國 立 交 通 大 學

電 機 與 控 制 工 程 研 究 所

碩 士 論 文

可變區塊大小移動估測之多模終止技術

On Multi-Mode Termination for Complexity Reduction of VBSME

指導教授：董蘭榮 博士

研究生：宋岳璋

中華民國九十五年七月

On Multi-Mode Termination for Complexity Reduction of VBSME

Advisor: Dr. Lan-Rong Dung

Graduate Student: Yue-Zhang Song

July 2006

Graduate Institute of Electrical and Control Engineering

National Chiao Tung University

Hsinchu, Taiwan, ROC

On Multi-Mode Termination for Complexity Reduction of VBSME

Graduate Student: Yue-Zhang Song Advisor: Dr. Lan-Rong Dung

Department of Electrical and Control Engineering

National Chiao Tung University

Abstract

H.264 is a video compression standard being jointly developed by the ITU-T video coding experts group and the ISO/IEC motion picture expert group. The main goal of this standardization effort is enhanced compression performance. The H.264 video coding standard achieves considerably higher coding efficiency than previous standards by using advanced intra-prediction and variable block size motion compensation with multiple reference frames at quarter-pel accuracy. One of the most challenging problems in implementing this encoder is variable block size motion estimation. Motion Estimation in h.264 takes 60%~80% process time. For each block size mode, motion estimation must be done for each partition block independently. For a macroblock, there are 41 partition search processes needed to be done. All modes need to do search once then select one best RDcost result for encoding. In this thesis, we propose an adaptive algorithm that adjusts search distance cap for motion estimation of each mode to skip unnecessary search point outside the search distance cap. This algorithm is implemented in JM10.1 with FME. Experimental results have shown that the proposed algorithm can adaptively predict the search distance cap to various video sequences with almost the same rate-distortion performance and save 20% - 50% motion estimation time compared to original fast motion estimation algorithm.

可變區塊大小移動估測技術之多模終止技術

學生：宋岳璋 指導教授：董蘭榮博士

國立交通大學

電機與控制工程學系研究所

摘要

H.264 是由 ITU-T 影像編碼組織與 ISO/IEC 影像圖片專家組織所共同研發出來的最新影像壓縮標準。這項標準的主要努力目的在加強影像壓縮的性能表現。H.264 影像壓縮標準藉著使用內插預測、多參考畫面的可變大小移動估測、其運動向量可以達 1/4 像素點精度，達成了超越以前任一個影像標準的高度編碼效率。對實作編碼器來說其中最居有挑戰性一項問題便是可變大小的移動估測。在 H.264 裡移動估測的處理時間佔據了壓縮時間的 60%-80%。 對每個宏塊的每種尺寸大小的分割，其移動估測是獨立完成，互不相關的。每個宏塊所有的尺寸大小總有用 41 塊分割，而這些分割都要做過移動估測，然後再選出一個其位元率與失真率總合最小的尺寸模式給編碼器做為最後的編碼使用。在這篇論文裡，我們提出了一個適應性調整搜尋距離上限的演算法用在移動估測上，當搜尋點超過這個上限時使其能略過不必要的搜尋點。這個演算法被實現在 h.264 的參考軟體模型版本 10.1 上，以快速移動估測演算法為實現的標的。實驗結果顯示我們所提出的演算法能夠對於不同的影像串列適應性的預測出其搜尋範圍，在與原本快速演算法其位元失真率曲線相當表現的情況下，節省 20%至 50%的移動估測時間。.

誌謝

能夠完成這份畢業論文，最感謝　董蘭榮老師的教誨與指導，讓我在這兩年的研究生生涯裡能夠從無到有，學習到了豐富的學問與知識。家裡的雙親提供我精神上與物質上的支持，讓我沒有後顧之憂，能在交大這裡完成這段出社會前的學業。實驗室裡的博班學長，同屆的四位同學與下屆的學弟，感謝你們！在這樣的時間裡，這樣的環境下，互相打氣加油。感謝大學同學們的鼓勵，讓我在低落的時刻，有溫暖的支持。

岳璋

Contents

List of Figures

List of Tables

Chapter 1 Introduction

H.264 is a video compression standard being jointly developed by the ITU-T video coding experts group and the ISO/IEC motion picture expert group. The main goal of this standardization effort is enhanced compression performance. The H.264 video coding standard achieves considerably higher coding efficiency than previous standards by using advanced intra-prediction and variable block size motion compensation with multiple reference frames at quarter-pel accuracy. One of the most challenging problems in implementing this encoder is variable block size motion estimation. Motion Estimation in h.264 takes 60%~80% process time. For each block size mode, motion estimation must be done for each partition block independently. For a marcoblock, there are 41 partition search processes needed to be done. All modes need to do search once then select one best RDcost result for encoding. This is the reason why the motion estimation in h.264 takes more time than previous video coding standard.

Doing search for all modes then selecting one best mode is optimal for final encoding. But it does cost a lot of computation. In order to reduce computational load in variable block size motion estimation, there are numbers of fast mode decision/prediction algorithm proposed for h.264/avc. In [1], they classify macroblock into homogeneous and stationary by whole frame edge map. By experimental result, setting edge vector amplitude threshold is used to classify macroblock type and to select mode. There is additional calculation to generate the edge direction histogram before doing motion search. In [2], early termination algorithm that detecting zero motion block by threshold value is proposed. Selecting the threshold of a set of clips by experimental result is adopted. This method is using threshold as an early termination criterion. Statistics result to determine the threshold is needed. If a MB is homogenous, it may tend to be coded as a larger block type. By using MAD (mean absolute difference) and MAFD (mean absolute frame difference) in [3], current MB is classified as homogenous or not. And the larger block size .mode motion search is chosen if the current MB is homogenous. These algorithms proposed in literature need to predefine threshold value setting by experimental result or perform calculation to gather information for mode prediction before motion estimation. They turn off certain block mode search options or early terminate search to reduce the computational load with the information gathering computation cost. In this thesis, we choose alternative way to reduce computational load by skipping search step but without overhead calculation for information.

In the motion estimation of h.264/avc, best search point for each partition is determined by choosing the candidate ones with the minimal value of the sum of SADcost and MVcost in the same search window. After motion estimation for each partition is done, one best block mode is selected in all modes by choosing search result with the minimal value of RDcost among all modes. The result of each mode except the lowest RDcost mode is finally discarded by mode decision making process.

It means that the rest best candidate search points in other modes are discarded by mode decision procedure. Because the RDO procedure lists the RDcost of all modes and chooses the lowest RDcost mode, every mode has chance to be discarded in the final RDO mode selection procedure. Doing the search becomes meaningless because the search result is discarded by final RDO stage: mode selection. It comes a thought:

Is it necessary to search all candidates even the result is finally abandoned?

Our observation find that the chance of the mode discarded is larger when the search point is further from start point. The result of the best mode is center-biased. When each mode doing search, compared to its candidate only, the minimal SAD and MV cost search point candidate is selected. But when doing mode selection procedure, mode result with the further distance search point candidate is less chance to be selected as the final encoding mode. It means that the longer MV length result one mode has, the more possibility the result is discarded in mode selection. This tendency makes us consider that the search may be skipped by distance check.

We also find that MV result has the similar distance distribution characteristic among all modes. The meaning of the similar distance distribution tells that the correlation about distance from start point for all block modes is high. This correlation among modes may reveal some information about search distance.

In order to save search steps in search process that may be discarded in mode selection procedure, we propose an adaptive algorithm that adjusts search step range for motion estimation of each mode to reduce the computational load.

The main ideas are:

1. Best mode result is center-biased; we try to find the range that mode result is discarded outside this range.
2. By using the neighbor motion vectors result, we predict the distance cap that limits the search steps.
3. When certain mode search is end but final encoding mode is not determined, there is some information from the search result of the search-end mode. From the observation, the similar motion vector distance distribution of each block help us to adjust this range.

In this thesis, a prediction–correction search range scheme for VBSME is proposed as follow. First, we predict the search range by neighbor MVs result by the concept of mv predictor and the correlation of neighbor marcoblock. Second, without limiting 16x16 mode search range, we do 16x16 mode search and get its mv length result. Then, we compare the predicted search range and mv length of 16x16 mode, take larger one as limited search range of other modes. When doing other mode search, if the search point is outside the limited search range, this search point is not searched. By using this algorithm, we can achieve 20%-50% me time saving compared to FME result with PSNR loss about 0.01-0.02 dB.

Chapter 2 Background

In this chap the technical overview of intermode decision and VBSME of H.264/avc will be introduced. The literatures about fast mode prediction / decision are reviewed as well.

2.1 Intermode Decision in H.264/AVC

As specified in H.264/AVC, there are in total 7 different block sizes (16x16, 16x8, 8x16, 8x8, 8x4, 4x8, and 4x4) that can be used in interframe motion estimation/compensation. These different block sizes actually form a two-level hierarchy inside a MB. The first level comprises block size of 16x16, 16x8, or 8x16. In the second level, the MB is specified as P8x8 type, of which each 8x8 block can be one of the subtypes such as 8x8, 8x4, 4x8 or 4x4. The relationship between these different block sizes is shown in Fig. 1. The procedure can be defined as follow:

$$J(s,c,MODE \mid \lambda_{MODE}) = SSD(s,c,MODE \mid QP) + \lambda_{MODE} \bullet R(s,c,MODE \mid \lambda_{MODE}) \quad (1)$$

where s and c are the source video signal and the reconstructed video signal, respectively, QP is the quantization parameter, λ_{mode} is the Lagrange multiplier, SSD is the sum of the squared differences between s and c, MODE indicates a MB mode which can be any one of 16x16, 16x8, 8x16 or P8x8. R(s,c,MODE|QP) is the number of bits associated with the chosen MODE and QP. There is also a SKIP mode in P slice referring to the 16x16 mode where no motion and residual information is encoded. SSD is given as

$$SSD(s,c,MODE\,|\,QP)=\sum_{x=1,y=1}^{16,16}(s_Y[x,y]-c_Y[x,y,MODE\,|\,QP)^2$$
$$+\sum_{x=1,y=1}^{8,8}s_U[x,y]-c_U[x,y,MODE\,|\,QP])^2 \quad (2)$$
$$+\sum_{x=1,y=1}^{8,8}s_V[x,y]-c_V[x,y,MODE\,|\,QP])^2$$

where sy[x,y] and cy[x,y,MODE|QP] represent the original and reconstructed luminance component; cu,cv, and su,sv are the corresponding chrominance components. The Lagrangian multiplier λ_{mode} is given by

$$\lambda_{MODE,P}=\begin{cases}0.85\times 2^{QP/3} & forP-frame\\ \max(2,\min(4,\frac{QP}{6}))\times\lambda_{MODE,P} & forB-frame\end{cases} \quad (3)$$

The mode decision for 8x8 sub-type is done similar to the MB mode decision by minimizing the Lagrangian function of (1), with MODE indicating a mode chosen from the set of potential prediction modes such as 8x8, 8x4, 4x8, and 4x4. Notice that intermode decision is an extremely time consuming process. For each position in the search window, motion estimation has to be performed in order to find the motion vector that minimizes

$$J(m,\lambda_{MOTION})=SA(T)D(s,c(m))+\lambda_{MOTION}\bullet R(m-p) \quad (4)$$

where m = $(mx,my)^T$ is the motion vector, p=$(px,py)^T$ is the prediction for the motion vector, and SA(T)D representing sum of absolute difference (SAD) or sum of absolute difference of Hadamard-transformed coefficients. The rate term R(m-p) represents the motion information only and is computed by a table-lookup.

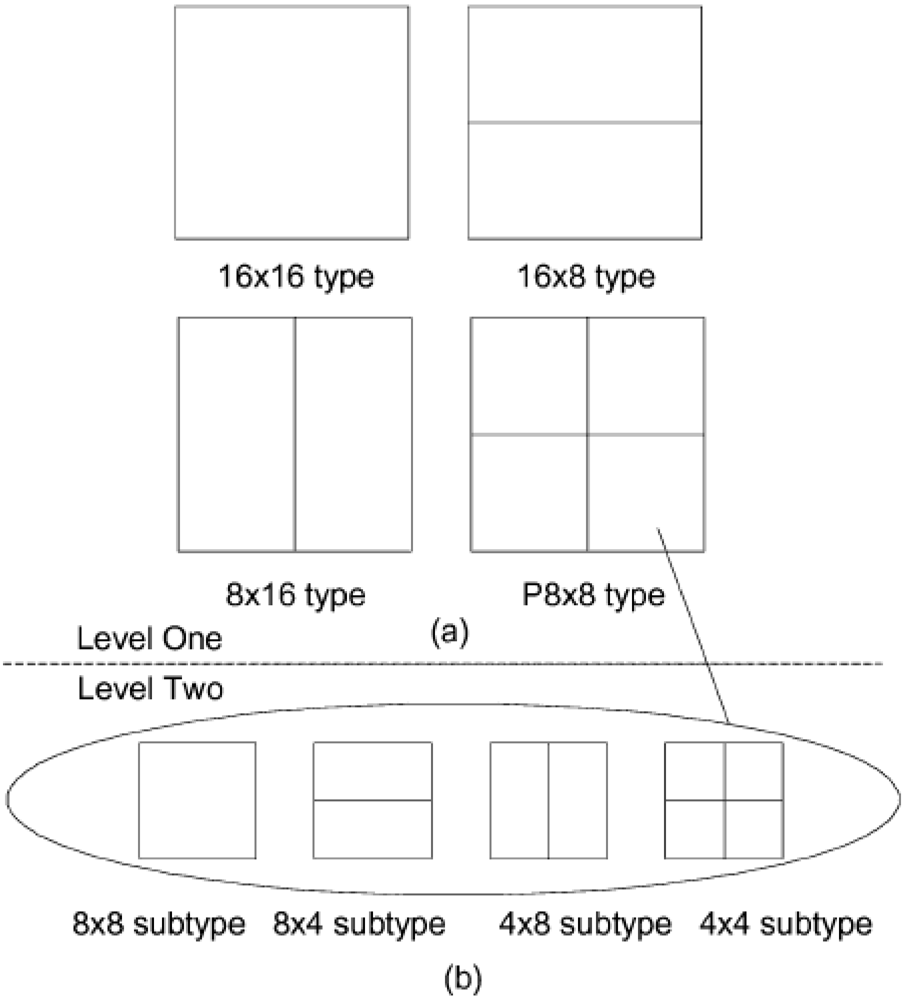

Figure 2.1 Different partitions in a MB. (a) MB partitions. (b) MB subpartitions

2.2 Variable block size motion estimation

Unlike the previous video coding standards such as MPEG 1 and H.263, the H.264 standard allows variable block sizes to be used in motion estimation. It permits seven different block sizes namely 16x16, 16x8, 8x16, 8x8, 8x4, 4x8 and 4x4. A separate motion vector is transmitted for each block size. Thus for a 16x16 macroblock, a maximum of 16 motion vectors can be sent assuming that 4x4 blocks are used. The blocks with higher motion detail can be coded using smaller block size that helps in improving the prediction by taking into consideration fine details in motion. Similarly, the blocks with less motion detail can be encoded using larger block sizes. It can be easily observed that as the block size is reduced, there is an increase in the number of encoded bits since the motion vector for each block has to be transmitted. The VBS scheme aims to produce the minimum number of bits for encoding all kinds of video sequence. This improved compression performance albeit comes with additional computational complexity. Currently, in the intermode RDO implementation of H.264/AVC, motion estimation is performed using ALL the possible block sizes to find the one with the least RD cost using Lagrangian multiplier. Therefore, only the motion vectors belonging to the best size blocks are actually used and the rest of the motion vectors are discarded at the end. Therefore, it is obviously a waste of computational resources by trying all the block sizes exhaustively.

2.3 Literatures review of mode prediction, fast mode decision

In order to save the huge computation of the VBSME, there are several algorithm proposed. In [4], the proposed algorithm uses the property of an all-zero coefficients block that is produced by quantisation and coefficient thresholding to effectively skip unnecessary modes. In [2], in order to reduce the complexity of motion estimation, an early termination algorithm is proposed. It predicts the best motion vector by examining only zero motion vector search point. With the proposed method, some of the motion searches can be stopped early, and then a large number of search points can be skipped. In [1], a fast intermode decision algorithm is proposed. It makes use of the spatial homogeneity and the temporal stationarity characteristics of video objects. Specifically, spatial homogeneity of a MB is decided based on the MB’s edge intensity, and temporal stationarity is decided by the difference of the current MB and its collocated counterpart in the reference frame. Based on the homogeneity and stationarity of the video objects, search modes are reduced in the RDO process. In [5], the proposed scheme is based on a 3D recursive search algorithm and takes into account the motion vector cost and previous frame information. The best mode for the current macroblock is obtained by analyzing the modes for neighbor four macroblocks in the current and previous frames. In [3], a fast intermode decision method that depends on only the absolute differences between consecutive frames is proposed. It uses the absolute frame difference information about the motions in successive frames. Large amplitudes will appear on the moving edges or boundaries of moving objects while small amplitudes in homogeneous areas. Therefore if the amplitudes in a MB are small, it is most likely that this MB belongs to a homogeneous region and using only larger block sizes in motion estimation will be accurate. Otherwise, this MB may contain complex motions and using more block types can achieve better rate distortion performance. These methods can be classified as two categories: using MB content information or using mode correlation of neighbor coded MBs. They predict the possible modes that may be encoded as best mode and turn off certain modes by using info provided by the two categories. The purpose that save computational load is achieved by reducing search modes.

Chapter 3 Proposed Algorithm

In this chap, we describe the proposed algorithm in detail. From the viewpoint of observation to experiment result, we collect the characteristic of the best mode result in the intermode decision. By using the feature of mode decision and motion search result, we propose a prediction-correction scheme algorithm to save the search point that is discarded in mode decision. Two methods are addressed and implemented in JM 10.1 reference model.

3.1 Observation

In Joint Model, the RDO (rate-distortion optimization) procedure includes two parts: mode decision and motion search. Motion search for each mode executes first and then mode decision. For each mode, motion search executes independently. There are 41 partitions for one macroblock to do motion search. After the motion search is done, mode decision procedure chooses the best mode that has lowest RDcost as encoding mode for a macroblock. When RDO procedure is end, the only one mode is selected. The mv length data and best mode result for each macroblock are collected. We observe the mv length distribution and best mode result. We take the “foreman” sequence as example. There are some characteristics:

Best mode is center-biased compared to the un-selected result. Selected result (square shape line) is more concentrate than un-selected result (diamond shape line) in figure 3.1. We illustrate mode 16x16 best mode and non-best result distribution over mv length here. Form the viewpoint of choosing minimal RDcost mode as the best encoding mode it is easy to understand that the search point near the start point has less mv coding bits. And the rate cost term in RDcost formula is smaller. The smaller rate cost term is preferred under the similar distortion cost.

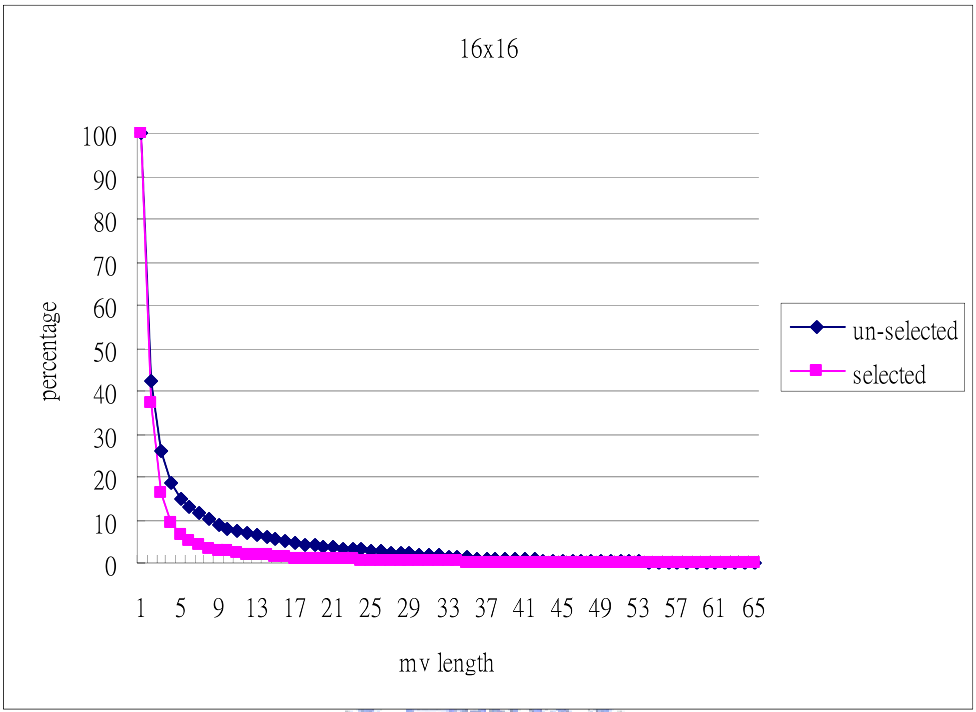

Figure 3.1 distribution on selected (best mode) and un-selected (non-best mode)

When search result exceeds certain distance, the possibility of the result discarded is high. Figure 3.2, 3.3, 3.4, 3.5 are shown that the further search point goes, the less chance it has to be selected. All macroblock partitions and sub-macroblock partitions have this tendency. Under the RDcost consideration, although these results are the best motion vector candidates of each partition during motion estimation procedure, the less rate cost one is preferred under the similar distortion cost level. In figure 3.2, we can find that there is 90% of mode 16x16 block search result is not selected as best mode when mv length exceeds 13. It is a lot of percentage that the motion estimation result is discarded in mode decision process. It means that the only 10% result is useful for encoding and the rest 90% search result is total a waste! The same tendency is also found in figure 3.3, 3.4, 3.5. the formula of the figure 3.2, 3.3, 3.4, 3.5 is

$$P_m = 100 \times \frac{\sum_{mv\_length=x}^{64} N_{not-selected}(x)}{\sum_{mv\_length=x}^{64} N_{total}(x)} \quad (5)$$

P is the percentage of mode m, m={16x16 16x8 8x16 8x8}

x is the mv length x=[0 64]

$N_{not\text{-}selected}(x)$ is the number of mode m not-selected as best result at mv length = x

N_{total} (x) is the number of mode m result at mv length = x

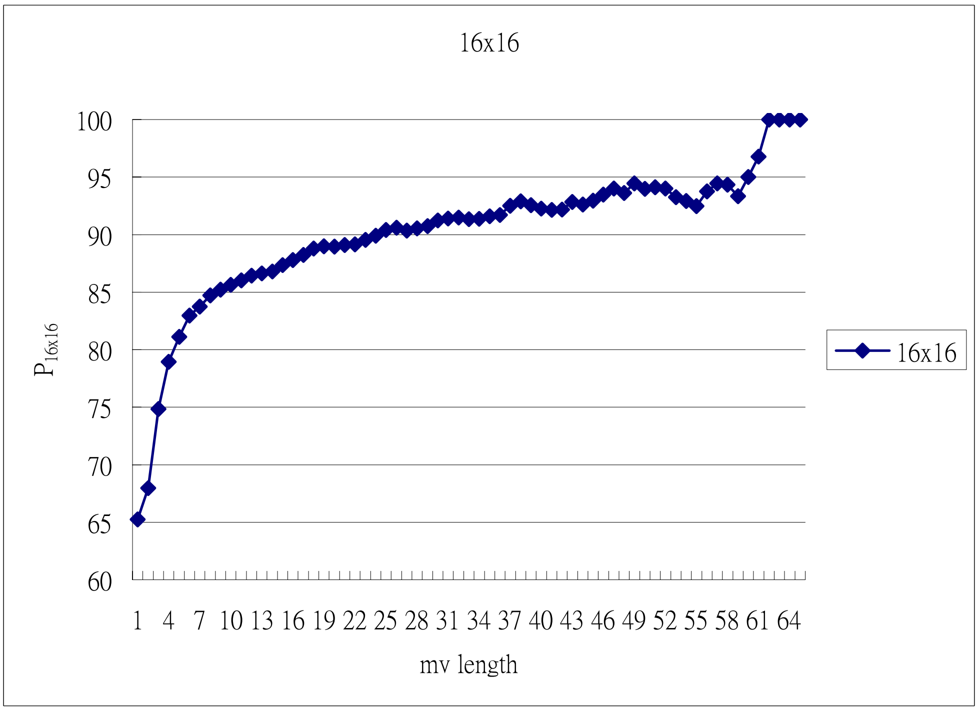

Figure 3.2 the percentage of mode 16x16 un-selected as best mode along mv length

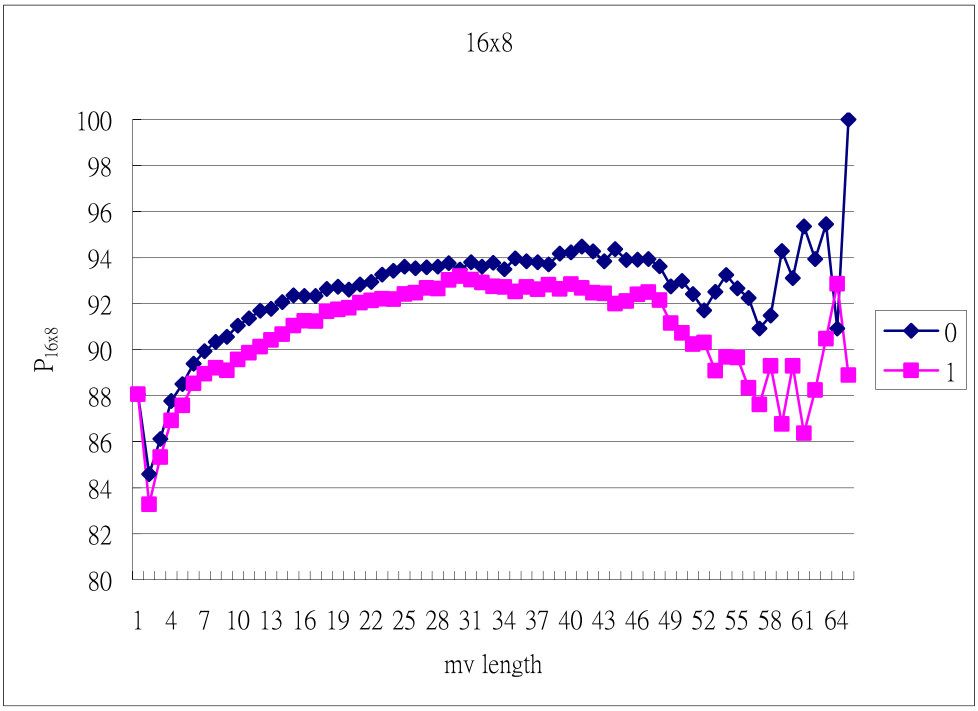

Figure 3.3 the percentage of mode 16x8 un-selected as best mode along mv length

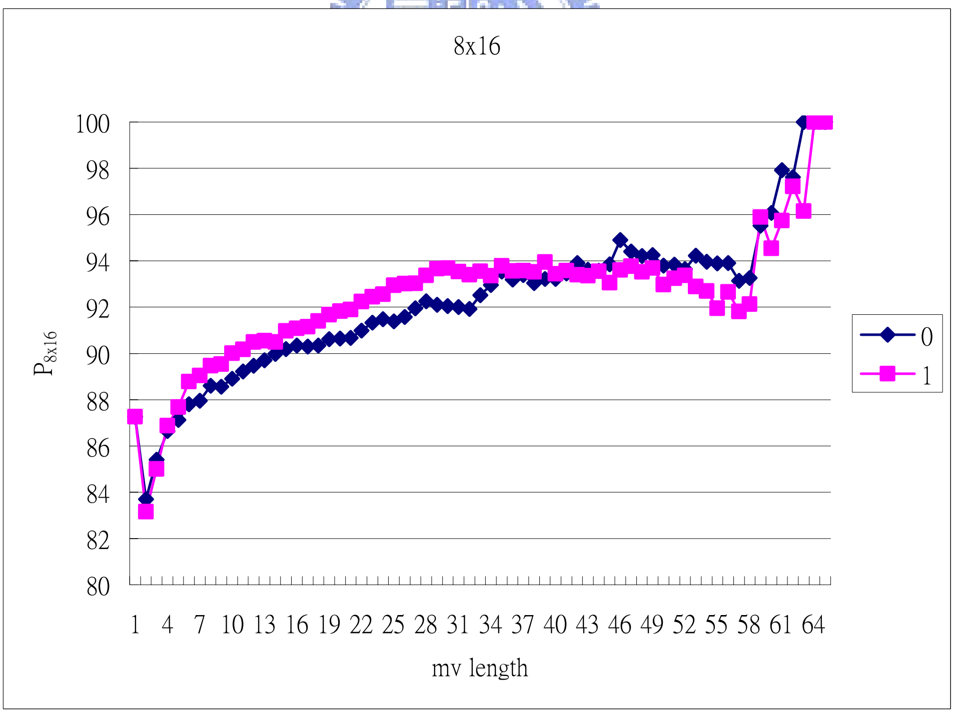

Figure 3.4 the percentage of mode 8x16 un-selected as best mode along mv length

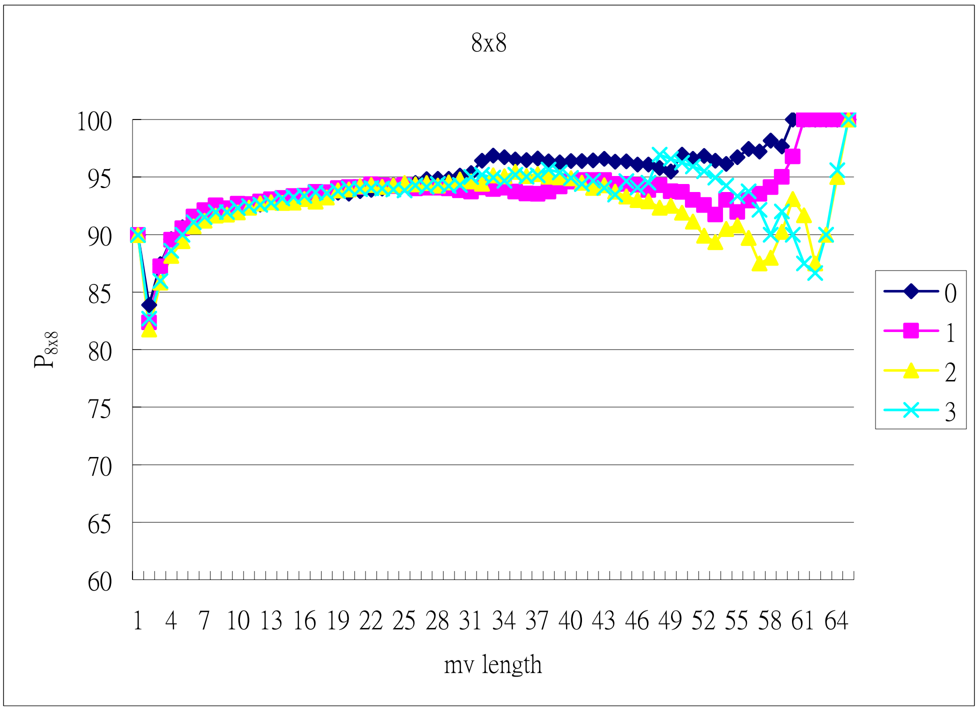

Figure 3.5 the percentage of mode 8x8 un-selected as best mode along mv length

Mode 16x16 has more possibility selected as the best mode when its best search point goes further than the rest modes. From the figure 3.6, we can find that the mode 16x16 result take a lot of percentage along mv length. For all mode decision result, mode 16x16 is 40% or more percent. The formula of figure 3.6 is

$$P_m = 100 \times \frac{\sum_{x}^{64} N_m}{\sum_{x}^{64} N_{16\times16} + \sum_{x}^{64} N_{16\times8} + \sum_{x}^{64} N_{8\times16} + \sum_{x}^{64} N_{8\times8}} \qquad \begin{array}{c} x = mv\_length \\ m = \{16\times16 \quad 16\times8 \quad 8\times16 \quad 8\times8\} \end{array}$$

N_m is the number of mode m selected as best result at mv length = x

Pm is the percentage of best mode beyond mv length = x

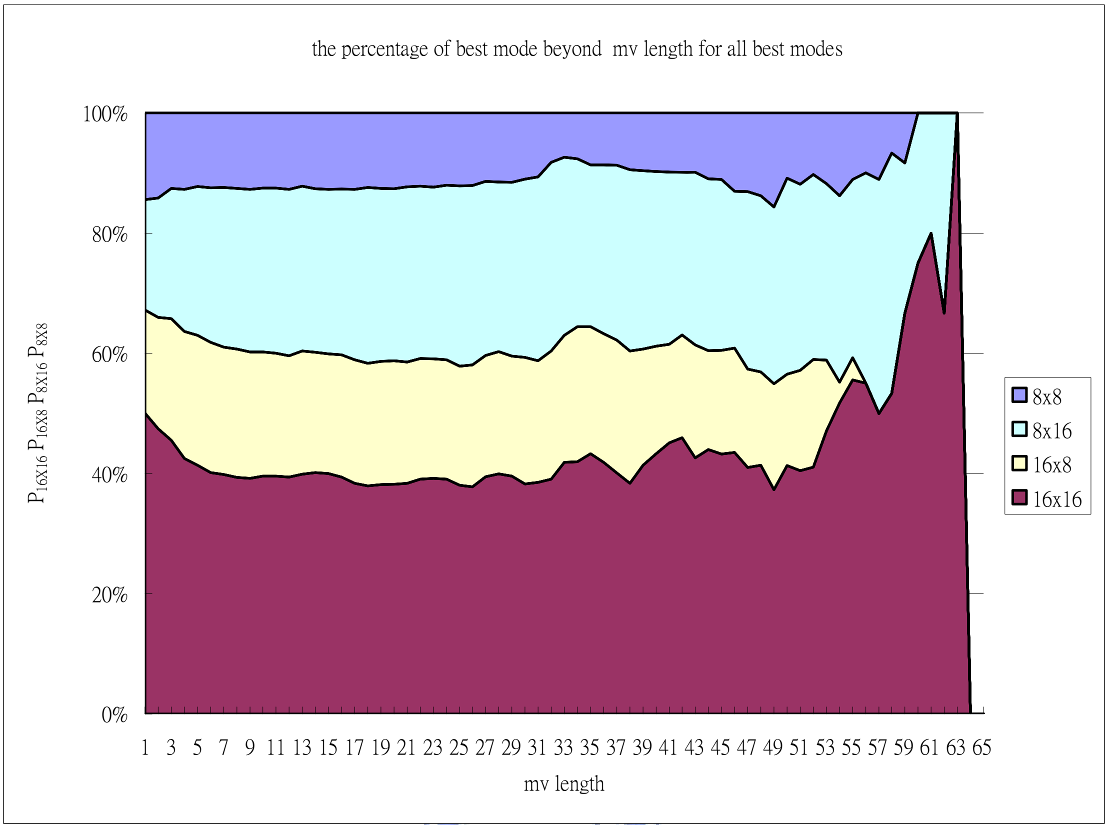

Figure 3.6 the percentage of each best mode in all modes along mv length

3.2 motivation

the mode prediction/fast mode decision algorithms in previous literatures use the indicator like SAD, edge vector, all zero coded block of the current macroblock or the coded mode correlation of the neighbor macroblocks. In order to reduce the computational load, these algorithms predict the possible modes for the current macroblock before motion estimation. After predicting the possible modes for the current macroblock, motion estimation executes. The main idea of these algorithms is that various block types are not efficient for each macroblock. Because each macroblock has different content and motion characteristic, so the search modes are predicted. As [1] mentions that if one macroblock is homogenous, the possibility of best mode is skip mode or 16x16 mode is high. Motion search of the other modes becomes meaningless. Such methods that turn off certain modes to reduce VBSME complexity are addressed versatilely. Based on our observation, efficient modes for motion search should be changed with the distance from start point. In other words, there should be different modes turned on with different distance between start point and search point. Which modes should be turned on is based on the distance. The idea of prediction mode based on the distance is implemented in motion estimation. That candidate search points are skipped or not depends on that the search point is outside or inside the predictive search distance. The skipped search points are probably discarded in mode decision procedure. Because of the skip mechanism, the complexity of VBSME is reduced.

3.3 proposed algorithm

3.3.1 The fixed search distance scheme

We first use the fixed search distance cap to limit the search step. The fixed search distance is used for motion estimation. When the search point is outside the fixed search distance, this search is skipped. We test this mechanism for 3 clips: “akiyo”, “foreman”, “stefan”. The experimental environment setting is listed at Table 4.1. The experimental results are shown in Figures 3.7, 3.8, 3.9. We find that scheme with the fixed search distance can save the unnecessary search but causes one problem. It causes that the difference PSNR degradation for different clips. Figures 3.7, 3.8, 3.9 show that fixed search distance causes difference of PSNR degradation for “akiyo”, “foreman”, “Stefan”. Especially in “stefan” case, the PSNR degradation is worse than other two. Because “Stefan” is fast motion clip, the motion vector length is longer. The fixed search distance causes erroneous search result because the motion search is limited in too small range. So the fixed search distance scheme is not suitable for all clips because of the difference motion characteristic among clips.

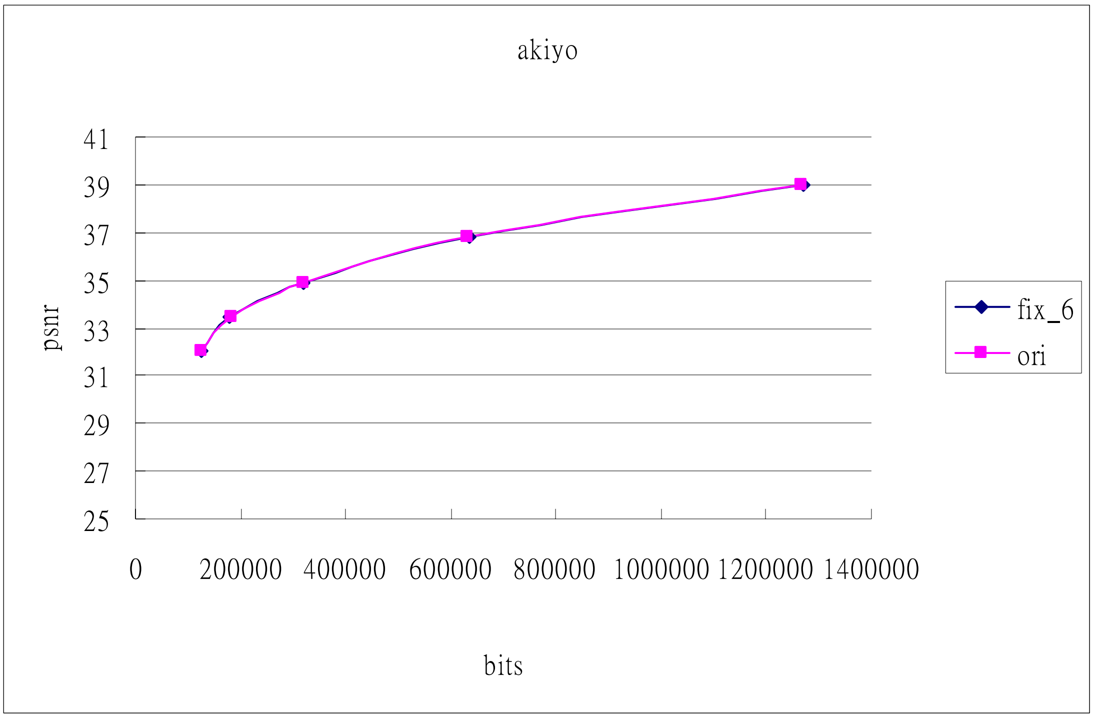

Figure 3.7 RD-curve of fixed search distance and origin search result for “akiyo”

(Fix_6 means that the fixed search distance is set as 6. Ori means that the search is original FME search.)

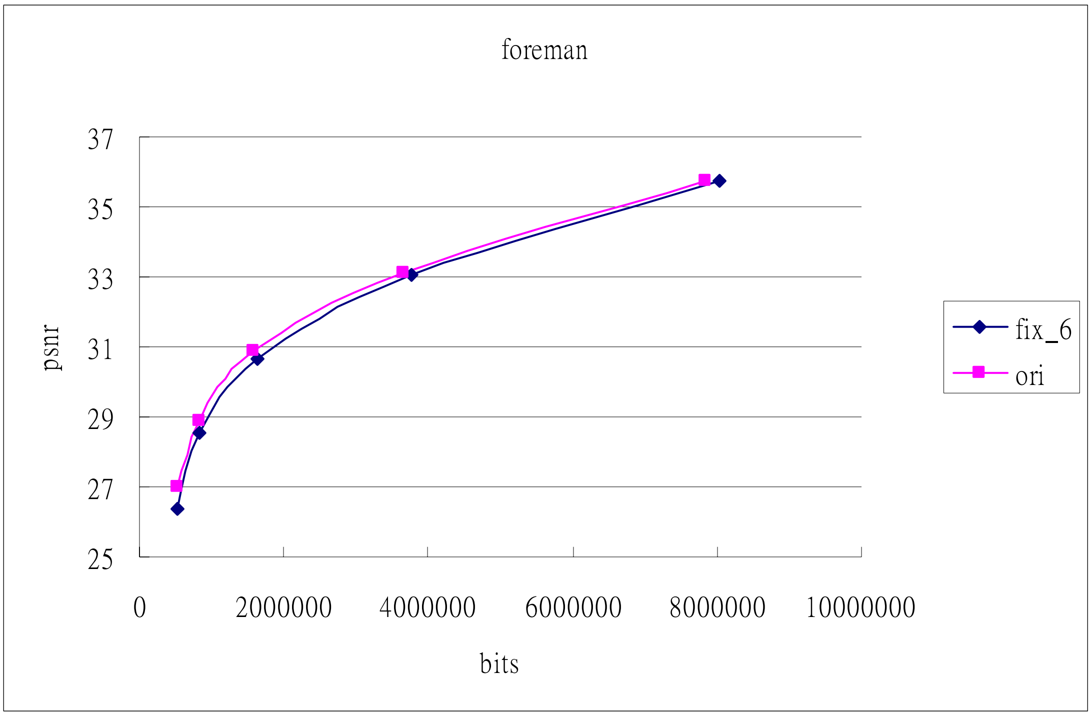

Figure 3.8 RD-curve of fixed search distance and origin search result for "foreman"

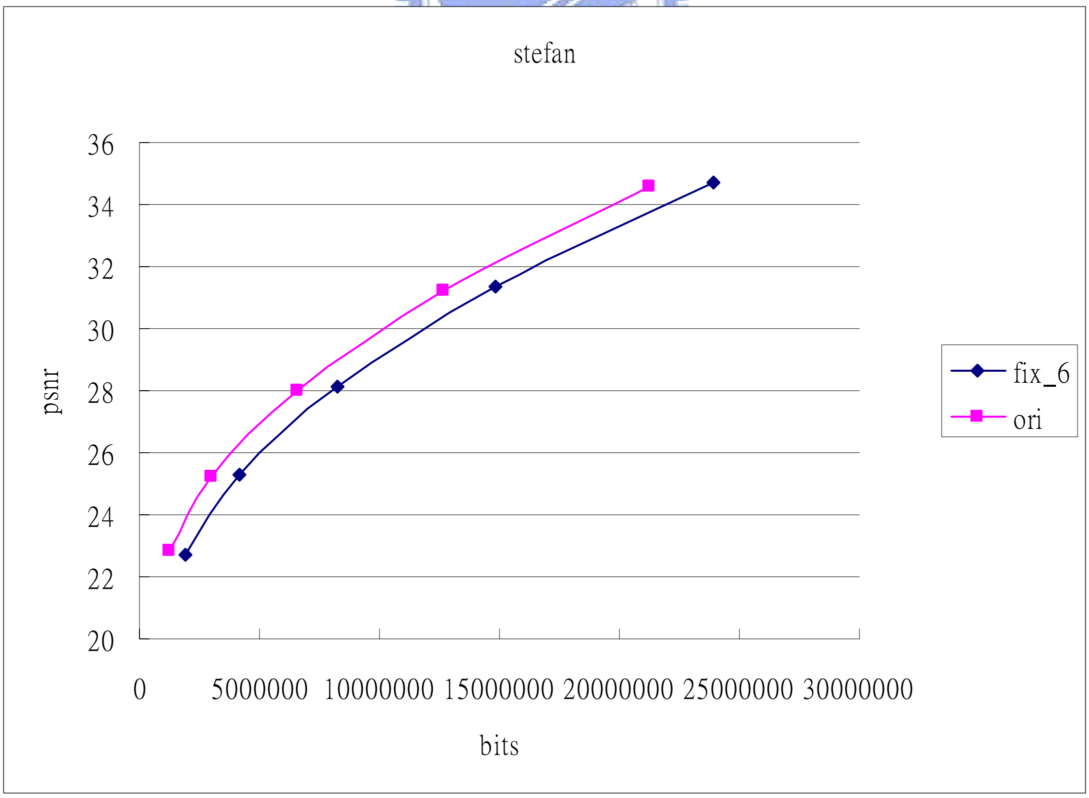

Figure 3.9 RD-curve of fixed search distance and origin search result for "stefan"

3.3.2 The predictive search distance scheme

Because the fixed search distance scheme is not suitable for different motion characteristic clips, determining the search distance cap adaptively is necessary for keeping the same quality. In order to save the search point that may be discarded in mode decision, the idea of the predictive search distance is first come to us. Just like motion vector predictor, the median of neighbor MVs is used as predictive search distance. The reason using predictor is based on the correlation of neighbor MBs motion information. There is a difference that motion vector predictor give motion search procedure a good initial start point. Here we use predictor to provide a cap to limit search steps. When each motion search begins, every search point needs to check the distance between start point and the predictive search distance. If the search point is inside the predictive search distance, the search continues. If not, the search point is skipped. Figure 3.10 show the concept that the search point marked as dash line arrow outside the predictive search distance is skipped and marked as the solid line arrow

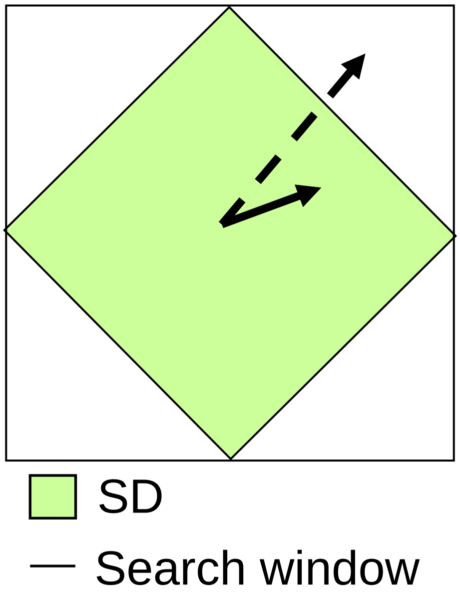

Figure 3.10 the diagram of the search distance cap

Again, we test the scheme for 3 clips: “akiyo”, “foreman”, “stefan”. The experimental environment setting is listed at table 4.1. The figure 3.11, 3.12, 3.13 show the RD-curve of the three clips: “akiyo”, “foreman”, “Stefan”. The prediction scheme is work well at “akiyo” and “foreman”. The RD-curve figure shows only a little bit PSNR loss at the same bits for both clips. But for “stefan” clip, RD-curve shows about 0.4-0.5 dB at the same bits. The reason that causes significant PSNR loss is prediction error of the search distance cap.

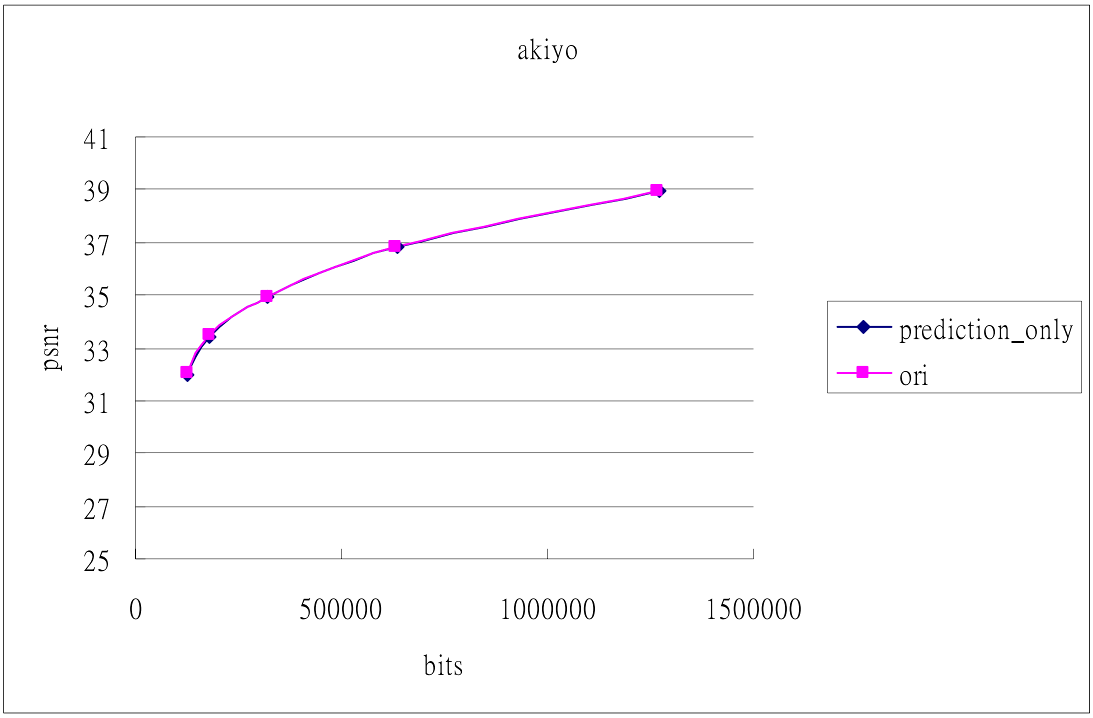

Figure 3.11 RD-curve of the predictive search distance and origin search result for "akiyo"

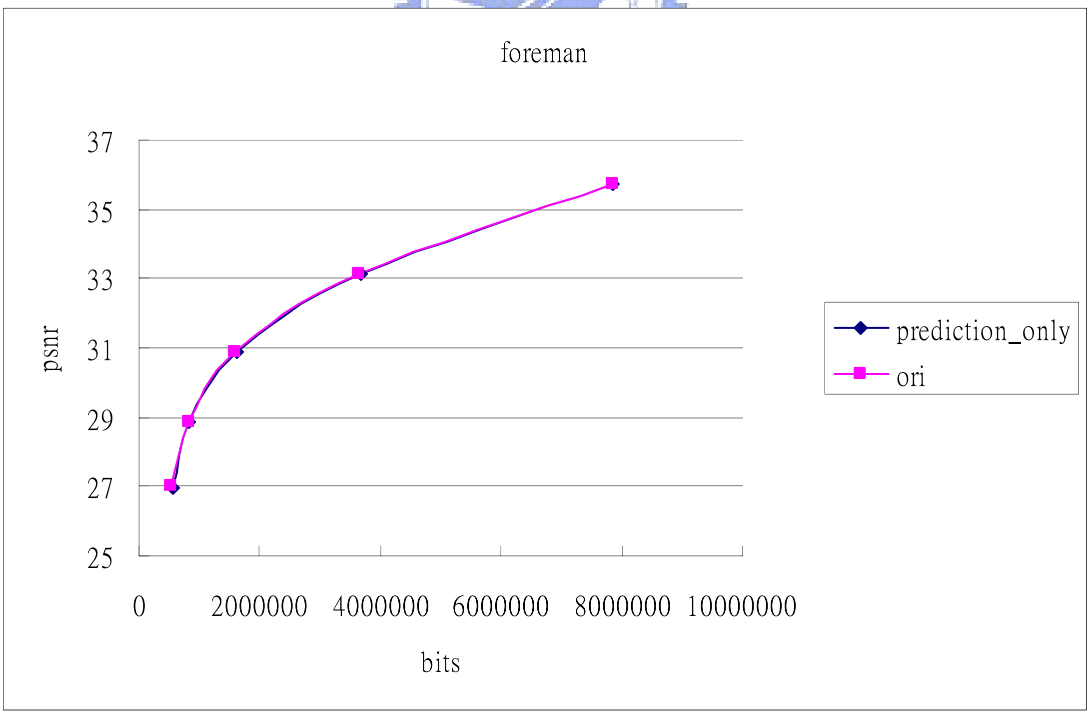

Figure 3.12 RD-curve of the predictive search distance and origin search result for "foreman"

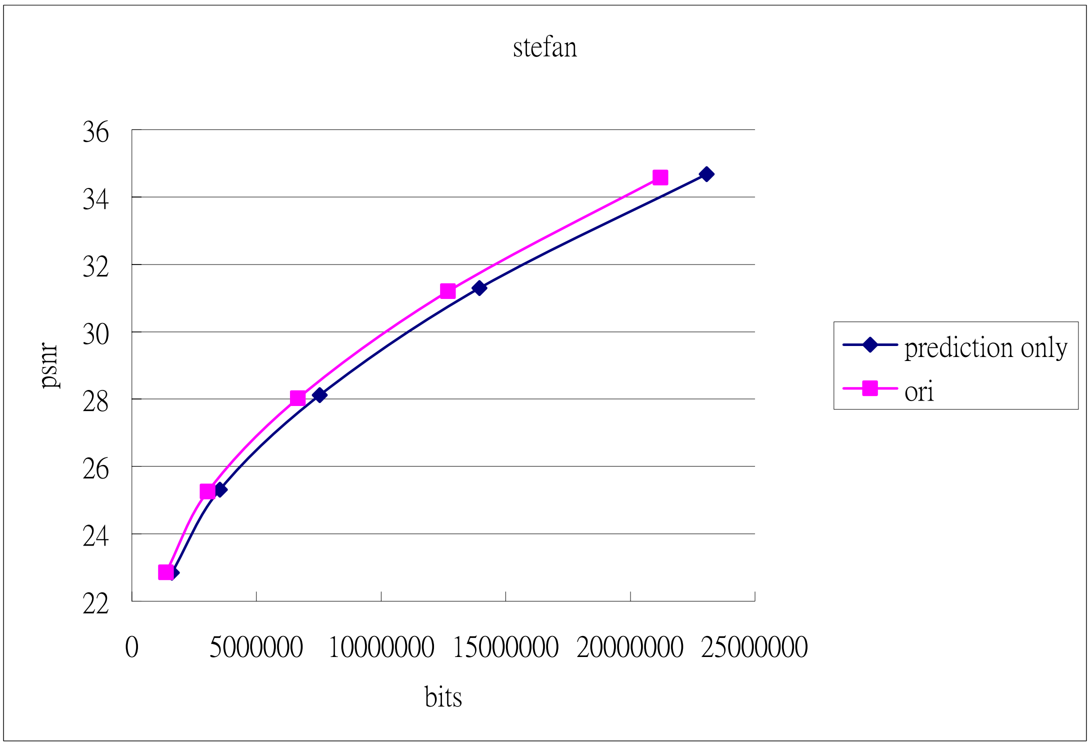

Figure 3.13 RD-curve of the predictive search distance and origin search result for "stefan"

The figure 3.14 explains how this error happens. When neighbor MB motion feature is slow motion, the prediction tends to predict a small search distance. If the current MB is fast motion, that it has longer mv length, the search distance cap limits motion search. It leads to erroneous search result. Even worse this error will propagate. When slow to fast transient happen then prediction error occur, the error search result cause succeeding MBs searching in the small search distance.

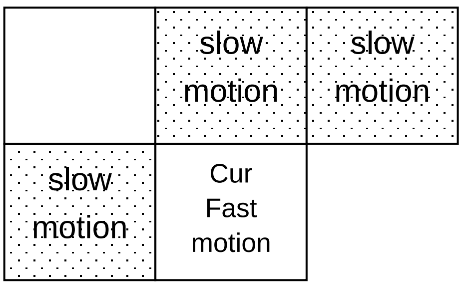

Figure 3.14 the motion transient diagram

3.3.3 The prediction-correction scheme

In order to avoid the prediction error condition and improve the performance for fast motion clip, the prediction-correction scheme is proposed. According to the observation for the figure 3.6 that 16x16 mode has the more possibility that is selected as best mode when the search point goes further than the result of the rest modes. The result of 16x16 mode for current macroblock is adopted as the correction factor to avoid the prediction error. The search distance cap is predicted by median value of MV_A, MV_B, and MV_C. Then the search distance cap is corrected when the MV length of 16x16 mode search result is longer than median value of MV_A, MV_B, MV_C. The search distance cap (SDC) is corrected by 16x16 mode search result because the current macroblock has fast motion property. When the current MB has slow motion property, the center-bias search algorithm will self-regulate the search distance to shorter MV. And the median of neighbor MV shrinks the SDC. The adaptive SDC mechanism is achieved by this prediction-correction scheme. This scheme doesn't limit the search distance of 16x16 mode because the 16x16 mode search result is the correction indicator to verify the motion property of the current macroblock. If we limit the search distance of 16x16 mode, the search result may be erroneous.

The motion estimation in h.264 joint model is two-stage search, that is, the integer pixel search is done in predefined search range and then the fractional pixel (half-pel and quarter-pel) search is used to refine the int. pel ME result. The proposed algorithm adjusts the search distance in integer pixel motion estimation. But we don't apply the SDC to the fraction pel motion estimation. Because the fractional pel motion estimation is local search. It affects the rate-distortion cost of the best mode but its search points are near the neighborhood of the integer pel search result. The integer pixel search result is not necessarily chosen as the best mode result. Our experiment result shows that. Figure 3.15, 3.16 show the RD-curve of the two cases: IME (integer pel motion estimation) only, and IME + FME (fractional pel motion estimation). The proposed and original algorithms are applied to experiment. The RD-curve performances of the two-stage ME for original and proposed scheme are close.

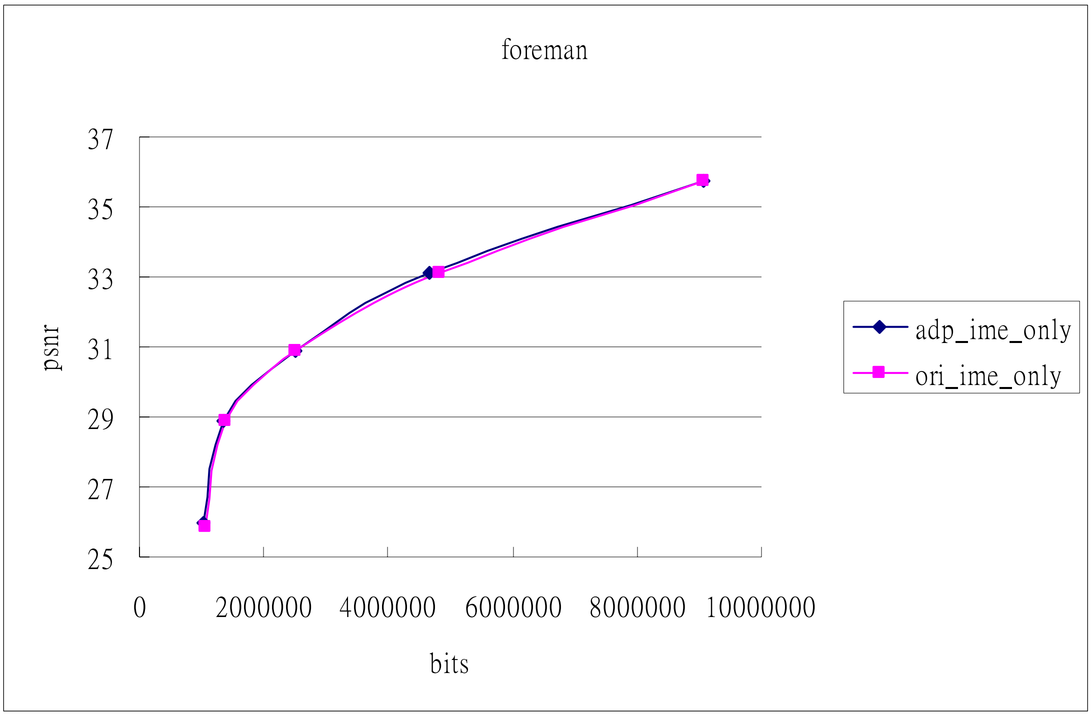

Figure 3.15 RD-curve performance of foreman with IME-only scheme

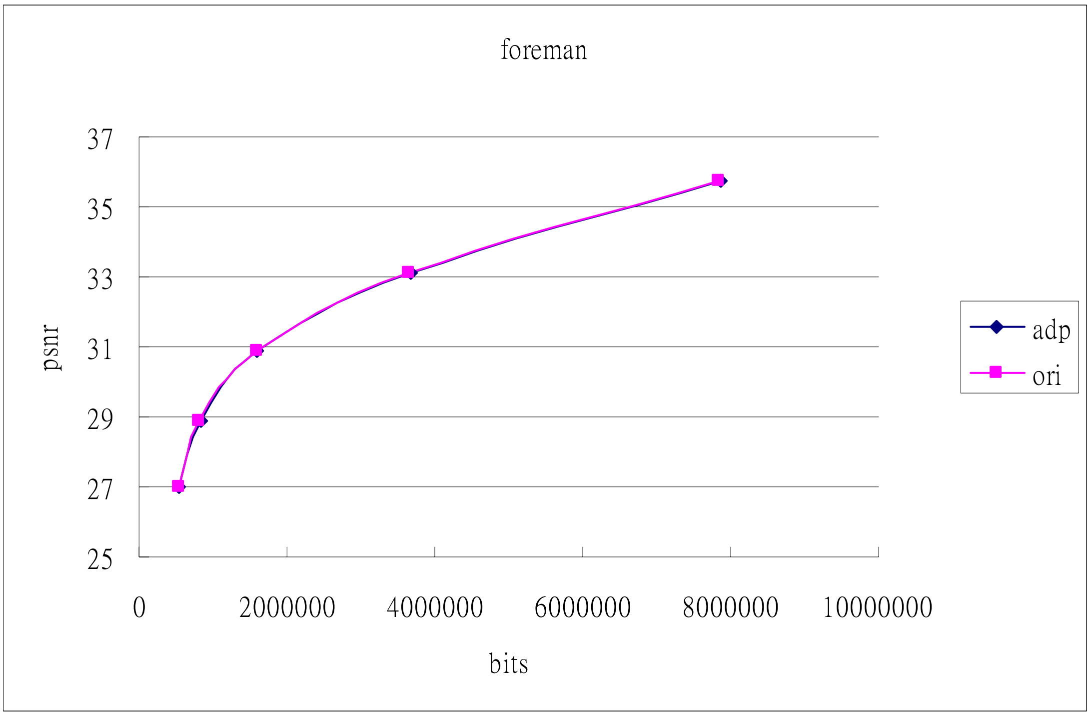

Figure 3.16 RD-curve performance of foreman with IME+FME scheme

The original system structure of ME in h.264 is two-stage motion estimation: integer pixel motion estimation and fractional pixel motion estimation. It is showed as figure 3.17(a). When encoding start, the coding procedure chooses the block size mode and then set the subblock number for integer pixel motion search loop. After the integer motion search, the coding procedure goes to the fractional pel motion search refinement. All modes of VBSME are looped to get their SAD and MVD cost by two-stage motion estimation. Then RDO procedure determines the best mode of current macroblock. We implement the proposed algorithm in h.264 joint model as figure 3.17(b). The proposed algorithm determines the search range distance cap by using the search result of 16x16 mode. So we separate the search progress into two parts: 16x16 mode and the rest modes. The search distance cap determination chooses a cap to limit the rest mode search distance. The mechanism of search distance cap determination block is described in figure 3.18. Unlike the original VBSME procedure, the SDC limit search distance of the rest search mode. When the search point goes further than the SDC, the search is stop.

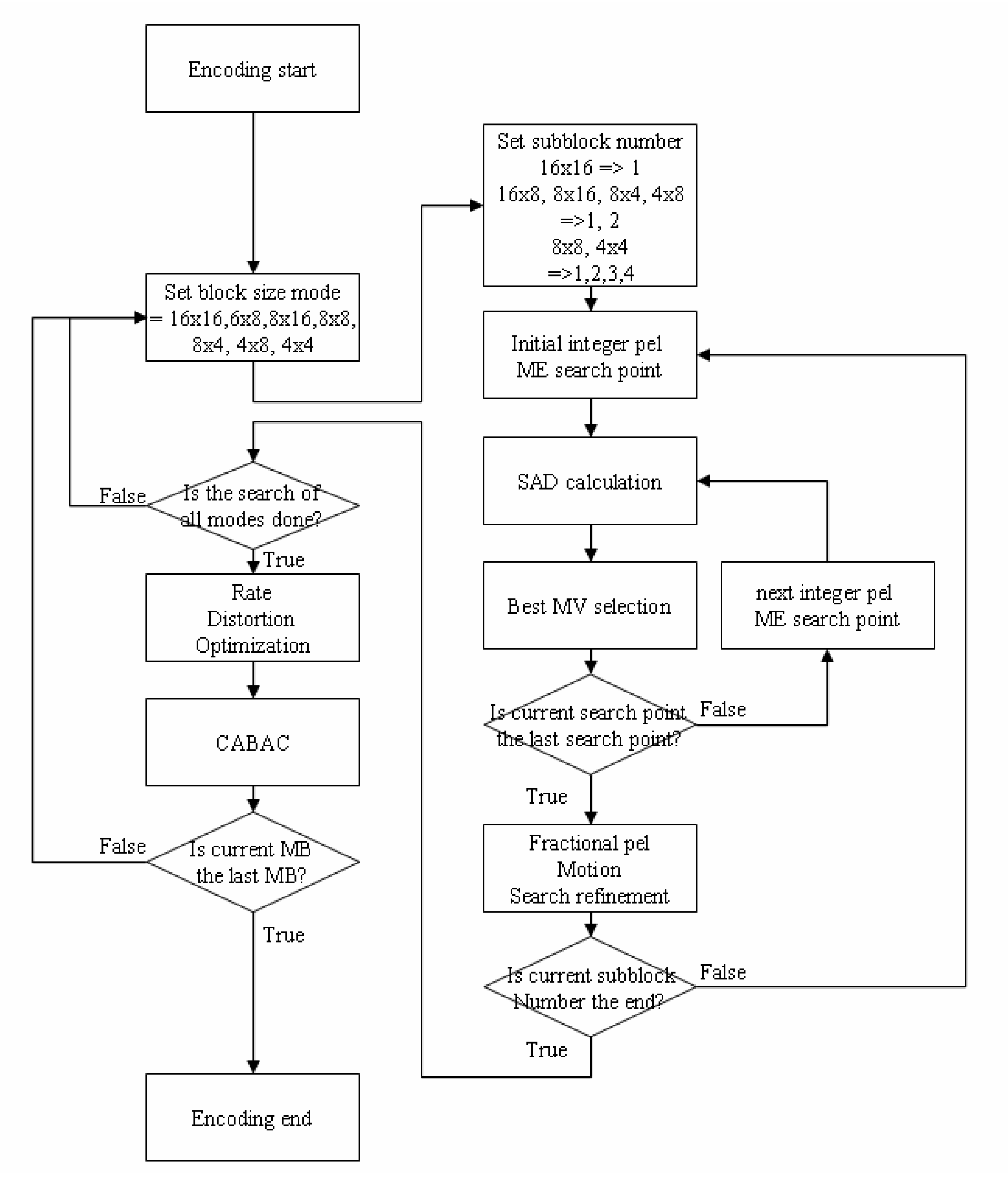

Figure 3.17 the flow chart of original VBSME (a)

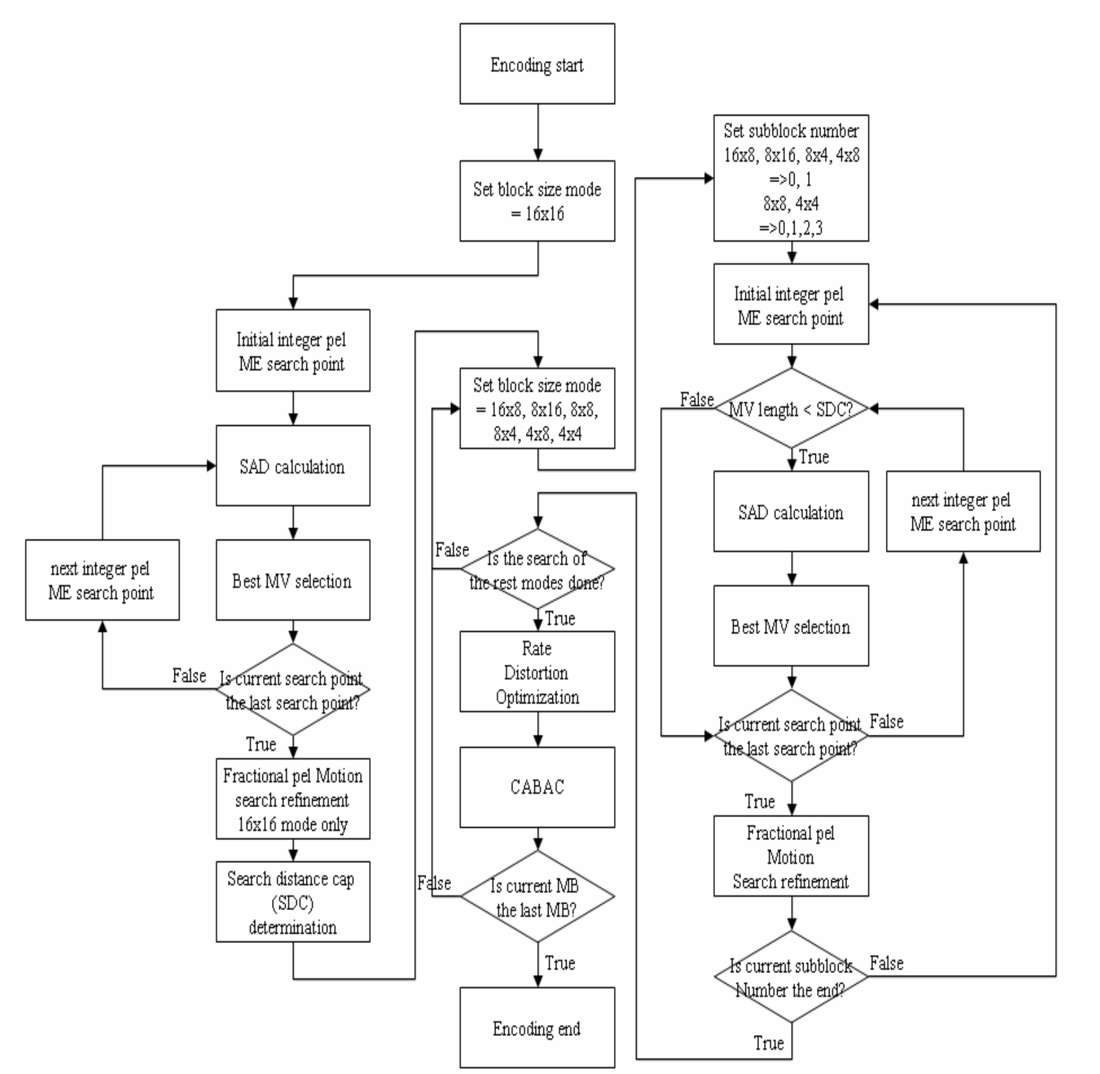

Figure 3.17 the flow chart of proposed algorithm (b)

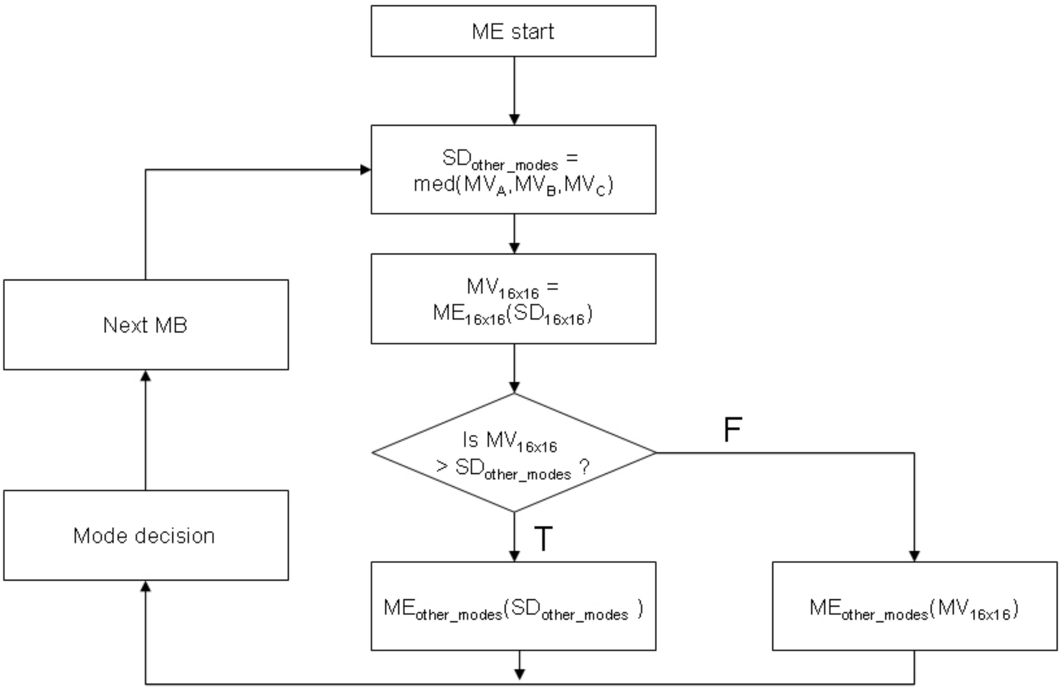

Figure 3.18 the flow chart of search distance cap determination

The flow chart of search distance cap determination is shown as figure 3.18. And the procedure of the adaptive search distance adjustment using prediction-correction scheme is described as follows.

Step 1: search distance prediction

Predict SD_{other_modes}, search distance cap of the all block mode except 16x16 type, and the median value of the neighbor macroblocks' motion vector length is chosen as SD_{other_modes}.

Step 2: motion estimation for 16x16 mode

Do motion estimation of 16x16 mode, and then MV_{16x16} value is got.

Step 3: search distance correction for rest modes

We use max criterion to choose SD_{others_mode} and MV_{16x16} as the search distance cap for rest modes. If SD_{others_mode} is chosen, then the procedure goes to step 5, else goes to step 4.

Step 4: motion search for the rest modes using the MV_{16x16} as cap.

When doing motion search procedure, if the distance between candidate search point and start point is longer than MV_{16x16}, this search point calculation is skipped. After motion search is done, the procedure goes to step 6.

Step 5: motion search for the rest modes using the SD_{other_modes} as cap.

When doing motion search procedure, the procedure is just like step 4 described. But the search distance cap is SD_{other_modes} to determine that the search point is skipped or not. When motion search is done, the procedure goes to step 6.

Step 6: mode decision

Compute the RDcost of all modes and select the lowest RDcost mode as encoding mode. And then the procedure goes to step 7.

Step 7: next macroblock

After encoding the macroblock, change the macroblock to next one. And then the procedure returns to step 1.

Chapter 4 Experimental Result

In our simulation, the proposed algorithm is simulated in H.264/MPEG-4 AVC with software model JM10.1 on sun ultra-20. The proposed algorithm is implemented with FME. All intermode is on. Motion vector precision is quarter-pel. The default search window for CIF (352x288 pels) is ±32, and for D1 (720x480 pels) is ±64 in both horizontal and vertical directions. Encoding frame type is P-type except the first I-frame. The tested clips are “akiyo”, “bus”, “container”, “coastguard”, “dancer”, “foreman”, “mobile”, “news” and “stefan” for CIF and “akiyo”, “container”, “coastguard”, “football” and “mobile” for D1. We scan QP value to encode each clip and get the result to draw RD-curve of each clip and record its motion estimation time in processor tick count. The experimental results of the proposed algorithm are evaluated by 3 factors as (6) (7) (8). Because we want to reduce the complexity of VBSME under the similar rate-distortion performance, so the terms: $\Delta PSNR$ and $\Delta bits(\%)$ are as small as possible with the term $\Delta me\_time(\%)$ is larger.

$$\Delta PSNR = PSNR_{original} - PSNR_{proposed} \quad (6)$$

$$\Delta bits(\%) = 100 \times \frac{(bits_{proposed} - bits_{original})}{bits_{original}} \quad (7)$$

$$\Delta me\_time(\%) = 100 \times \frac{me\_time_{original} - me\_time_{proposed}}{me\_time_{original}} \quad (8)$$

The definition of $\Delta PSNR$, $\Delta bits(\%)$ and $\Delta me\_time(\%)$ is listed above.

The meaning of $\Delta PSNR$ is the degradation of PSNR between original FME and FME with proposed algorithm under the same QP. If $\Delta PSNR$ is positive value, it means that the proposed result has worse quality compared to the original result. The meaning of $\Delta bits(\%)$ is percentage of bits-increment to the original FME result. If

$\Delta bits(\%)$ is positive value, the proposed one produces more bits to the original one. The meaning of $\Delta me\_time(\%)$ is percentage of motion estimation time saved to the original FME result. The me time is the processor computation tick consumed in the motion estimation procedure. If $\Delta me\_time(\%)$ is positive value, it means that the proposed algorithm saves more ticks compared to original one.

Table 4.1 simulation condition list

clip	Frame size	Frame numbers	Encode frame type	Motion estimation algorithm	Search window	Motion vector precision	Inter-mode	QP range
akiyo	CIF 352x288	300	IPPP	FME (UMHexgon)	±32	1/4 pel	16x16 16x8 8x16 8x8 8x4 4x8 4x4	28 32 36 40 44
bus		150						
container		300						
coastguard		300						
dancer		250						
foreman		300						
mobile		300						
news		300						
stefan		300						
akiyo	D1 720x480	300			±64			
container		300						
coastguard		300						
football		260						
mobile		300						

The figure 4.1, 4.2, 4.3 are the RD-curve of “akiyo”, “foreman”, and “stefan”. We can find that the performance between the original FME and the FME applied with the proposed algorithm is close. Under the same bits, the degradation of PSNR for “akiyo” is about 0.02 dB, for “foreman” is about 0.01 dB and for “stefan” is about 0.02 dB. The saved motion estimation time is 16.86 % compared to original algorithm for “akiyo” clip, 26.32% for “foreman” clip, and 32.168% for “stefan” clip.

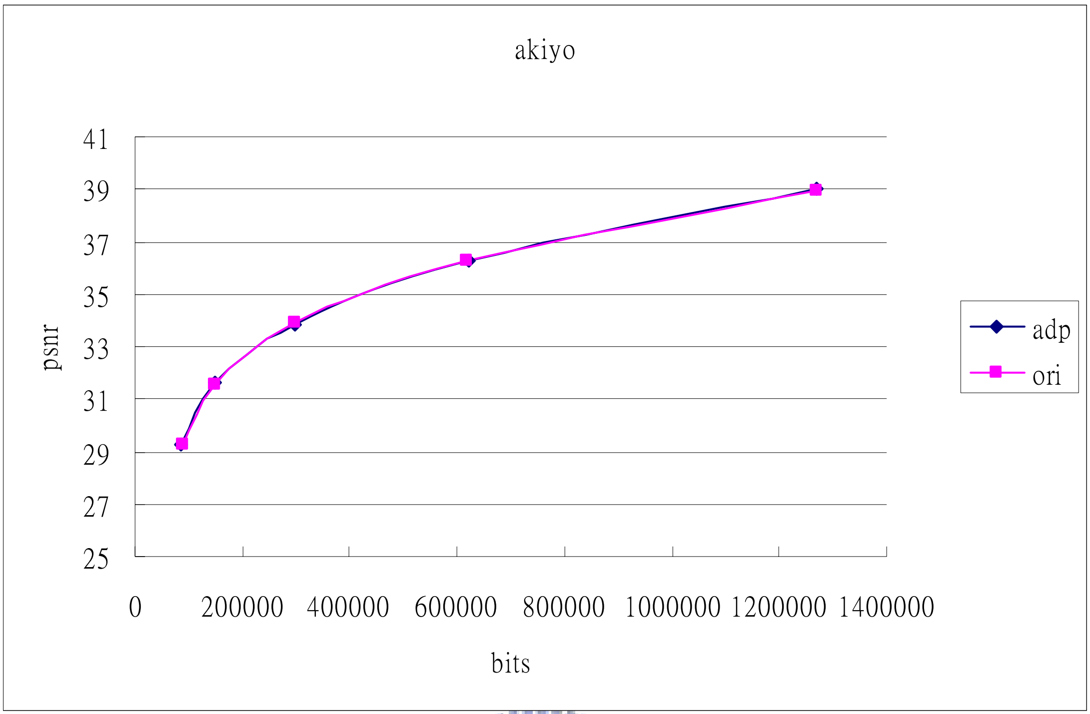

Figure 4.1 RD-curve of the adaptive scheme and the origin one for “akiyo”

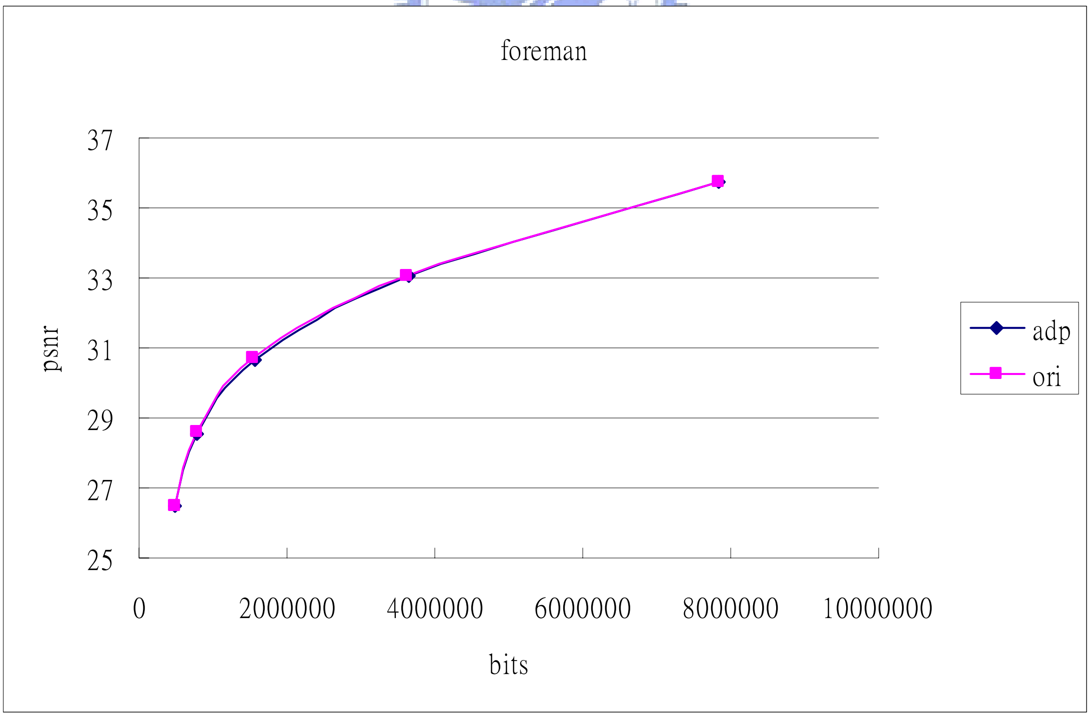

Figure 4.2 RD-curve of the adaptive scheme and the origin one for "foreman"

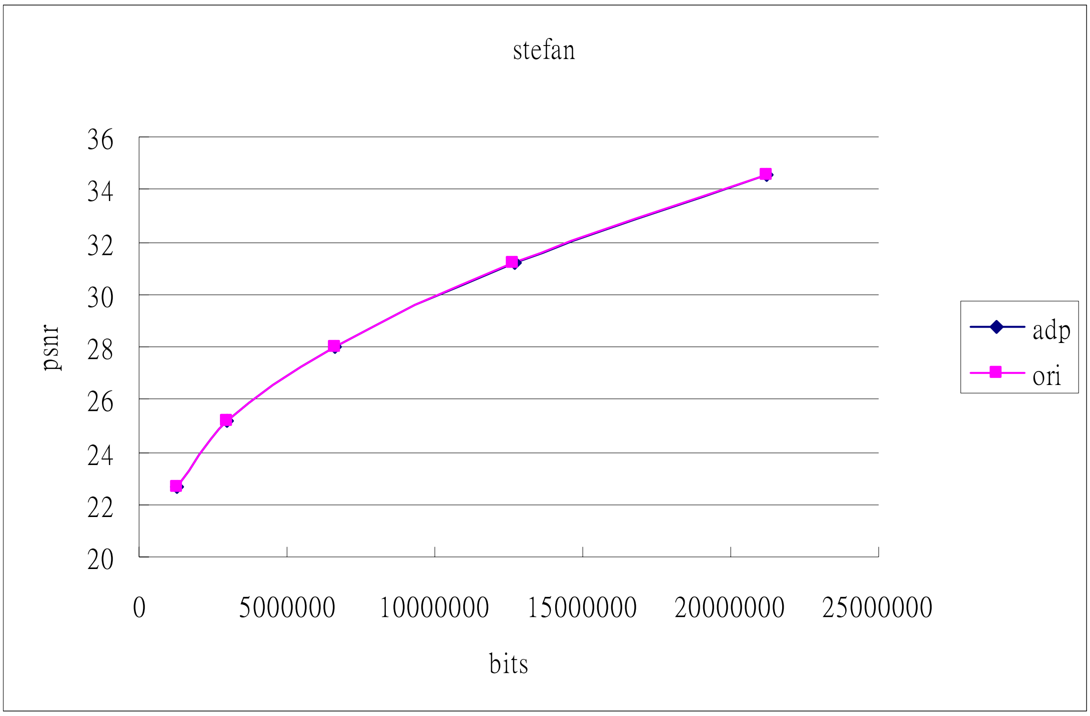

Figure 4.3 RD-curve of the adaptive scheme and the origin one for “stefan”

The all test sequence results are all listed at table 4.2 ~ 4.10. Take CIF format result as example. If we classify all CIF clips into three categories: fast, medium, slow motion characteristic types, “akiyo” and “container” are slow ones. And “coastguard” and “foreman” are medium motion clips. Fast motion clips are “bus”, “dancer”, “mobile” and “stefan”. We can find that tests of fast motion clips with the proposed algorithm are saving more percentage of the motion estimation time than tests of the medium and slow motion clips. Tests of medium motion clips save more time than tests of slow motion ones. The explanation of this phenomenon is just the advantage of adaptive search distance cap. As we know the motion vector length of fast motion clips is longer, it means that search takes longer time to find the best position of each mode for fast motion estimation algorithm. Because the results of the rest modes except the best mode are discarded, the search time consumed is a waste. But the proposed algorithm suppresses this waste. The original fast motion estimation scheme just let the search go and finally discard the result. From the RD performance of the tests, the proposed algorithm did good performance and saved the motion estimation time.

Table 4.2 PSNR degradation of all CIF format clips

△PSNR(dB)					
clip\QP	28	32	36	40	44
akiyo	-0.01	0.00	0.03	-0.04	0.04
bus	0.00	0.00	-0.01	0.01	0.02
container	-0.01	-0.01	0.00	0.00	0.01
coastguard	0.00	0.00	0.01	0.00	0.02
dancer	-0.01	0.01	0.01	0.01	0.01
foreman	0.00	0.00	0.01	0.02	0.00
mobile	0.00	0.00	0.00	0.01	0.02
stefan	0.00	0.01	0.00	0.00	0.01

Table 4.3 motion estimation time saved of all CIF format clips

△me time(%)					
clip\QP	28	32	36	40	44
akiyo	16.14	12.74	12.30	20.05	18.22
bus	35.13	29.19	32.28	35.09	33.01
container	20.26	26.67	31.02	28.90	29.83
coastguard	27.83	29.55	35.19	26.32	28.97
dancer	30.91	27.88	28.88	19.52	32.73
foreman	26.41	26.08	24.88	26.95	27.26
mobile	34.51	31.10	40.00	34.15	35.21
stefan	32.38	32.37	32.31	32.84	30.94

Table 4.4 bits increment of all CIF format clips

△bits(%)					
clip\QP	28	32	36	40	44
akiyo	0.18	0.62	-0.28	1.01	-0.27
bus	0.26	0.17	0.20	0.45	0.61
container	-0.04	-0.07	-0.20	-0.55	-0.78
coastguard	0.04	-0.05	-0.35	0.10	-0.72
dancer	0.33	0.16	0.21	0.43	0.13
foreman	-0.03	0.11	0.79	0.27	0.56
mobile	0.05	0.18	0.11	0.10	0.60
stefan	0.19	0.33	0.22	0.31	1.12

Table 4.5 PSNR degradation of all D1 format clips

△PSNR(dB)					
seq.\QP	28	32	36	40	44
akiyo	0	0	-0.01	0.01	0.01
container	0	0	0	-0.01	0
coastguard	0	0.01	0.01	0	0.01
football	0.01	0.01	0.01	0.02	0.06
mobile	0	0	0.01	0.02	0.01

Table 4.6 motion estimation time saved of all D1 format clips

△me time(%)					
seq.\QP	28	32	36	40	44
akiyo	15.68	22.11	26.40	21.99	28.50
container	41.85	43.35	41.62	37.82	39.83
coastguard	52.70	51.99	51.13	49.46	46.87
football	47.36	47.79	47.36	45.15	42.61
mobile	49.27	48.83	48.92	48.09	47.87

Table 4.7 bits increment of all D1 format clips

Δbits(%)					
seq.\QP	28	32	36	40	44
akiyo	0.339	-0.465	0.862	0.235	0.440
container	0.035	0.077	-0.141	0.272	-1.079
coastguard	0.017	0.064	0.096	0.007	-0.415
football	0.287	0.242	0.126	0.340	0.268
mobile	0.137	0.251	0.200	0.372	0.339

Table 4.8 PSNR degradation of all HD format clips

ΔPSNR(dB)					
seq.\QP	28	32	36	40	44
mobcal	0	0.01	0.02	0.01	0.03
parkrun	0	0.01	0	0	-0.02
shields	0	0	0.01	0	0.01
stockholm	0	0	0	0	0.01

Table 4.9 motion estimation time saved of all HD format clips

Δme time(%)					
seq.\QP	28	32	36	40	44
mobcal	43.30	41.66	40.76	40.11	38.94
parkrun	45.10	44.66	44.46	43.97	43.17
shields	40.51	39.95	39.35	38.94	38.15
stockholm	38.86	37.13	36.61	36.67	36.50

Table 4.10 bits increment of all HD format clips

Δbits(%)					
seq.\QP	28	32	36	40	44
mobcal	0.206	0.248	0.269	0.272	0.215
parkrun	0.004	0.000	0.064	-0.022	-0.001
shields	0.008	0.119	0.083	-0.205	0.162
stockholm	0.061	0.169	-0.122	-0.646	0.771

The ME execution time is CPU occupation time of ME process. It contains the data loading time and computation time. The other complexity measure unit is search point counts. The search point counts exclude the data loading time. They reflect the real computation load for the usage of CPU. The search point results are listed at Tables 4.11~4.19. Tables 4.11, 4.14, 4.17 are the search point counts of the original FME algorithm for the three frame formats. Tables 4.12, 4.15, 4.18 are the search point counts of the proposed algorithm for three frame formats. Tables 4.13, 4.16, 4.19 are the percentage of the search point counts saved by proposed algorithm for the three frame formats. The percentage of the search points saved by proposed algorithm is better than the percentage of motion estimation time saved for all format. The search point counts can be considered as a more precise indicator to represent the actual computation loading.

Table 4.11 the search point count of the original VBSME for CIF format

clip\qp	28	32	36	40	44
akiyo	24579931	27174905	29646785	33054389	33300304
bus	50895213	50294033	50503794	51192035	50425643
container	39810571	41353513	43193423	45504705	46207518
coastguard	86316257	88136021	89408780	84677799	78210876
dancer	77810756	73211137	69652856	66619466	61921852
foreman	75714710	73606852	73551977	74053605	76689236
mobile	1.02E+08	1.02E+08	1.03E+08	1.03E+08	1.03E+08
stefan	1.01E+08	98627136	97341252	94444397	93666806

Table 4.12 the search point count of the proposed algorithm for CIF format

clip\qp	28	32	36	40	44
akiyo	15892992	16745185	17630384	18485618	18530538
bus	25913272	25810599	25952039	25941349	25120671
container	20997096	21710904	22545474	23544738	23762065
coastguard	43217457	43781612	44063793	41969050	38316320
dancer	36023645	34313212	32769153	31646937	30096468
foreman	39302324	38429091	38333612	38235450	38432205
mobile	50165945	49938036	50070438	49972409	49351950
stefan	49793313	49004757	48618251	47458440	46891287

Table 4.13 the percentage of the search point saved by proposed algorithm for CIF format

clip\QP	28	32	36	40	44
akiyo	35.34159	38.37997	40.53189	44.07515	44.35325
bus	49.08505	48.68059	48.61368	49.32542	50.18275
container	47.25749	47.49925	47.80346	48.25867	48.57533
coastguard	49.93127	50.32495	50.71648	50.43677	51.00896
dancer	53.70351	53.13116	52.95361	52.49596	51.39605
foreman	48.09156	47.79142	47.88228	48.36787	49.88579
mobile	50.83378	50.85826	51.29839	51.41219	52.00829
stefan	50.77362	50.31311	50.05381	49.74986	49.9382

Table 4.14 the search point count of the original VBSME for D1 format

clip\QP	28	32	36	40	44
akiyo	83079340	84286690	93311870	103118540	121195140
container	222406460	202736130	196019860	195132050	192414840
coastguard	577841830	588426749	580971922	539526711	472794942
football	687919154	680753189	671905556	599602180	508226435
mobile	572104821	564269992	567303604	564714883	558513153

Table 4.15 the search point count of the proposed algorithm for D1 format

clip\QP	28	32	36	40	44
akiyo	47180600	46643280	49019140	50145930	53942240
container	77855740	74666590	74070320	74300450	73846300
coastguard	185949742	187004029	185303581	174114464	155160244
football	235947975	227044732	219546846	199958524	176683877
mobile	199870967	197414243	197214650	195174739	191416750

Table 4.16 the percentage of the search point saved by proposed algorithm for D1 format

clip\QP	28	32	36	40	44
akiyo	43.210189	44.661156	47.467412	51.370597	55.491417
container	64.993939	63.170556	62.212849	61.92299	61.621307
coastguard	67.819958	68.219659	68.104555	67.728296	67.182338
football	65.701206	66.648011	67.324746	66.651468	65.235205
mobile	65.063925	65.014223	65.236489	65.438358	65.727441

Table 4.17 the search point count of the original VBSME for HD format

clip\QP	28	32	36	40	44
mobcal	2145182047	2144642469	2030339643	1919390013	1868048734
parkrun	2015963221	1935207386	1954974772	1988722670	1977699979
shields	1830692900	1817177498	1843539751	1856337737	1926092016
stockholm	1585841341	1452619488	1468439366	1504667930	1581782406

Table 4.18 the search point count of the proposed algorithm for HD format

clip\QP	28	32	36	40	44
mobcal	797849353	749403276	714531441	674360040	644933868
parkrun	670864757	650867855	653100119	654263284	649684212
shields	658396635	651588011	658381868	657971170	665347916
stockholm	596383656	561255906	562451741	563761632	570419070

Table 4.19 the percentage of the search point saved by proposed algorithm for HD format

clip\qp	28	32	36	40	44
mobcal	62.80738252	65.05695999	64.807295	64.8659191	65.47553304
parkrun	66.72237122	66.367023	66.5929132	67.10133123	67.14950605
shields	64.03565912	64.14285276	64.28708046	64.555417	65.45606801
stockholm	62.39323313	61.36249647	61.69731253	62.53248835	63.9382087

In [1], edge calculation and previous frame difference is an important indicator to classify what kinds of image the current frame is. These two factors must be calculated for each frame. The overhead to mode prediction is significant. The overhead is needed to consider as a computational complexity loading.

The edge vector:

$$\overrightarrow{D_{i,j}} = \{dx_{i,j}, dy_{i,j}\} = \{p_{i-1,j+1} + 2\times p_{i,j+1} + p_{i+1,j+1} - p_{i-1,j-1} - 2\times p_{i,j-1} - p_{i+1,j-1},\ p_{i+1,j-1} + 2\times p_{i+1,j} + p_{i+1,j+1} - p_{i-1,j-1} - 2\times p_{i-1,j} - p_{i-1,j+1}\}$$

This operator contains 10 times addition operation. N_{dx}, N_{dy} are operation counts of the edge vector. The total overhead operations can formulate as follow:

Edge detection: $M\times N\times(N_{dx}+N_{dy})+(M+N)$

Stationary region detection: $M\times N$

Edge operation: $N_{dx}=5$ $N_{dy}=5$

M x N is frame size dimension. All units are addition operator. N_{dx} and N_{dy} are number of edge calculating operation.

The overhead formula for each frame in proposed algorithm as follow:

Median of neighbor MVs: $3\times\frac{M\times N}{256}$

Search distance cap determination: $\frac{M\times N}{256}$

M x N is frame size dimension. All units are addition operator.

Table 4.20 overhead operation list (unit: add ops.)

algorithm\format	CIF	D1	HD
[1]	1115776	3802800	10139600
proposed	1584	5400	14400

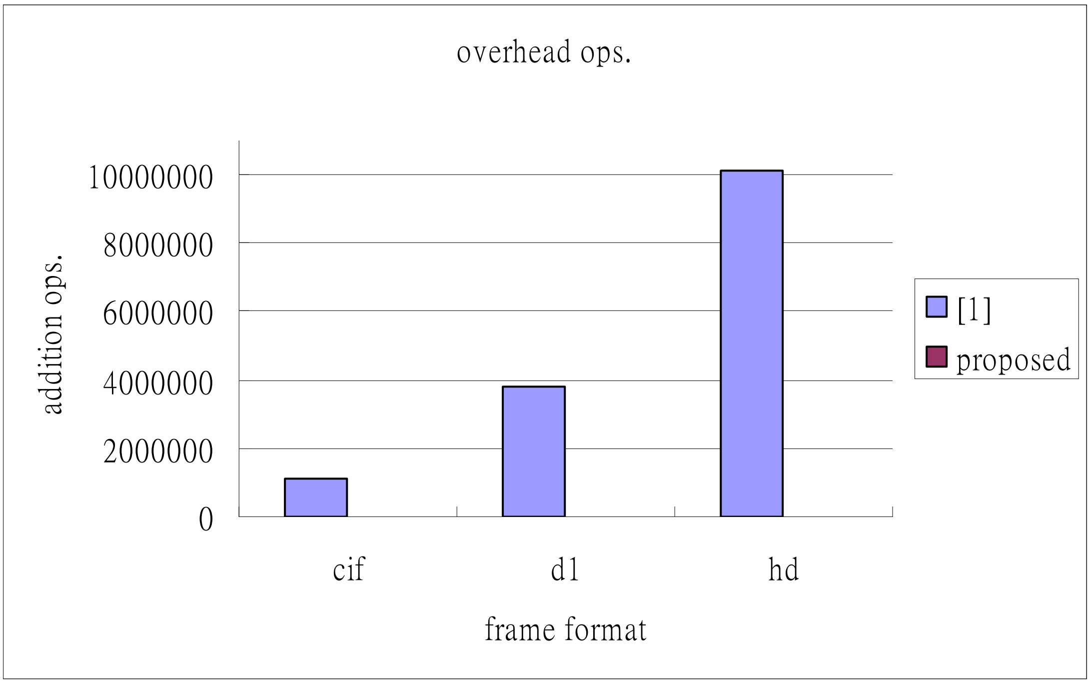

Figure 4.4 comparisons of overhead operation between proposed algorithm and ref. [1]

we can find the overhead of proposed algorithm is few. Under the same computational efficiency of motion estimation, proposed algorithm has smaller overhead calculation than [1].

We simulate our algorithm in H.264 software model JM 10.1 for FME. We can save 20% ~ 50% motion estimation time compared to the original fast motion estimation time with the close RD-curve performance to the original JM software for FME. Therefore, the proposed algorithm can effectively reduce the complexity of the VBSME by saving the unnecessary search outside the search distance cap.

Chapter 5 Conclusion

In h.264/avc, variable block size is the feature for better coding efficiency to previous coding standard. For searching the best mode of one macroblock, variable block size motion estimation does the 41 times search procedure for each macroblock. Therefore VBSME becomes the heaviest computational load part of an encoder. In order to reduce this loading, many methods to predict the most possible modes that one macroblock may be encoded are addressed to reduce the loading of VBSME. Unlike these proposed methods trying to predict the most possible modes for a macroblock before motion estimation, we propose a prediction-correction scheme to determine search distance cap adaptively for search procedure. This algorithm is based on the observation of the relationship of best mode result and its motion vector length distribution. The unnecessary search point is skipped if the location of the search point is outside the search distance cap. The algorithm we proposed can be easily implemented to fast motion estimation algorithm without any overhead calculation. All information for determining search distance cap already exists. From the experimental result, our algorithm can achieve the motion estimation execution time saving about 20-50% compared to original FME result under almost the same RD-curve performance. And our algorithm can merge existing mode-prediction algorithm to achieve the better result to ease VBSME complexity under the close RD-curve performance.

BIBLIOGRAPHY

[1] D. Wu, F. Pan, K. P. Lim, S. Wu, Z. G. Li, X. Lin, S. Rahardja, and C. C. Ko“, fast intermode decision in h.264/avc video coding”, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, VOL. 15, NO. 6, JULY 2005.

[2] Libo Yang, Keman Yu, Jiang Li, and Shipeng Li, “an effective variable block-size early termination algorithm for h.264 video coding”, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, VOL. 15, NO. 6, JUNE 2005.

[3] X. Jing and L.-P. Chau, “fast approach for h.264 inter mode decision”, ELECTRONICS LETTERS 19th August 2004 Vol. 40 No. 17

[4] Y.-H. Kim, J.-W. Yoo, S.-W. Lee, J. Shin,J. Paik and H.-K. Jung, “Adaptive mode decision for H.264 encoder”, ELECTRONICS LETTERS 16th September 2004 Vol. 40 No. 19.

[5] A. Ahmad, N. Khan, S. Masud and M.A. Maud, “Efficient block size selection in H.264 video coding standard”, ELECTRONICS LETTERS 8th January 2004 Vol. 40 No. 1.

[6] Gwo-Long LI, Mei-Juan CHEN, “Adaptive Search Range Decision and Early Termination for Multiple Reference Frame Motion Estimation for H.264”, IEICE TRANS. COMMUN., VOL.E89–B, NO.1 JANUARY 2006.

[7] Shu-Fa Lin', Che-Yu Chang', Che-Chun Sul, Yi-Lun Lin2, Chia-Ho Pan3, and Homer Chen, “Fast Multi-Frame Motion Estimation and Mode Decision for H.264 Encoders”, 2005 International Conference on Wireless Networks, Communications and Mobile Computing.

[8] Andy C. Yu, “Efficient block-size selection algorithm for inter-frame coding in h.264/mpeg-4 avc”, CASSP 2004.

[9] Jeyun Lee' and Byeungwoo Jean, “Fast mode decision for H.264”, 004 IEEE International Conference on Multimedia and Expo (ICME).

[10] “Draft ITU-T recommendation and final draft international standard of joint video specification (ITU-T rec. H.264/ISO/IEC 14496-10 AVC)”, Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T, JVTG050, 2003.

[11] Qionghai Dui ,Dongdong Zhu and Rong Ding, “Fast mode decision for inter prediction in h.264”, 2004 International Conference on Image Processing (ICIP)

[12] Zhang Fan Zhang Xudong, “Fast macroblock mode decision in h.264”, ICSP’O4 Proceedings

[13] LIAU Ming, QUANZi Yi, MENAi Dong, “Enhanced fast mode decision based on edge map And motion detail analysis for h.264/jvt”, IEEE Int. Workshop VLSl Design & Video Tech.

[14] Jianjun Guo, Kui Dai, Yun Cheng and Zhiying Wang, “Research on Fast Block Partition Mode Selection Algorithm in H.264”, Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science (ICIS’05)

[15] Xiang-Jun Zhu and Shan-An Zhu, “Fast Mode Decision and Reduction of Reference Frames for H.264 Encoder”, 2005 International Conference on Control and Automation (ICCA2005)

[16] Lidong Xut Xinggang Lin, “Fast Mode Decision for Inter Frames in JJ.264/AVC”, Proceedings of ISCIT2005

[17] Zhu Hong, Wu Cheng-ke, Wang Yang-li and Fang Yong, “Fast Mode Decision for H.264/AVC based on Macroblock Correlation”, Proceedings of the AINA’05

[18] Christos Grecos, Ming Yuan Yang, "Fast Inter Mode Prediction for P Slices in the H264 Video Coding Standard", IEEE TRANSACTIONS ON BROADCASTING, VOL. 51, NO. 2, JUNE 2005

[19] Bin Feng, Guang-xi Zhu, Wen-yu Liu, "Fast Adaptive Inter Mode Decision Method in H.264 Based on Spatial Correlation", Proceedings of the Seventh IEEE International Symposium on Multimedia (ISM'05)

[20] Li Wang, Qi Wang, Yu Liu, Wei Lu, "A fast Intra Mode Decision Algorithm for MPEG-2 to H.264 Video Transcoding", IEEE Consumer Electronics, June, 2006, pp. 1-5.

[21] Gao Shen, Lu Tie-jun, "Fast mode decision algorithm in H.264 for video communications", ASICON 2005. 6th International Conference, Oct, 2005, Volume 2, pp 1107-1111.

[22] Bin Feng, Guang-xi Zhu, Wen-yu Liu, "Fast adaptive inter-prediction mode decision method for H.264 based on spatial correlation", ISCAS Proceedings, May, 2006, pp 21-24.

[23] Shen Gao, Tiejun Lu, "An improved fast mode decision algorithm in H.264 for video communications", IEEE proceedings on Systems and Control in Aerospace and Astronautics, Jan, 2006, pp. 19-21.

[24] Dongming Zhang, Yanfei Shen, Shouxun Lin, Yongdong Zhang, "Fast inter frame encoding based on modes pre-decision in H.264", IEEE Multimedia and Expo, 2005, 2005, pp. 6-9.

[25] Libo Yang, Yu, K., Jiang Li, Shipeng Li, "An effective variable block-size early termination algorithm for H.264 video coding", IEEE transactions on Circuits and Systems for Video Technology, June 2005, Volume 15, Issue 6, pp. 784-788.

[26] Jianjun Guo, Kui Dai, Yun Cheng, Zhiying Wang, "Research on fast block partition mode selection algorithm in H.264", IEEE Computer and Information

Science, 2005, pp. 111 - 113

[27] Zhi-Yi Mai, Chun-Ling Yang, Sheng-Li Xie, "Improved best prediction mode(s) selection methods based on structural similarity in H.264 I-frame encoder", IEEE transaction on Systems, Man and Cybernetics, Volume 3, 10-12 Oct. 2005 pp. 2673 – 2678.

[28] Xiaoan Lu, Tourapis, A.M., Peng Yin, Boyce, J., "Fast mode decision and motion estimation for H.264 with a focus on MPEG-2/H.264 transcoding", IEEE transaction on Circuits and Systems, 2005, Volume 3, pp.2673-2678.

[29] Changsung Kim, Kuo, C.-C.J., "A feature-based approach to fast H.264 intra/inter mode decision", IEEE transaction on Circuits and Systems, 2005, Volume 1, pp. 308-311

[30] Hoi-Ming Wong, Au, O.C., Chang, A., Shu-Kei Yip, Chi-Wang Ho, "Fast mode decision and motion estimation for H.264 (FMDME)", IEEE proceedings on Circuits and Systems, 2006, May, pp. 21-24