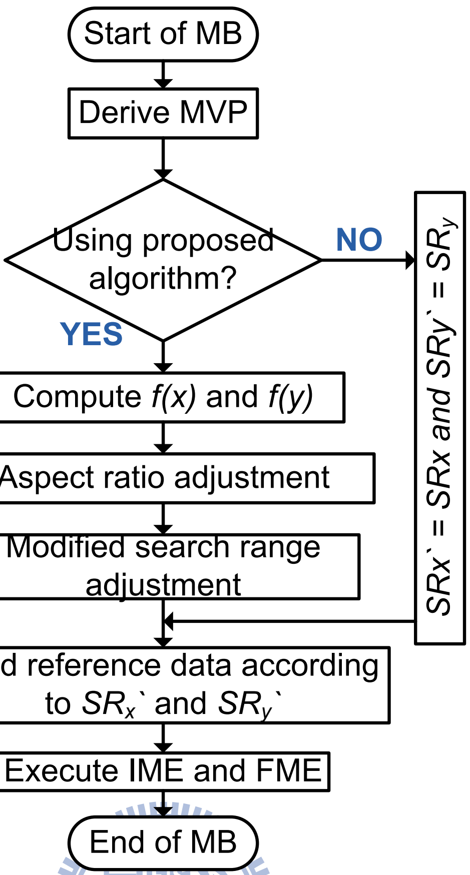

Start of MB
Derive MVP
Using proposed algorithm?
NO
YES
Compute f(x) and f(y)
Aspect ratio adjustment
Modified search range adjustment
SRx` = SRx and SRy` = SRy
Load reference data according to SRx` and SRy`
Execute IME and FME
End of MB

Fig. 5.10. Flowchart of our proposed algorithms combination

Table 5-XIV PSNR difference between proposed search range adjust and combined algorithm (CIF resolution) (Unit: dB)

PSNR Comparison (CIF)			Flower	Football	Foreman	Soccer	Stefan
SR8	QP18	SR Adjust	44.89	44.98	44.43	44.54	45.00
		Combined	44.96	44.99	44.45	44.56	45.00
		ΔPSNR	**0.07**	**0.01**	**0.02**	**0.02**	**0.00**
	QP28	SR Adjust	35.04	37.25	36.74	36.73	36.67
		Combined	35.07	37.23	36.75	36.72	36.71
		ΔPSNR	**0.03**	**-0.02**	**0.01**	**-0.01**	**0.04**
	QP38	SR Adjust	25.47	30.29	30.23	30.78	28.17
		Combined	25.47	30.26	30.23	30.74	28.18
		ΔPSNR	**0.00**	**-0.03**	**0.00**	**-0.04**	**0.01**
SR16	QP18	SR Adjust	44.96	44.97	44.44	44.56	45.00
		Combined	44.96	44.96	44.45	44.56	45.00
		ΔPSNR	**0.00**	**-0.01**	**0.01**	**0.00**	**0.00**
	QP28	Combined	35.07	37.21	36.75	36.70	36.71
		SR Adjust	35.07	37.20	36.75	36.71	36.71
		ΔPSNR	**0.00**	**-0.01**	**0.00**	**0.01**	**0.00**
	QP38	SR Adjust	25.46	30.25	30.24	30.72	28.18
		Combined	25.46	30.24	30.24	30.74	28.20
		ΔPSNR	**0.00**	**-0.01**	**0.00**	**0.02**	**0.02**

Table 5-XV PSNR difference between proposed search range adjust and combined algorithm (4CIF resolution) (Unit: dB)

PSNR Comparison (4CIF)			City	Coastguard	Football	Foreman	Mobile
SR8	QP18	SR Adjust	44.59	46.20	47.24	46.61	45.56
		Combined	44.61	46.22	47.15	46.60	45.56
		ΔPSNR	**0.02**	**0.02**	**-0.09**	**-0.01**	**0.00**
	QP28	SR Adjust	35.99	38.97	40.59	39.99	37.56
		Combined	36.11	39.01	40.46	39.96	37.57
		ΔPSNR	**0.12**	**0.04**	**-0.13**	**-0.03**	**0.01**
	QP38	SR Adjust	29.08	31.26	33.56	33.50	29.71
		Combined	29.13	31.26	33.48	33.45	29.72
		ΔPSNR	**0.05**	**0.00**	**-0.08**	**-0.05**	**0.01**
SR16	QP18	SR Adjust	44.61	46.22	47.12	46.60	45.56
		Combined	44.61	46.22	47.06	46.59	45.56
		ΔPSNR	**0.00**	**0.00**	**-0.06**	**-0.01**	**0.00**
	QP28	Combined	36.10	39.01	40.44	39.95	37.57
		SR Adjust	36.11	39.01	40.36	39.94	35.57
		ΔPSNR	**0.01**	**0.00**	**-0.08**	**-0.01**	**-2.00**
	QP38	SR Adjust	29.15	31.26	33.46	33.44	29.72
		Combined	29.15	31.27	33.41	33.41	29.72
		ΔPSNR	**0.00**	**0.01**	**-0.05**	**-0.03**	**0.00**

Table 5-XVI Bitrate difference between proposed search range adjust and combined algorithm (CIF resolution) (Unit: kbits)

Bitrate Comparison (CIF)			Flower	Football	Foreman	Soccer	Stefan
SR8	QP18	SR Adjust	6255.34	4430.29	2702.96	2783.17	5124.41
		Combined	6188.27	4374.49	2680.99	2737.41	4929.14
		ΔBitrate	**-1.07%**	**-1.26%**	**-0.81%**	**-1.64%**	**-3.81%**
	QP28	SR Adjust	2024.90	1499.21	473.92	733.07	1586.38
		Combined	1989.59	1468.83	467.92	712.45	1447.30
		ΔBitrate	**-1.74%**	**-2.03%**	**-1.27%**	**-2.81%**	**-8.77%**
	QP38	SR Adjust	291.00	407.04	95.53	184.71	316.07
		Combined	285.31	393.34	92.61	170.94	271.94
		ΔBitrate	**-1.96%**	**-3.37%**	**-3.06%**	**-7.45%**	**-13.96%**
SR16	QP18	SR Adjust	6189.66	4346.72	2675.14	2722.02	4925.08
		Combined	6190.64	4349.20	2678.69	2728.02	4914.08
		ΔBitrate	**0.02%**	**0.06%**	**0.13%**	**0.22%**	**-0.22%**
	QP28	Combined	1990.35	1445.82	465.74	704.20	1442.77
		SR Adjust	1988.84	1450.89	467.76	706.31	1434.34
		ΔBitrate	**-0.08%**	**0.35%**	**0.43%**	**0.30%**	**-0.58%**
	QP38	SR Adjust	284.70	385.53	91.26	166.91	268.94
		Combined	284.85	385.79	91.85	166.25	266.75
		ΔBitrate	**0.05%**	**0.07%**	**0.65%**	**-0.40%**	**-0.81%**

Table 5-XVII Bitrate difference between proposed search range adjust and combined algorithm (4CIF resolution) (Unit: kbits)

Bitrate Comparison (4CIF)			City	Coastguard	Football	Foreman	Mobile
SR8	**QP18**	**SR Adjust**	16437.43	9121.39	9542.55	6532.09	13032.69
		Combined	16016.86	9059.30	9254.29	6321.72	12997.72
		ΔBitrate	**-2.56%**	**-0.68%**	**-3.02%**	**-3.22%**	**-0.27%**
	QP28	**SR Adjust**	1923.67	2607.77	3208.93	1161.08	3675.89
		Combined	1845.87	2591.61	3017.22	1062.70	3654.97
		ΔBitrate	**-4.04%**	**-0.62%**	**-5.97%**	**-8.47%**	**-0.57%**
	QP38	**SR Adjust**	226.28	491.09	1020.12	287.70	592.72
		Combined	215.71	485.59	964.50	257.25	584.92
		ΔBitrate	**-4.67%**	**-1.12%**	**-5.45%**	**-10.58%**	**-1.32%**
SR16	**QP18**	**SR Adjust**	16048.87	9058.24	9180.15	6305.04	12995.58
		Combined	16035.44	9054.17	9055.14	6285.58	12995.29
		ΔBitrate	**-0.08%**	**-0.04%**	**-1.36%**	**-0.31%**	**0.00%**
	QP28	**Combined**	1847.24	2590.58	2973.10	1049.97	3655.84
		SR Adjust	1847.24	2590.28	2895.19	1039.82	3652.95
		ΔBitrate	**0.00%**	**-0.01%**	**-2.62%**	**-0.97%**	**-0.08%**
	QP38	**SR Adjust**	215.29	484.81	945.20	252.18	586.18
		Combined	214.66	484.14	914.76	238.07	585.56
		ΔBitrate	**-0.29%**	**-0.14%**	**-3.22%**	**-5.60%**	**-0.11%**

5.6. Summary

In this chapter, we consider the rate distortion performance degradation problem resulted by the FME skipping due to the reference data absence. To solve this problem, a search range adjust algorithm is proposed by using the mathematical approach. In our approach, the relationship between motion vector predictor and non-overlapping area size is observed and modeled. Via the help of modeling results, a search range adjust algorithm is thus proposed. Simulation results demonstrate that our proposed search range adjust algorithm can achieve ten time PSNR performance improvement when compared to FME skipping approach. For the bitrate comparisons, up to 90.56% bitrate can be saved when compared to FME skipping method for CIF sequences. Furthermore, we further propose a search range aspect ratio adjust algorithm which adjusts the search range aspect ratio under the same search range area constraint. By solving the mathematical equations, we can individually derive the aspect ratio for the new adjusted search range. Simulation results show that our proposed search range aspect ratio adjust algorithm results in almost the same PSNR performance when compared to JM. For the bitrate performance, our proposed algorithm can achieve 1.55% bitrate decreasing on average when compared to JM for 4CIF sequences. In summary, our proposed algorithm can efficiently solve the rate distortion performance degradation problem caused by FME skipping approach. In addition, under the same search range area constraint, our proposed search range aspect ratio adjustment algorithm results in better rate distortion performance when compared to JM.

Chapter 6

Conclusions and Future Works

6.1. Conclusions

To improve the video coding system performance drop caused by intensive data access and high computational complexity of motion estimation, this dissertation proposed several data access bandwidth and computational complexity reduction algorithms for integer and fractional motion estimation in H.264/MPEG4-AVC and its scalable extension.

For high data access bandwidth issue in integer motion estimation, we proposed a rate distortion bandwidth efficient motion estimation algorithm to reduce the amount of data accesses. In this algorithm, the relationship between rate distortion cost and data bandwidth is observed first and the observed results are used to derive a model to describe the relationship between rate distortion cost and data bandwidth. Afterwards, a low data bandwidth motion estimation algorithm is thus proposed based on the modeling. In addition, through the aid of the modeling, we further proposed a bandwidth aware motion estimation algorithm to dynamically allocate the data bandwidth for motion estimation under the certain available data bandwidth constraint. Simulation results show that our proposed low data bandwidth motion estimation algorithm can achieve 78.82% data access bandwidth saving on average.

For high data access bandwidth and computational complexity issues caused by additional adopted Inter-layer prediction modes in scalable video coding, we proposed several data and computational unit reuse algorithms to reduce the data access bandwidth and computational complexity of Inter-layer predictions by observing the relationship between successive spatial layers. In addition, an adaptive motion estimation switching algorithm is also proposed in this dissertation to dynamically select the data efficient motion estimation algorithm according to the frame resolution. Through our proposed data efficient Inter-layer prediction algorithms, at least 50.55% can be saved. In addition, the computation units for calculating the distortion costs can be shared by different prediction modes so that the hardware costs of computation units can be reduced in our proposed algorithms.

In addition to the intrinsic high computation complexity, the extra adopted Inter-layer prediction modes in scalable video coding also heavily increase the computation complexity of fractional motion estimation and thus result in the insufficient timing budget in hardware implementation. To reduce the high computational complexity of fraction motion estimation in scalable video coding, we proposed a mode pre-selection algorithm for fractional motion estimation. In this algorithm, the relationship between rate distortion costs of integer and fractional motion estimation is observed and analyzed. According to the observing and analytical results, several rules are thus proposed to filter out the potentially ignorable prediction modes before entering the fraction motion estimation process. Simulation results demonstrate that our proposed mode pre-selection algorithm for fractional motion estimation can achieve 65.97% prediction mode reduction on average.

To improve the rate distortion performance drop caused by the fractional motion estimation skipping mechanism, this dissertation proposed a search range adjust algorithm to adjust the search range by means of observing the relationship between motion vector predictor and non-overlapping area size. Through the observing, we proposed a mathematical equation to describe the relationship between motion vector predictor and non-overlapping area size. Afterwards, the mathematical equation is thus been used to calculate the new search range size so that the adjusted search range size can cover the required reference data of fractional motion estimation as much as possible. Via the proposed search range adjust algorithm, 90.56% bitrate increasing can be reduced when compared to fractional motion estimation skipping approach. In addition, a search range aspect ratio adjust algorithm is also proposed to further improve the rate distortion performance by means of solving the mathematical equation. Simulation results prove that our proposed search range aspect ratio adjust algorithm can result in better rate distortion performance when compared to JM under the same search range size constraint.

In summary, this dissertation proposed several data access bandwidth and computational complexity reduction algorithms to lighten the problems of intensive data access and high computational complexity of motion estimation. Through the algorithms proposed in this dissertation, the data access bandwidth and computation complexity both in integer and fractional motion estimation can be reduced significantly and thus achieve the overall video coding system performance improvement.

6.2. Future Works

In the future, the following directions can be considered as the research direction.

More data bandwidth reduction:

The problems of high data access bandwidth requirement and computational complexity will still be the major bottleneck in improving the overall video coding system performance. Therefore, there are some possible directions that can be studied to further improve the problems mentioned above in the future.

1. More accurate motion vector predictor estimation: In existing video coding standards, the motion vector predictor is commonly used as the search center for motion estimation. If more accurate motion vector predictor can be estimated, the search range size can be thus reduced. As a result, not only the computational complexity but also the data access bandwidth can be decreased.

2. Dynamic search range decision: If the search range can be dynamically decided according to the exact video motion behavior, the unnecessary reference data loading can be avoided so that both of the data access bandwidth and computational complexity can be reduced.

3. Reference data re-compression: If the reconstructed reference data can be further compressed before storing into the external memory, the data amount for accessing reference data can be thus reduced.

Considering technology advancing:

In addition, the advancing of manufacturing technology should be further considered in the future. For example, the development of 3D IC and wide IO technology may result in expectable data transmission rate increasing so that the intensive data access problem of motion estimation will be very different. From this point, the design considerations for video coding system have to be adjusted according to the effects of technology advancing.

Power consumption issue:

In the future, the requirement of low power electronic system will be the trend and thus lead to the electronic system designers to further consider the power consumption when designing the systems. Therefore, power consumption can be further discussed when proposing or designing the video coding systems.

Bibliography

[1] Iain E. G. Richardson, "Video Codec Design," John Wiley & Sons, Ltd, 2002.

[2] ISO/IEC 11172-2, ‘Information technology-coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s-part 2: Video’, 1993.

[3] ITU-T Recommendation H.261, Video Codec for Audiovisual Services at *p* x 64 kbit/s, March 1993.

[4] ISO/IEC 13818-2, “Information technology: generic coding of moving pictures and associated audio information: Video 1995.

[5] ISO/IEC 14996-2, ‘Information technology-coding of audio-visual objects-part 2: Visual’, 1998.

[6] ITU-T Recommendation H.263, Video Coding for Low Bit Rate Communication, version 1, Nov. 1995; version2, Jan. 1998; version 3, Nov. 2000.

[7] ISO/IEC International Standard 14496-10, Information technology – Coding of audio-visual objects – Part 10: Advanced Video Coding, third edition, Dec. 2005, corrected version, March 2006.

[8] "Advanced video coding for generic audiovisual services," ITU-T Recommendation H.264, March 2010.

[9] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.13, no.7, pp. 560-576, July 2003.

[10] K. R. Rao, Z. S. Bojkovic, and D. A. Milovanovic, "Multimedia communication systems," Upper Saddle River, NJ: Prentice-Hall, 2002.

[11] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of scalable extension of H.264/MPEG-4 AVC video coding standard," *IEEE Transactions on Circuit and Systems for Video Technology*, vol.17, no.9, pp.103-112, September 2007.

[12] C. A. Segall and G. J. Sullivan, "Spatial scalability within the H.264/AVC scalable video coding extension," *IEEE Transactions on Circuits and Systems for Video Technology*,

vol.17, no.9, pp. 1121-1135, September 2007.

[13] Z. He, Y. Liang, L. Chen, I. Ahmad, and D. We, "Power-rate-distortion analysis for wireless video communication under energy constraints," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.15, no.5, pp.645-658, May 2005.

[14] T.-C. Chen, S.-Y. Chien, Y.-W. Huang, C.-H. Tsai, C.-Y. Chen, T.-W. Chen, and L.-G. Chen, "Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC encoder," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.16, no.6, pp.673-688, June 2006.

[15] T.-Y. Chen, G.-L. Li, and T.-S. Chang, "Memory analysis for H.264/AVC scalable extension encoder," in Proceeding of *IEEE International Symposium on Circuits and Systems*, pp.361-364, May 2009.

[16] J.-C. Tuan, T.-S. Chang, and C.-W. Jen, "On the data reuse and memory bandwidth analysis for full-search block-matching VLSI architecture," *IEEE Transactions on Circuits Systems for Video Technology*, vol.12, no.1, pp. 61–72, January 2002.

[17] C.-H. Hsieh and T.-P. Lin, "VLSI architecture for block-Matching motion estimation algorithm," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.2, no.2 pp. 169-175, Jun 1992.

[18] W.-F. He, Y.-L. Bi, and Z.-G. Mao, "Efficient frame-level pipelined array architecture for full-search block-matching motion estimation," in Proceeding of *IEEE International Symposium on Circuits and Systems*, pp.2887-2890, vol.3, May 2005.

[19] Y.-W. Huang, S.-Y. Chien, B.-Y. Hsieh, and L.-G. Chen, "Global elimination algorithm and architecture design for fast block matching motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.14, no.6, pp.898-907, June 2004.

[20] M. Miyama, J. Miyakoshi, Y. Kuroda, K. Imamura, H. Hashimoto, M. Yoshimoto, "A sub-mW MPEG-4 motion estimation processor core for mobile video application," *IEEE Journal of Solid-State Circuits*, vol.39, no.9, pp.1562-1570, September 2004.

[21] S. Kappagantula and K. R. Rao, "Motion compensated interframe image prediction," *IEEE Transactions on Communications*, vol.33, no.9, pp.1011-1015, September 1985.

[22] L. M. Po and W. C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.6, no.3, pp.313-317, June 1996.

[23] S. Zhu and K.-K Ma, "A new diamond search algorithm for fast block-matching motion estimation," *IEEE Transactions on Image Processing*, vol.9, no.2, pp.287-290, February 2000.

[24] Zhu, X. Lin, and L.-P. Chau, "Hexagon-based search pattern for fast block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.12, no.5, pp.349-355, May 2002.

[25] P.-L. Tai, S.-Y. Huang, C.-T. Liu, and J.-S. Wang, "Computation-aware scheme for software-based block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.13, no.9, pp.901-912, September 2003.

[26] Z. He, Y. Liang, L. Chen, I. Ahmad, and D. Wu, "Power-rate-distortion analysis for wireless video communication under energy constraints," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.15, no.5, pp.645-658, May 2005.

[27] W.-M. Chao, C.-W. Hsu, Y.-C. Chang, and L.-G. Chen, "A novel hybrid motion estimator supporting diamond search and fast full search," in Proceeding of *IEEE International Symposium on Circuits and Systems*, pp.492-495, vol.2, May 2002.

[28] C.-Y. Chen, C.-T. Huang, Y.-H. Chen, and L.-G. Chen, "Level C+ data reuse scheme for motion estimation with corresponding coding orders," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.16, no.4, pp.553-558, April 2006.

[29] T.-C. Chen, C.-Y. Tsai, Y.-W. Huang, and L.-G. Chen, "Single reference frame multiple current macroblocks scheme for multiple reference frame motion estimation in H.264/AVC," *IEEE Transaction on Circuits and Systems for Video Technology*, vol.17,

no.2, pp.242-247, February 2007.

[30] H. Shim, K. Kang, and C.-M. Kyung, "Search area selective reuse algorithm in motion estimation," in Proceeding of *IEEE International Conference on Multimedia and Expo,* pp.1611-1614, July 2007.

[31] M.-C. Lin and L.-R. Dung, "Two-step windowing technique for wide range motion estimation," in Proceeding of *IEEE Asia Pacific Conference on Circuits and Systems*, pp.1478-1481, November 2008.

[32] S.-H Wang, W.-L. Tao, C.-N. Wang, W.-H. Peng, and T. Chiang, "Platform-based design of all binary motion estimation with bus interleaved architecture," in Proceeding of *IEEE International Symposium on VLSI Design, Automation and Testing*, pp.241-244, April 2005.

[33] S.-H. Wang, S.-H. Tai, and T. Chiang, "A low-power and bandwidth-efficient motion estimation IP core design using binary search," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.19, no.5, pp.760-765, May 2009.

[34] T.-C. Wang, Y.-W. Huang, H.-C. Fang, and L.-G. Chen, "Performance analysis of hardware oriented algorithm modification in H.264," in Proceeding of *IEEE International Conference on Multimedia and Exp*, pp. 601-604, vol. 3, July 2003.

[35] M. Gallant, G. Gote, and F. Kossentini, "An efficient computation-constrained block-based motion estimation algorithm for low bit rate vide coding," *IEEE Transactions on Image Processing*, vol.8, no.12, pp.1816-1823, December 1999.

[36] JM11 http://iphome.hhi.de/suehring/tml/download/

[37] J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Transactions on Communications*, vol.COM-29, no.12, pp.1799-1808, December 1981.

[38] R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," *IEEE Transactions on Communications*, vol.COM-33, no.8, pp.888-896, August 1985.

[39] Y.-H. Chen and T.-D. Chuang and Y.-J. Chen, and L.-G. Chen, "Bandwidth-efficient encoder framework for H.264/AVC scalable extension," in Proceeding of *IEEE International Symposium on Multimedia*, pp.401-406, December 2007.

[40] T.-C. Chen and Y.-H. Chen and S.-F. Tsai and S.-Y. Chien, and L.-G. Chen, "Fast algorithm and architecture design of low-power integer motion estimation for H.264/AVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 5, pp. 568-577, May 2007.

[41] L.-K. Liu, and B. Feig, "A block-based gradient descent search algorithm for block motion estimation in video coding," *IEEE Transactions on Circuits and Systems for Video Technol*ogy, vol. 6, no. 4, pp. 419-422, August 1996.

[42] http://www.micron.com/products/dram/ddr/partlist.aspx

[43] G.-S. Yu and T.-S. Chang, "Optimal data mapping for motion compensation in H.264 video decoding," in Proceeding of *IEEE Workshop on Signal Processing Systems*, pp.505-508, October 2007.

[44] H. Li, Z. Li, and C. Wen, "Fast mode decision algorithm for inter-frame coding in fully scalable video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 7, pp.889-895, July 2006.

[45] B. Lee, M. Kim, S. Hahm, C. Park, and K. Park, "A fast selection scheme in inter-layer prediction of H.264 scalable video coding," in Proceeding of *IEEE Symposium on Broadband Multimedia Systems and Broadcasting*, April 2008.

[46] H. Li, Z. G. Li, C. Wen, and L. P. Chau, "Fast mode decision for spatial scalable video coding," in Proceeding of *IEEE International Symposium on Circuits and Systems*, pp.3005-3008, May 2006.

[47] P.-C. Wang, G.-L. Li, S.-F. Huang, M.-J. Chen, and S.-C. Lin, "Efficient mode decision algorithm based on spatial, temporal and inter-layer correlation coefficients for scalable video coding," *Electronics and Telecommunication Research Institute Journal*. vol.32,

no.4, pp.577-587, August 2010.

[48] H.-S. Huang, G.-L. Li, and T.-S. Chang, "Low memory bandwidth prediction method for H.264/AVC scalable video extension," in Proceeding of *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pp.294-298, October 2009.

[49] Y. -H. Chen, T. -D. Chuang, Y. -J. Chen, and L. -G. Chen, "Bandwidth-efficient encoder framework for H.264/AVC scalable extension," in Proceeding of *IEEE International Symposium on Multimedia Workshops*, pp.401-406, December 2007.

[50] C.-C. Lin, Y.-K. Lin, and T.-S. Chang, "PMRME: A parallel multi-resolution motion estimation algorithm and architecture for HDTV sized H.264 video coding," in Proceeding of *IEEE International Conference on Acoustics, Speech and Signal Proceeding,* vol.2, pp.II-385-II-388, April 2007.

[51] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," in Proceeding of *IEEE National Telecommunication Conference*, pp. G5.3.1-G5.3.5, December 1981.

[52] J. Reichel, H. Schwarz, and M. Wien, ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6: JVT-V202 'JSVM 9 software' 22nd Meeting: Marrakech, Morocco, January 2007.

[53] T.-C. Chen, Y.-W. Huang, and L.-G. Chen, "Fully utilized and reusable architecture for fractional motion estimation of H.264/AVC," in Proceeding of *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp.V-9-V-12, May 2004.

[54] T.-C. Wang, Y.-W. Huang, H.-C. Fang, L.-G. Chen, "Performance analysis of hardware oriented algorithm modifications in H.264," in Proceeding of *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp.493-496, vol.2, April 2003.

[55] K. Minoo and T. Q. Nguyen, "Reverse, sub-pixel block matching: applications within H.264 and analysis of limitations," in Proceeding of *International Conference on Image Processing*, pp.3161-3164, October 2006.

[56] Y.-K. Lin, D.-W. Li, C.-C. Lin, T.-Y. Kuo, S.-J. Wu, W.-C. Tai, W.-C. Chang, and T.-S. Chang, "A 242mW, 10mm^2 1080p H.264/AVC high profile encoder chip," in Proceeding of *International Solid-State Circuits Conference*, pp. 314-315, February 2008.

[57] H. Nisar and T.-S. Choi, "Fast and efficient fractional pixel motion estimation for H.264/AVC video coding," in Proceeding of *IEEE International Conference on Image Processing*, pp.1561-1564, October 2008.

[58] C.-Y. Kao, C.-L. Wu, and Y.-L. Lin, "A high-performance three-engine architecture for H.264/AVC fractional motion estimation," *IEEE Transactions on Very Large Scale Integration System*, vol.18, no.4, pp.662-666, April 2010

[59] Y.-J. Wang, C.-C. Cheng, and T.-S. Chang, "A fast algorithm and its VLSI architecture for fractional motion estimation for H.264/MPEG-4/AVC video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.17, no.5, pp.578–583, May 2007.

[60] G. Kim, J. Kim, and C.-M. Kyung, "A low cost single-pass fractional motion estimation architecture using bit clipping for H.264 video codec," in Proceeding of *IEEE International Conference on Multimedia and Expo*., pp.661-662, July 2010.

[61] C.-C. Yang, K.-J. Tan, Y.-C. Yang, and J.-I. Guo, "Low complexity fractional motion estimation with adaptive mode selection for H.264/AVC," in Proceeding of *IEEE International Conference on Multimedia and Expo*., pp.673-678, July 2010.

[62] C.-C. Lin, Y.-K. Lin, and T.-S. Chang, "A fast algorithm and its architecture for motion estimation in MPEG-4 AVC/H.264," in Proceeding of *Asia Pacific Conference on Circuits and Systems*, pp.1250-1253, December 2006.

[63] ITU-T and I. JTC1. (2008) JSVM Software version JSVM 9.17.

[64] S. Y. Yap and J. V. McCanny, "A VLSI architecture for variable block size video motion estimation," *IEEE Transactions on Circuits and Systems,-II: Express Brief*, vol.15, no.7, pp.384-389, July 2004.

[65] C.-M. Ou, C.-F. Le, and W.-J. Hwang, "An efficient VLSI architecture for H.264 variable block size motion estimation," *IEEE Transactions on Consumer Electronics*, vol.15, no.4, pp.1291-1299, November 2005.

[66] Y. Song, Z. Liu, S. Goto, and T. Ikenaga, "Scalable VLSI architecture for variable block size integer motion estimation in H.264/AVC," *IEICE Transactions on Fundamentals*, vol.E89-A, no.4, pp.979-988, April 2006.

[67] C.-Y. Chen, S.-Y. Chien, Y.-W. Huang, T.-C. Chen, T.-C. Wang, and L.-G. Chen, "Analysis and architecture design of variable block-size motion estimation for H.264/AVC," *IEEE Transactions on Circuits and Systems-I: Regular Papers*, vol.53, no.2, pp.578-593, February 2006.

[68] M. Abramowitz and I.A. Stegun, "Handbook of mathematical functions," Applied Mathematics Series, 55, Notional Bureau of Standards.

[69] Z. Chen, Y. Song, T. Ikenaga, and S. Goto, "A dynamic search range algorithm for variable block size motion estimation in H.264/AVC," in Proceeding of *IEEE International Conference on Information, Communications & Signal Processing*, pp.1-4, December 2007.

[70] M. G. Sarwer and Q. M. J. Wu, "An efficient search range decision algorithm for motion estimation in H.264/AVC," *International Journal of Circuits, Systems and Signal Processing*, pp.173-180, 2009, vol.3, no.4.

[71] S.-W. Lee, S.-M. Park, and H.-S. Kang, "Fast motion estimation with adaptive search range adjustment," *Optical Engineering Letter*, vol. 46, no. 4, April 2007.

[72] C.-C. Yang, G.-L. Li, M.-C. Chi, M.-J. Chen, and C.-H. Yeh, "Prediction error prioritizing strategy for fast normalized partial distortion motion estimation algorithm," *IEEE Transactions on Circuits and Systems for Video Technology*. vol.20, no.7, pp.1150-1155, August 2010.

[73] G.-L. Li and M.-J. Chen, "Adaptive search range decision and early termination for

multiple reference frame motion estimation for H.264," *IEICE Transactions on Communications***,** vol.E89-B, no.1, pp.250-253 January 2006.

[74] JM12.2 http://iphome.hhi.de/suehring/tml/download/