國 立 交 通 大 學

電子工程學系 電子研究所

博 士 論 文

搜尋樣型之區塊移動估計研究：模型、演算法設計與視訊編碼應用

Pattern-based Block Motion Estimation: Modeling, Algorithm Design and Video Coding Applications

研 究 生： 蔡 彰 哲

指導教授： 杭 學 鳴

中華民國九十九年九月

搜尋樣型之區塊移動估計研究：
模型、演算法設計與視訊編碼應用

Pattern-based Block Motion Estimation:
Modeling, Algorithm Design and Video Coding Applications

研 究 生：蔡 彰 哲　　Student: Jang-Jer Tsai

指導教授：杭 學 鳴 博士　　Advisor: Dr. Hsueh-Ming Hang

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

& 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

September 2010

Hsinchu, Taiwan, Republic of China

中華民國九十九年九月

搜尋樣型之區塊移動估計研究：
模型、演算法設計與視訊編碼應用

研究生：蔡彰哲　　指導教授：杭學鳴 博士

國立交通大學 電子工程學系 電子研究所博士班

摘 要

基於搜尋樣型之區塊移動估計 (Pattern-based Block Motion Estimation, PBME) 演算法是現今視訊編碼系統最常採用的壓縮工具之一。儘管許多研究者已經探討過 PBME，卻甚少研究是關於「解釋 PBME 的工作原理與機制」之理論模型。

在這篇論文中，我們提出一個 PBME 的統計模型。這個模型包含兩個元件：1)移動向量 (Motion Vector) 的統計分布機率函數，2)一個搜尋演算法可以達到的最小搜尋點數函數，我們稱為「權重函數 (Weighting Function, WF)」。藉由檢視實驗資料，我們驗證此統計模型的正確性。然後我們展示兩個此模型的應用範例。由建立理想的 WF 中，我們設計基因式稜型搜尋演算法 (Genetic Rhombus Pattern Search, GRPS)。模擬結果顯示，與其他著名的搜尋演算法相比，GRPS 減少 20%的搜尋點數同時維持類似的壓縮影像 PSNR (Peak Signal-to-Noise Ratio) 品質。更進一步，我們提出的模型能可靠預測一個 PBME 演算法應用在一新影像序列上的運算複雜度。

我們將此模型應用在 PBME 的設計上，檢視一個典型 PBME 演算法中每個元件，然後系統化調整主要元件，來達成最佳或接近最佳的結果。首先我們使用解析模型來分析並設計基因式樣型搜尋 (Genetic Pattern Searches)。然後我們提出一個適應性搜尋樣型切換策略 (Adaptive Pattern Switching Strategy)，此一樣型切換策略會動態的在兩個樣型搜尋中切換。第三，我們延伸提出的 PBME 模型來評估起始搜尋點的效率，並漸進式建立一個接近最佳的起始搜尋點集合 (Starting Point Set)。第四，我們檢視早期終止方法 (Early Termination)，並建議一個選取有效門檻值的量度值。藉此，我們建立一個有精確門檻值的早期終止機制。

結合上述的技術，我們發展出一個完整的 PBME 演算，其效能超過許多現存的演算法。

儘管 WF 相當符合「確定性樣型搜尋方法 (Deterministic Pattern Search Scheme) 」，然而，WF 不能精確的預測「機率性樣型搜尋方法 (Probabilistic Pattern Search Scheme) 」 (如基因式樣型演算法 (Genetic Pattern Search))的效能。因此，我們提出「改良權重函數 (Refined Weighting Function, RWF) 」。在「單調象限函數與平滑象限邊界 (Quadrant Monotonic function with Smooth quadrant Border, QMSB) 的比對誤差表面 (Matching Error Surface) 」假設下，RWF 可以更精確描述基因式與非基因式樣型搜尋演算法。RWF 代表一個搜尋演算法在 QMSB 比對誤差平面中可以達到的平均搜尋點數函數。在建立 RWF 的過程中，我們學習如何更進一步加速樣型搜尋演算法，並設計兩個動量指引 (Momentum-directed) 基因式樣型搜尋演算法。此演算法給予每個可能的移動向量變異 (MV Mutation) 不同的優先權 (Priority) ，平均而言加速之前提出的基因式樣型搜尋演算法 5%到 7%。其中，不同移動向量變異的優先權是按照之前成功的搜尋來給定。

此外，我們檢視整個「可以預測樣型搜尋方法之平均搜尋點數」的改良模型。配合 RWF，改良模型的預測精確度隨之提升。因此，我們重新檢視適應性樣型搜尋演算法中的編碼工具。我們特別專注在兩個編碼工具的影響：樣型切換策略與起始點選擇。我們研究這些工具中的最佳參數選擇，以及其對整體效能的影響。實驗結果顯示，我們改良的搜尋樣型切換策略可以再加速搜尋流程，並且將視覺品質保持到跟其構成的樣型搜尋方法相同。

總體而言，這篇論文建立一個 PBME 的分析模型，並且示範如何使用此一模型來建立樣型搜尋演算法與適應性樣型搜尋方法。我們並改良此模型，增加其精確性，也依此設計更好的快速搜尋演算法。

Pattern-based Block Motion Estimation: Modeling, Algorithm Design and Video Coding Applications

Student: Jang-Jer Tsai Advisor: Dr. Hsueh-Ming Hang

Department of Electronics Engineering and Institute of Electronics
National Chiao Tung University

Abstract

Pattern-based block motion estimation (PBME) is one of the most widely adopted compression tools in the contemporary video coding systems. However, despite that many researchers have studied PBME, few have attempted to construct an analytical model that can explain the underneath principle and mechanism of various PBME algorithms.

In this dissertation, we propose a statistical PBME model that consists of two components: 1) the statistical probability distribution of the motion vectors, and 2) the minimal number of search points (called *weighting function,* WF) achieved by a search algorithm. We verify the accuracy of the proposed model by checking the experimental data. Then, two application examples using this model are proposed. Starting from an ideal weighting function, we devise a novel genetic rhombus pattern search (GRPS) to match the design target. Simulations show that comparing to the other popular search algorithms, GRPS reduces the average search points for more than 20% and, in the meanwhile, it maintains a similar level of coded image peak signal-to-noise ratio (PSNR) quality. Furthermore, the proposed model can reliably predict the performance of a PBME algorithm applied to a new video sequence.

With the aid of the proposed model, we design new PBMEs by looking into every component of a typical PBME algorithm and fine-tuning the major components systematically to achieve the optimal or nearly optimal results. First, we use the aforementioned analytic model in analyzing and designing effective genetic-algorithm-based pattern searches. Then, we propose an adaptive switching strategy that dynamically switches between two pattern searches. Third, we extend our PBME model to evaluate the efficiency of starting (initial search) points. A near

optimal set of starting points is identified through iterative steps. Fourth, we study the early termination threshold technique and suggest a metric in selecting an effective threshold. An early termination mechanism with accurate threshold is thus constructed. Combining all these techniques, we develop a PBME algorithm that outperforms many existing algorithms.

Although the WF matches the deterministic search schemes quite well, however, the WF fails to give a precise search point prediction when a probabilistic search method such as a genetic pattern search is involved. Therefore, we propose a *refined weighting function* (RWF) that describes both genetic and non-genetic pattern searches more accurately under the assumption that the matching error surface is a *quadrant monotonic function with smooth quadrant border* (QMSB). In the process of constructing RWF, we further accelerate the pattern searches and two momentum-directed genetic pattern search algorithms are devised. These new algorithms assign priorities to the candidate mutations based on the information provided by the preceding successful searches and this can further reduce the computational complexity of the previously proposed genetic pattern searches by 5% to 7% in average.

With refined RWF, the prediction accuracy of the refined model is significantly improved. Consequently, we re-examine the coding tools in the adaptive pattern search scheme. We focus on two components, the pattern switching strategy and the starting point selection. We investigate the optimal parameter selection issue in these tools and their impacts on the overall coding performance. Experimental results show that our refined pattern switching schemes can further accelerate the search process and in the meanwhile keep the visual quality comparable to the best of their constituent pattern searches.

In summary, we propose an analytical model for PBME and demonstrate a methodology for developing new pattern-based search algorithms and the adaptive pattern search schemes by using our proposed model. One step further, we refine the original model, improve its accuracy and then design better fast search algorithms accordingly.

Dedication

To My Father in Heaven

Acknowledgement

This dissertation is the result of more than six years of work. During these exciting years, I received the help and support from numerous people, without whom this dissertation would have never been begun and completed.

First and foremost I want to thank my advisor, Prof. Hsueh-Ming Hang, for supporting me over the years, and for giving me so much freedom to explore and discover new areas of video coding. He has taught me, both consciously and unconsciously, what a good research should achieve, and how a good scholar should act. It has been an honor to be under his supervision.

I like to thank my former bosses and colleagues in Pixart Imaging Inc. and Computer and Communication Laboratory (CCL), Industrial Technology Research Institute (ITRI). I had the pleasure of working together with them over the past 10 years. Special thanks are due to President Sen-Huang Huang and Director Tsi-Yi Chao. Without their assistance and kindness, I would never ever join the Ph.D. program, let alone finishing it. During my days in ITRI, Director Chih-Yuan Liu sent me to several exhibitions and conferences, domestic and abroad. These experiences broaden my view and I am deeply appreciated. I am especially grateful to Mr. Chih-Hsin Lin, Dr. Hsin-Chia Chen, Prof. Shou-Te Wei, Mr. Wen-Hsin Chuang, Mr. Chien-Hsing Hsieh, Mr. Chuan-Ching Lin, Mr. Chun-Neng Wang, Mr. Shih-Wei Kuo, Mr. Wen-Cheng Liao, Mr. Chia-Lin Wang, Director Cheng-Kuang Sun, Mr. Ching-Chin Huang, Ms. Ming-Chu Chen, Ms. Shu-Fang Huang, Mr. Yuan-Chu Tai, and Ms. Tzu-Fang Li for their help. I bear in mind how they aided me through the tough times in these years.

The members of our laboratory have contributed immensely to my personal and academic time in my Ph.D pursuit. The group has been a good source of knowledge as well as collaboration and friendship. I thank Dr. Kun-Chien Hung, Dr. Chia-Yang Tsai, Mr. Chao-Hsiung Hung, Ms. Hai-Wei Wang, Mr. Ssu-Hsien Wu, Mr. Cheng-Wei Chou, Mr. Chung-Hao Wu, Mr. Chao-Hsuan Li, Mr. Chun-Yen Ko, Mr. Shu-Wei Teng, Mr. Fu-Kai Yang, Ms. Yu-Ting Weng and Ms. Su-Min Chu. I particularly thank Dr. Hung-Chih Lin for his tutoring on the qualification exam subjects and quite a few courses.

Before the beginning of my Ph.D program, I benefited very much from Prof. Kai-Kuang Ma, Dr. Yao Nie and Prof. Kenneth Barner. I like to thank them for generously offering me advices, which consolidate my knowledge in this area. In the developing of this dissertation, Prof. Thomas Wiegand, Prof. Oscar Au and many anonymous reviewers have provided me valuable comments and constructive feedbacks regarding the submitted journal and conference manuscripts related to this work. I like to acknowledge their time and efforts, which make this dissertation robust and

solid. Moreover, I like to thank all professors in my dissertation committee, Prof. Pao-Chi Chang, Prof. Ja-Ling Wu, Prof. Homer H. Chen, Prof. Jia-Shung Wang, Prof. Sheng-Jyh Wang, Prof. Wen-Nung Lie, and Prof. Yong-Sheng Chen. They offer me invaluable questions, comments and suggestions to this thesis and my future researches. I appreciate their insightful advices very much.

Prof. Han-Ching Wu, Mr. Jung-Tang Lin, Ms. Mei-Hui Chen, Ms. Hsiu-Ying Wang, Ms. Hsiu-Lien Hsiao and the members in C.A. Club always share their life experience with me and take good care of me. I am more than grateful to them.

I further like to express special thanks to my classmates and long-term friends, Mr. Hsin-Chieh Chuang, Prof. Chin-Lung Yang, Dr. Wen-Chang Yeh, Dr. Han-Ting Lu, and Prof. Wen-Hsiao Peng. Their joy and enthusiasm they have for their own careers and life are contagious and motivational. I enjoy the days we hang out together. Mr. Yueh-Heng Tu, Mr. Sheng-Tai Liao, Mr. Shun-Pin Yang, Mr. Po-Chun Fan and Mr. Chao-Ming Wu constantly share their perspectives and insights with me. They open a window for me from time to time when I feel at a dead end.

At last, I would like to thank my family for their love, consideration and help. I owe a great deal to my parents and my grandmother. They raise me up, provide me a good education and stand by me unconditionally. Besides I gratefully thank my brothers and sisters-in-law for taking care of the family affairs. They are “the winds beneath my wings”, which carry me forward. In particular, I am indebted to Chia-Chia for putting up with my days in developing these works and for supporting me to pursuit my Ph.D. degree. Thank you all.

Jang-Jer Tsai,
National Chiao-Tung University,
September 2010

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

The explicit use of motion compensation to improve video compression efficiency can be traced back to the late 1960s, a patent filed by Haskell and Limb [1] and a conference paper by Rocca [2], both in the year of 1969 [3]. The necessary operation in the video encoder to enable motion compensation in the video decoder is motion estimation (ME). A technique of motion estimation, block motion estimation (BME), has been widely adopted by the contemporary video coding systems [4][5][6][7] because it is an effective means in reducing the inter-frame correlation for image sequence coding. Block motion estimation schemes partition the current frame into non-overlapping blocks and find the block with the minimal block-matching cost in the reference frame. The most straightforward implementation of BME, the so-called full search (FS), evaluates the matching costs of every motion vector candidate in the search area and finds the motion vector (MV) of the best-matched block. Yet, it requires a huge amount of computing power particularly for sophisticated coding algorithms that include multiple reference frames and variable size block motion estimations. Since 1980s, the pattern-based block motion estimation (PBME) algorithms [17][18][44] have been developed to alleviate the computational burden and to minimize the impact on the coding quality. Static PBME [26][27][28][29][31][32], which use fixed search patterns, got popular in late 1990s and early 2000s. However, because the characteristics of an image sequence vary with time, no one single search pattern can handle the entire sequence well. In 2000s, the adaptive pattern search algorithms [39][40][43], which dynamically switch search patterns, were devised. In this dissertation, we focus only on PBME and its variations.

Section 1.1 Motivation

Despite that many fast algorithms have been proposed to reduce the computational complexity of PBME, most of them are devised based on experimental data or heuristic ideas. Few researchers,

to our knowledge, have tried to construct an accurate mathematical model for the PBME process. Therefore, one purpose of this dissertation is to construct an analytical model for PBME, and the other purpose is to design new fast algorithms by using the proposed model.

To be specific, we like to propose a model that reveals the relationship among the video sequences, the search methods, and the computational complexity. Essentially, we want to propose answers to the following questions: Why does one pattern search outperform the other? What is the underlying mechanism of its search efficiency? Is there a pattern search that handles nearly all sequences well? If not, how can we adaptively choose the proper pattern searches? What is the impact of starting points on the coding performance? Is it possible to portray all these problems by using one single model?

Section 1.2 Research Contributions

Table 1-1 The contributions of this dissertation and the related publications

Theory	Application	Publication	
		Journal	Conference
MV PDF*		T.CSVT09[54]*	
WF*	Genetic pattern searches*	T.CSVT09[54]*	ISCAS07[51]
Complete Model*	Performance prediction*	T.CSVT09[54]*	
1st Method*	Selection of the initial search point set+	T.CSVT10[55]+	
2nd Method*	Adaptive pattern switching strategy+	T.CSVT10[55]+	ICASSP07[52]
	Optimization of early termination mechanism+	T.CSVT10[55]+	
RWF~	Momentum-directed genetic pattern searches~	Submitted[56]~	ICIP08[53]
Refined Complete Model~	Improved prediction accuracy~	Submitted[56]~	
1st Method~	Influences on the selection of starting point set#	Submitted[57]#	
2nd Method~	Influences on the pattern switching strategy#	Submitted[57]#	

Table 1-1 shows the contributions of this dissertation and the related publications. Items with the

same superscript marker are published in the same journal manuscript. Items in the same row are included in the same conference paper. In the theory part, we build a statistical probability distribution function of the motion vectors (MV PDF) [54], and the minimal number of search points achieved by a pattern-based search algorithm, the so called *weighting function* (WF) [51]. Combining them together, we propose a statistical model for PBME (The Complete Model) [54]. There are two methods to train the parameters in the model. We further replace the WF by the *refined weighting function* (RWF) [53], which better describes the behavior of the probabilistic PBME. Thus, we have the refined model (Refined Complete Model) [56], which similarly has two training methods to acquire its parameters. In the application part, we devise the genetic pattern searches [51][54] based on our observation of WF. With the complete model, we are able to predict the performance of a search algorithm on an image sequence [54]. Based on the 1st method, we construct the high-performance initial search point set [55]. Based on the 2nd method, we select properly a good threshold for pattern switching mechanism [52][55]. Another application is the optimization of the early termination mechanism. Combining all these tools together, we construct the adaptive pattern search schemes [55]. In addition, hinted by the shape of RWF, we further design the momentum-directed genetic pattern searches [53][56]. With the refined model, the prediction accuracy is improved. Accordingly, we re-examine its influence on the selection of starting point set and the pattern switching strategy [57].

Section 1.3 Dissertation Organization

The rest of this dissertation is organized as follows. First, we review the development of PBME and describe several popular pattern-based search algorithms in Chapter 2. Then, we introduce the model for PBME and present two of its applications – pattern search design and performance prediction - in Chapter 3. Chapter 4 extends the applications of the model to the design of adaptive pattern search schemes, which include three major coding tools – pattern switching

strategy, starting point selection, and early termination. In Chapter 5, we further refine our proposed model and re-examine all the coding tools in a PBME. Finally, we conclude this dissertation by summarizing our contributions and pointing out some possible future works in Chapter 6.

Chapter 2 Pattern-based Block Motion Estimation

Modern video compression systems convert the huge digitized video data into a small-size sophisticated bitstream by using the well-known structure – block-based hybrid coding (BHC) [4][5]. A BHC scheme divides an image frame into blocks, reduces the inter-frame dependence among image frames by ME, removes the intra-frame redundancy by intra prediction, discrete cosine transform and entropy coding techniques, and packs the image essential information into a comprehensive representation. In general, a BHC video system comprises two major modules: intra frame coding and inter frame coding. Block Motion Estimation (BME), a ME technique, has been widely adopted by modern video coding standards [4], such as the H.26X series [5] and MPEG-1/2/4 [7].

Although many algorithms have been developed to accelerate BME, however BME remains the most computation-intensive component in the video encoders. As the coding algorithm progresses, the more sophisticated ME tools are invented, such as variable block size ME and multiple reference frames. The most intuitive BME algorithm is FS, which examines all the candidates (checking points) in the search area by calculating the block matching cost between the current block and the reference block and find the motion vector (MV) with the smallest block matching cost. Because FS consumes a tremendous amount of computing power, many researchers have devised fast BME schemes to reduce computation without sacrificing the coding efficiency. According to [8], fast BME algorithms can be classified mainly into two categories: 1) reducing the number of checking (search) points and 2) lowering the computational complexity in calculating the block-matching criterion for each checking (search) point. This dissertation focuses on the algorithms in the first category.

PBME is the most popular scheme in the first fast BME category. It typically consists of three sets of tools for reducing the search points: 1) an operative threshold for early decision

mechanisms [9][14][30][31], 2) the selection of good initial search points [14][15][16][30][31], and 3) an effective set of search patterns [17][26][27] [28][29][31][32][33]. Combing all these tools, the latest PBME algorithms achieve a dramatic speed-up in finding the near-optimal candidate motion vectors while maintaining a desired level of quality. The first set and the second set of speed-up tools make use of the data correlation inside one frame (intra-frame) or between nearby frames (inter–frame). The third set of tools (search patterns) is effective when the matching cost surface is nearly monotonic. Among these tools, the search pattern plays a key role in deciding the performance of a search algorithm especially when the data correlation is low.

Four step search [26], diamond search [27][28], hexagonal search [29] and their improved versions [30][31][32][33][34], have been the most popular and effective methods in fast PBME. However, since the contents of an image sequence vary quite drastically along with time, one single search pattern often can not handle well the diverse characteristics of the entire sequence. Thus, the adaptive PBME [35][36][37][38][39][40][41][42][43], which mainly comprise multiple pattern searches and a pattern switching mechanism, have been proposed. Note that, the overall performance of an adaptive pattern search algorithm is still bounded by its constituent pattern searches.

Section 2.1 Initial Search Point

The initial search point of the PBME has crucial effect on the search point performance as reported in [14][15][16]. Conventionally, the most common used initial search point is zero motion vector (ZMV, (2.1)) and predicted motion vector (PMV, (2.2)). Herein, we adopt the prediction formula specified by the MPEG-4 standard for PMV. Unless explicitly stated otherwise, we use PMV as the predetermined initial search point for the conventional pattern searches for two reasons. First, the motion vector differences between the best MV and PMV is coded in the bitstream on our simulation platform – a MPEG-4 SP@L3 encoder. That is, when the best MV is

PMV, it takes 0 bits to code the best MV. Clearly, the simulation platform favors PMV because it consumes the least bits. Second, PMV is likely to be the MV with the smallest block-matching error in statistics. Thus, if we choose PMV as the initial search point, the resulting number of search point acquired by a typical PBME is small. Further discussions on the best initial search points are in Section 4.3.

$$ZMV = (0,0). \tag{2.1}$$

$$PMV = Median(MV^{U}, MV^{L}, MV^{UR}), \tag{2.2}$$

where the location of MV^{U}/MV^{L}/MV^{UR} and PMV are illustrated by **Fig. 2-1**. MV^{U} is the adjacent upper block of the current block, MV^{L} is the adjacent left block of the current block, and MV^{UR} is the neighboring up-right block of the current block.

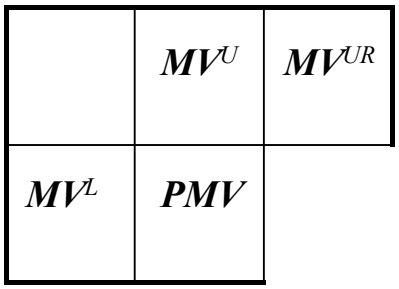


Fig. 2-1 Predicted motion vector.

Section 2.2 Some Popular Pattern-based Search Algorithms

Four representative pattern-based search methods, Four Step Search (FSS) [26], Diamond Search (DS) [27][28], Enhanced Hexagonal Search (EHS) [32], and Easy Rhombus Pattern Search (ERPS), are used to illustrate the construction of the PBME model. These pattern-based search algorithms are chosen because of their well-recognized performance. Among the existing PBME algorithms, EHS performs rather well particularly on high motion sequences, and ERPS is more suitable for low motion sequences.

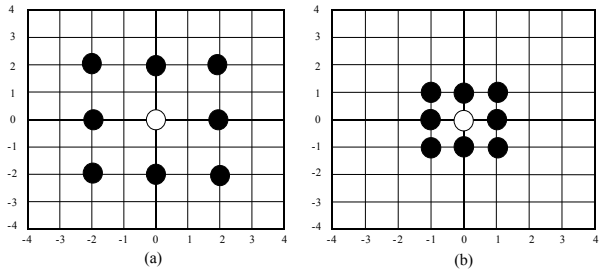


Fig. 2-2 Search patterns used in FSS.

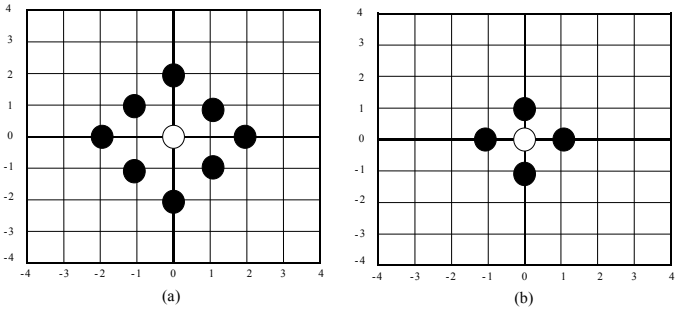


Fig. 2-3 Search patterns used in DS.

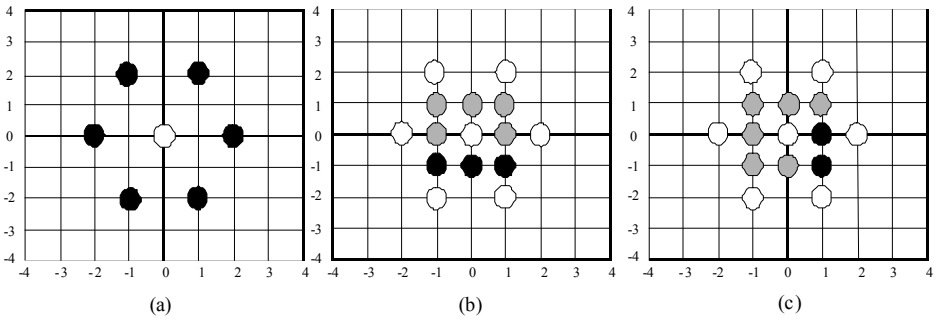


Fig. 2-4 Search patterns used in EHS.

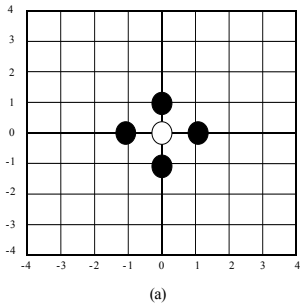


Fig. 2-5 Search patterns used in ERPS.

FSS and DS both consist of two specific search patterns, as shown in **Fig. 2-2** and **Fig. 2-3**, respectively. The large search pattern, **Fig. 2-2**(a) or **Fig. 2-3**(a), is used for the coarse regular searches, while the small search pattern, **Fig. 2-2**(b) or **Fig. 2-3**(b), is used for fine ending search. Their procedures can be summarized as follows.

Step 1) Check the predetermined starting point, PMV, in the predefined search window, as well as the points in the large pattern, which centers at the predetermined starting point, PMV. If the minimal block distortion (MBD) point is found to be at the center of the large search pattern, proceed to Step 3; otherwise, proceed to Step 2.

Step 2) Set the MBD point in the previous search step as the center, and a new large pattern is formed. New search points generated by the new large pattern are checked if they were not examined in the previous large pattern. Thus, the new MBD point is again identified. If the MBD point is the center point of the latest large search pattern, go to Step 3; otherwise, repeat this step continuously.

Step 3) Switch the search pattern from the large pattern to the small one. The points covered by the small pattern are evaluated to compare with the current MBD point. The new MBD point is the final motion vector.

Instead of using one single small ending search pattern, EHS uses two small ending search patterns as well as one large coarse regular search pattern as shown in **Fig. 2-4**. Its algorithm is the same as the one described above except for step 3. EHS switches the large hexagonal search pattern to one of the partial square patterns. The pattern in **Fig. 2-4**(b) is used when the smallest block distortion sum of two neighboring points in the previous-searched hexagonal pattern is in the vertical direction and the pattern in **Fig. 2-4**(c) is used otherwise. The two or three points covered by the newly formed partial square pattern are evaluated to compare with the current MBD point, and the new MBD point is the final motion vector.

Unlike other algorithms mentioned above, ERPS uses only one rhombus search pattern in **Fig. 2-5**, for both coarse search and fine ending search. This particular rhombus pattern is also known as “small diamond” in [27][28] and “cross pattern” in [19]. ERPS is a simplified version of

adaptive rood pattern search (ARPS[31]). It is ARPS without initial rood patterns as well as various motion vector predictors; PMV is the sole starting search point. The algorithm of ERPS is as follows.

Step 1) Check the predetermined starting point, PMV, in the predefined search window, as well as the points in the rhombus pattern, which centers at the predetermined starting point. If the MBD point is found to be at the center of the rhombus pattern, the MBD point is the final motion vector; otherwise, proceed to Step 2.

Step 2) Set the MBD point in the previous search step as the center, and a new rhombus pattern is formed. Three or two new candidate points are checked, and the MBD point is again identified. If the new MBD point is the center point of the latest rhombus pattern, the new MBD point is the final motion vector; otherwise, repeat this step continuously.

Chapter 3 Modeling of Pattern-based Block Motion Estimation

Many researches have proposed fast PBME to reduce the computational requirement of the highly computation-intensive BME. However, few researchers, to our knowledge, have tried to construct an accurate model for the PBME process. To be specific, it is a model that unveils the relationship among the video sequences, the search methods, and the computational complexity. Our aim is to construct an explicit mathematical model for PBME.

Recent research works on PBME often collect the statistics of motion vectors and design good search patterns based on experiences. Few papers are able to provide a systematic way in modeling and designing the search pattern. Among the existing search patterns, the rhombus patterns are known quite effective for low motion sequences [19][20][31], and the hexagonal patterns are very powerful for high motion sequences [29][32][33]. Combining these two sets of search patterns, [21] uses rhombus pattern for initial searches and switches to hexagonal pattern for the succeeding regular searches. One step further, [39] and [40] select the search patterns adaptively according to a set of criteria. Typically these papers use only the experimental data to show the effectiveness of the corresponding search patterns. In this chapter, we like to further explore the following problems. Why does one search pattern outperform the other? What is the underlying mechanism behind it? Is there a search pattern that handles nearly all sequences well? Moreover, can we construct a mathematical model that describes the underlying mechanism? An attempt is made in this chapter to answer these questions.

In this study, we are going to construct a simple and yet effective statistical model for PBME. Using this statistical model, we can predict the performance of one search algorithm when it is applied to a test sequence. Also, based on this model, a novel genetic PBME algorithm is devised.

The rest of this chapter is organized as follows. Section 3.1 presents the probability

distribution functions of the motion vectors acquired by FS. In Section 3.2, we analyze the search points of several representative PBME algorithms and formulate the *weighting functions* (WF, first mentioned in Section 1.2). Based on the proposed probability distribution function for motion vectors and the WFs of different PBME algorithms, Section 3.3 builds a statistical model for PBME. To demonstrate the use of this model, a new genetic rhombus pattern search is presented in Section 3.4, which shows good performance for both low motion and relative high motion sequences. Section 3.5 describes the second example of using our model: predicting the performance of applying a specific search algorithm to a specific video sequence. Lastly, we summarize this chapter by Section 3.6.

Section 3.1 Probability Distribution of Motion Vectors

In order to design a good search pattern set, many papers discussed the nature of motion vectors. [19], [20] and [21] empirically gather the statistics of the motion vectors around the initial search point and develop their search algorithms. [43] assumes that the motion vector distribution can be approximated by either Gaussian or Laplace probability distributions. So far, we have not found an attempt of inventing a probability distribution function (PDF) that provides a quite precise match to the motion vectors.

We select a few representative training sequences at various bit rates under the settings given in **Table 3-1** for generating motion vectors. The selected sequences are encoded by a MPEG-4 SP@L3 encoder using FS. All the sequences are in CIF (352X288) format. Only the first frame is coded as I frame, and all the remaining frames are coded as P frames. The motion vector search range is set to 16, the initial quantizer step size is set to 15, and the block size is set to 16x16. When the quantization step varies to achieve the desired bit rate, the PSNR quality of the coded video sequence ranges from 26dB (poor but acceptable) to 40dB (visually the same as original).

Table 3-1 The selected training sequences and their settings.

Abbreviation	Sequence	Bit rate (K bps)	Frame rate (fps)	Number of frames	PSNR
CT256	container	256	7.5	300	39
CT40	container	40	7.5	300	32
HL40	hall	40	7.5	300	33
MD96	mother and daughter	96	10	300	40
CG112	coastguard	112	30	300	29
FM512	foreman	512	30	300	34
FM1024	foreman	1024	30	300	36
FB1024	football	1024	30	90	35
FG768	flower garden	768	30	250	26
ST1024	Steven	1024	30	300	29

3.1.1 Motion Vector Distributions

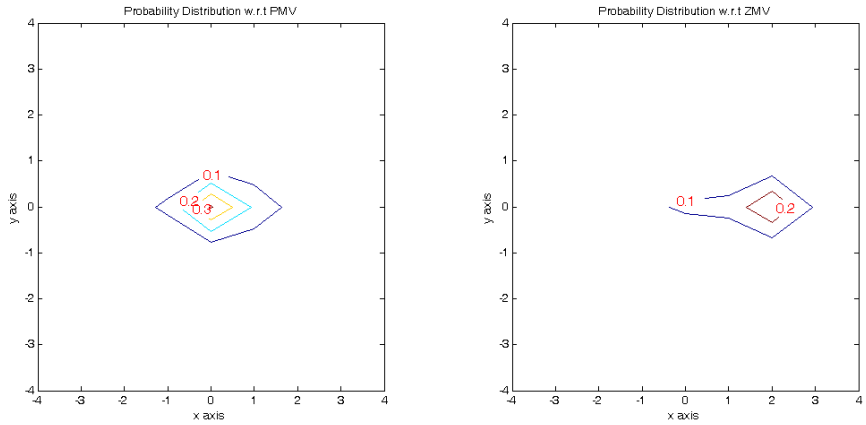


Fig. 3-1 Contour plots of the motion vector probability distribution of video sequence CG112 (partial).

In our experiments, we test two kinds of initial motion vectors (origins of PBME search), namely, ZMV and PMV. **Fig. 3-1** shows the probability distributions of the motion vectors obtained by applying FS with a search region [-16~+15, -16~+15] to a video sequence, coast guard at 112K (CG112). Only the probability distribution in region [-4~+4, -4~+4] is shown. The left plot is the motion vector probability distribution with respect to (w.r.t.) PMV, and the right one is the motion vector probability distribution w.r.t. ZMV. Herein, ZMV is defined by (2.1), PMV is defined by (2.2), and the label on the contour shows the probability of motion vectors.

From the motion vector distributions obtained by applying FS to a video sequence, **Fig. 3-1** for example, the motion vector distributions with respect to (w.r.t.) PMV generally have a more symmetric shape as compared to the motion vector distributions w.r.t. ZMV. In addition, the PMV-based motion vectors have a smaller standard deviation. They cluster better. Therefore, the motion vector distribution w.r.t. PMV is thus used in the rest of this chapter.

The statistics of the motion vectors w.r.t. PMV of all the selected training sequences show that the horizontal mean values (μ_x) and vertical mean values (μ_y) both are close to zero. Thus, these motion vector distributions are zero-biased w.r.t. PMV. For a particular sequence, the variance of the horizontal motion vectors (σ^2_x) is often larger than that of the vertical motion vectors (σ^2_y). Furthermore, the correlations (ρ_{xy}) between the horizontal components and the vertical components of motion vectors are nearly zero for all our training sequences in **Table 3-1**.

3.1.2 Normalized Independent 2D Distribution

Based on the above observations, three popular zero-mean normalized independent 2D distributions are considered as candidates for modeling MV distribution: 1) Gaussian distribution function, 2) Laplace distribution function and 3) Cauchy distribution function.

A Gaussian probability distribution with mean μ and variance σ^2 is shown by (3.1); a Laplace probability distribution with mean μ and variance $2b^2$ is shown by (3.2); and a Cauchy probability distribution with median m and the full width at half maximum Γ can be expressed by (3.3), where $x \in (-\infty, +\infty)$.

$$G_{1D}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{(x-\mu)^2}{2\sigma^2}} \tag{3.1}$$

$$L_{1D}(x) = \frac{1}{2b} e^{\frac{|x-\mu|}{b}} \tag{3.2}$$

$$C_{1D}(x) = \frac{1}{\pi} \frac{\Gamma/2}{(x-m)^2 + (\Gamma/2)^2} \tag{3.3}$$

Because the correlations between the horizontal components of motion vectors, x, and the vertical components of motion vectors, y, are almost zero, it is reasonable to assume that these two random variables in the motion vectors, x and y, are independent. The independent 2D Gaussian probability distribution can be defined as (3.4), where $x \in (-\infty,+\infty)$ and $y \in (-\infty,+\infty)$.

$$G_{independent\_2D}(x,y) = \frac{1}{\sqrt{2\pi\sigma_x^{\,2}}} e^{-\frac{(x-\mu_x)^2}{2\sigma_x^{\,2}}} \frac{1}{\sqrt{2\pi\sigma_y^{\,2}}} e^{-\frac{(y-\mu_y)^2}{2\sigma_y^{\,2}}} \tag{3.4}$$

Furthermore, since the mean values of motion vectors are nearly (0,0), one may set $\begin{cases}(\mu_x,\mu_y)=(0,0)\\(\sigma_x^{\,2},\sigma_y^{\,2})=(\lambda_x,\lambda_y)\end{cases}$. Thus, (3.4) becomes (3.5).

$$G_{2D}(x,y) = \frac{1}{\sqrt{2\pi\lambda_x}} e^{-\frac{x^2}{2\lambda_x}} \cdot \frac{1}{\sqrt{2\pi\lambda_y}} e^{-\frac{y^2}{2\lambda_y}} \tag{3.5}$$

Although these probability distributions are defined in the domains of $x \in (-\infty,+\infty)$ and $y \in (-\infty,+\infty)$, the actual distributions of motion vectors are confined in a search area, A. Consequently, we need to normalize the probability distributions as shown by (3.6). Then, the sum of probabilities in the search area would equal 1.

$$G(x,y) = \frac{G_{2D}(x,y)}{\sum_{(x',y')\in A} G_{2D}(x',y')} \tag{3.6}$$

Consequently, the zero-mean normalized independent 2D Gaussian Distribution G(x,y) is defined by (3.7). Similarly, the zero-mean normalized independent 2D Laplace Distribution L(x,y) is defined by (3.8), and the zero-mean normalized independent 2D Cauchy Distribution C(x,y) is defined by (3.9). Note that, $(x,y) \in A$, and A is the geographical area of [-32~+31, -32~+31] in our experiments.

$$G\ (x,y) = \frac{e^{-\frac{x^2}{2\lambda_x}} \cdot e^{-\frac{y^2}{2\lambda_y}}}{\sum_{(x',y')\in A} e^{-\frac{x'^2}{2\lambda_x}} \cdot e^{-\frac{y'^2}{2\lambda_y}}} \tag{3.7}$$

$$L(x,y)=\frac{e^{-\frac{|x|}{b_x}}e^{-\frac{|y|}{b_y}}}{\sum_{(x',y')\in A}e^{-\frac{|x'|}{b_x}}e^{-\frac{|y'|}{b_y}}} \tag{3.8}$$

$$C(x,y)=\frac{\frac{1}{x^2+\eta_x^2}\frac{1}{y^2+\eta_y^2}}{\sum_{(x',y')\in A}\frac{1}{x'^2+\eta_x^2}\frac{1}{y'^2+\eta_y^2}} \tag{3.9}$$

Remark: Strictly speaking, the zero correlation between the horizontal components and vertical components of motion vectors does not imply that they are statistically independent. However, we will justify the correctness of these probabilistic models using the goodness-of-fit test [10][11] as follows.

To find out which of the three PDFs best approximates the PDF of motion vectors acquired by FS, a well-known *goodness-of-fit test*, 2D KS test [12][13], is adopted. The *statistic D* defined in [13] is used as the measure of similarity between the hypothesized PDF and the observed PDF (data). To be more specific, the *statistic D* is the maximum absolute difference between two cumulative probability distributions. The smaller *statistic D* is, the better the hypothesized PDF matches the observed PDF.

In the 2D KS test, the motion vector probability distributions acquired by FS, PDF_{FS}, are tested against the hypothesized zero-mean normalized independent 2D Gaussian (3.7), Laplace (3.8) and Cauchy distributions (3.9) with the same variances. Therefore, we need to adjust the parameter values of these three distributions so that the variances of the hypothesized distributions equal those of PDF_{FS}. Those fitted hypothesized zero-mean normalized independent 2D Gaussian, Laplace and Cauchy distributions are called $G_{FS}(x,y)$, $L_{FS}(x,y)$ and $C_{FS}(x,y)$, respectively.

Table 3-2 shows the 2D KS test results between $PDF_{FS}(x,y)$ and the three hypothesized distributions. Clearly the normalized independent 2D Cauchy distribution $C_{FS}(x,y)$ generally has the smallest *statistic D* values. However, according to [13], the values of *statistic D* in our

experiments are so large that it is improper to claim that any of these three 2D distributions has a good match to the target $PDF_{FS}(x,y)$.

Table 3-2 *Statistic D* of 2D KS test.

Sequences	$G_{FS}(x,y)$	$L_{FS}(x,y)$	$C_{FS}(x,y)$
CT256	0.48	0.38	0.08
CT40	0.42	0.35	0.14
HL40	0.36	0.30	0.08
MD96	0.38	0.32	0.12
CG112	0.41	0.32	0.12
FM512	0.39	0.30	0.07
FM1024	0.40	0.30	0.06
FB1024	0.28	0.23	0.19
FG768	0.40	0.32	0.12
ST1024	0.39	0.33	0.17
Average	0.39	0.32	0.11

3.1.3 A Fitted Probability Distribution

To further reduce the difference between $C_{FS}(x,y)$ and $PDF_{FS}(x,y)$, we extend C(x,y) and propose a new form of PDF denoted by T(x,y), which is defined by (3.10). For each of the selected training sequences, τ_x and τ_y are optimized such that the maximum discrepancy between $PDF_{FS}(x,y)$ and T(x,y) is minimized, and ξ_x and ξ_y are adjusted such that the variances of T(x,y) are the same as those of the training sequences. T(x,y) with those fitted parameters matching the $PFD_{FS}(x,y)$ becomes $T_{FS}(x,y)$. The fitted parameters and their corresponding 2D KS test results are shown in **Table 3-3**. One may note that τ_x and τ_y vary from 1.13 to 2.2. This indicates the variations among the test sequences are considerably large.

$$T(x,y)=\frac{\dfrac{1}{|x|^{\tau_x}+\xi_x}\dfrac{1}{|y|^{\tau_y}+\xi_y}}{\sum_{(x',y')\in A}\dfrac{1}{|x|^{\tau_x}+\xi_x}\dfrac{1}{|y|^{\tau_y}+\xi_y}} \tag{3.10}$$

Despite the large individual differences among the training sequences, **Table 3-3** shows that

τ_x and τ_y generally are around 1.67. We thus choose $\begin{cases}(\tau_x,\tau_y)=(\frac{5}{3},\frac{5}{3})\\(\xi_x,\xi_y)=(\zeta_x,\zeta_y)\end{cases}$ to simplify T(x,y). The resultant distribution is called S(x,y) as defined by (3.11).

$$S(x,y)=\frac{\frac{1}{|x|^{5/3}+\zeta_x}\frac{1}{|y|^{5/3}+\zeta_y}}{\sum_{(x',y')\in A}\frac{1}{|x|^{5/3}+\zeta_x}\frac{1}{|y|^{5/3}+\zeta_y}} \tag{3.11}$$

Table 3-3 Parameters of $T_{FS}(x,y)$ for the training sequences and their corresponding KS test results.

T(X,Y)	CT256	CT40	HL40	MD96	CG112	FM512	FM1024	FB1024	FG768	ST1024
xi_x(ξ_x)	0.01	0.04	0.13	0.11	0.01	0.25	0.21	0.12	0.09	0.10
xi_y (ξ_y)	0.03	0.13	0.11	0.10	0.55	0.24	0.20	0.43	0.40	0.19
tau_x (τ_x)	1.84	1.70	1.73	1.58	1.46	1.94	1.99	1.13	1.75	1.79
tau_y (τ_y)	1.50	1.54	1.72	1.83	2.21	1.82	1.88	1.18	1.98	1.31
max_pdf_diff	0.01	0.01	0.02	0.01	0.03	0.03	0.03	0.02	0.08	0.01
Statistic D	0.01	0.05	0.06	0.05	0.07	0.05	0.05	0.14	0.12	0.03

The 2D KS test shows that $S_{FS}(x,y)$ on the average has a smaller statistic D in comparison with $G_{FS}(x,y)$, $L_{FS}(x,y)$, and $C_{FS}(x,y)$. Note that the parameters (ζ_x, ζ_y) of $S_{FS}(x,y)$ are obtained by numerical methods so that the variances of $S_{FS}(x,y)$ match the data statistics. In summary, after several attempts, we found that $S_{FS}(x,y)$ is a rather good model to describe the probability distribution of the motion vectors derived by using FS. It constitutes the first element of our complete model.

Section 3.2 Search Points in Pattern-based Search Algorithms

Search patterns are generally devised based on the assumption that the matching cost surface is a unimodal one; in other words, the distortion associated with a search point closer to the global

minimum has a smaller value. Under this assumption, the number of search points is defined as the minimal number of search points in all possible paths leading to the best-matched point from the starting (initial) point. The search point number in this definition depends on the search pattern. And, for a given search pattern, it is determined by the shortest path between the initial point and the best-matched point. Therefore, it is a discrete function of the location and will be called *weighting function*. By examining the search process of a PBME, we can construct its corresponding WF.

Note that the global uni-modal cost surface assumption is so strong that it is not always true for most video sequences [32]. Typically it is valid within a small neighborhood of the global minimal point. Consequently, the WF does not represent the actual number of search points. Indeed, it represents the lower bound of the number of search points. But the statistics also show that the number of actual search points is highly correlated with our defined WF.

3.2.1 Weighting Function of Pattern-based Search Algorithms

By examining the search algorithms in Section 2.2, we can construct their WFs. Fig. 3-2 shows two examples of the FSS search process. Fig. 3-2(a) is the case of the minimum search point. There are only two steps. The initial coarse search examines 9 points and the fine ending search examines 8 points. The initial point happens to be the best-matched point. Fig. 3-2(b) shows a typical search process. In addition to the 9 initial search points and the 8 ending search points, FSS checks 3 new points if it moves horizontally or vertically, and it checks 5 new points if it moves diagonally. In the WF of FSS (3.12), '9' represents the coarse initial search points, and '8' represents the fine ending search points. The parameter $M_{FSS}(x,y)$ is '5' if it moves diagonally and '3' otherwise. And $n_{FSS}(x,y)$ denotes the number of movements.

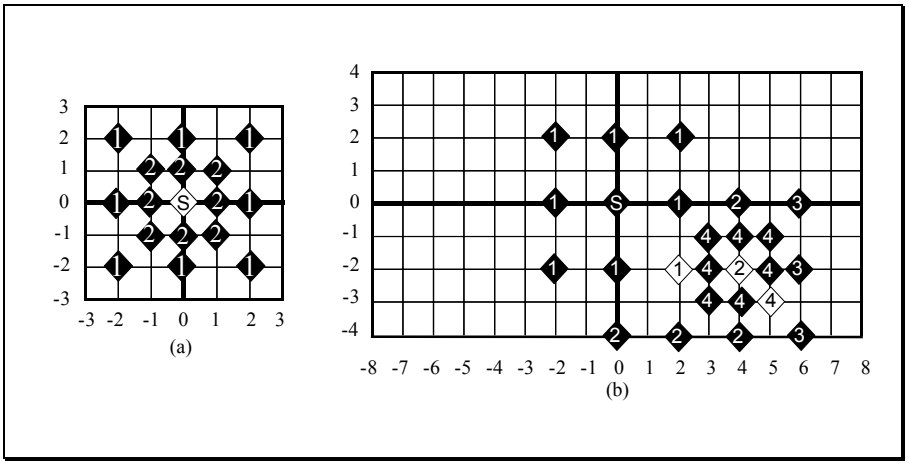


Fig. 3-2 Examples of FSS process.

$$WF_{FSS}(x,y) = 9 + M_{FSS}(x,y) \times n_{FSS}(x,y) + 8 \quad (3.12)$$

$$WF_{DS}(x,y) = 9 + M_{DS}(x,y) \times n_{DS}(x,y) + 4 \quad (3.13)$$

$$WF_{EHS}(x,y) = 7 + 3 \times n_{EHS}(x,y) + K_{EHS}(x,y) \quad (3.14)$$

$$WF_{ERPS}(x,y) = 1 + M_{ERPS} \times n_{ERPS}(x,y) + 4 \quad (3.15)$$

Likewise, by examining the search steps of the other three PBME algorithms, the WFs of DS, EHS, and ERPS can also be formulated by (3.13), (3.14), and (3.15), respectively. In Eqs (3.13) to (3.15), $M_{DS}(x,y)$ is either 5 or 3, $K_{EHS}(x,y)$ is either 3 or 2, M_{ERPS} is either 3 or 2, all depending on the search direction. And the value of $n_{DS}(x,y)$. $n_{EHS}(x,y)$, and $n_{ERPS}(x,y)$ are decided by the number of movements.

From (3.12) to (3.15), it is clear that in the minimum case ERPS checks only 5 points, while FSS checks 17 points, DS checks 13 points and EHS checks 9 points. Thus, ERPS has the smallest number of search points among the four algorithms for the minimum cases. The minimum cases refer to the situations that the best-matched motion vector is located at the starting point.

Fig. 3-3 shows the contour plots of the WFs of FSS, DS, EHS, and ERPS, respectively. The value on a contour represents the least number of search points for a search algorithm to move from the origin to a point (location) on the contour. Because EHS moves faster than any other

algorithms, EHS surpasses the other algorithms at distant locations. Therefore, by looking into the WF of a search algorithm, we understand why it works better for a particular situation (fast motion or slow motion). Use ERPS as an example: because $WF_{ERPS}(x,y)$ has the smallest values around the starting point, it has advantages for slow motion situations. On the other extreme, $WF_{EHS}(x,y)$ has the smallest values at distant locations. WF forms the second element of our complete model.

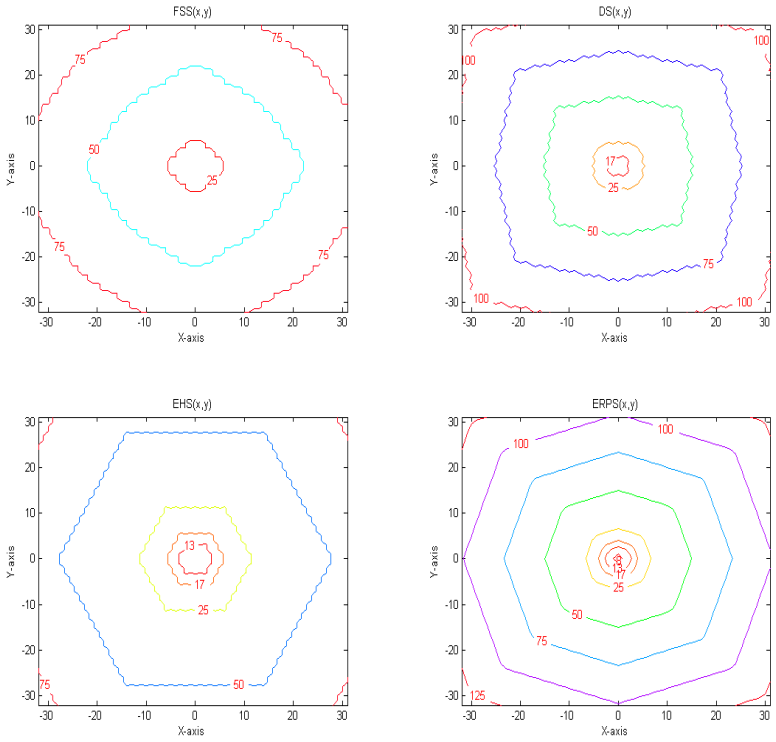


Fig. 3-3 Contour plots of the WFs of FSS, DS, EHS, and ERPS, respectively.

Section 3.3 Statistical Model for Pattern-based Block Motion Estimation

Based on the problem formulation in Section 3.1 and Section 3.2, the total average search points (ASP) for a sequence can be represented by (3.16). It depends on both search algorithm (SA) and

the video sequence. It is the sum of the products of the number of search points and the motion vector probability distributions at all locations within the search area, where $SPF_{SA}(x,y)$ denotes the number of search points, $PDF_{SA}(x,y)$ denotes the motion vector distribution acquired by a specific algorithm, and A is the search area.

$$ASP = \sum_{(x,y)\in A} PDF_{SA}(x,y) \times SPF_{SA}(x,y) \tag{3.16}$$

When we apply a specific algorithm to a specific sequence, we obtain the ASP directly from the experiments without the need of calculating (3.16), which requires the knowledge of $PDF_{SA}(x,y)$ and $SPF_{SA}(x,y)$. Our goal is to construct a generic model in which the dependency on SA and video sequence is separable. In other words, the PDF is sequence dependent but not SA dependent. And the search point function is SA dependent but not sequence dependent. That is, we would like to replace $PDF_{SA}(x,y)$ by $PDF_{FS}(x,y)$, and $SPF_{SA}(x,y)$ by $WF_{SA}(x,y)$ in (3.16). Thus, (3.16) becomes (3.17). Herein, $PDF_{FS}(x,y)$ denotes the PDF of the motion vector acquired by FS, and $WF_{SA}(x,y)$ denotes the *weighting function* of a specific algorithm discussed previously. With (3.17), we can thus predict ASP before actually applying a search algorithm to a video sequence, as long as we know the motion vector PDF acquired by FS and the WF of a specific SA.

$$ASP = \sum_{(x,y)\in A} PDF_{FS}(x,y) \times WF_{SA}(x,y) \tag{3.17}$$

However, (3.17) differs from (3.16) due to a few reasons. First, because the block-matching cost surface typically is not globally monotonic in the search area, the actual search process from time to time does not take the shortest path to the location of the best-matched motion vector. Thus, the average number of actual search points, $SPF_{SA}(x,y)$, is higher than $WF_{SA}(x,y)$, the shortest-path (minimal) search points. Second, the motion vectors found by a specific algorithm sometimes differ from the ones found by FS. Consequently, the motion vector PDF of this specific algorithm, $PDF_{SA}(x,y)$, is not the identical to that of the full search, $PDF_{FS}(x,y)$.

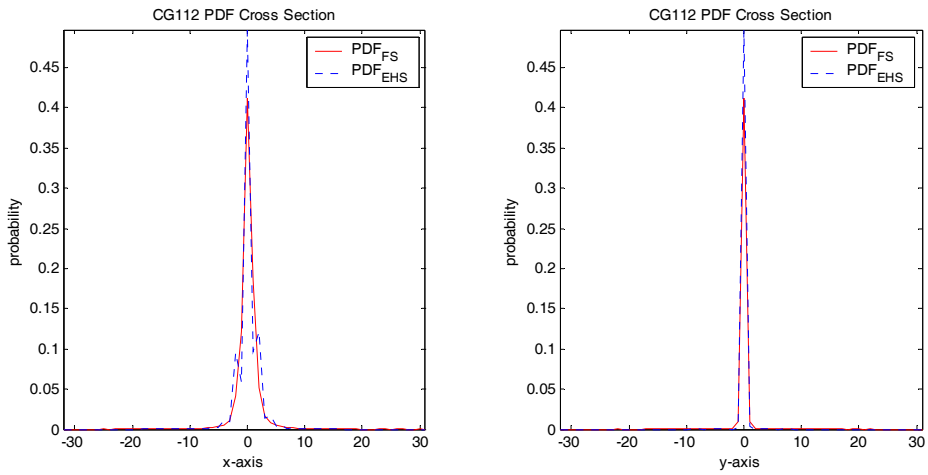


Fig. 3-4 PDF shift between PDF acquired by FS and that acquired by EHS (CG112).

Fig. 3-4 shows the cross sections of PDFs acquired by FS and those acquired by EHS for the video sequence, CG112. It is clear that these two PDFs are not identical. PDF shift refers to the phenomenon that $PDF_{FS}(x,y)$ differs from $PDF_{SA}(x,y)$. The main causes are: 1) the search pattern is relatively small, thus the search is trapped by a local optimal; 2) the early decision mechanism terminates the search when a near-optimal solution is found; and 3) the starting point of SA disagrees to that of FS, PMV, in our formulation.

Fig. 3-5 shows the theoretical WF and the empirical SPF obtained by applying EHS to the video sequence, FB1024, in the region [-10~+10, -10~+10]. Herein, on the left plot (the theoretical WF), the value on a contour represents the shortest-path search points for EHS to move from the origin to a point (location) on the contour and, on the right plot (the empirical SPF), it represents the average number of actual search points. For the empirical SPF, the contour is not continuous, because some motion vectors never happen when we apply a SA to a specific sequence. Thus there indeed exists some differences between the theoretical WF and the empirical SPF and therefore we called it WF drift. It happens because the search algorithm does not always follow the shortest path in the search process as discussed earlier.

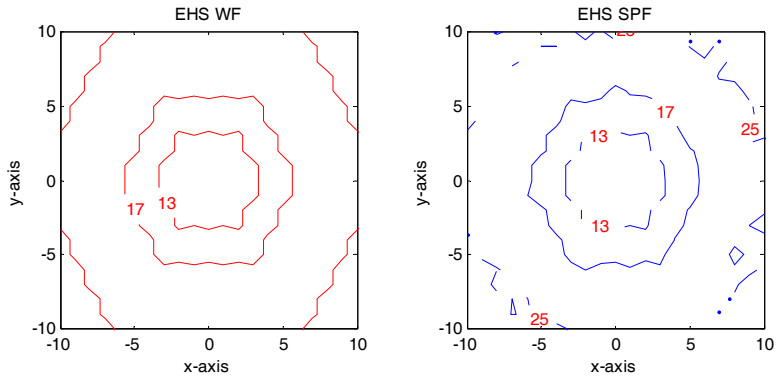


Fig. 3-5 The contour plots of the theoretical WF and the empirical SPF by applying EHS to FB1024 (partial).

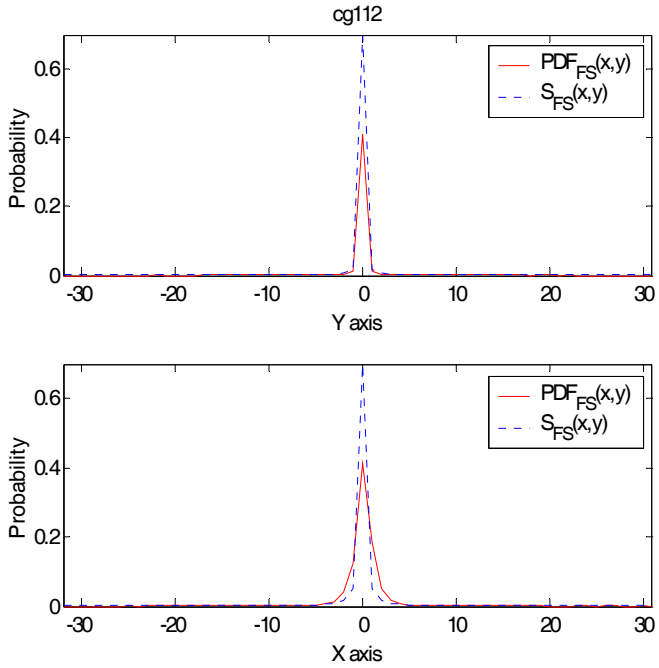


Fig. 3-6 PDF differences between $PDF_{FS}(x,y)$ and $S_{FS}(x,y)$ of CG112.

Moreover, Section 3.1 suggests that the distribution $S(x,y)$ best approximates $PDF_{FS}(x,y)$. We can thus substitute $S(x,y)$ for $PDF_{FS}(x,y)$ in (3.17) as long as its variances are known. Thus, $S(x,y)$ becomes $S_{FS}(x,y)$, the $S(x,y)$ that matches the motion vectors acquired by FS. However, the substitution of $S_{FS}(x,y)$ also induces new PDF matching error. **Fig. 3-6** shows the PDF differences between $S_{FS}(x,y)$ and $PDF_{FS}(x,y)$ of the video sequence, CG112.

$$ASP = C_1 \times \sum_{x,y \in A} S_{FS}(x,y) \times WF_{SA}(x,y) + C_2 \tag{3.18}$$

Therefore, Eq.(3.17) needs adjustment to compensate for various shifts, drifts and model errors. Eq.(3.18) is a modified formula for modeling ASP. Two additional terms, C_1 and C_2, are included in Eq.(3.18). We propose that ASP is a linear function of the sum of the products of $S_{FS}(x,y)$ and $WF_{SA}(x,y)$. By tuning the values of C_1 and C_2, we can reduce the WF drift error, the PDF shift error and the PDF mismatch error. Consequently, with the pre-analysis of $WF_{SA}(x,y)$ for a specific SA and pre-calculation of $S_{FS}(x,y)$ for a specific sequence, one may use Eq.(3.18) to estimate the ASP values of another SA when it is applied to this specific sequence.

We need to justify the above model is valid for real data. There are two methods to decide C_1 and C_2. In the first method, we apply a fixed SA to a set of training sequences to compute C_1 and C_2 by the regression method. Our aim is that the model with trained C_1 and C_2 can predict the ASP of a new sequence accurately. In the second method, we apply a few search algorithms (the training algorithms) to a specific sequence, and then calculate C_1 and C_2 based on the acquired data. In this case the goal is that the model with trained C_1 and C_2 can predict the ASP values of a new algorithm.

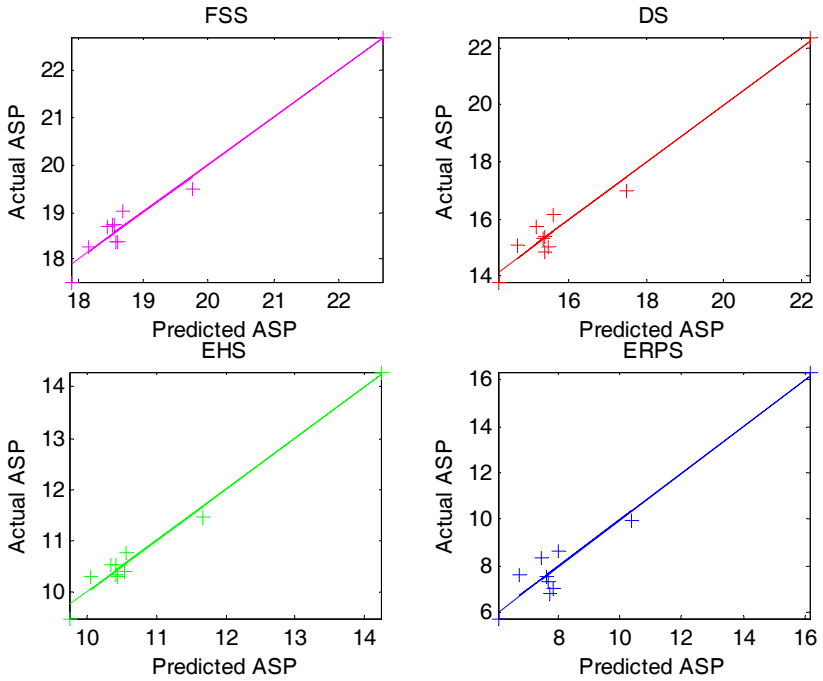


Fig. 3-7 The actual ASP and the predicted ASP pairs for 4 popular search algorithms (1st method)

In the first method, C_1 and C_2 are acquired from a set of training sequences with one specific search algorithm. **Fig. 3-7** shows the pairs of the actual ASP and predicted ASP of various sequences for the four popular search algorithms. Each training sequence is represented by a plus-sign mark, the solid line represents the case that the predicted ASP is exactly the same as the actual ASP. The X-axis represents the predicted ASP and the Y-axis represents the actual ASP. **Table 3-4** displays the C_1 and C_2 values for each search algorithm. The last column is the correlation coefficient between the actual ASP and the predicted ASP. One may notice that the correlation coefficients for all algorithms are very close to 1, which means that the predicted ASPs are nearly the same as the actual ASPs.

Table 3-4 Regression parameters (C_1 and C_2) and the correlation coefficients between the model-predicted ASP and the real data. (1st method).

BME	C_1	C_2	ASP correlation
FSS	0.42	10.38	0.98
DS	0.46	7.59	0.98
EHS	0.42	5.63	0.99
ERPS	0.44	2.97	0.98

In the second method, C_1 and C_2 can be acquired by applying a set of search algorithms (training algorithms) to a specific sequence. We then predict the ASP value of a new algorithm by using the proposed model. **Fig. 3-8** shows the actual ASP versus the predicted ASP pairs for 10 sequences. Each training algorithm is represented by a cross-sign mark, the dash line shows the case that the predicted ASP is exactly the same as the actual ASP, and the X-axis represents the predicted ASP and the Y-axis represents the actual ASP. **Table 3-5** displays the C_1 and C_2 values for the 10 sequences and the correlation coefficients between the predicted ASP and the actual ASP. The correlation coefficients are very close to 1 for all sequences except for FB1024, which has a value of 0.73. This may be due to the high motion nature of FB1024. In spite of the small number of training algorithms, the coherence between the predicted ASP and the actual ASP is very high for all 10 sequences.

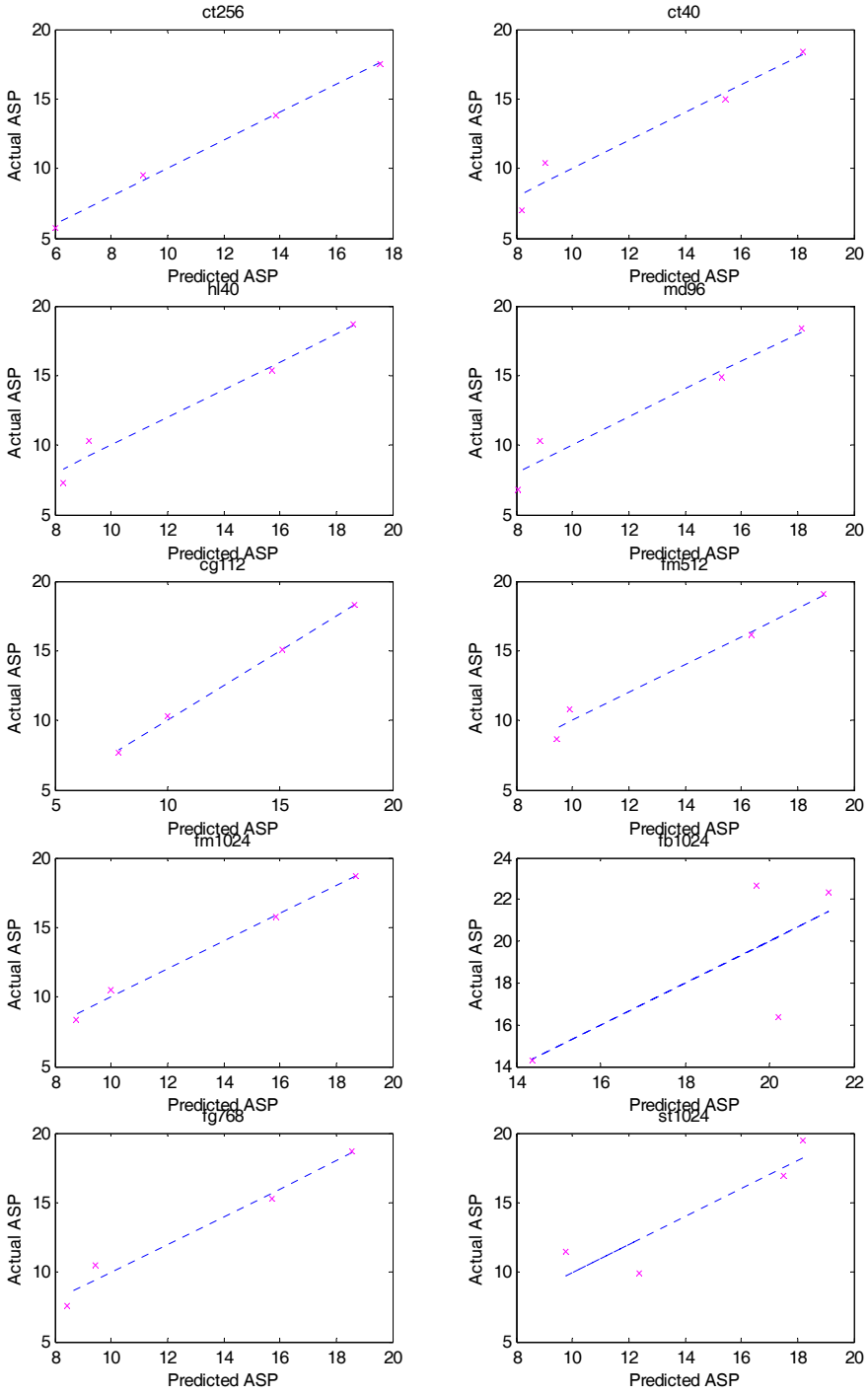


Fig. 3-8 The actual ASP and predicted ASP pairs for 10 training sequences (2nd method).

Table 3-5 Regression parameters (C_1 and C_2) and the correlation coefficients between model-predicted ASP and the actual data (2nd method).

Sequence	C_1	C_2	ASP correlation
CT256	1.07	-1.42	1.00
CT40	1.17	-4.70	0.98
HL40	1.19	-4.35	0.99
MD96	1.17	-4.52	0.97
CG112	1.05	-1.05	1.00
FM512	1.15	-3.60	0.99
FM1024	1.10	-2.36	1.00
FB1024	0.62	1.66	0.73
FG768	1.15	-3.76	0.98
ST1024	1.08	-5.82	0.91

The first method and the second method are designed for different scenarios. The first method is used to predict the ASP of a new sequence (for a given specific search algorithm), while the second method is used to predict the ASP of a new search algorithm (for a given specific sequence). Due to different sizes of training samples and purposes, the accuracy comparison between these two methods may not be meaningful.

In the following two sections, we will show how this model, (3.18), can be used to inspire the design of a new search algorithm as well as it can be used to predict the search performance of a new video sequence or a new search algorithm.

Section 3.4 Application I: Pattern-based Search Algorithm Design

How can we devise a new pattern-based search algorithm with the help of the previous analysis? We do this in three steps. We first construct a target WF based on the analysis in the past two sections. Then, we devise a search pattern that hopefully achieves the desired WF. At last, we evaluate its performance by simulation on real pictures.

The first step of designing a new search algorithm is to find a WF that has the smallest possible values at all locations, because, in our proposed model, ASP depends on WF and the smaller the WF is, the faster the corresponding search algorithm is.

Most effective PBME algorithms typically consist of two stages: 1) coarse regular search stage and 2) fine ending search stage. The purpose of the regular search stage is to fast locate the potential optimal motion vectors, and the ending stage is to determine the best-matched point in a small neighborhood. Each stage may use one or several search patterns. In the regular search stage, because the shortest path between two points in a plane is the strait line, the fastest search path for a search algorithm is the strait line from the starting point directly to the best-matched motion vector. Based on the previous experiments, we suspect that a doable search method moves at most one unit distance horizontally or vertically per step. As shown in **Fig. 3-9**(a), the minimal number of search points for reaching the motion vector (x,y) is '*abs(x)+abs(y)+1*'. In the ending stage, to decide precisely the location of the best candidate motion vector generally requires to search at least the neighboring 4 points and the current point (center) itself, as shown in **Fig. 3-9**(b). Consequently, the minimal number of search points for motion vector (x,y) is '*Max(5, 4+abs(x)+abs(y))*'. Thus, the ideal WF (x,y) is expressed as (3.19) and its contour plot is depicted in **Fig. 3-11**.

$$WF_{GRPS}(x,y) = \mathrm{Max}(5, 4 + \mathrm{abs(x)} + \mathrm{abs(y)}) \tag{3.19}$$

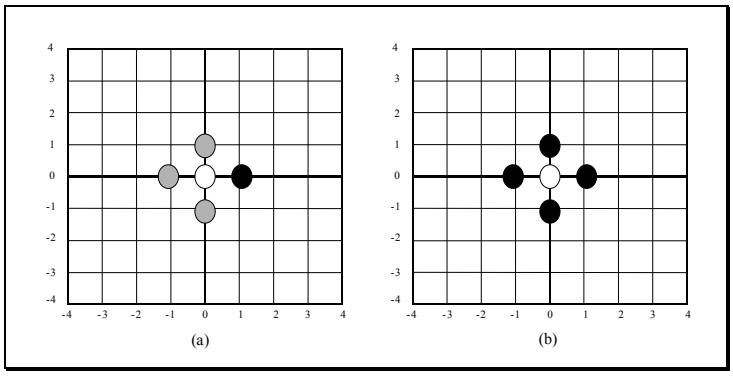


Fig. 3-9 Search patterns for GRPS.

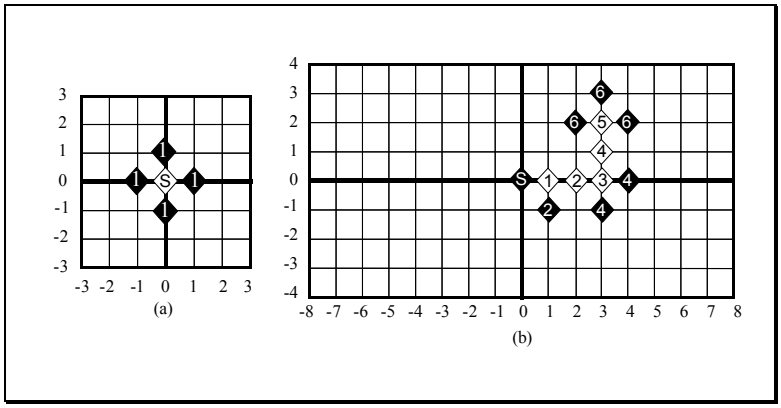


Fig. 3-10 Examples of GRPS search process.

The second step is to choose proper search patterns that fulfill the desired WF. By simplifying the genetic search algorithms in [22], [23] and [24] as well as combining the rhombus search pattern, we propose a genetic rhombus pattern search (GRPS) to match the WF in (3.19). The algorithm is described below.

1. Initial:
   Check the starting point, PMV, and set it as the parent point.
2. Mutation:
   Select randomly a next generation point (the mutation point) from the untested points of a rhombus pattern centered at the parent. That is, check one (black dot, for example) of the four solid points (black and gray dots) in the coarse search pattern in **Fig. 3-9**(a).
3. Competition:
   Select the survivor between the parent and its mutation based on their matching costs.
   a. If the mutation is better than the parent, the mutation is the survivor (the next parent). Go to step 2.
   b. If the parent is better than its mutation, the parent is the survivor (the next parent) and check if there is any remaining untested mutation point in the four points of a rhombus pattern. If there is one, go to step 2; otherwise, (that is, all points in the ending search pattern, **Fig. 3-9**(b), are checked,) go to step 4.
4. End:
   Set the current survivor as the final motion vector.

Note that, we do not adopt the complete genetic algorithm here. Only the concept of genetic optimization is used in the proposed algorithm.

Fig. 3-10(a) shows the fastest search case of GRPS, and **Fig. 3-10**(b) shows a typical search case of GRPS. From **Fig. 3-10**, we conclude that the WF of the proposed GRPS (3.19) can be expressed by (3.20), where M_{GRPS} is 1, 2, 3 or 4, depending on whether the mutation is successful

and the $n_{GRPS}(x,y)$ value depends on the number of movements. Comparing **Fig. 3-11**, $WF_{GRPS}(x,y)$ contour plot, with the previously discussed known algorithms, we can find that GRPS has the same or smaller number of search points as ERPS near the starting point, and it has a smaller number of search points than EHS in locations away from the starting point. In other words, it achieves the smallest number of search points at nearly all locations, compared to the four popular search algorithms.

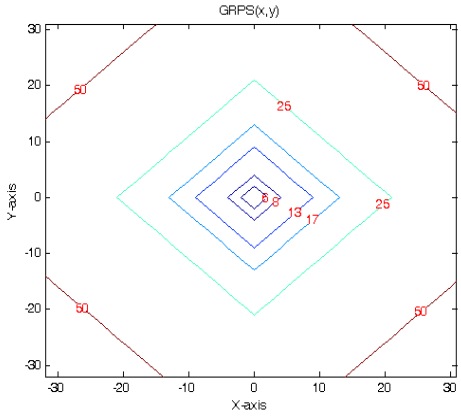


Fig. 3-11 WF of GRPS.

$$WF_{GRPS}(x,y) = 1 + M_{GRPS} \times n_{GRPS}(x,y) \tag{3.20}$$

In the last step of designing a new search algorithm, we evaluate the performances of the proposed GRPS by conducting experiments on the training sequences. The results are shown in **Table 3-6** (Average number of search points), **Table 3-7**(Peak signal noise ratio), and **Table 3-8** (Performance comparison). It is compared with FS and the four representative search algorithms described in Section 2.2. In **Table 3-8**, the computing gain (CG) is defined as the ratio of ASP minus one, and the quality gain (QG) is defined as the PSNR difference. In summary, the ASP of GRPS on average is 22% faster than that of ERPS, 56% faster than EHS, 130% faster than DS, 172% faster than FSS, and 145 times faster than FS. On the other hand, the PSNR of GRPS is on average better than all other search algorithms, except for ERPS. Compared with ERPS, the quality loss of GRPS is very small, around 0.01dB. Therefore, GRPS outperforms all the other search algorithms in terms of ASP for all training sequences, and its coding quality is comparable

with all the other algorithms.

Table 3-6 ASP (Average number of search points).

ASP	GRPS	ERPS	EHS	DS	FSS	FS
CT256	5.36	5.75	9.59	13.81	17.53	1024.00
CT40	5.98	7.04	10.42	15.03	18.38	1024.00
HL40	6.35	7.33	10.34	15.38	18.72	1024.00
MD96	5.98	6.83	10.32	14.85	18.37	1024.00
CG112	6.08	7.63	10.31	15.09	18.25	1024.00
FM512	7.13	8.65	10.76	16.17	19.03	1024.00
FM1024	6.94	8.32	10.54	15.76	18.71	1024.00
FB1024	11.89	16.36	14.29	22.36	22.70	1024.00
FG768	6.38	7.57	10.55	15.30	18.73	1024.00
ST1024	7.65	9.95	11.48	16.96	19.47	1024.00
Average	6.97	8.54	10.86	16.07	18.99	1024.00

Table 3-7 PSNR (Peak Signal Noise Ratio).

PSNR	GRPS	ERPS	EHS	DS	FSS	FS
CT256	39.49	39.50	39.48	39.51	39.49	39.56
CT40	32.21	32.08	31.46	31.92	31.69	32.04
HL40	34.49	34.60	34.27	34.25	34.17	33.55
MD96	40.08	40.09	39.87	39.99	39.93	39.80
CG112	29.14	29.16	29.07	29.14	29.13	29.08
FM512	34.05	34.10	33.94	34.06	34.02	34.06
FM1024	36.52	36.61	36.46	36.59	36.48	36.56
FB1024	34.87	34.88	34.86	34.93	34.94	35.28
FG768	26.17	26.19	26.15	26.18	26.16	26.33
ST1024	29.39	29.31	29.47	29.44	29.35	29.48
Average	33.64	33.65	33.50	33.60	33.54	33.57

Table 3-8 Coding Performance Comparison.

	GRPS over ERPS		GRPS over EHS		GRPS over DS		GRPS over FSS		GRPS over FS	
Gain	CG	QG	CG	QG	CG	QG	CG	QG	CG	QG
CT256	0.07	-0.01	0.79	0.02	1.58	-0.01	2.27	0.00	190.04	-0.07
CT40	0.18	0.13	0.74	0.74	1.51	0.28	2.07	0.51	170.24	0.16
HL40	0.15	-0.11	0.63	0.22	1.42	0.24	1.95	0.32	160.26	0.94
MD96	0.14	-0.02	0.73	0.20	1.48	0.08	2.07	0.15	170.24	0.27
CG112	0.25	-0.02	0.70	0.07	1.48	0.00	2.00	0.01	167.42	0.06
FM512	0.21	-0.05	0.51	0.12	1.27	-0.01	1.67	0.03	142.62	-0.00
FM1024	0.20	-0.08	0.52	0.07	1.27	-0.06	1.70	0.04	146.55	-0.04

FB1024	0.38	-0.01	0.20	0.01	0.88	-0.06	0.91	-0.06	85.12	-0.41
FG768	0.19	-0.02	0.65	0.02	1.40	-0.00	1.94	0.01	159.50	-0.15
ST1024	0.30	0.07	0.50	-0.08	1.22	-0.06	1.55	0.04	132.86	-0.09
Average	0.22	-0.01	0.56	0.14	1.30	0.04	1.72	0.11	145.83	0.07

In principle, a classical pattern search checks all possible candidates in its search pattern. Comparatively, its corresponding genetic pattern search randomly picks up one candidate in the search pattern. When the matching error surface is uni-modal and monotonic, half of the points in the search pattern should have smaller matching discrepancies than that of the center of the search pattern. Thus, in statistics, a genetic pattern search moves faster than its corresponding classical pattern searches. Yet, the classical pattern search moves along the steepest descent path, but the genetic pattern search may take a longer path. In addition, when the best MV is the starting point, the minimal numbers of search points required by both algorithms are the same. In this case, a genetic pattern search has the same computation performance as its corresponding classical pattern search. Therefore, as a whole, GRPS is faster than ERPS but its acceleration ratios are much less than 100%, as shown in **Table 3-6**.

In **Table 3-7**, the "sub-optimal" fast searches sometimes outperform the "optimal" FS in their PSNR qualities. The phenomena are particularly noticeable when the percentages of MV differential coding bits in the total bits are very high. FS looks for the motion vectors that minimize prediction errors without bit rate consideration; FS does not look for the motion vectors that produce the optimal rate-distortion outputs, which includes both distortion and bit rates. When the prediction error differences between the sub-optimal blocks and best matched blocks are small and this situation occurs in a large proportion of the coded images, the motion vectors have a significant influence on the coded picture quality. Although the prediction errors produced by a fast algorithm may be higher but the bits needed to encode its motion vectors may be fewer. It is thus possible that the final PSNR is higher for a sub-optimal search algorithm. To achieve the best overall rate-distortion results, the motion vectors sometimes should be replaced by the

"sub-optimal" motion vectors, which have larger estimation errors but fewer bits. In one way, we can do a sophisticated R-D optimization scheme, which is out of our scope in this paper. Alternatively, we can implicitly take the bitrate into consideration in the search process when we search the possible candidates according to the MV differential coding bits. Our proposed GRPS follows this concept.

Section 3.5 Application II: Performance Prediction

Our proposed model can also be used to predict the performance of applying a certain search method to an image sequence.

First, we demonstrate the prediction of the ASP value for a new test sequence using the parameters acquired by the first method. With the model parameters obtained from the training sequences, **Fig. 3-12** depicts the predicted performances on the new extra test sequences with FSS, DS, EHS and ERPS, respectively. Herein, the new extra test sequences and their settings in this experiment are listed in **Table 3-9**. In **Fig. 3-12**, X-axis represents the number of predicted ASP and the Y-axis represents the number of actual ASP, the training sequences are in plus-sign marks, the test (outside) sequences are in diamond-shape marks and the solid reference line represents that the predicted performance is exactly the same as actual performance. It is clear that the predicted performances are quite accurate.

Table 3-9 The extra test sequences and their coding settings.

Abbreviation	Sequence	Bitrate (K bps)	Frame rate (fps)	Number of frames
SI96	silent	96	10	300
TT512	table tennis	512	30	300
MB1024	mobile calendar	1024	30	300
TM768	tempete	768	30	260
NE40	news	40	7.5	90

Moreover, GRPS can be used to verify the validness of the proposed model for a new search

algorithm (which is outside the training algorithms). With the modeling parameters obtained from the training sequences by the first method, the performance of GRPS between the predicted ASP and actual ASP are compared in **Fig. 3-13**, wherein the plus-sign marks are the training sequences, the diamond-shape marks are the test sequences, and the solid line is the reference case when the predicted ASP matches exactly as actual ASP. In this figure, the X-axis represents the number of predicted ASP and the Y-axis represents the number of actual ASP. **Fig. 3-13** shows that 1) the proposed model indeed offers a good performance prediction for a new search pattern. One may observe that the diamond-shape mark (the ASPs of the test sequences) can be accurately predicted by the proposed model, and 2) GRPS has a high performance for both low motion and relative high motion sequences as predicted by our proposed model and the experiment results verify this prediction.

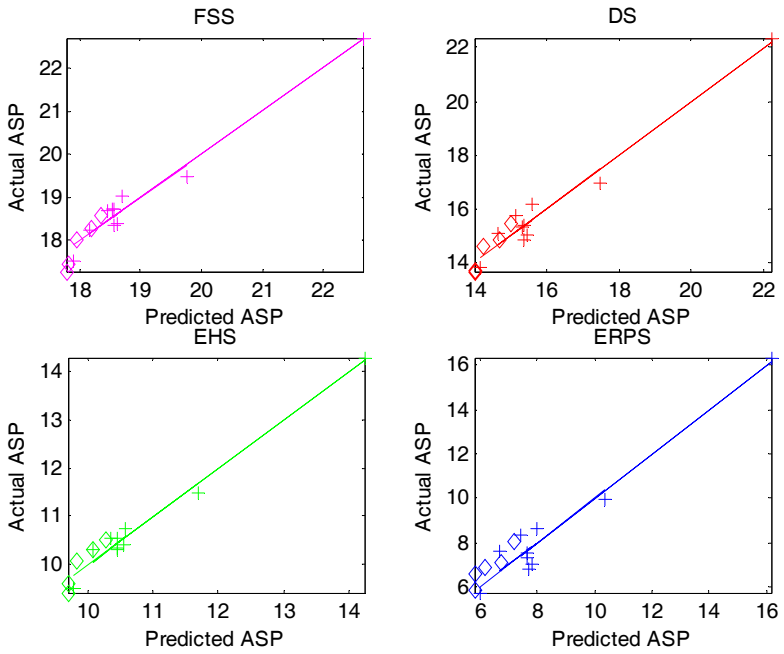


Fig. 3-12 Relation charts between the predicted ASP and the actual ASP (1st Method).

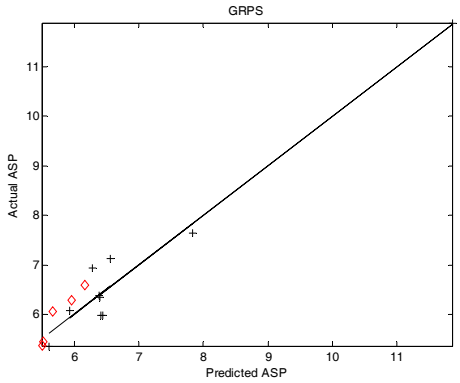


Fig. 3-13 Performance prediction for GRPS (for various test sequences, 1st method).

Furthermore, we evaluate the predictive ASP value for the proposed GRPS by using the second method. The performance of GRPS on a sequence can be predicted by the proposed model with the modeling parameters estimated based on the four popular search algorithms as the training data. In **Fig. 3-14**, the X-axis is the predicted ASP value, and the Y-axis is actual ASP. The cross-sign marks are the training search algorithms; they are FSS, DS, EHS, and ERPS. The diamond-shape marks represent the proposed GRPS, and the dash line indicates that the predicted performance is exactly the same as the actual performance. One may see that the performance prediction is generally very accurate for all 10 sequences.

Thus, the performance prediction can be used to select an effective search pattern set in a practical video coding system with adaptive selection of search pattern sets as suggested by [39] and [40]. The variances of motion vectors of a video clip can be known by either the predicting method (from the previous video clip) or the pre-analysis method (a two-pass process). Then, the PDF of motion vectors for the video clip is constructed. Combining with the WF of different search pattern sets, we know which search pattern set is more effective for this video clip. The video clip can be a video sequence, part of a video sequence (some video frames), part of a frame (a few image blocks), and the predicting method can use information from the spatial or temporal neighboring video clips.

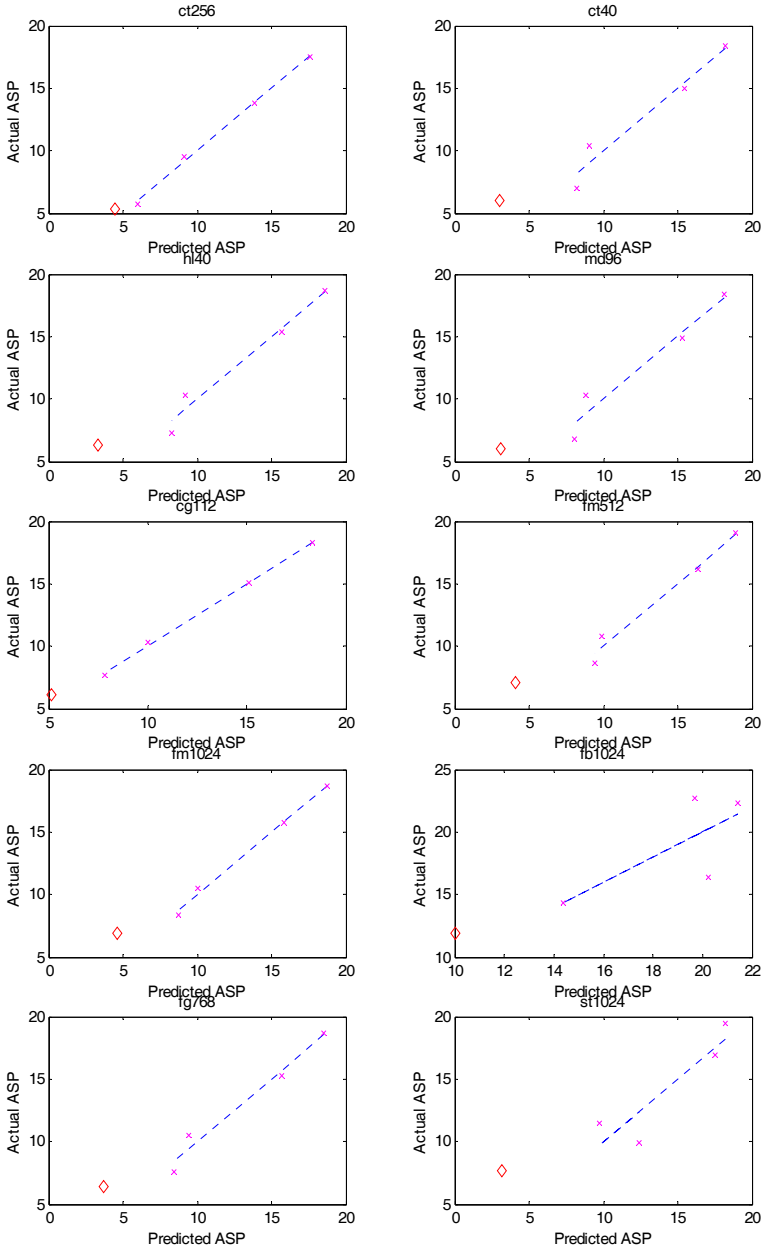


Fig. 3-14 Performance prediction of GRPS as a new search algorithm (2nd method)

Section 3.6 Chapter Summary

A systematic approach is taken in this chapter for constructing a mathematical model for the PBME algorithms. With the assistance of goodness-of-fit tests, we propose a statistical PDF for the motion vectors. It matches well the real motion vector PDF produced by FS when their

variances are adjusted identical. We then propose a weighting function model that describes the minimal search points of a search algorithm. The WF of a certain PBME algorithm is estimated by analyzing the search process of that PBME. The complete PBME model includes these two elements: the statistical PDF derived from a video sequence and the WF derived from a search algorithm.

With the proposed model, we can predict the performance of a new pattern search without actually applying the search algorithm to a video sequence. Thus, it helps us in constructing new search patterns (algorithms). Two application examples are given. Starting from an ideal WF target, we proposed the GRPS algorithm, which outperforms all other popular search algorithms. In addition, because this model can be used to predict the performance of a PBME algorithm, it can assist in selecting search patterns adaptively as the video sequence changes its characteristics along the time. Therefore, an adaptive (switchable) pattern search algorithm is made possible with a small amount of computational overhead.

Chapter 4 Design of Pattern-based Block Motion Estimation Algorithms

To relieve the computational burden of BME, a myriad of fast BME algorithms have thus been proposed. Recent proposals in BME constitute several heuristic components and each has a number of possible parametric structures and values. The overall system is very complicated and how to optimally choose these parameters/structures are not yet fully explored. After studying many algorithms devised in the past, we develop a systematic approach to find the optimal or nearly optimal solutions to these problems and, at the end, a new BME algorithm including all techniques is proposed.

According to [8], fast BME algorithms are classified mainly into two categories: 1) reducing the number of search (checking) points and 2) lowering the computational complexity in calculating the block-matching cost for each search point. In this study, we focus on the first category. The most popular algorithm in the first category is the PBME algorithm, which is typically a multi-step process. Often, three sets of tools are included: 1) search patterns [26][27][28][29][30][31][32][33], 2) starting points [14][15][16][30][31], and 3) early termination thresholds [9][14][30][31]. Adopting a divide-and-conquer approach in this study, we first scrutinize the underneath mechanism in each tool by using the analytical model [54] in Chapter 3. Then, we study and improve the techniques used for each processing step. At the end, we combine these tools together and form a very effective PBME algorithm.

The rest of this chapter is organized as follows. Section 4.1 reviews our previously proposed analytical model for PBME algorithms, which consists of 1) a minimal check point profile associated with a search pattern and 2) a statistical MV probability distribution associated with an image sequence. In Section 4.2, we propose two sets of the genetic-algorithm-based search patterns for different types of moving image sequences. Then, we design a pattern switching

strategy, which dynamically changes search patterns based on the real-time video statistics. Section 4.3 examines the impact of starting (initial) point set and suggests a starting point set that produces outstanding search results. Section 4.4 suggests a threshold predictor that can be used in the early termination algorithm. Combining all these techniques together, Section 4.5 presents a complete PBME algorithm and its performance. At last, a brief concluding summary is given in Section 4.6.

Section 4.1 Review of the PBME Model

In Chapter 3 [54], the proposed mathematical model (expressed by (4.1), identical to (3.18)) can predict the ASP produced by a PBME. This model consists of two components: a statistical probability distribution function $S_{FS}(x,y)$ of MVs (approximated by (4.2)), and the minimal number of search points for a MV located at (x,y), $WF_{SA}(x,y)$, (called *weighting function*) produced by a search algorithm. In (4.1), (x,y) are the relative coordinates of which the origin is the *PMV*, defined by (2.2). The parameters (C_1, C_2) are obtained experimentally by training methods. Note that C_1 is always positive because *ASP* and the sum of products of $S_{FS}(x,y)$ and $WF_{SA}(x,y)$ are always positively correlated.

$$ASP = C_1 \times \sum_{x,y \in A} S_{FS}(x,y) \times WF_{SA}(x,y) + C_2 \tag{4.1}$$

$$S_{FS}(x,y) = \frac{\frac{1}{|x|^{5/3} + \zeta_x} \frac{1}{|y|^{5/3} + \zeta_y}}{\sum_{(x',y') \in A} \frac{1}{|x'|^{5/3} + \zeta_x} \frac{1}{|y'|^{5/3} + \zeta_y}} \tag{4.2}$$

Eq.(4.2) is derived from (3.11) based on the experimental data. In (4.2), (x,y) and (x',y') are relative coordinates with respect to (w.r.t.) *PMV*, and A is the search area. The parameters (ζ_x, ζ_y) are obtained by numerical methods such that the variances of $S_{FS}(x,y)$ match those of the MVs acquired by performing FS on a specific sequence.

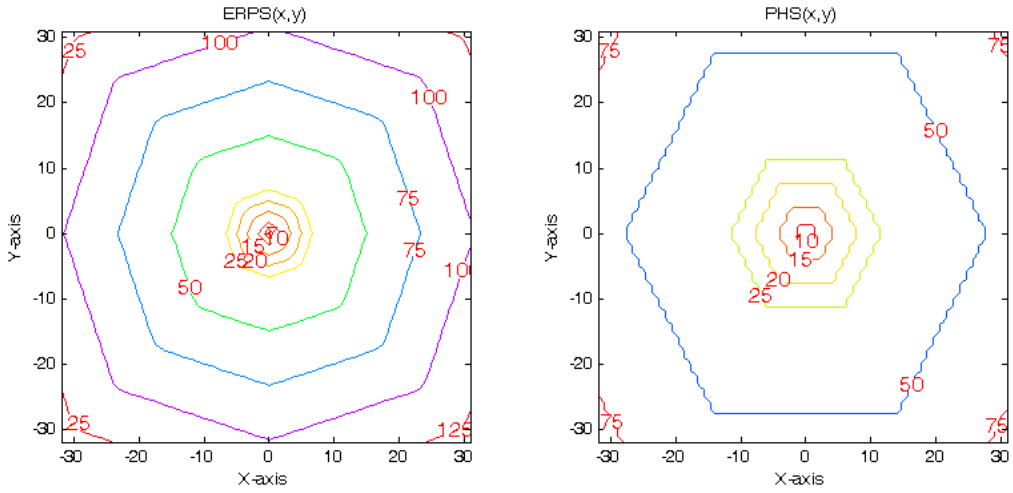


Fig. 4-1 Contour plots of the WFs of ERPS and PHS.

The weighting function, $WF_{SA}(x,y)$, is the minimal number of search points produced by a specific PBME algorithm when the argument (x,y) is the target MV. The weighting function can be obtained by analyzing the search procedure. **Fig. 4-1** exemplifies WF of ERPS and PHS (Point-oriented Hexagonal Search [33]).

Note that, the simulation platform, the test sequences and their settings adopted in this chapter are the same as described in Section 3.1 and **Table 3-1**.

Section 4.2 Adaptive Pattern Search Algorithms

4.2.1 Genetic Pattern Searches

A preferred pattern search should have the following desirable properties: 1) it consumes less computing power, 2) it does not degrade the video quality, and 3) it costs fewer bits in coding the MV vectors.

In [51] and [54], after analyzing the weighting function of several popular search algorithms (**Fig. 3-3** and **Fig. 4-1**), we find that ERPS has the smallest ASP values for the MVs near the

origin (PMV) and PHS has the smallest ASP for the points away from the origin. These observations are consistent with the well-known facts that PHS moves faster than many other algorithms and thus it quickly reaches the distant locations, and ERPS examines fewer points when the target MV is close to the origin. These observations suggest that a good PBME algorithm should have small weighting function values for all locations in the search area, particularly for the high probability target MVs.

A search algorithm degrades the video quality when it is trapped into a local optimum point. To reduce such cases, a search algorithm shall check all the neighboring points of the target when it decides to terminate the search process. The dilemma is that the increased checking points also increase computation. To achieve a balance between speed and quality, a PBME algorithm shall carefully select the number and the locations of check points at the termination step.

A search algorithm should make good use of the uneven MV distribution to reduce the entropy coding bits. For example, if the (best) MVs cluster around a predictable location, it takes fewer bits in encoding MVs and less computing power in finding MVs. Because the probability density function of typical MVs peaks at around the PMV, a PBME algorithm with small weighting function near the starting point (PMV) would consume less computing power and fewer coding bits on the average. For convenience, therefore, our PBME model is centered at PMV.

Based on the above design considerations, we adopt the genetic algorithms [23][24] to modify the traditional PBME algorithms. The simplest genetic algorithm contains only a mutation-and-competition loop. When a survivor (parent) produces a mutant (a child), the survivor competes with its own mutant to decide the next survivor (next-stage parent). The process stops when the survivor beats all its mutants. In contrast, the traditional PBME algorithms check all points in the search pattern and move the center (origin) to the winner until the central point beats all the other points in the search pattern. Thus, on the average, the traditional methods

check more points than the genetic-based methods.

A traditional PBME algorithm typically consists of two search patterns, the *large search pattern* and the *small search pattern*. The large search pattern is used for the coarse (regular) search and the small search pattern is used for the fine (terminating) search. In converting a traditional search algorithm into a genetic one, we blend the genetic algorithm into the coarse search stage. The central point (which is the winner of the previous search step) in the search pattern is the parent in the genetic search and all the other points are the child candidate set. Instead of calculating the block matching cost of all child candidates and deciding the best MV, we randomly select a point (a *mutant*) from the child candidate set, calculate its block-matching cost, compare its cost with the parent's cost (*competition*), and decide the survivor (next *parent*). This process continues until all the points in the current child set are examined. If the parent beats all its children, it is then declared to be the winner. In addition, a typical terminating search checks all the points in the small search pattern to avoid trapping into the local minimum. But recent studies [32][33] suggest that it is often sufficient to check only the candidate points near the smallest error points in the large pattern. This phenomenon can be explained using the monotonic error surface assumption below.

Conceptually, the genetic PBME algorithm can reach the optimal point under the assumption that the matching-error (cost) function is monotonic and uni-modal. That is, the block matching error surface decreases monotonically as the checking point moves closer to the global minimum. Therefore, we can reach the global minimum point by moving the search pattern center to the current minimal point step-by-step. Under the monotonic and uni-modal matching-error surface assumption, typically, half of the candidate child points (mutants) in a large pattern have higher matching cost than the current point (parent). Therefore, on the average, the genetic algorithm saves about 50% computation for moving one step when compared to its non-genetic sibling. As for the chance of being trapped in the local minimum, a genetic algorithm has roughly a similar

behavior as its non-genetic sibling. The reason is that both of them end the (coarse) search stage when the matching error of the center point is smaller than that of any other point in the (large) search pattern. But they may be trapped into different sub-optimum locations.

Because of the computational advantage of the genetic algorithm, we convert ERPS into GRPS (genetic-based ERPS) and PHS into GPHS (genetic-based PHS), respectively. The flow chart of GRPS is shown in **Fig. 4-2**, and its associated search pattern is shown in **Fig. 4-3**. In step 2 (S2), it randomly checks one point (black, for example) among all search points in **Fig. 4-3**(a). The condition of step 3B (S3B) is whether all the (black) points in **Fig. 4-3**(b) have been checked.

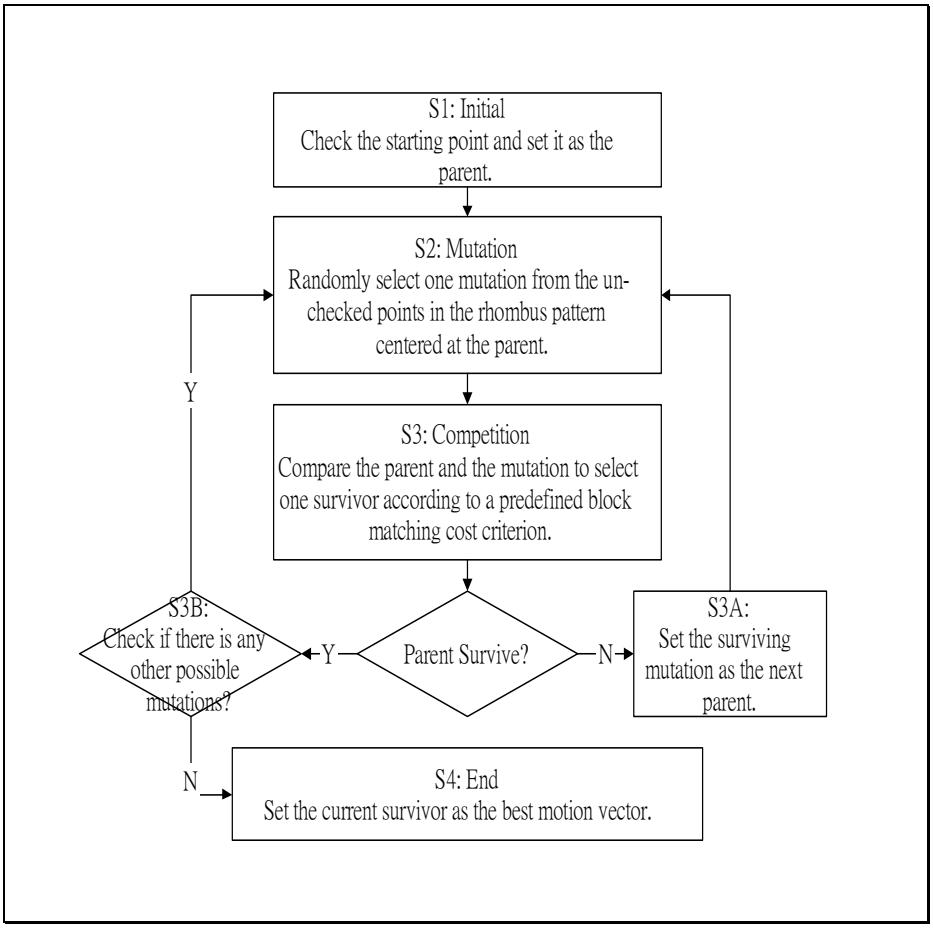


Fig. 4-2 The flowchart of GRPS

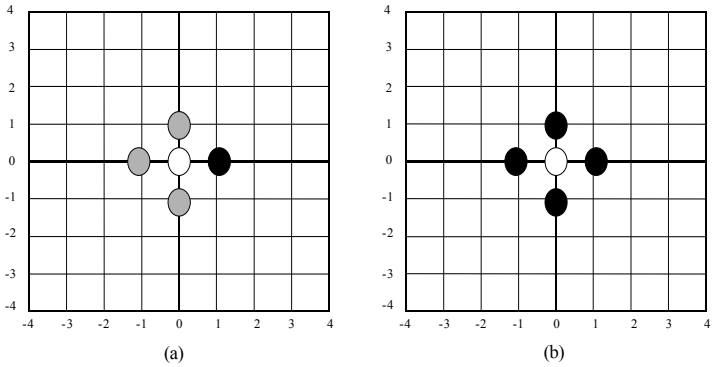


Fig. 4-3 The search patterns for GRPS

The flowchart of GPHS is shown in **Fig. 4-4** and its associated search patterns are shown in **Fig. 4-5**. Steps 2 and 3 are similar to those of GRPS but with a different large search pattern. In Step 4 (S4, Refinement), as suggested in [33], we first calculate the cost function, so-called Normalized Group Distortion (NGD) defined by (4.3), of all the grey points in **Fig. 4-5**(b). Then, we select the smallest NGD point from points a to f in **Fig. 4-5**(c) and the smaller NGD point from points g and h in **Fig. 4-5**(d). These two points constitute the small search pattern. Herein, the NGD of points a to h is calculated, respectively, based on the SADs in the groups A to H as defined by **Fig. 4-5**(c) and **Fig. 4-5**(d). This last step is biased to the horizontal direction because most pixels in nature image sequences have a higher probability in moving horizontally.

$$NGD = \sum_{i=1}^{N} \frac{SAD_i}{d_i} = \sum_{i=1}^{N} \frac{SAD_i}{\sqrt{(x_i - x)^2 + (y_i - y)^2}}, \tag{4.3}$$

where (x,y) is the point to be evaluated, (x_i, y_i) is its i-th neighbor in its group, which can be one of A to H in **Fig. 4-5**(c) and **Fig. 4-5**(d), and N is the total point number in each group. Then, SAD_i denotes the SAD of the neighbor i, and d_i denotes the distance between (x,y) and (x_i, y_i).

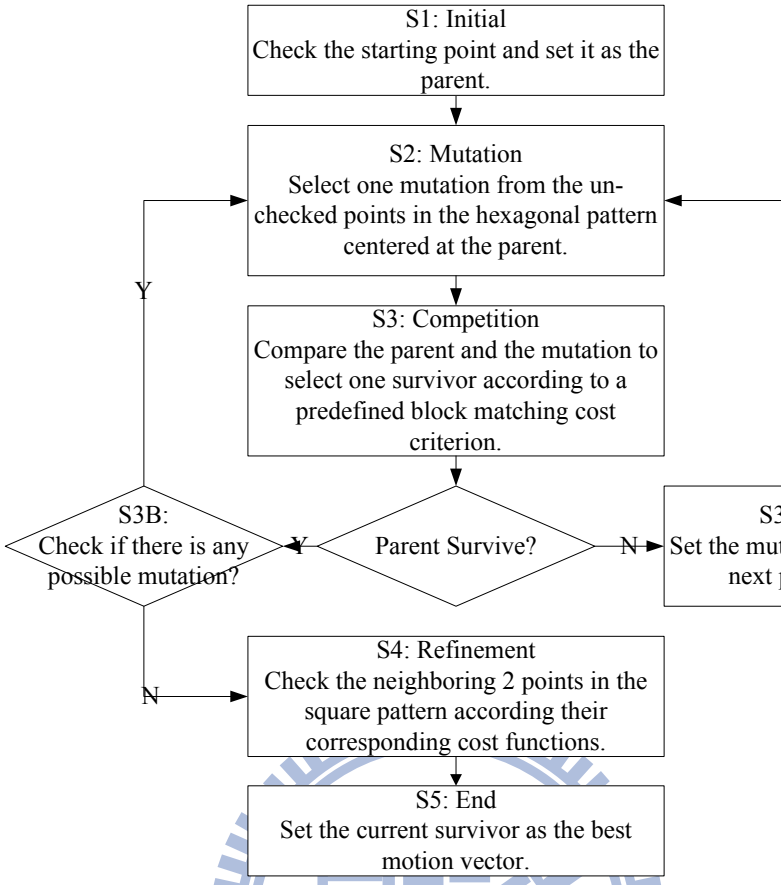


Fig. 4-4 The flow chart of GPHS

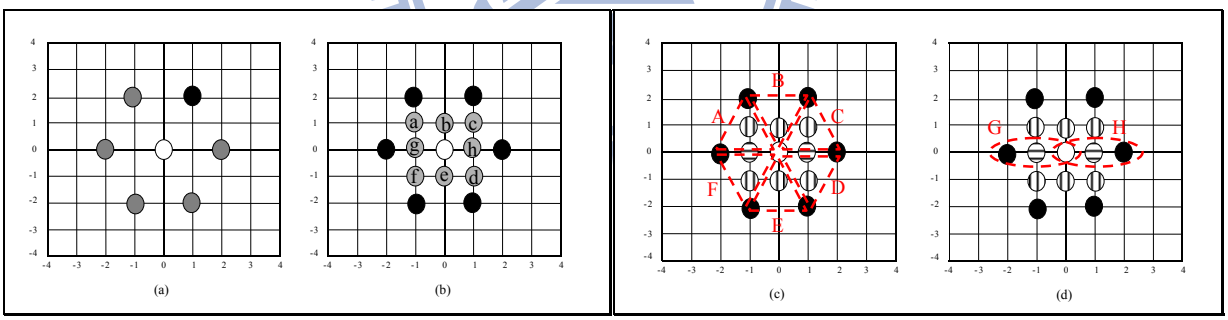


Fig. 4-5 The search patterns of GPHS

Fig. 4-6 shows the weighting functions of GRPS and GPHS. Compared with the weighting functions of the other PBME algorithms in **Fig. 4-1**, GRPS has the smallest values around the center but GPHS has the smallest values at far-away locations. As discussed earlier, we predict that they should outperform their non-genetic siblings.

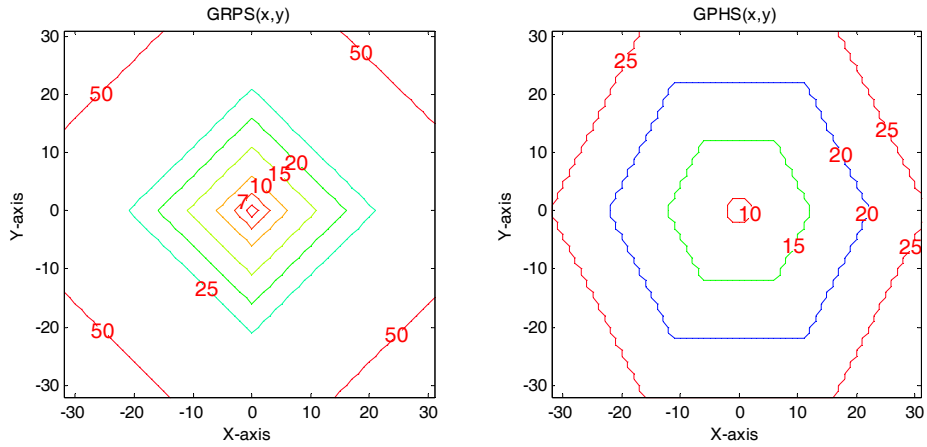


Fig. 4-6 Contour plots of the weighting function for GRPS and GPHS.

Table 4-1 shows the performance of FS, DS, ERPS, PHS, GRPS and GPHS when they are applied to the representative sequences under the settings in Section 3.1 and **Table 3-1**. Herein, 'ASP' denotes the average number of search points per block, and 'PSNR' denotes the average frame PSNR in a coded sequence. On the average, GRPS is faster than ERPS by 23% and GPHS is faster than PHS by 6%, and their PNSR values are about the same at the same bit rates. When they are compared to their non-genetic siblings, their average PSNR drop is about 0.01dB and 0.06dB, respectively. When compared to the other popular search algorithms, GRPS outperforms DS by 131% in speed with a 0.04dB PSNR gain and it outperforms FS by 146 times in speed with a 0.07dB PSNR gain. GPHS outperforms DS by 58% in speed with a 0.19dB PSNR loss and beats FS by 100 times in speed with a 0.16dB PSNR loss. Overall, it shows that we can accelerate the search process by applying the genetic algorithm to the traditional pattern search and the resulting PSNR quality loss is negligible.

Moreover, the computational overhead of the genetic based algorithms is very small because we did not use the entire conventional genetic algorithm. Other than a few additional comparisons of the matching errors, the only computational overhead is the random selection of a mutant from the child set and this process can be implemented by a simple pseudo number generator.

Table 4-1 The performance of FS, DS, ERPS, PHS, GRPS, and GPHS

Normal	FS		DS		ERPS		PHS		GRPS		GPHS	
Sequence	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR
CT256	1024	39.56	13.81	39.51	5.75	39.50	9.52	39.44	5.36	39.49	9.38	39.43
CT40	1024	32.04	15.03	31.92	7.04	32.08	10.30	31.48	5.98	32.21	9.89	31.21
HL40	1024	33.55	15.38	34.25	7.34	34.57	10.10	34.17	6.35	34.49	9.68	34.09
MD96	1024	39.80	14.85	39.99	6.82	40.10	10.03	39.85	5.98	40.08	9.65	39.80
CG112	1024	29.08	15.09	29.14	7.64	29.12	10.26	29.03	6.08	29.14	9.76	29.00
FM512	1024	34.06	16.17	34.06	8.65	34.10	10.57	33.92	7.13	34.05	10.00	33.89
FM1024	1024	36.56	15.76	36.59	8.32	36.61	10.35	36.44	6.94	36.52	9.85	36.46
FB1024	1024	35.28	22.36	34.93	16.36	34.88	14.18	34.87	11.89	34.87	12.75	34.73
FG768	1024	26.33	15.30	26.18	7.57	26.19	10.34	26.17	6.38	26.17	9.95	26.15
ST1024	1024	29.48	16.96	29.44	9.95	29.31	11.40	29.33	7.65	29.39	10.56	29.42
Average	1024	33.57	16.07	33.60	8.54	33.65	10.71	33.47	6.97	33.64	10.15	33.41

4.2.2 Adaptive Pattern Switching Strategy

Because the contents of video sequences vary drastically, one single pattern search may not produce the best result in terms of speed and PSNR. Thus, the adaptive pattern-switching search algorithms were proposed [35][36][37][38][39][40][41][42][43]. These algorithms are empirically constructed and the switching criterion is often based on block (feature) classification. Few papers have clear and strong evidence as why certain block features can be used as the switching criterion. Also, there are few discussions on how to optimally choose the pattern search set. Therefore, we like to explore these issues based on our previous study [52].

We look for an adequate index that can be used to decide the right instant to switch between two pattern searches. The target is lowering the computational complexity. That is, if search algorithm 1 (SA1) is in use, would the average search points be fewer than that produced by using search algorithm 2 (SA2)? Based on our ASP model (Eq.(4.1)), the difference in ASP is expressed by (4.4).

$$D_{ASP} = C_1 \times \sum_{x,y \in A} S_{FS}(x,y) \times (WF_{SA1}(x,y) - WF_{SA2}(x,y)) \tag{4.4}$$

Note that WF_{SA1} and WF_{SA2} depend on search algorithms only. But because S_{FS} is a function of the MV variance, D_{ASP} is thus picture-dependent. The parameter C_1 is fixed for a video sequence; dividing D_{ASP} by C_1, we obtain the *switching index* (I_{ASP}) defined by (4.5).

$$I_{ASP} = D_{ASP} / C_1 \tag{4.5}$$

ERPS and PHS are chosen as the basic pattern searches owing to their short range and long range search performance. Then, the I_{ASP} between ERPS and PHS, drawn against two variables, MV variance and MV standard deviation, are shown in **Fig. 4-7**(a) and **Fig. 4-7**(c). In **Fig. 4-7**(a), the X-axis is the MV variance of the horizontal component and the Y-axis is that of the vertical component. In **Fig. 4-7**(c), the axes are the MV standard deviations along the horizontal direction and the vertical direction, respectively. When $I_{ASP} > 0$, ERPS outperforms PHS in terms of *ASP*, and when $I_{ASP} < 0$, PHS is better. Therefore, the switching criterion can be the MV variance value, at which I_{ASP} equals zero. For the case of ERPS and PHS pair, the threshold, $I_{ASP}=0$, is approximately a straight line in the MV *variance* coordinates. That is, (4.6) is used to decide the pattern search in use, wherein P, Q, and R are determined by the numerical methods. In our experiments, $P = 1$, $Q = 1$ and $R = 20$.

$$P \cdot VAR_X + Q \cdot VAR_Y = R\,. \tag{4.6}$$

When GRPS and GPHS are the two basic pattern searches, their I_{ASP} are shown in **Fig. 4-7**(b) and **Fig. 4-7**(d). But in this case, we find that in the MV *standard deviation* domain, $I_{ASP}=0$ is better approximated by a straight line. That is, (4.7) is used to decide the pattern search and $U = 1$, $V = 1$ and $W = 12$ in our experiments.

$$U \cdot STD_X + V \cdot STD_Y = W\,. \tag{4.7}$$

Indeed, our analysis in the above agrees with the commonly accepted principle that the small search patterns are more suitable for the 'low motion' sequences or 'low MV variance' sequences and the large search patterns are more suitable for the 'high motion' sequences or 'high MV variance' sequences. In other words, the key in determining the relative search cost (namely, ASP) between two pattern searches is not the magnitudes of MV, but the randomness of MV, which can be measured by the MV variance or standard deviation.

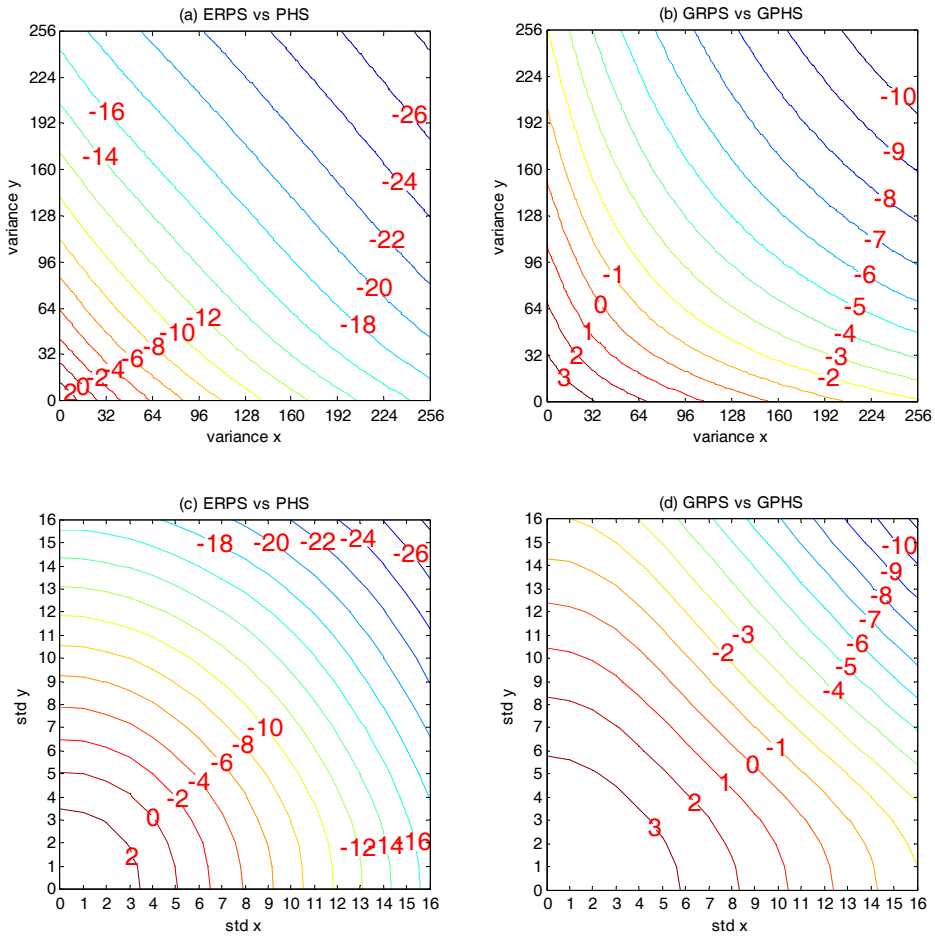


Fig. 4-7 The I_{ASP} between ERPS and PHS w.r.t. MV variance or MV standard deviation, and that between GRPS and GPHS w.r.t. MV variance or MV standard deviation

An adaptive pattern switching strategy is thus developed based on the threshold equation defined by (4.6) or (4.7). To ease the following discussions, the adaptive algorithm using GRPS and GPHS is called *adaptive genetic pattern search* (AGPS). Its flow chart is shown in **Fig. 4-8**. A similar algorithm is developed for the ERPS and PHS pair and is called *adaptive pattern search* (APS), which has a similar procedure but replaces (4.7) by (4.6) in Step S3 in **Fig. 4-8** and, of course, GRPS and GPHS are replaced by ERPS and PHS, respectively.

In real-time applications, the MV variances or standard deviations of the current frame are not available before its MVs are calculated. Fortunately, the motion characteristics in an image sequence typically change gradually [30]; therefore, the MV variances in the neighboring spatial or temporal areas are generally similar. After testing a few MV variance predictors, we found that the MV variance of the previous frame is a good prediction to its value in the current frame.

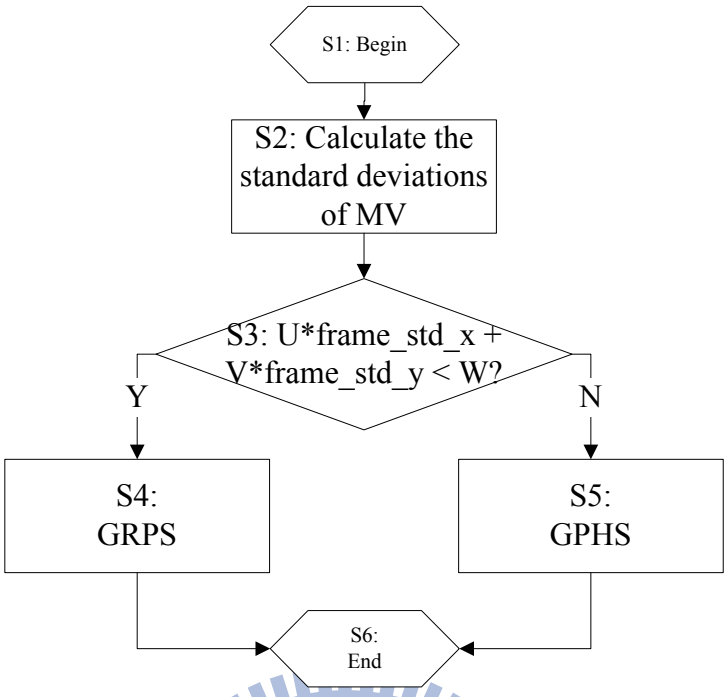


Fig. 4-8 The flow chart of AGPS.

Furthermore, using one single pattern search for the entire frame is a rough strategy. The MV characteristics may vary in different parts of a frame. Hence, we can switch the pattern search for each block. Because the MV characteristics in the nearby spatial/temporal area tend to be similar, after a few try-and-errors, three neighboring blocks in the current and previous frame is used in calculating the MV variance and standard deviation as shown by (4.8) and (4.9). The locations of these three blocks are defined by **Fig. 4-12**.

$$VAR^{MV} = \frac{(MV^{L} - MV^{mean})^2 + (MV^{U} - MV^{mean})^2 + (MV^{P} - MV^{mean})^2}{3}. \tag{4.8}$$

$$STD^{MV} = \sqrt{VAR^{MV}}\ . \tag{4.9}$$

$$MV^{mean} = \frac{MV^{L} + MV^{U} + MV^{P}}{3}, \tag{4.10}$$

where MV^{L}, MV^{U} and MV^{P} are the motion vectors of the left and the upper block neighbors to the current block, and the collocated block in the previous frame, respectively, as illustrated by **Fig. 4-12**.

The so-called *double level pattern switching strategy* for AGPS (abbr. DL AGPS) is thus proposed and its flow chart is shown in **Fig. 4-9**. If the previous frame has small MV standard

deviations, we incline towards using GRPS as the pattern search with the exception that the MV standard deviations derived from the nearby blocks are very large. On the other hand, if the previous frame has large MV standard deviations, GPHS is often chosen unless the MV standard deviations derived from the neighboring blocks are very small. The parameter values of U, V, W_{frame}, W_{block1}, and W_{block2} are derived from data by using the numerical method. In our experiments, $U = 1$, $V = 1$, $W_{frame} = 12$, $W_{block1} = 8$, and $W_{block2} = 16$. Likewise, the flow chart of the double level pattern switching strategy for APS (abbr. DL APS) is similar but the MV standard deviation is replaced by MV variance in choosing the pattern search. Also, (4.6) is in use and $P = 1$, $Q = 1$, $R_{frame} = 20$, $R_{block1} = 8$, and $R_{block2} = 32$.

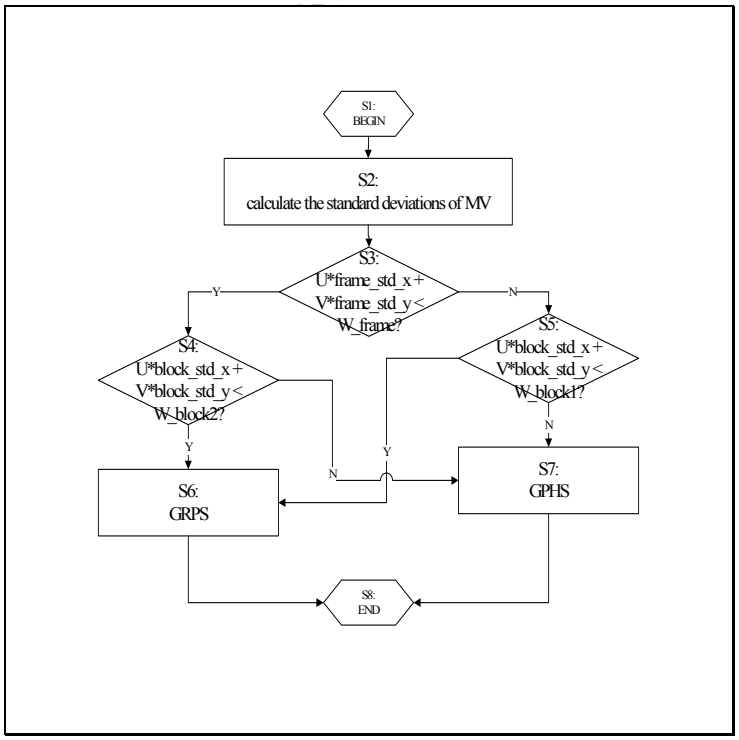


Fig. 4-9 Flow chart of the double level adaptive genetic pattern search (DL AGPS).

The computational overhead of the proposed adaptive pattern selection strategy is very small. At the frame level, the frame MV variance/standard deviation is calculated once per frame. At the block level, we only use the upper, the left, and the co-located block motion vectors to calculate the MV variance/standard deviation. In computer simulation, the run time profiling shows that the overhead of the proposed adaptive strategy consumes only about 2% computation of the entire

ME module.

Table 4-2 shows the performance of GRPS, GPHS, AGPS and DL AGPS, and **Table 4-4** shows the performance of ERPS, PHS, APS and DL APS, when they are tested on the sequences under the settings given by **Table 3-1** ('normal' speed). To test the extreme cases, we generate new test sequences consisting of the odd frames of the sequences in **Table 3-1**. These new sequences (denoted as '2X') thus run at twice the speed of their originals. **Table 4-3** and **Table 4-5** show the performance of various pattern search algorithms tested on the 2X sequences under the settings given by **Table 3-1**. In **Table 4-2** to **Table 4-5**, 'ASP' is the average number of search points per block, 'PSNR' is the average frame PSNR of a sequence, and 'RATIO' is the frequency (probability) ratio of GRPS or ERPS used.

Table 4-2 (normal sequences) shows that both AGPS and DL AGPS outperform GPHS by around 46% in terms of ASP and they have similar PSNR performance as GRPS. Both the single level pattern switching strategy and the double level pattern switching strategy tend to use GRPS as the pattern search because most natural image sequences have relative small MV standard deviation. However, in the extreme cases such as football (FB1024), AGPS outperforms GRPS by 1.3% in ASP (0.01dB PSNR loss) and outperforms GPHS by 8.6% (0.13dB PSNR gain). And the DL AGPS further outperforms AGPS by 0.9% (0.04dB PSNR loss).

In **Table 4-3** (2X sequences), AGPS outperforms GRPS by 1.5% in ASP (0.01dB PSNR loss) and outperforms GPHS by 33.7% (0.25dB PSNR gain), and DL AGPS further outperforms AGPS by 0.5% (0.02dB PSNR gain). In general, AGPS has some advantages on fast motion sequences and DL AGPS adds in a slightly better gain in ASP and PSNR quality.

Without the genetic search feature, **Table 4-4** (normal sequences) shows that APS outperforms ERPS by 3.4% in ASP (0.00dB PSNR gain) and outperforms PHS by 29.6% (0.17dB PSNR gain). And DL APS further outperforms AGPS by 0.7% (0.01dB PSNR gain). On the 2X sequences (**Table 4-5**), APS outperforms ERPS by 8% in ASP (0.01dB PSNR loss) and

outperforms PHS by 20% (0.19dB PSNR loss), and DL APS further outperforms AGPS by about 0.8% (0.01dB PSNR gain).

Overall, the adaptive pattern switching strategy is effective. It does not hurt the slow motion sequences but reduce the computation quite significantly on the fast motion sequences. With the adaptive pattern switching scheme, the proposed algorithm outperforms the 'single' pattern search algorithms. Clearly, the genetic version, AGPS, is much better than the non-genetic version, APS. Though marginally, the double level strategy further improves in both PSNR quality and speed especially for sequences with high MV variances.

To examine the correctness of the switching strategy, we show the I_{ASP} (represented by the diagonal curves) for the 'normal' sequences and the '2X' sequences in **Fig. 4-10** and **Fig. 4-11**. In these figures, a dot denotes an MV variance or standard deviation (STD) pair of an image frame and the cross denotes the MV variance/STD for the entire sequences. The dashed line is the pattern switching threshold. It is clear that for most sequences, the MV variance or STD of the current frame is highly correlated with that of the preceding frames and its value changes slowly across frames. Overall, ERPS and GRPS are generally adequate for coding these sequences. However, in the extreme case, such as football (FB1024) and foreman (FM512/FM1024), PHS and GPHS stand out.

Table 4-2 Performance of GRPS, GPHS, AGPS, and DL AGPS on the normal speed sequences.

Normal	GRPS		GPHS		AGPS			DL AGPS		
Sequence	ASP	PSNR	ASP	PSNR	ASP	PSNR	RATIO	ASP	PSNR	RATIO
CT256	5.35	39.50	9.38	39.43	5.35	39.50	100%	5.36	39.51	100%
CT40	5.98	32.20	9.89	31.21	5.98	32.20	100%	6.04	32.07	100%
HL40	6.35	34.45	9.68	34.09	6.35	34.45	100%	6.35	34.45	100%
MD96	5.98	40.06	9.65	39.80	5.98	40.06	100%	5.98	40.04	100%
CG112	6.08	29.11	9.76	29.00	6.08	29.11	100%	6.08	29.11	100%
FM512	7.13	34.05	10.00	33.89	7.13	34.05	100%	7.10	34.04	100%
FM1024	6.94	36.52	9.85	36.46	6.94	36.52	100%	6.93	36.53	100%
FB1024	11.89	34.87	12.75	34.73	11.74	34.86	89%	11.64	34.82	93%
FG768	6.38	26.17	9.95	26.15	6.38	26.17	100%	6.37	26.17	100%
ST1024	7.65	29.39	10.56	29.42	7.65	29.39	100%	7.62	29.43	100%
Average	6.97	33.63	10.15	33.41	6.96	33.63		6.95	33.62	

Table 4-3 Performance of GRPS, GPHS, AGPS and DL AGPS on the 2X sequences.

2x forward	GRPS		GPHS		AGPS			DL AGPS		
Sequence	ASP	PSNR	ASP	PSNR	ASP	PSNR	RATIO	ASP	PSNR	RATIO
CT256	5.62	38.65	9.51	38.52	5.62	38.65	100%	5.59	38.79	100%
CT40	6.60	30.28	10.34	29.22	6.60	30.28	100%	6.60	30.27	100%
HL40	6.51	33.31	9.74	32.95	6.51	33.31	100%	6.51	33.31	100%
MD96	6.40	38.66	9.85	38.37	6.40	38.66	100%	6.38	38.67	100%
CG112	7.36	27.43	10.64	27.24	7.36	27.43	100%	7.34	27.45	100%
FM512	9.07	32.34	11.01	32.19	8.80	32.33	93%	8.73	32.35	93%
FM1024	8.85	35.25	10.79	35.12	8.49	35.22	92%	8.47	35.25	93%
FB1024	15.75	33.22	14.62	33.12	15.13	33.18	84%	14.92	33.23	83%
FG768	7.01	25.51	10.35	25.42	7.01	25.51	100%	7.03	25.51	100%
ST1024	9.28	27.99	11.73	27.87	9.28	27.99	100%	9.27	27.92	100%
Average	8.25	32.27	10.86	32.00	8.12	32.26		8.08	32.28	

Table 4-4 Performance of ERPS, PHS, APS and DL APS on the normal sequences.

Normal	ERPS		PHS		APS			DL APS		
Sequence	ASP	PSNR	ASP	PSNR	ASP	PSNR	RATIO	ASP	PSNR	RATIO
CT256	5.75	39.50	9.52	39.44	5.75	39.50	100%	5.75	39.49	100%
CT40	7.04	32.08	10.30	31.48	7.04	32.08	100%	6.97	32.15	99%
HL40	7.34	34.57	10.10	34.17	7.34	34.57	100%	7.31	34.55	100%
MD96	6.82	40.10	10.03	39.85	6.82	40.10	100%	6.80	40.10	100%
CG112	7.64	29.12	10.26	29.03	7.64	29.12	100%	7.64	29.13	100%
FM512	8.65	34.10	10.57	33.92	8.47	34.06	92%	8.33	34.10	94%
FM1024	8.32	36.61	10.35	36.44	8.23	36.60	93%	8.07	36.65	95%
FB1024	16.36	34.88	14.18	34.87	14.02	34.86	51%	13.88	34.90	60%
FG768	7.57	26.19	10.34	26.17	7.57	26.19	100%	7.57	26.19	99%
ST1024	9.95	29.31	11.40	29.33	9.72	29.33	85%	9.68	29.30	85%
Average	8.54	33.65	10.71	33.47	8.26	33.64		8.20	33.65	

Table 4-5 Performance of ERPS, PHS, APS and DL APS on the 2X sequences.

2X forward	ERPS		PHS		APS			DL APS		
Sequence	ASP	PSNR	ASP	PSNR	ASP	PSNR	RATIO	ASP	PSNR	RATIO
CT256	6.35	38.68	9.74	38.51	6.35	38.68	100%	6.35	38.67	99%
CT40	8.15	30.22	10.89	29.54	8.15	30.22	100%	8.13	30.26	98%
HL40	7.57	33.38	10.22	33.02	7.57	33.38	100%	7.58	33.34	100%
MD96	7.56	38.66	10.38	38.44	7.56	38.66	100%	7.49	38.72	99%
CG112	9.54	27.53	11.48	27.34	9.62	27.50	99%	9.54	27.49	97%
FM512	11.70	32.45	12.02	32.23	10.58	32.41	79%	10.35	32.43	82%
FM1024	11.36	35.29	11.75	35.21	10.23	35.31	80%	9.94	35.29	82%
FB1024	22.32	33.24	17.15	33.22	17.31	33.24	9%	17.29	33.25	21%
FG768	8.69	25.53	10.83	25.48	8.69	25.53	100%	8.65	25.54	98%
ST1024	12.45	27.93	13.00	27.88	11.81	27.87	66%	11.78	27.88	67%
Average	10.57	32.29	11.75	32.09	9.79	32.28		9.71	32.29	

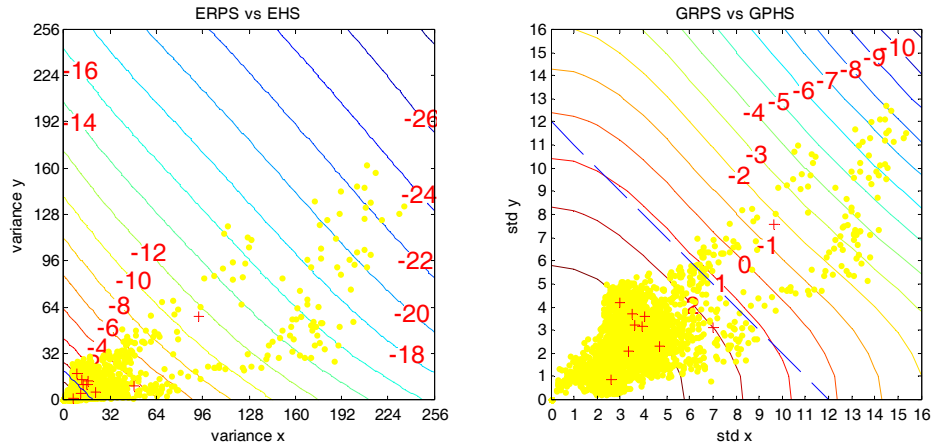


Fig. 4-10 Pattern switching threshold (dash line), I_{ASP} (solid line) and the frame MV variance/STD of the normal sequences.

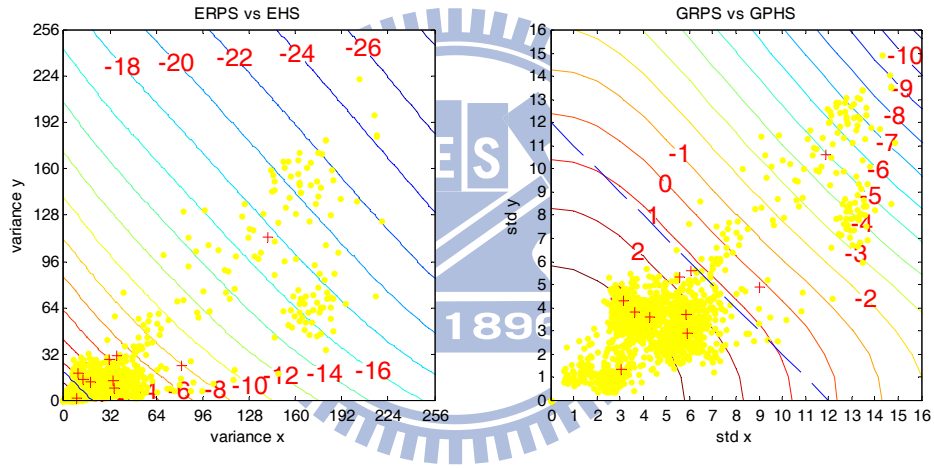


Fig. 4-11 Pattern switching threshold (dash line), I_{ASP} (solid line) and the MV variance/STD for the 2X sequences.

Section 4.3 Starting Point Selection

The impact of starting points or initial points on fast search algorithms have been studied by many researchers such as [14], [16], [30] and [31]. Typically the starting point is predicted by using a combination of the MVs of a few neighboring blocks. The most probable MV estimated by this type of MV *predictor* is used as the starting point for PBME algorithms. Although many MV predictors have been proposed, most of them are derived based on heuristic experiments. We like

to design a criterion that evaluates the effectiveness of MV predictors and propose a systematical approach that constructs the optimal *Starting Point Set* (SPS). The DL AGPS and DL APS discussed in the previous section are the search algorithms used to test our starting point set in this section.

We assume that the proposed PBME model (first method in Section 3.3 [54]) is valid for different starting point selection. Then, because the MV field acquired by FS is fixed for a given video sequence, a different starting point only does a translational shift on the motion vector distribution. Given two starting points, SP1 and SP2, their difference in *ASP* (E_{ASP}) can be represented by (4.11).

$$E_{ASP} = C_1 \times \sum_{x,y \in A} ((S_{FS\_SP1}(x,y) - S_{FS\_SP2}(x,y)) \times WF_{SA}(x,y)) \tag{4.11}$$

Let SP2 be a fixed starting point for comparison purpose; (4.11) thus becomes (4.12), in which η is a constant.

$$E_{ASP} = C_1 \times \sum_{x,y \in A} (S_{FS\_SP1}(x,y) \times WF_{SA}(x,y)) - \eta \tag{4.12}$$

Rearrange (4.12), we obtain G_{ASP} defined by (4.13), which is proportional to the ASP using SP1. Thus, it is used as the performance assessment criterion for starting point evaluation.

$$G_{ASP} = (E_{ASP} + \eta) / C_1 \tag{4.13}$$

Because *WF* is fixed for a specific algorithm and only $S_{FS\_SP1}(x,y)$ may vary, G_{ASP} is a function of MV characteristics. Herein, the MV characteristics are either the MV variances or MV standard deviations calculated from the MV w.r.t. a specific starting point (SP1). And the MVs are acquired by using FS on the selected sequence.

Fig. 4-12 shows the MV candidates that are often considered in starting point selection. They are the MVs of the neighboring spatial/temporal neighboring blocks. And the most commonly used mathematical function includes median(.) and mean(.). Combining them together, (4.14)-(4.25) are some representative MV predictors under our investigation.

$$MV^{pred1..14} = \text{one of} \begin{pmatrix} MV^U, MV^L, MV^{UL}, MV^{UR}, \\ MV^P, MV^{PL}, MV^{PR}, MV^{PU}, MV^{PD} \\ MV^{PUL}, MV^{PUR}, MV^{PDL}, MV^{PDR}, \text{or } MV^{PP} \end{pmatrix} \tag{4.14}$$

$$MV^{pred15} = (0,0) = ZMV \tag{4.15}$$

$$MV^{pred16} = median(MV^L, MV^U, MV^{UR}) = PMV \tag{4.16}$$

$$MV^{pred17} = mean(MV^L, MV^U, MV^{UR}) \tag{4.17}$$

$$MV^{pred18..19} = median \text{ or } mean(MV^L, MV^U, MV^{UL}) \tag{4.18}$$

$$MV^{pred20..21} = median \text{ or } mean(MV^L, MV^U, MV^P) \tag{4.19}$$

$$MV^{pred22..23} = median \text{ or } mean(MV^L, MV^U, MV^P, MV^P) \tag{4.20}$$

$$MV^{pred24} = MV^P + (MV^P - MV^{PP}) = AMV \tag{4.21}$$

$$MV^{pred25..26} = median \text{ or } mean(MV^P, MV^{PL}, MV^{PR}, MV^{PU}, MV^{PD}) \tag{4.22}$$

$$MV^{pred27..28} = median \text{ or } mean \begin{pmatrix} MV^P, MV^{PL}, MV^{PR}, MV^{PU}, MV^{PD}, \\ MV^{PUL}, MV^{PUR}, MV^{PDL}, MV^{PDR} \end{pmatrix} \tag{4.23}$$

Notation: A×(*a*,*b*) means (a×A_x, b×A_y), A is a vector, and (a,b) is a pair of scalar; for example,

$median(MV^L, MV^U, MV^{UR})\times((2,0))$ means $(2\times median(MV_x^L, MV_x^U, MV_x^{UR}),0)$

$$MV^{pred29..32} = median(MV^L, MV^U, MV^{UR})\times((1,0), (0,1), (2,0) \text{ or } (0,2)) \tag{4.24}$$

$$MV^{pred33..36} = max \begin{pmatrix} abs(median(MV_x^L, MV_x^U, MV_x^{UR})), \\ abs(median(MV_y^L, MV_y^U, MV_y^{UR})) \end{pmatrix} \times((1,0), (-1,0), (0,1), \text{or } (0,-1)) \tag{4.25}$$

Table 4-6 to **Table 4-9** show the G_{ASP} of some of the well-known and best performed MV predictors ((4.14)-(4.25)) applied to the test sequences using the weighting functions of ERPS, PHS, GRPS and GPHS, respectively. We find that MV^{pred21} (mean value of MV^U, MV^L, and MV^P), MV^{pred23} (mean value of MV^U, MV^L, and two MV^P) and MV^{pred28} (mean value of MV^{PU}, MV^{PD}, MV^{PL}, MV^{PR}, MV^{PUL}, MV^{PUR}, MV^{PDL}, MV^{PDR}, and MV^P) have the smallest average G_{ASP} among all the MV predictors. Together with the well-known PMV (MV^{pred16}, identical to (2.2)) and ZMV (MV^{pred15}, identical to (2.1)), these 5 MV predictors form the candidate set for the starting points.

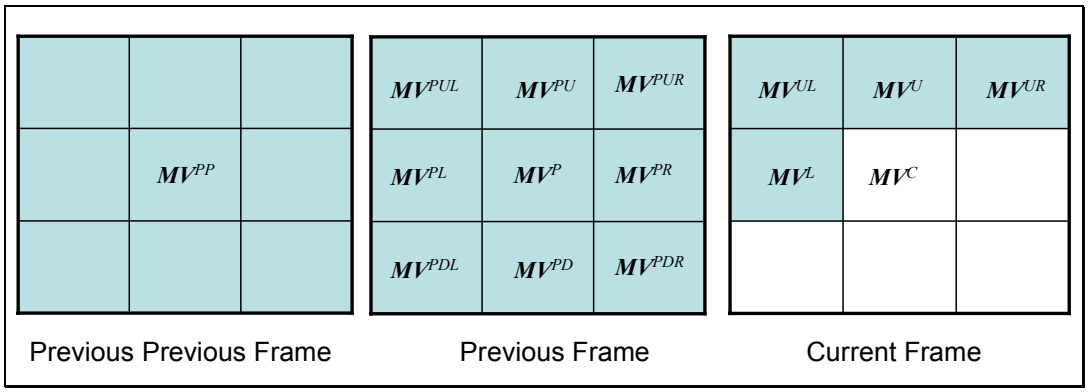


Fig. 4-12 Motion vector predictor candidates in the current frame, the previous frame and the frame before previous frame.

Table 4-6 G_{ASP} of the MV predictors applied to the test sequences using WF_{ERPS}.

MV	CT256	CT40	HL40	MD96	CG112	FM512	FM1024	FB1024	FG768	ST1024	Average
pred15	**6.22**	9.99	**9.87**	**10.07**	8.78	14.59	13.75	32.21	**9.56**	19.84	13.49
pred16	6.47	10.73	10.67	10.71	8.55	11.47	10.20	30.84	10.16	17.46	12.73
pred21	6.31	10.52	10.51	10.47	8.50	**10.66**	**9.47**	**27.92**	9.90	15.24	**11.95**
pred23	6.33	10.51	10.66	10.60	8.54	10.77	9.60	29.06	9.89	15.40	**12.14**
pred27	**6.22**	**9.98**	10.20	10.43	**8.20**	11.08	10.02	30.80	9.67	15.86	12.25
pred28	6.29	10.37	10.54	10.60	8.38	10.95	9.84	28.56	9.88	**15.15**	**12.06**
MIN	6.22	9.98	9.87	10.07	8.20	10.66	9.47	27.92	9.56	15.15	**11.95**

Table 4-7 G_{ASP} of the MV predictors applied to the test sequences using WF_{PHS}.

MV	CT256	CT40	HL40	MD96	CG112	FM512	FM1024	FB1024	FG768	ST1024	Average
pred15	**9.49**	**11.05**	**11.00**	**11.08**	10.55	13.03	12.66	21.59	**10.87**	15.59	12.69
pred16	9.59	11.36	11.34	11.35	10.45	11.67	11.14	20.88	11.12	14.41	12.33
pred21	9.53	11.27	11.27	11.25	10.43	**11.33**	**10.83**	**19.41**	11.01	13.36	**11.97**
pred23	9.54	11.27	11.33	11.31	10.44	11.38	10.89	19.98	11.01	13.43	**12.06**
pred27	**9.49**	**11.05**	11.13	11.23	**10.31**	11.51	11.06	20.87	10.91	13.66	12.12
pred28	9.52	11.21	11.28	11.31	10.38	11.45	10.98	19.73	11.00	**13.32**	**12.02**
MIN	9.49	11.05	11.00	11.08	10.31	11.33	10.83	19.41	10.87	13.32	**11.97**

Table 4-8 G_{ASP} of the MV predictors applied to the test sequences using WF_{GRPS}.

MV	CT256	CT40	HL40	MD96	CG112	FM512	FM1024	FB1024	FG768	ST1024	Average
pred15	**5.30**	**6.28**	**6.27**	**6.32**	5.96	7.70	7.43	14.39	**6.18**	9.27	7.51
pred16	5.36	6.48	6.50	6.50	5.90	6.73	6.36	13.87	6.35	8.53	7.26
pred21	5.32	6.43	6.45	6.43	5.89	**6.50**	**6.16**	**12.69**	6.28	7.83	**7.00**
pred23	5.32	6.42	6.50	6.47	5.90	6.53	6.20	13.15	6.28	7.88	**7.06**
pred27	**5.30**	**6.28**	6.36	6.42	**5.81**	6.62	6.31	13.83	6.21	8.01	7.11
pred28	5.31	6.38	6.46	6.47	5.85	6.58	6.26	12.93	6.27	**7.79**	**7.03**
MIN	5.30	6.28	6.27	6.32	5.81	6.50	6.16	12.69	6.18	7.79	**7.00**

Table 4-9 G_{ASP} of the MV predictors applied to the test sequences using WF_{GPHS}.

MV	CT256	CT40	HL40	MD96	CG112	FM512	FM1024	FB1024	FG768	ST1024	Average
pred15	**9.16**	**9.68**	**9.67**	**9.69**	9.52	10.34	10.22	13.20	**9.62**	11.20	10.23
pred16	9.20	9.79	9.78	9.78	9.48	9.89	9.71	12.96	9.71	10.80	10.11
pred21	9.18	9.76	9.76	9.75	9.48	**9.78**	**9.61**	**12.47**	9.67	10.45	**9.99**
pred23	9.18	9.76	9.78	9.77	9.48	9.79	9.63	12.66	9.67	10.48	**10.02**
pred27	**9.16**	**9.68**	9.71	9.74	**9.44**	9.84	9.69	12.96	9.64	10.55	10.04
pred28	9.17	9.74	9.76	9.77	9.46	9.82	9.66	12.58	9.67	**10.44**	**10.01**
MIN	9.16	9.68	9.67	9.69	9.44	9.78	9.61	12.47	9.62	10.44	**9.99**

We adopt the initial candidate set approach. That is, the proposed BME algorithm examines all MV candidates in the candidate set and then uses the best candidate as the starting point for the subsequent search procedure. As shown by (4.26), the total search point number (N_{TSP}) equals to the size of starting point set (N_{SPS}) plus the number of average search points (N_{ASP}) produced by a specific search algorithm minus one, where "minus one" represents the initial point count included in the N_{ASP}.

$$N_{TSP} = N_{SPS} + N_{ASP} - 1 \tag{4.26}$$

A well-designed starting point set should decrease N_{ASP} more than the increased size of starting point set (N_{SPS}). We develop a systematic approach to find the optimal SPS. It is an add-on approach. At the beginning, there is only one MV in the SPS. We calculate its N_{TSP} using a certain search algorithm. After a number of simulations, we retain a few best performers. We then add a second MV into each of these sets and evaluate their N_{TSP} again. We continue adding new points until the N_{TSP} does not decrease with additional MV in that set. This procedure is described by the flow chart in **Fig. 4-13**. In theory, this procedure does not guarantee that the final best set is globally optimal because our set is progressively constructed. However, our experiments indicate that the results are quite good.

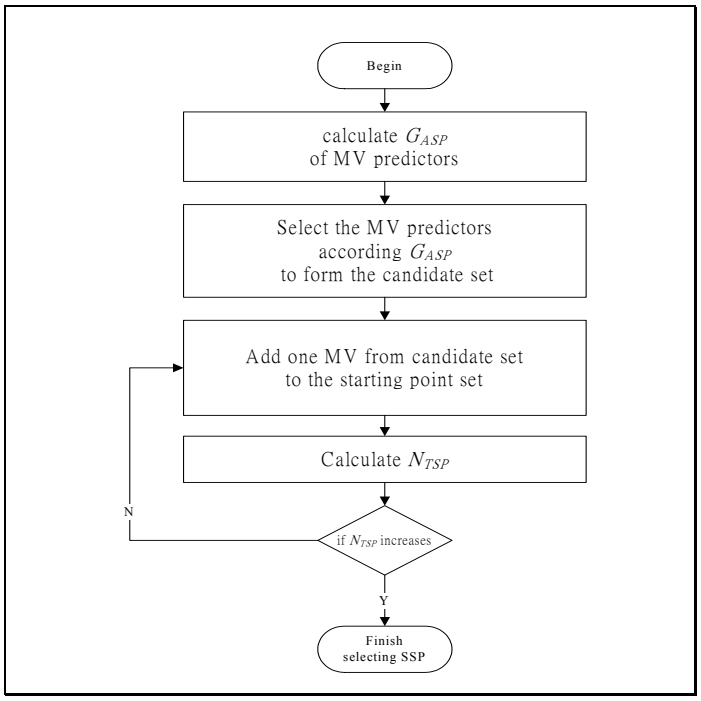


Fig. 4-13 The flow chart of constructing SPS.

We show the performance of DL APS with various starting point sets here. Due to limited space, only the better performed ones are shown. **Table 4-10** is the results of DL APS with one starting point. We find that DL APS with MV^{pred21}, MV^{pred23} or *PMV* are the best. Use each of these three MVs as the first element in three separated sets, we add a second MV. Their performance is shown in **Table 4-11**. The better performer for both speed and quality is the set of MV^{pred23} plus *PMV*. Based on this selection, we add one more MV into the starting point set and the results are on **Table 4-12**. The set of MV^{pred23} plus *PMV* and MV^{pred28} is the best. If we add one more MV into SPS, N_{TSP} increases. Therefore, our SPS for DL APS is $\{MV^{pred23}, PMV, MV^{pred28}\}$. The order in the set is the order in search. We repeat the same SPS identification procedure for DL AGPS and the best result is *PMV* plus MV^{pred23}, which gives an average ASP of 6.61. The experimental results used in constructing the set for DL AGPS are shown in **Table 4-13** and **Table 4-14**.

Table 4-10 The performance of DL APS with only one starting point.

Normal	PMV		Pred21		Pred23		Pred28		ZMV	
Sequence	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR
CT256	5.75	39.49	5.58	39.57	5.57	39.60	5.50	39.58	5.48	39.55
CT40	6.97	32.15	6.89	32.11	6.71	32.49	6.51	32.70	6.47	32.76
HL40	7.31	34.55	7.27	34.54	7.33	34.48	7.15	34.54	7.10	34.55
MD96	6.80	40.10	6.76	40.12	6.73	40.11	6.79	40.15	7.01	40.07

CG112	7.64	29.13	7.59	29.11	7.58	29.10	7.87	29.11	10.49	28.99
FM512	8.33	34.10	8.44	34.04	8.49	34.03	9.03	33.97	11.83	33.27
FM1024	8.07	36.65	8.24	36.53	8.28	36.53	8.85	36.49	11.71	36.00
FB1024	13.88	34.90	13.61	34.94	14.06	34.92	16.16	34.71	19.85	34.04
FG768	7.57	26.19	7.60	26.18	7.56	26.17	7.70	26.18	9.68	26.17
ST1024	9.68	29.30	9.57	29.37	9.41	29.44	10.06	29.49	13.81	28.23
Average	8.20	33.65	8.16	33.65	8.17	33.69	8.56	33.69	10.34	33.36

Table 4-11 The performance of DL APS when there are two points in the starting point set.

Normal	PMV +Pred21		PMV +Pred23		PMV +Pred28		PMV +ZMV		Pred21 +Pred23		Pred21 +PMV		Pred21 +Pred28		Pred21 +ZMV		Pred23 +Pred21		Pred23 +PMV		Pred23 +Pred28		Pred23 +ZMV	
Sequence	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**
CT256	5.52	39.57	5.50	39.55	5.47	39.56	5.45	39.57	5.52	39.61	5.51	39.53	5.46	39.61	5.45	39.55	5.53	39.57	5.49	39.59	5.45	39.63	5.45	39.56
CT40	6.53	32.45	6.43	32.69	6.40	32.69	6.39	32.74	6.58	32.61	6.53	32.50	6.42	32.71	6.38	32.72	6.57	32.63	6.46	32.68	6.41	32.72	6.36	32.73
HL40	7.00	34.56	6.99	34.51	6.95	34.56	6.98	34.54	7.17	34.50	7.00	34.56	6.99	34.58	6.98	34.59	7.16	34.55	6.98	34.52	7.00	34.56	6.98	34.56
MD96	6.56	40.10	6.56	40.09	6.55	40.11	6.62	40.12	6.66	40.11	6.56	40.11	6.59	40.11	6.64	40.13	6.66	40.11	6.52	40.10	6.60	40.10	6.63	40.11
CG112	7.00	29.12	6.96	29.12	6.97	29.13	7.71	29.12	7.37	29.10	7.01	29.12	7.13	29.12	7.75	29.09	7.36	29.11	6.94	29.12	7.13	29.12	7.78	29.06
FM512	7.86	34.10	7.78	34.12	7.92	34.11	8.48	34.05	8.33	34.03	7.86	34.08	8.23	34.06	8.77	33.98	8.29	34.04	7.80	34.13	8.28	34.06	8.81	33.97
FM1024	7.63	36.60	7.56	36.50	7.71	36.61	8.24	36.55	8.10	36.54	7.62	36.57	8.02	36.51	8.58	36.50	8.11	36.54	7.57	36.64	8.12	36.56	8.60	36.50
FB1024	12.70	34.98	12.48	35.02	12.94	34.99	14.55	34.91	13.64	34.94	12.70	35.00	13.65	34.91	15.41	34.75	13.67	34.96	12.52	35.05	14.03	34.85	15.61	34.71
FG768	7.09	26.21	7.05	26.21	7.00	26.19	7.34	26.18	7.49	26.17	7.08	26.18	7.25	26.18	7.60	26.19	7.49	26.18	7.06	26.20	7.24	26.18	7.58	26.18
ST1024	8.83	29.48	8.71	29.46	8.77	29.52	10.06	29.23	9.41	29.36	8.87	29.44	9.15	29.43	10.42	29.09	9.41	29.33	8.71	29.44	9.05	29.43	10.38	29.13
Average	7.67	33.72	7.60	33.73	7.67	33.75	8.18	33.70	8.03	33.70	7.67	33.71	7.89	33.72	8.40	33.66	8.03	33.70	7.61	33.75	7.93	33.72	8.42	33.65

Table 4-12 The performance of DL APS when there are three points in the starting point set, MV^{pred23} is the first starting point, and PMV is the second starting point.

Normal	Pred23+PMV+Pred21		Pred23+PMV+Pred28		Pred23+PMV+ZMV	
Sequence	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**
CT256	5.48	39.61	5.45	39.53	5.44	39.57
CT40	6.44	32.64	6.36	32.69	6.31	32.74
HL40	6.97	34.58	6.91	34.54	6.90	34.53
MD96	6.50	40.17	6.49	40.14	6.53	40.18
CG112	6.90	29.12	6.79	29.12	7.29	29.11
FM512	7.80	34.10	7.80	34.12	8.13	34.08
FM1024	7.57	36.62	7.59	36.59	7.92	36.59
FB1024	12.79	35.01	12.81	35.02	13.60	34.90
FG768	7.06	26.19	6.93	26.19	7.20	26.19
ST1024	8.73	29.47	8.59	29.55	9.39	29.39
Average	7.62	33.75	7.57	33.75	7.87	33.73

Table 4-13 The performance of DL AGPS with only one starting point.

Normal	PMV		Pred21		Pred23		Pred28		ZMV	
Sequence	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**	**ASP**	**PSNR**
CT256	5.36	39.51	5.26	39.55	5.26	39.56	5.22	39.55	5.22	39.62
CT40	6.04	32.07	5.83	32.47	5.78	32.61	5.71	32.67	5.70	32.70
HL40	6.35	34.45	6.34	34.42	6.35	34.46	6.25	34.50	6.22	34.52
MD96	5.98	40.04	5.95	40.04	5.94	40.07	5.95	40.11	6.09	40.04
CG112	6.08	29.11	6.04	29.10	6.11	29.04	6.22	29.09	7.77	28.96
FM512	7.10	34.04	7.10	33.97	7.11	33.96	7.42	33.90	9.66	33.20
FM1024	6.93	36.53	6.98	36.49	6.98	36.45	7.29	36.41	9.60	35.98
FB1024	11.64	34.82	11.61	34.83	11.95	34.72	13.55	34.53	16.50	33.99
FG768	6.37	26.17	6.33	26.16	6.31	26.16	6.27	26.18	7.34	26.16
ST1024	7.62	29.43	7.31	29.43	7.23	29.47	7.54	29.46	10.37	28.22

Average	6.95	33.62	6.88	33.64	6.90	33.65	7.14	33.64	8.45	33.34

Table 4-14 The performance of DL AGPS when there are two points in the starting point set.

Normal	Pred21 +Pred23		Pred21 +PMV		Pred21 +Pred28		Pred21 +ZMV		PMV +Pred21		PMV +Pred23		PMV +Pred28		PMV +ZMV		Pred23 +Pred21		Pred23 +PMV		Pred23 +Pred28		Pred23 +ZMV	
Sequence	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR
CT256	5.24	39.54	5.26	39.55	5.22	39.57	5.21	39.58	5.25	39.52	5.24	39.62	5.22	39.59	5.22	39.60	5.24	39.59	5.24	39.56	5.21	39.56	5.20	39.55
CT40	5.76	32.60	5.76	32.52	5.69	32.68	5.68	32.73	5.78	32.40	5.72	32.67	5.69	32.61	5.67	32.73	5.75	32.62	5.72	32.61	5.68	32.70	5.67	32.72
HL40	6.30	34.53	6.26	34.53	6.21	34.51	6.21	34.55	6.24	34.51	6.25	34.54	6.21	34.57	6.20	34.55	6.30	34.54	6.24	34.53	6.23	34.57	6.21	34.55
MD96	5.95	40.07	5.89	40.07	5.90	40.10	5.95	40.08	5.91	40.08	5.91	40.12	5.89	40.09	5.94	40.09	5.92	40.11	5.88	40.14	5.90	40.10	5.93	40.09
CG112	5.98	29.11	5.89	29.13	5.94	29.12	6.40	29.08	5.89	29.12	5.88	29.12	5.89	29.11	6.41	29.11	5.99	29.11	5.87	29.11	5.97	29.11	6.43	29.08
FM512	7.05	34.00	6.82	34.10	7.03	34.02	7.52	33.91	6.84	34.07	6.78	34.09	6.89	34.08	7.39	34.01	7.03	34.00	6.78	34.09	7.06	33.99	7.52	33.91
FM1024	6.92	36.49	6.68	36.53	6.89	36.51	7.40	36.48	6.65	36.54	6.61	36.57	6.74	36.55	7.21	36.56	6.91	36.51	6.62	36.57	6.95	36.54	7.41	36.47
FB1024	11.61	34.80	10.66	35.00	11.69	34.86	13.00	34.69	10.61	34.95	10.55	35.02	10.85	35.00	12.23	34.80	11.65	34.86	10.49	35.03	11.93	34.83	13.11	34.69
FG768	6.28	26.18	6.13	26.18	6.16	26.19	6.39	26.18	6.14	26.18	6.14	26.20	6.09	26.19	6.35	26.19	6.28	26.16	6.11	26.18	6.15	26.18	6.38	26.18
ST1024	7.20	29.45	7.19	29.48	7.17	29.44	8.27	29.13	7.15	29.49	7.11	29.46	7.19	29.52	8.19	29.20	7.24	29.41	7.13	29.47	7.16	29.51	8.25	29.14
Average	6.83	33.68	6.65	33.71	6.79	33.70	7.20	33.64	6.65	33.69	6.62	33.74	6.67	33.73	7.08	33.68	6.83	33.69	6.61	33.73	6.82	33.71	7.21	33.64

Table 4-15 is a comparison of DL APS and DL AGPS with and without SPS. As discussed earlier, "DL APS + SPS" uses the 3-point SPS, "DL AGPS + SPS" uses the 2-point SPS, and the other algorithms use only PMV as the sole starting point. We find that DL APS with SPS outperforms DL APS by 8.3% in ASP (0.09dB PSNR gain). The DL AGPS with SPS outperforms DL AGPS by 5.0% in ASP (0.12dB PSNR gain).

In summary, the best SPS we identify for DL APS is $\{MV^{pred23}, PMV, MV^{pred28}\}$, and the best SPS for DL AGPS, $\{PMV, MV^{pred23}\}$. With SPS, DL APS and DL AGPS can reduce ASP with slightly increased PSNR. In these two cases, the SPS size of DL AGPS is smaller. Our conjecture is that a fast-moving pattern search needs only a small SPS because the search algorithm can cover a large search area quickly without the help of additional starting points. The experiments also indicate that a 2-point (or 3-point) SPS is generally better than the single-point SPS (PMV).

Table 4-15 The effects of SPS on DL APS and DL AGPS.

Normal	DL APS		DL APS + SPS		DL AGPS		DL AGPS + SPS	
Sequence	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR
CT256	5.75	39.49	5.45	39.53	5.36	39.51	5.24	39.62
CT40	6.97	32.15	6.36	32.69	6.04	32.07	5.72	32.67
HL40	7.31	34.55	6.91	34.54	6.35	34.45	6.25	34.54

MD96	6.8	40.1	6.49	40.14	5.98	40.04	5.91	40.12
CG112	7.64	29.13	6.79	29.12	6.08	29.11	5.88	29.12
FM512	8.33	34.1	7.8	34.12	7.1	34.04	6.78	34.09
FM1024	8.07	36.65	7.59	36.59	6.93	36.53	6.61	36.57
FB1024	13.88	34.9	12.81	35.02	11.64	34.82	10.55	35.02
FG768	7.57	26.19	6.93	26.19	6.37	26.17	6.14	26.2
ST1024	9.68	29.3	8.59	29.55	7.62	29.43	7.11	29.46
Average	8.2	33.65	7.57	33.75	6.95	33.62	6.62	33.74

Section 4.4 Early Termination Mechanism

The *early termination* mechanism terminates the search process when the block-matching error produced by a MV (in the search area) is smaller than a pre-chosen threshold. And in this case, this MV is accepted as the best MV. Clearly, there is a trade-off between the MV quality (matching error) and the computational speed. Thus, the challenge is to find the termination threshold that maximizes the speed gain and minimizes the quality degradation. In this section, we set up a systematic method to find the nearly optimal early termination threshold (ETT) [50].

The most commonly used block matching error is the sum of absolute difference (*SAD*). Due to the correlation among the spatial/temporal nearby blocks, [14] proposed a general form (4.27) for ETT. It suggests that the threshold is a function of the SAD and the MV of the neighboring blocks.

$$T = \min\left\{\max\left\{f\left(\begin{matrix}SAD_1,\ldots,SAD_i,\ldots,SAD_n,\\ MV_1,\ldots,MV_i,\ldots MV_n\end{matrix}\right),T_{\min}\right\},T_{\max}\right\}, \tag{4.27}$$

where SAD_i and MV_i, respectively, are the *SAD* and MV of a neighboring block labeled by i, and T_{min} and T_{max} stand for the lower and the upper bounds of the threshold, respectively. In practice, most researches use only the SAD predictor. For example, [16] suggests (4.28) and [30] suggest (4.29).

$$T = a \times \min(SAD_1, SAD_2, ..., SAD_n) + b\,, \tag{4.28}$$

where a, b are fixed values.

$$T = SAD^p + \delta\ , \tag{4.29}$$

where SAD^p is the SAD of the co-located block in the previous frame (**Fig. 4-14**) and δ is a bias parameter.

To find the best threshold predictor, we use the correlation coefficient between the SAD predictor (SAD^{pred}) and the best SAD acquired using FS (SAD^c, as shown in **Fig. 4-14**) as the measure for the effectiveness of this threshold. First, we perform FS on the test sequences in **Table 3-1** to obtain the SAD values of all blocks. For each of the SAD predictors, we calculate its correlation with the actual SAD (SAD^C) of the corresponding block. The one with highest correlation coefficient (closer to 1) is the best SAD predictor. By using the regression method, we find an approximation function (predictor) that best describes the relation between the predicted SAD and SAD^C. Also, we set up an upper bound for the threshold estimate to prevent the quality loss in the high ETT cases. And at last, we fine-tune the predictor coefficients (slope and offset) to achieve the desired speed and quality trade-off. This fine-tuned function thus serves as the early termination threshold.

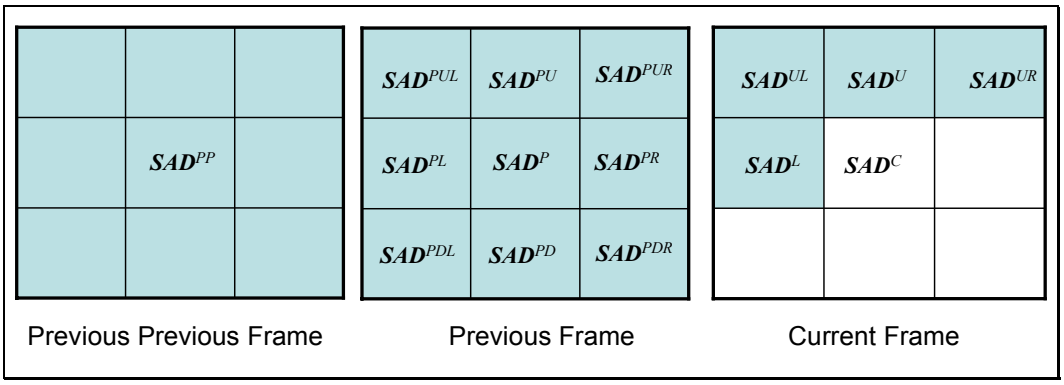


Fig. 4-14 The SAD candidates in the current frame, the previous frame and the frame before the previous frame.

An ETT predictor often consists of two elements: 1) a selected SAD set of nearby blocks, and 2) a mathematical function operating on the selected SAD set. The most commonly used

mathematical functions are mean(.), median(.), min(.) and max(.). The most commonly used 14 neighboring SADs are shown in **Fig. 4-14**. Combining them together, there are 65532 possibilities. $((\sum_{i=1}^{14} C_i^{14}) \times 4 = (2^{14} - C_0^{14}) \times 4 = 65532)$. Moreover, we can insert different weighting before each block SAD, which leads to enormous forms of the SAD predictors. In our study, we select some representative SAD predictors ((4.30)-(4.43)). **Table 4-16** shows the correlation coefficient between a few selected SAD predictors and SAD^C. Limited by space, only the better ones are shown there. Among the 55 SAD predictors under consideration, SAD^{pred15} (mean SAD of the upper and left blocks) is the best predictor in 2D cases and SAD^{pred35} (median SAD of the upper, left, and two previous blocks) is the best predictor in all cases (2D and 3D cases). Herein, the 2D cases only use the SADs of the blocks in the same frame, and the 3D cases can also use the SADs of the blocks in the current frame and the previous frame SADs.

$$SAD^{pred1..14} = \text{one of} \begin{pmatrix} SAD^U, SAD^L, SAD^{UL}, SAD^{UR}, \\ SAD^P, SAD^{PL}, SAD^{PR}, SAD^{PU}, SAD^{PD}, \\ SAD^{PUL}, SAD^{PUR}, SAD^{PDL}, SAD^{PDR}, or \cdot SAD^{PP} \end{pmatrix} \quad (4.30)$$

$$SAD^{pred15..17} = mean,\ min, \text{or } max(SAD^U, SAD^L) \quad (4.31)$$

$$SAD^{pred18..21} = mean, median, min, \text{or } max(SAD^U, SAD^L, SAD^{UL}) \quad (4.32)$$

$$SAD^{pred22..25} = mean, median, min, \text{or } max(SAD^U, SAD^L, SAD^{UR}) \quad (4.33)$$

$$SAD^{pred26..29} = mean, median, min, \text{or } max(SAD^U, SAD^L, SAD^{UL}, SAD^{UR}) \quad (4.34)$$

$$SAD^{pred30..33} = mean,\ median,\ min, \text{ or } max(SAD^P, SAD^U, SAD^L) \quad (4.35)$$

$$SAD^{pred34..35} = mean \text{ or } median(SAD^P, SAD^P, SAD^U, SAD^L) \quad (4.36)$$

$$SAD^{pred36..37} = mean \text{ or } median(SAD^P, SAD^P, SAD^P, SAD^U, SAD^L) \quad (4.37)$$

$$SAD^{pred38..41} = mean,\ median,\ min, \text{ or } max(SAD^P, SAD^{PU}, SAD^{PD}, SAD^{PL}, SAD^{PR}) \quad (4.38)$$

$$SAD^{pred42..45} = mean,\ median,\ min, \text{ or } max\begin{pmatrix} SAD^U, SAD^L, SAD^{UL}, SAD^{UR}, \\ SAD^P, SAD^{PU}, SAD^{PD}, SAD^{PL}, SAD^{PR} \end{pmatrix} \quad (4.39)$$

$$SAD^{pred46..48} = mean,\ min, \text{ or } max\begin{pmatrix} mean(SAD^U, SAD^L, SAD^{UL}, SAD^{UR}), \\ mean(SAD^P, SAD^{PU}, SAD^{PD}, SAD^{PL}, SAD^{PR}) \end{pmatrix} \quad (4.40)$$

$$SAD^{pred49..51} = mean,\ \ min, \text{ or } max\begin{pmatrix} median(SAD^U, SAD^L, SAD^{UL}, SAD^{UR}), \\ median(SAD^P, SAD^{PU}, SAD^{PD}, SAD^{PL}, SAD^{PR}) \end{pmatrix} \quad (4.41)$$

$$SAD^{pred52..53} = mean, \text{ or } max\begin{pmatrix} \min(SAD^U, SAD^L, SAD^{UL}, SAD^{UR}), \\ \min(SAD^P, SAD^{PU}, SAD^{PD}, SAD^{PL}, SAD^{PR}) \end{pmatrix} \quad (4.42)$$

$$SAD^{pred\,54..55} = mean, \text{ or } min\begin{pmatrix}\max(SAD^{U}, SAD^{L}, SAD^{UL}, SAD^{UR}),\\ \max(SAD^{P}, SAD^{PU}, SAD^{PD}, SAD^{PL}, SAD^{PR})\end{pmatrix} \quad (4.43)$$

Table 4-16 The correlation coefficients between the selected SAD predictors and the actual block SAD.

SAD	CT256	CT40	HL40	MD96	CG112	FM512	FM1024	FB1024	FG768	ST1024	Average	All
pred15	**0.725**	**0.809**	**0.760**	0.711	**0.767**	**0.748**	**0.743**	**0.727**	**0.926**	**0.844**	**0.776**	**0.886**
pred16	0.698	0.754	0.698	**0.715**	0.675	0.658	0.651	0.653	0.887	0.747	0.714	0.831
pred30	0.856	0.908	0.891	0.850	0.883	0.835	0.825	**0.765**	0.958	0.908	0.868	0.932
pred34	0.892	0.938	0.934	0.890	0.918	0.850	**0.835**	0.753	0.966	0.924	0.890	0.942
pred35	**0.906**	**0.957**	0.961	**0.914**	**0.938**	0.848	0.828	0.714	**0.970**	**0.930**	**0.897**	**0.945**
pred36	0.903	0.950	0.951	0.905	0.931	**0.851**	0.832	0.737	0.968	0.928	0.896	0.945
pred37	0.888	0.957	**0.974**	0.904	0.936	0.813	0.781	0.635	0.960	0.914	0.876	0.931

To produce a better SAD predictor on SAD^{C}, we have tried the multi-dimensional regression method. But we find that the linear regression is sufficient to have a pretty accurate approximation. Consequently, (4.44) is the predictor of choice.

$$SAD_{th}^{Linear\_predicted} = K_1 \times SAD^{pred} + K_2, \quad (4.44)$$

Table 4-17 shows the coefficients of the best 2D/3D predictors for various test sequences. The 'Average' row denotes the average values of all sequences. The 'All' row shows the values calculated using all sequences as data samples. To check the effectiveness of these predictors, we calculate the mean and the standard deviation (STD) of both the best 2D and 3D SAD prediction errors. In **Fig. 4-15** and **Fig. 4-16**, each dot represents the SAD pair (SAD^{pred}, SAD^{C}) of a block. The star mark at the center of a vertical bar represents the mean of SAD^{C}, and the bar length represents the standard deviation of prediction errors. It is obvious that the standard deviation becomes larger as the value of SAD^{pred} increases. This implies that for large predicted SAD values, their prediction accuracy is lower. Hence, to ensure a high MV quality, we propose an upper bound in (4.45) using the average SAD of all coded block in the same frame.

$$SAD_{th}^{Upper\_bounded} = \frac{\sum_{i=1}^{N_C-1} SAD_i}{N_C - 1} + K_3, \quad (4.45)$$

where SAD_i is the SAD of the i-th block in the current frame, K_3 is the allowed maximum early termination error offset, and N_c denotes the current block index in a frame. Finally, the early termination threshold (ETT) is defined below by (4.46).

$$T = SAD_{th} \equiv \min(SAD_{th}^{Linear\_predicted}, SAD_{th}^{Upper\_bounded}), \tag{4.46}$$

The parameter values are empirically decided: K_1 is set to 1, K_2 is set to 384 and K_3 is set to 512. Under this setting, we achieve a good balance between speed and quality.

Table 4-17 Regression coefficients for the best 2D and 3D SAD predictors.

Predictor	Pred 15 (best 2D)		Pred35 (best 3D)	
	K_1	K_2	K_1	K_2
CT256	0.84	77.20	0.98	13.83
CT40	0.92	95.43	1.02	-11.95
HL40	0.88	90.38	1.04	-27.50
MD96	0.80	83.77	0.96	27.77
CG112	0.86	320.99	0.98	70.65
FM512	0.81	249.69	0.85	192.79
FM1024	0.79	239.74	0.83	200.46
FB1024	0.69	549.12	0.64	660.09
FG768	0.99	216.38	0.97	146.71
ST1024	0.95	165.75	0.93	202.48
Average	0.85	208.85	0.92	147.53
All	0.97	66.65	0.96	76.53

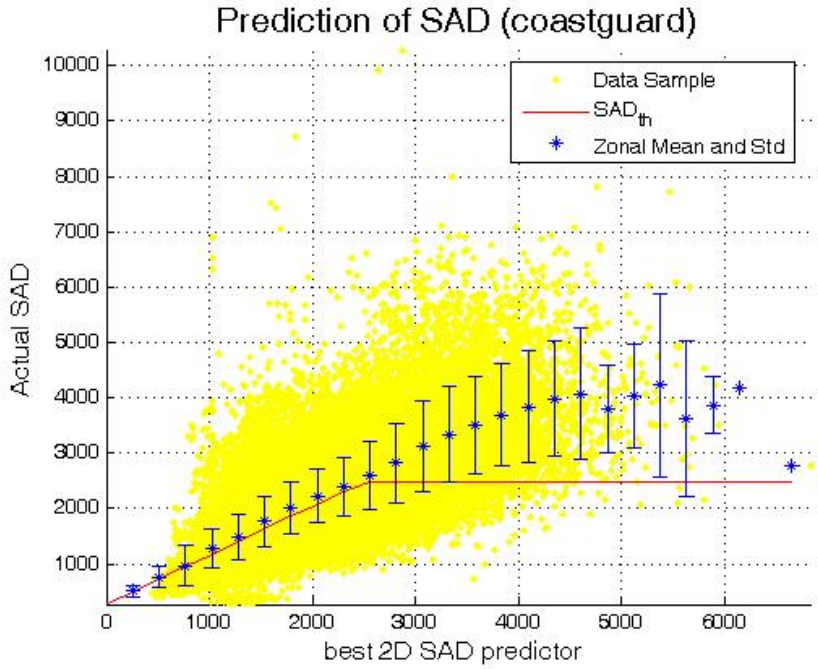


Fig. 4-15 Best 2D SAD predictor versus SAD^C

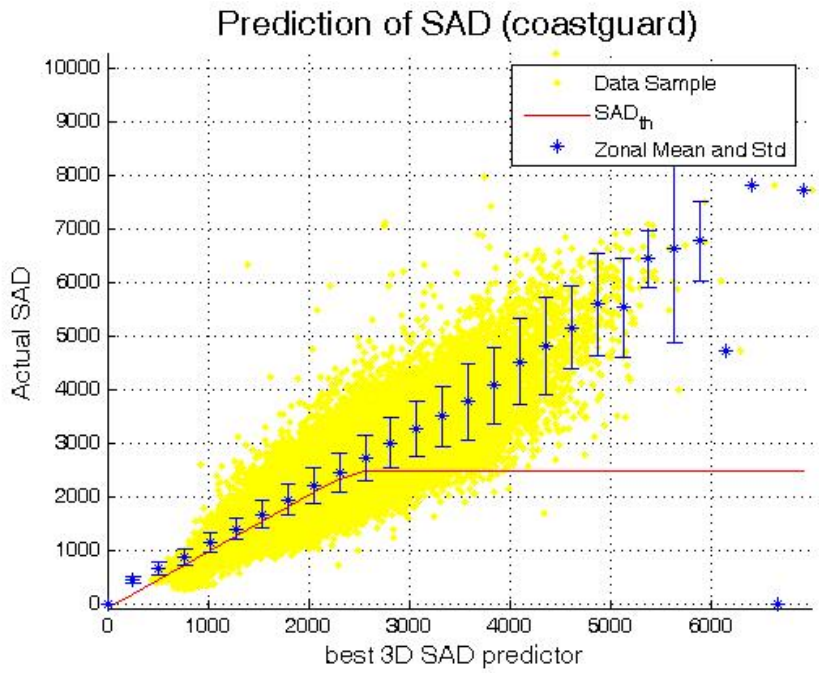


Fig. 4-16 Best 3D SAD predictor versus SAD^C

The computational overhead of our proposed early termination mechanism is negligible when compared to the speed gain. In the memory requirement, it only needs to record the SAD of roughly a row of blocks in the 2D case and the SAD of roughly a frame of blocks in the 3D case. As for the computing power requirement, it needs a few 'compare', one 'shift', one 'multiply', and one 'divide' operations for each block.

Table 4-18 shows the performance of DL AGPS with SPS and several early termination mechanisms. As suggested by their proponents, parameter a is set to 1.2 and b is set to 128 in (4.28), and δ is set to 50 in (4.29). We find that the DL AGPS with our best 2D ETT outperforms the plain DL AGPS scheme by 154% in average search points (0.02dB PSNR gain), and it outperforms (4.28) by 10% (0.01dB PSNR loss) and outperforms (4.29) by 11% (0dB PSNR gain). And the DL AGPS with our best 3D ETT outperforms the plain DL AGPS scheme by 162% in average search points (0.02dB PSNR gain), and it outperforms (4.28) by 14% (0.01dB PSNR loss), outperforms (4.29) by 15% (0dB PSNR gain) and finally it further outperforms our best 2D ETC by 4% (0dB PSNR gain).

Table 4-18 The performance of DL AGPS with SPS and various early termination mechanisms.

Normal	NO ETT		(4.28)		(4.29)		Best 2D ETT		Best 3D ETT	
Sequence	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR
CT256	5.24	39.62	1.59	39.59	1.57	39.54	1.38	39.55	1.36	39.63
CT40	5.72	32.67	2.00	32.85	1.63	32.83	1.71	32.89	1.63	32.87
HL40	6.25	34.54	2.14	35.05	2.01	34.90	1.64	35.05	1.56	35.03
MD96	5.91	40.12	1.89	40.24	1.78	40.25	1.52	40.23	1.48	40.21
CG112	5.88	29.12	2.64	29.02	2.91	29.09	2.43	29.01	2.23	28.99
FM512	6.78	34.09	3.27	33.98	3.34	34.00	2.70	33.91	2.54	33.93
FM1024	6.61	36.57	3.15	36.52	3.34	36.53	2.57	36.48	2.48	36.49
FB1024	10.55	35.02	5.44	34.78	5.88	34.95	5.06	34.79	5.09	34.85
FG768	6.14	26.20	3.08	26.18	2.81	26.18	3.98	26.17	3.84	26.18
ST1024	7.11	29.46	3.40	29.47	3.76	29.33	3.15	29.52	2.97	29.40
Average	6.62	33.74	2.86	33.77	2.90	33.76	2.61	33.76	2.52	33.76

We also test our proposed ETT on *outside* sequences, which are sequences not in the training set. These 4 extra sequences and their settings are in **Table 4-19**. The performance of DL AGPS with SPS and various early termination mechanisms on these sequences is shown in **Table 4-20**.

Table 4-19 The extra sequences and their settings.

Abbreviation	Sequence	Bitrate (K bps)	Frame rate (fps)	Number of frames
st96	silent	96	10	300
tt512	table tennis	512	30	300
mb1024	mobile calendar	1024	30	300
ne40	news	40	7.5	90

In **Table 4-20**, we find that the DL AGPS with our best 2D ETT outperforms the plain DL AGPS scheme by 151.1% in average search points (0.03dB PSNR loss) and it outperforms (4.28) by 11.6% (0.03dB PSNR gain). And it has about the same performance as (4.29) in both speed and quality. And the DL AGPS with our best 3D ETT outperforms the plain DL AGPS scheme by 166.8% in average search points (0.06dB PSNR loss), and it outperforms (4.28) by 18.5% (0.00dB PSNR loss), outperforms (4.29) by 6.3% (0.03dB PSNR loss) and outperforms our best 2D ETT by 6.3% (0.03dB PSNR loss). Overall, the results of the outside sequences are consistent with the training sequences and, therefore, the proposed ETT is rather effective.

Table 4-20 The performance of DL AGPS with SSP and various early termination mechanisms on the extra sequences in **Table 4-19**.

Normal	NO ETC		(4.28)		(4.29)		Best 2D ETT		Best 3D ETT	
Sequence	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR	ASP	PSNR
st96	5.86	35.25	2.34	35.26	2.06	35.27	2.00	35.28	1.98	35.28
tt512	6.02	35.15	2.23	35.02	2.31	35.08	2.07	35.05	2.03	35.01
mb1024	5.32	27.54	2.93	27.52	3.02	27.53	2.87	27.52	2.53	27.51
ne40	5.40	34.49	2.54	34.40	1.61	34.42	2.06	34.46	1.93	34.39
Average	5.65	33.11	2.51	33.05	2.25	33.07	2.25	33.08	2.12	33.05

Section 4.5 A PBME Algorithm with All Features

We have discussed in the previous three sections three techniques that reduce computations of a PBME algorithm. They are 1) adaptive genetic pattern search, 2) starting point set and 3) early termination mechanism. We now examine the performance of the PBME scheme with all the best selected techniques.

Table 4-21 The ASP performance of FS, DS, AIPS-MP, ARPS-ZMP and our proposed best algorithm.

Type	Sequence	FS	DS	AIPS-MP	ARPS-ZMP	Ours
1X	CT256	1,024	13.81	1.37	3.58	1.36
	CT40	1,024	15.03	1.64	5.59	1.63
	HL40	1,024	15.38	1.62	5.14	1.56
	MD96	1,024	14.85	1.70	3.62	1.48
	CG112	1,024	15.09	2.96	9.88	2.22
	FM512	1,024	16.17	3.64	9.59	2.55
	FM1024	1,024	15.76	3.55	9.22	2.49
	FB1024	1,024	22.36	7.78	18.86	5.06
	FG768	1,024	15.30	5.04	7.07	3.84
	ST1024	1,024	16.96	4.54	10.63	2.98
2X	CT256	1,024	14.15	1.43	4.36	1.42
	CT40	1,024	16.05	1.77	6.71	1.75
	HL40	1,024	15.62	1.88	5.53	1.74
	MD96	1,024	15.44	2.35	4.82	1.89
	CG112	1,024	17.04	4.45	11.95	2.90
	FM512	1,024	18.72	6.07	13.31	3.89
	FM1024	1,024	18.26	5.92	12.99	3.83
	FB1024	1,024	27.39	10.16	23.35	7.15
	FG768	1,024	16.30	6.73	8.57	4.24
	ST1024	1,024	19.49	5.73	13.14	4.07
	Average	1,024	16.96	4.02	9.40	2.90

Table 4-22 The PSNR performance of FS, DS, AIPS-MP, ARPS-ZMP and our proposed best algorithm.

Type	Sequence	FS	DS	AIPS-MP	ARPS-ZMP	Ours
1X	CT256	39.57	39.51	39.62	39.59	39.62
	CT40	32.04	31.92	32.90	32.88	32.88
	HL40	33.55	34.25	35.07	34.87	35.05
	MD96	39.80	40.00	40.15	40.25	40.22
	CG112	29.08	29.14	28.48	29.15	29.03
	FM512	34.06	34.06	33.56	34.00	33.92
	FM1024	36.56	36.58	36.31	36.51	36.50
	FB1024	35.28	34.93	34.01	34.82	34.82
	FG768	26.20	26.18	26.17	26.19	26.18
	ST1024	29.48	29.44	29.40	29.18	29.40
2X	CT256	38.95	38.60	38.99	39.01	38.94
	CT40	29.81	29.94	31.18	31.16	31.12
	HL40	32.33	33.08	33.88	33.68	33.91
	MD96	38.41	38.60	38.63	38.77	38.73
	CG112	27.36	27.51	26.54	27.45	27.28
	FM512	32.42	32.38	31.86	32.37	32.23
	FM1024	35.28	35.24	35.01	35.20	35.20
	FB1024	33.44	33.28	32.83	33.12	33.15
	FG768	25.51	25.53	25.50	25.54	25.52
	ST1024	28.11	27.96	27.67	27.68	27.91
	Average	32.86	32.91	32.89	33.07	33.08

Table 4-23 The sizes (number of bytes) of the coded bitstreams by FS, DS, AIPS-MP, ARPS-ZMP and our proposed best algorithm.

Type	Sequence	FS	DS	AIPS-MP	ARPS-ZMP	Ours
1X	CT256	1138576	1154328	1148264	1156088	1148108
	CT40	207006	206694	206660	206858	206576
	HL40	209118	208978	207572	207758	207434
	MD96	369588	369794	370496	369780	370022
	CG112	433944	433866	433836	433932	434006
	FM512	653302	654332	654126	654218	654466
	FM1024	1269206	1279634	1280938	1275726	1277238
	FB1024	390320	388370	393910	390680	391420
	FG768	822462	822476	822514	822486	822486
	ST1024	1149726	1164650	1216952	1164104	1174556
2X	CT256	646044	625926	648724	637682	641638
	CT40	105318	104618	104460	104714	104304
	HL40	106892	106708	106414	106562	106550
	MD96	185516	185396	186060	185458	185552
	CG112	219324	219258	230076	219274	219240
	FM512	328032	327970	327630	327956	328200
	FM1024	644290	642322	646118	641534	642620
	FB1024	193800	195098	197144	197026	197410
	FG768	412942	412944	412902	412930	412976
	ST1024	618942	626372	634250	621654	628812
	Average	505217	506487	511452	506821	507681

The performance of FS, DS [27][28], ARPS-ZMP[30], AIPS-MP[31] and the DL AGPS with SPS and the 3D ETT (our proposed best algorithm with all features) are shown in **Table 4-21**, **Table 4-22**, and **Table 4-23**. Experimental results show that the proposed best algorithm outperforms ARPS-ZMP by 224% in average search points, AIPS-MP by 38%, DS by 485%, and FS by 353 times while the average PSNR quality is slightly better (0.01dB~0.22dB) than all the other algorithms including FS and the average sizes of the coded bitstreams are very similar (-0.49%~+0.74%). This may be due to the fact that our scheme often prefers a smaller value MV, which requires fewer bits in coding. Thus, a few additional bits are available for texture (DCT coefficients) coding, which results in better overall PSNR.

For our proposed algorithm with all the best techniques, each component contributes to the overall computation gain and the PSNR quality. In average, the adaptive pattern search outperforms its constituent pattern searches up to 34% with roughly the same PSNR quality, the optimal starting point set further provides 5% computation gain with 0.1dB PSNR increment, and the early termination mechanism offers up to 167% computation acceleration and roughly the same PSNR quality. Clearly, the early termination mechanism provides the most gain in the computation complexity, and the optimal starting point set offers the least gain. Yet, in theory, the video quality may degrade if we further accelerate the computation by the early termination mechanism. Comparatively, the video quality can be slightly improved by the optimal starting point set because it may reduce the variances of MV PDF and use less bits to code MV.

We may examine the overhead of the tools in our proposed complete algorithm one by one. For the genetic pattern search and adaptive pattern switching strategy, the run time profiling shows that the overhead is about 2% of the total computation time used for motion estimation. For the starting point calculation, it is fixed for each pattern search algorithm and thus it is negligible. For the early termination criterion calculation, the parameters in Eq. (4.46) are fixed and Eq. (4.46) uses simple arithmetic operations performed on a small amount of data. The additional computing

time is also negligible. Overall, the run time overhead of our proposed PBME algorithm is very small. Note that, this profiling is conducted on the personal computer with an Intel CPU. It may not perfectly portray the reality. For an embedded multimedia system, we may use a DSP or an ASIC for the video encoding and decoding. The extra computation of our proposed algorithm can be calculated in parallel by some auxiliary hardware.

Note that, the three proposed tools are not necessary coupled together. Any of the three tools can be adopted and combined by the other BME schemes. Because there are numerous possible combinations, it is beyond our capability to explore all the possibilities.

4.5.1 The Rate-Distortion Performance

To further understand the rate-distortion performance of our proposed best algorithm, we further select one slow motion sequence, container, and one fast motion sequence, foreman, and code them by FS and our proposed best algorithm at four different bitrates under the settings in **Table 4-24**.

Table 4-24 The rate-distortion test sequences and their settings.

Abbreviation	Sequence	Bit rate (K bps)	Frame rate (fps)	Number of frames
CT256	container	256	7.5	300
CT128	container	128	7.5	300
CT64	container	64	7.5	300
CT32	container	32	7.5	300
FM1536	foreman	1536	30	300
FM1024	foreman	1024	30	300
FM512	foreman	512	30	300
FM256	foreman	256	30	300

Fig. 4-17 shows the ASP performance of DS, AIPS-MP, and our proposed best algorithm. We do not show the ASP of FS here, because they are fixed to 1024. The ASPs of FS are very large when compared with all other fast algorithms. Our proposed best algorithm noticeably outperforms all other algorithms. **Fig. 4-18** shows the rate-distortion performances of FS, DS,

AIPS-MP and our proposed best algorithm. For both low and high motion sequences, our proposed best algorithm shows rather good rate-distortion performance. Quantitatively, we show the BDPSNR and BDRate [58][59] comparisons between FS, DS, AIPS-MP and our proposed best algorithm in **Table 4-25**. Thus, it is quiet clear that our proposed best algorithm provides substantial gain in computing complexity and keeps comparable rate-distortion performance with other algorithms including FS.

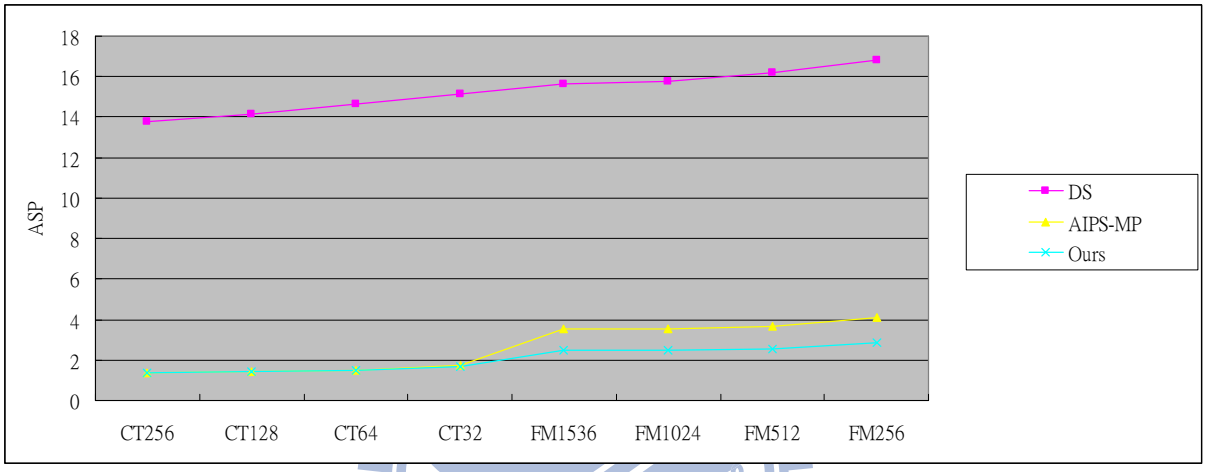


Fig. 4-17 The ASP performances of DS, AIPS-MP and our proposed best algorithm.

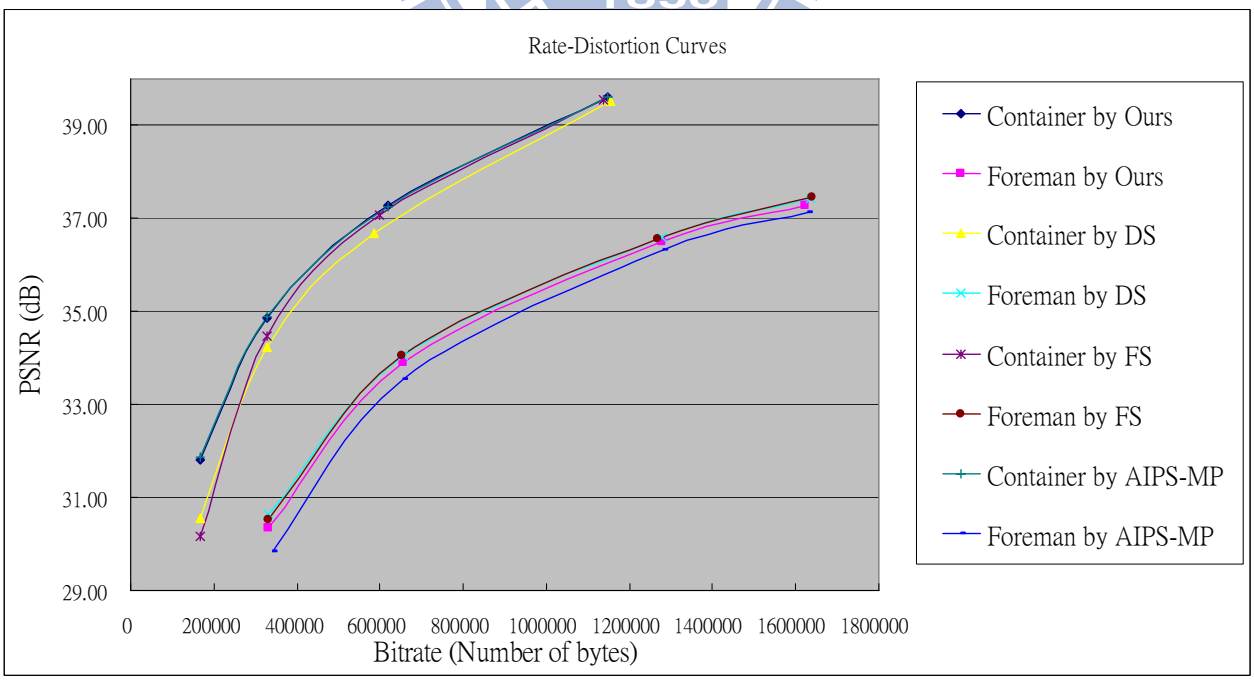


Fig. 4-18 The rate-distortion performances of FS, DS, AIPS-MP and our proposed best algorithm.

Table 4-25 The BDPSNR and BDRate comparisons between FS, DS, AIPS-MP and our proposed best algorithm.

	Ours to FS		Ours to DS		Our to AIPS	
Sequence	BDPSNR(dB)	BDRate(%)	BDPSNR(dB)	BDRate(%)	BDPSNR(dB)	BDRate(%)
Container	0.40	-6.84	0.55	-11.34	-0.02	0.40
Foreman	-0.14	3.60	-0.15	3.78	0.36	-7.58
Average	0.13	-1.62	0.20	-3.78	0.17	-3.59

Section 4.6 Chapter Summary

In this chapter, three important techniques have been investigated for reducing complexity of pattern-based block motion estimation (PBME). They are adaptive pattern switch [35][36][37][38][39][40][41][42][43], starting point selection [14][16][30][31] and early termination [14][30][31][50]. The prior arts in designing these schemes often based on heuristic reasoning and/or speculation on the collected data. The contribution of this study is to re-examine these techniques using a systematic approach. Optimal or nearly optimal solutions are thus proposed. Based on our previous motion estimation model and pattern search analysis ([51] and [54]), we impose the genetic search structure on the conventional ERPS and PHS schemes to reduce computation. Furthermore, a pattern switching strategy based on the on-line MV statistics is proposed. A well-chosen starting point set indeed reduces the average number of search points. A step-by-step procedure is proposed to find the best starting point set. The so-called early termination can further improve the search speed. We suggest a metric (correlation coefficient) to identify the best predictor for determining the termination threshold. At last, a PBME algorithm combining all the above features is examined. Simulations show that the search speed of the proposed algorithm is much faster than any previous search algorithm and its coding quality is kept at about the same PSNR level.

Chapter 5 Refined Model and Its Impact on Video Coding

In Chapter 3 [54], we propose a model for describing the pattern-based block motion estimation behavior. Our proposed model uses the notion of weighting function to characterize the efficiency of a pattern search algorithm. WF is defined as the *minimum number* of search points that a specific pattern search algorithm can achieve when the matching error surface is monotonic. Therefore, its values depend on the search patterns. Given the motion vector probability distribution of a video sequence, our complete model (expressed in (3.18)) can predict the performance (number of search points) of a PBME algorithm by using its WF and the motion vector probability distribution of a particular video sequence.

Yet, because our proposed genetic algorithms are stochastic in nature, the afore-mentioned WF cannot accurately characterize their average performance (number of search points). The difference is due to the fact that the genetic pattern searches randomly pick up the search direction but the classical pattern searches move along the steepest descent path on the matching error surface directly toward the best matching point. One purpose of this study is to construct a more accurate model for the genetic pattern searches and thus we can predict more precisely the performance of a new search algorithm. Accordingly, the *refined weighting function* (RWF, first mentioned in Section 1.2) is proposed. Also, inspired by the RWF, we devise a new type of genetic pattern searches that further reduce the computation.

Besides, we re-examine two critical coding tools in the adaptive pattern search scheme with the *refined model*. One is the impact of the component pattern searches of an adaptive pattern search on the switching threshold and the other one is the impact of the starting points on a search algorithm's performance. For comparison purpose, the conventional adaptive pattern search schemes and the genetic adaptive pattern search schemes are designed and tested. For each type of

pattern searches (the conventional type and the genetic type), we design a single level adaptive scheme and a double level adaptive scheme. Thus we totally propose four schemes.

The remaining parts of this chapter are organized as follows. Section 5.1 analyzes and models the behavior of a genetic pattern search by the RWF, which replaces the original WF. Based on the analysis, new search algorithms are proposed in Section 5.2. In Section 5.3, we adopt RWF in our refined model and compare its prediction accuracy with the original model using WF. Based on the refined model and the selected constituent searches, Section 5.4 re-examines the optimal threshold selection in the pattern switching mechanisms and the construction of the starting search point set for different search algorithms. Finally, we summarize this chapter in Section 5.5.

Section 5.1 Analysis on Genetic Pattern Searches

In this section, we dissect how the RWF characterizes the genetic pattern searches. First, we examine the assumption on the matching error surface required in the building of RWF. Accordingly, we demonstrate the construction of RWF by using two genetic pattern searches as the examples.

Matching Error Surface

To ensure the convergence to the optimal point of a fast BME search algorithm, most previous researches assume that the matching error (distortion) surface has a bell-like shape. In history, Jain and Jain [44] first suggested that the matching error surface satisfies the *quadrant-monotonicity* condition [45]. Let the origin point $O=(0,0)$ be the global minimum point (GMP) of a two-dimensional function $F_{QM}(x,y)$. Function $F_{QM}(x,y)$ is said quadrant monotonic (QM) if $F_{QM}(A) \le F_{QM}(B)$ for any two points satisfying the following conditions: 1) Points $A=(x_A,y_A)$ and $B=(x_B,y_B)$ are located in the same quadrant within the search range, and 2) $|x_A| \le |x_B|$ and $|y_A|<|y_B|$, or $|x_A|<|x_B|$ and $|y_A| \le |y_B|$. This condition requires only the monotonic ordering

relationship for any two points located inside the same quadrant. It does not specify the relationship for two points resided on different quadrants. Although a loose condition may cover a large range of real data, it also excludes powerful fast search techniques that assume a reasonable distance-dependent ordering relationship for two points near the quadrant boundaries.

The other extreme makes a strong uniform monotonicity (SM) assumption on the error surface. Examples are the recent studies [46][47] on examining the matching error surface. In [46], a mathematical model (5.1) of the matching error surface is proposed and verified by data fitting on the nature image sequences. In their proposed model (5.1), $F_{SM}(r)$ denotes the matching error of a search point with a distance of "r" away from GMP, $F_{SM}(0)$ is the global minimal matching error value of GMP, r is the chess board distance of (x,y), and h is an image-dependent constant. Although [46] shows that their statistical data fit this model well, in practice the error surface of individual block (not the entire image sequence) is processed by the search algorithm. In this case, model (5.1) is too strong and the individual block error surface does not match it well from time to time.

$$F_{SM}(r) = F_{SM}(0) + F_{SM}(0) \times \sqrt{h \cdot r}, \text{ where } r = |x| + |y|. \tag{5.1}$$

In this study, we assume that the matching error (distortion) surface $F_{QMSB}(x,y)$ is a quadrant monotonic function with smooth quadrant border (QMSB), which is specified by the following two properties.

Property 1: The error surface $F_{QMSB}(x,y)$ is *quadrant-monotonic* (QM).

Property 2: Let $C = (x_C, y_C)$ and $D = (x_D, y_D)$ be any two points inside the search region and located in different quadrants and ($|C - D| < R_{border}$, (e.g., R_{border} =3)). If $|C - O| > |D - O|$ implies $F_{QMSB}(C) > F_{QMSB}(D)$, then this quadrant monotonic function $F_{QMSB}(.)$ has a smooth quadrant border.

With this QMSB assumption, we can compare two points located at different quadrants if they are not far away from each other. On the other hand, it is a more general and relaxed model

than the model (5.1). In (5.1), the matching error surface is strictly symmetric, but QMSB only assumes smooth borders. Thus, it can cover broader cases.

Construction of the Refined Weighting Function

Because WF does not accurately describe the random nature of genetic pattern searches, herein we propose a RWF under the QMSB assumption. The RWF, $RWF(x,y)$, represents the average number of search points needed to reach the GMP located at (0,0) from the starting point (x,y). In contrast, the weighting function, $WF(x,y)$, represents the minimal number of search steps required to reach GMP from the starting point (x,y).

We use two examples to show how the averaged search point numbers are produced in a genetic search algorithm. In the following examples, we assume a parent point has N possible mutations (children) and m out of the N candidates have smaller matching error than the parent. Our purpose is to find the mutation with a matching error smaller than the parent. If we check one mutation at one step, it takes at most (N-m+1) steps (search points) to identify a solution. **Fig. 5-1** shows all possible search sequences for the case of a parent point with 4 possible mutations (A, B, C, and D) and only one of them, denoted as D, has the smaller matching error (than the parent). In the first branch, point A is picked up at the first step. Because its matching error is higher than the parent, we continue to pick up another mutation among B, C, and D. At any step, if D is picked, it becomes the new parent. Another example is shown in **Fig. 5-2**, in which two mutations (out of 4 candidates), denoted as C and D, have smaller matching errors. In this case, the new parent is produced when either C or D is checked. At the end, based on the entire search sequence tree in each case, the expected number of search points (ESP) needed to move from a parent to a smaller matching error mutation is E_m^N, as shown by (5.2).

$$E_m^N = \frac{m}{N} + \frac{m}{N}\sum_{j=1}^{N-m}((j+1)\times\prod_{i=1}^{j}(\frac{(N-m)-(i-1)}{N-i})) = \frac{N+1}{m+1}, \text{ where } N > m. \quad (5.2)$$

Herein we assume that the probability for selecting each mutation is equal. On a QMSB

matching error surface, the number of the admissible mutations, m, is decided by the relative location pattern formed by the current parent and the global optimal point. And N is decided by the search pattern, the parent location and its type (a starting parent or an intermediate parent). Eqs. (5.3) and (5.4) exemplify the calculations of the ESP values for finding an admissible mutation (it becomes the next parent) for the cases of **Fig. 5-1** and **Fig. 5-2**, respectively. Similarly, the ESP values of the other m and N values can be calculated. **Table 5-1** lists the ESP values, E_m^N, for N=3,...,6, and m=1,...,N.

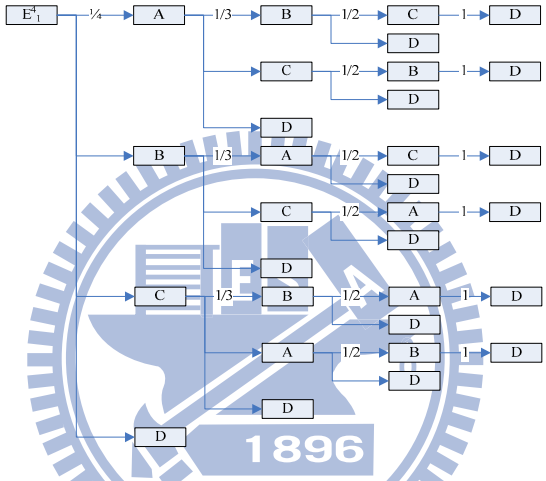


Fig. 5-1 All possible search sequences for a parent point with N=4 and m=1.

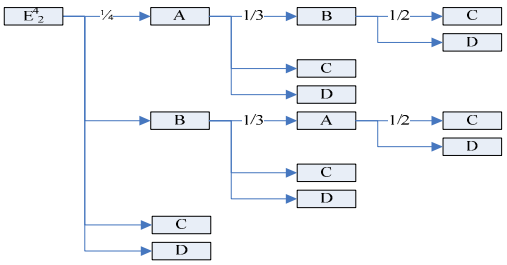


Fig. 5-2 All possible search sequences for a parent point with N=4 and m=2.

$$E_1^4 = \left[\left[\left[\left[\frac{1}{4}\times\frac{1}{3}\times\frac{1}{2}\times 1\times 4+\right]\times 2+\right.\right.\right.\left.\left.\left.\left[\frac{1}{4}\times\frac{1}{3}\times\frac{1}{2}\times 3\right]\right]\times 3+\right.\left.\left[\frac{1}{4}\times\frac{1}{3}\times 2\right]\right]\left.\left[\frac{1}{4}\right]\right] = \frac{5}{2} \tag{5.3}$$

$$E_2^4 = \left[\left[\left[\frac{1}{4}\times\frac{1}{3}\times\frac{1}{2}\times 3\times 2+\right]\times 2+\right.\right.\left.\left.\left[\frac{1}{4}\times\frac{1}{3}\times 2\times 2\right]\right]\left.\left[\frac{1}{4}\times 1\times 2\right]\right] = \frac{5}{3} \tag{5.4}$$

Table 5-1 The *ESP* values

E_m^N		m					
		1	2	3	4	5	6
N	3	2.00	1.33	1.00			
	4	2.50	1.67	1.25	1.00		
	5	3.00	2.00	1.50	1.20	1.00	
	6	3.50	2.33	1.75	1.40	1.17	1.00

5.1.1 RWF of the Genetic Rhombus Pattern Search

As an example, we construct the RWF for GRPS, defined in Subsection 4.2.1 [54]. Assuming the matching error surface is QMSB, then we are able to get the number of *small* distortion points (the value of m in E_m^N) in this search pattern. For GRPS, there are two types of starting search point cases (S_1^{GRPS} and S_2^{GRPS}) and two types of intermediate search point cases (M_1^{GRPS} and M_2^{GRPS}), as shown in **Fig. 5-3**. Herein, points A, B, C and D are the search candidates (mutations) and point

GMP denotes the best matching point. In **Fig. 5-3**(a), assuming S_1^{GRPS} is the starting point of a new search, only 1 out of the 4 points belonging to the GRPS pattern centering at S_1^{GRPS} have a smaller matching error than S_1^{GRPS} when point GMP has the same horizontal or vertical coordinate as S_1^{GRPS}. Otherwise, it is the **Fig. 5-3**(b) case, in which 2 out of the 4 points in the pattern centering at S_2^{GRPS} have smaller errors. Similarly, for the intermediate steps in **Fig. 5-3**(c), only 1 out of the 3 points centering around M_1^{GRPS} has a smaller matching error when point GMP has the same horizontal or vertical coordinate as M_1^{GRPS}. Otherwise, in the case of **Fig. 5-3**(d), 2 out of the 3 points centering around M_2^{GRPS} have smaller errors. Therefore, the average numbers of search points needed to move from S_1^{GRPS}, S_2^{GRPS}, M_1^{GRPS} and M_2^{GRPS} to a legitimate next point are E_1^4 (=5/2), E_2^4 (=5/3), E_1^3 (=4/2) and E_2^3 (=4/3), respectively, as listed in **Table 5-1**.

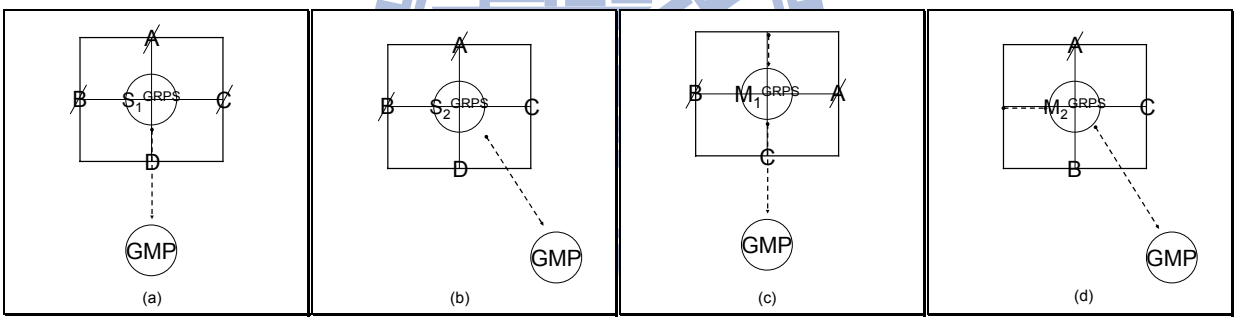


Fig. 5-3 Two cases of starting search points, (a) and (b), and two cases of intermediate search points, (c) and (d), in the search process of GRPS when the matching error surface is QMSB.

We next consider the averaged search points of multiple moves from the starting point (*x*,*y*) to the best matching point (0,0). This *average number* of search points is defined to be $RWF_{GRPS}(x,y)$. When the starting point is chosen and thus it belongs to one of the quadrants defined by GMP, the GRPS procedure moves along the search points only located inside that quadrant, as shown by **Fig. 5-3**(b) or **Fig. 5-3**(d). If the starting point is located on a quadrant boundary, **Fig. 5-3**(a) and **Fig. 5-3**(c) show the cases that it moves along that quadrant boundary. Thus, the $RWF_{GRPS}(x,y)$ function is quadrant-symmetric. We only need to study one quadrant. **Fig.**

5-4 shows the recursive procedure of calculating the RWF of GRPS based on the above analysis. And **Fig. 5-5** shows the contour plot of $RWF_{GRPS}(x,y)$.

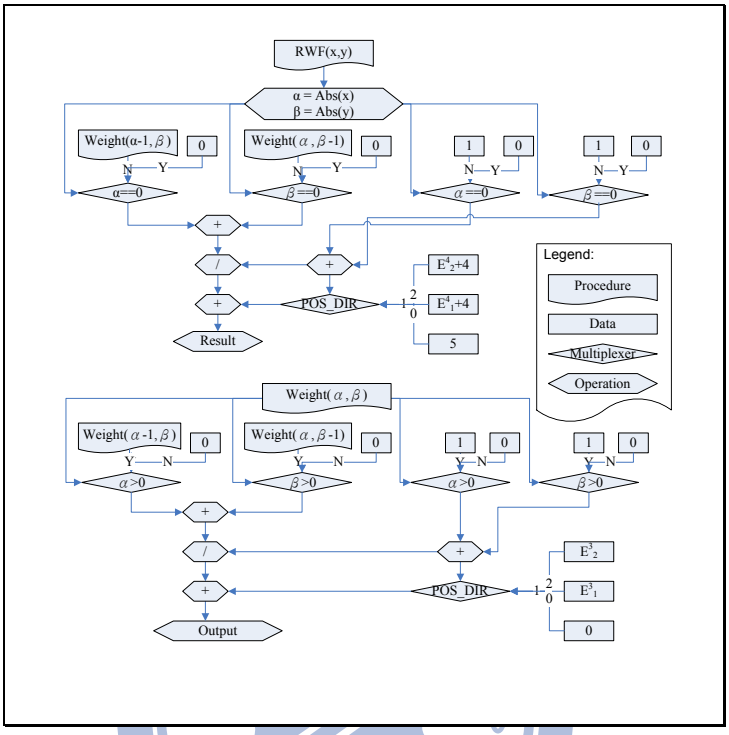


Fig. 5-4 The construction of RWF

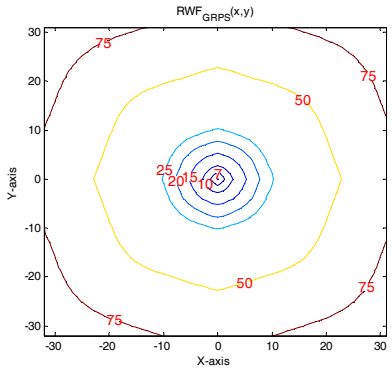


Fig. 5-5 The RWF of GRPS

Let GMP be (0,0). In **Fig. 5-4**, $RWF(x,y)$ denotes the final RWF value from the starting point (*x*,*y*) to GMP, and *Weight(α,β)* denotes the recursively accumulated ESP from the intermediate point (α,β) to GMP. Because $RWF(x,y)$ is quadrant-symmetric, we only need to

consider the non-negative quadrant, thus $\alpha=Abs(x)$ and $\beta=Abs(y)$. Abs(.) denotes the absolute value operation. When $\alpha=0$ and $\beta=0$, this is the last step and $RWF(x,y)$ is 5. The other cases include intermediate points. If $\alpha\neq0$ and $\beta\neq0$, it is **Fig. 5-3**(b) and thus, $RWF(x,y)$ is E_2^4+4 plus the average of *Weight(α-1,β)* and *Weight(α,β-1)*. Otherwise, either $\alpha\neq0$ or $\beta\neq0$; thus, it is **Fig. 5-3**(a) and $RWF(x,y)$ is E_1^4+4 plus either *Weight(α-1,β)* or *Weight(α,β-1)*.

In calculating the intermediate point ESP, *Weight(α,β)*, we adopt the recursive approach. If α=0 and β=0, this is the final step and *Weight(α,β)* is 0. If α≠0 and β≠0, it is the Fig. 5-3(d) case and *Weight(α,β)* is E_2^3 plus the average of *Weight(α-1,β)* and *Weight(α,β-1)*. Otherwise, either α≠0 or β≠0; thus, it is Fig. 5-3(c) and *Weight(α,β)* is E_1^3 plus either *Weight(α-1,β)* or *Weight(α,β-1)*.

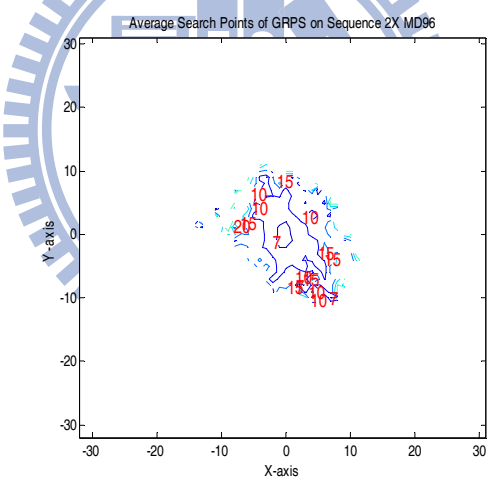


Fig. 5-6 The real average search points of GRPS when it is applied on the sequence '2X MD96'.

Fig. 5-6 shows the real average search points of GRPS when it is applied to the sequence '2X MD96'. The outer ring of **Fig. 5-6** is empty because these points never become the best matching points. When we compare **Fig. 5-6** (the real average search points of GRPS) with **Fig. 3-11** (the WF of GRPS) and **Fig. 5-5** (the RWF of GRPS), the inner contour shape in **Fig. 5-6** is more similar to that in **Fig. 5-5** than to that in **Fig. 3-11.** Evidently, RWF is a better representation for the real average search point than WF.

5.1.2 RWF of the Genetic Point-oriented Hexagonal Search

As another example, we construct the RWF for GPHS, defined Subsection 4.2.1 [55]. For the ease of the derivation of RWF for GPHS, let the matching error surface $H_{QMSB}(x,y)$ be a *quadrant monotonic function with smooth border* (QMSB), $O=(0,0)$ be the optimum search point, and $P=(x_P,y_P)$ and $Q=(x_Q,y_Q)$ be any two points in the search range, and $|P-Q|<R_{nbd}$, (R_{nbd}=3). That is, $|P-O|>|Q-O|$ implies $H_{QMSB}(P)>H_{QMSB}(Q)$. Then, using a similar procedure in deriving $RWF_{GRPS}(x,y)$, we construct $RWF_{GPHS}(x,y)$ by computer simulations and show its contour plot in **Fig. 5-7**.

We like to add some remarks here on the necessity of the QMSB assumption. In Subsection 5.1.1 [53], the less rigorous QM assumption is sufficient for the derivation of RWF for GRPS. Yet, in the derivation of RWF for GPHS, the more rigorous QMSB assumption is required. It is because that a large search pattern contains possibly two nearby points in different quadrants. For different search patterns, we can adjust R_{nbd} in the QMSB assumption to match the maximum distance between any two points in the search pattern. A special case is R_{nbd}=2; the QMSB assumption becomes the QM assumption in this case. On the other hand, when R_{nbd} approaches infinity, the QMSB assumption becomes the SM assumption. In this study, the more general QMSB assumption is used to expand our previously proposed model [53] for modeling the genetic pattern searches.

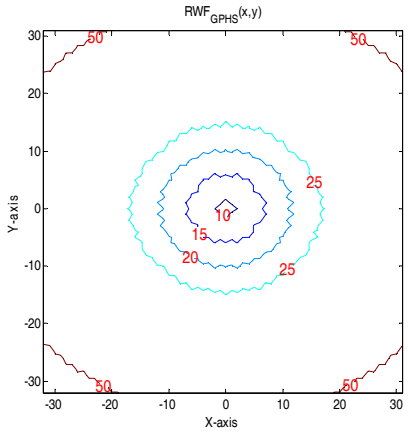


Fig. 5-7 The RWF of GPHS.

Section 5.2 Proposed Momentum-directed Genetic Pattern Searches

We propose two momentum-directed genetic pattern searches (MD-GPS) in this section. They are the momentum-directed version of GRPS and GPHS, respectively.

Observing the operation of the current GRPS, we find a way to speed it up: the algorithm should move directly towards the direction of best matching point. Statistically, the successful direction of the previous search is often the correct search direction at the current point. Our QMSB error surface model certainly leads to this conclusion too. Therefore, instead of selecting randomly one mutation from the candidate child set, we select the mutation based on its preceding successful mutations. That is, it tends to move along the same direction of the prior successful search. On the other hand, it can change the search directions when the assumption of QMSB matching error surface is not valid.

The flow chart of the proposed algorithm, namely, *momentum-directed* GRPS (MD-GRPS), is described by **Fig. 5-9**. Its RWF contour plot is in **Fig. 5-8**. In **Fig. 5-10**, C is the current parent, P is obtained using the last successful mutation direction, and PP is obtained using the

second-to-the-last successful mutation direction differing from P. Arrows show the search order. It indicates shows the search order (priority) of candidate mutations in MD-GRPS, which is 1) the mutation with the same direction as the last successful mutation (P), 2) the mutation with the same direction as the second-to-the-last successful mutation (PP), 3) the mutation with the opposite direction to the second-to-the-last successful mutation, and 4) the mutation with the opposite direction to the last successful mutation. In **Fig. 5-10**, P cannot be in the opposite direction of PP. If they are in the opposite direction, the search process returns to its previous parent. Also, the fourth (also the least) priority point is never searched, because the opposite direction to the last successful mutation is the previous parent point that has been checked in the early search step.

Likewise, by adopting the momentum-directed search order in the genetic pattern search, we convert GPHS to a momentum-directed one. **Fig. 5-11** shows its RWF contour plot. The flow chart of the proposed algorithm, namely, the momentum-directed GPHS (MD-GPHS), is described by **Fig. 5-12**. And the search order (priority) of candidate mutations in MD-GPHS is shown in **Fig. 5-13**. In **Fig. 5-13**, C is the current parent, P is obtained using the last successful mutation direction, and PP is obtained using the second-to-the-last successful mutation direction but is different from P. Arrows show the search order. **Fig. 5-13** shows two possible cases, which are decided by the relative direction of the last successful mutation direction (P) and the second-to-the-last successful mutation direction (PP). Similar to the discussions in MD-GRPS, we do not consider the case that P and PP are in the opposite direction. And the search order is determined by the following principles. The first priority candidate is the mutation with the same direction to the last successful mutation (P), the second is the one with the same direction to the second-to-the-last successful mutation (PP) but it must differ from P, and the last candidate is the one with the opposite direction to the last successful mutation. As for the search order of the remaining three candidates, it is determined by their distance to the last successful mutation, P (the smaller one is checked first). When they are identical, the order is decided by their distance to

the second-to-the-last successful mutation, PP. Similar to the discussion in MD-GRPS, the sixth (also the least) priority point is never searched, because the opposite direction to the last successful mutation is the previous parent point that has been checked.

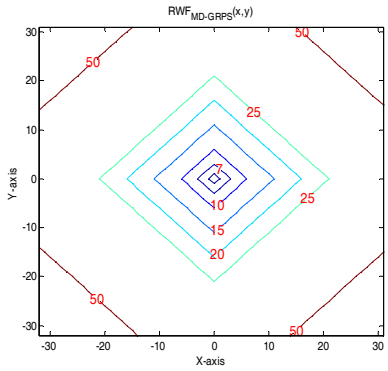


Fig. 5-8 The RWF of MD-GRPS

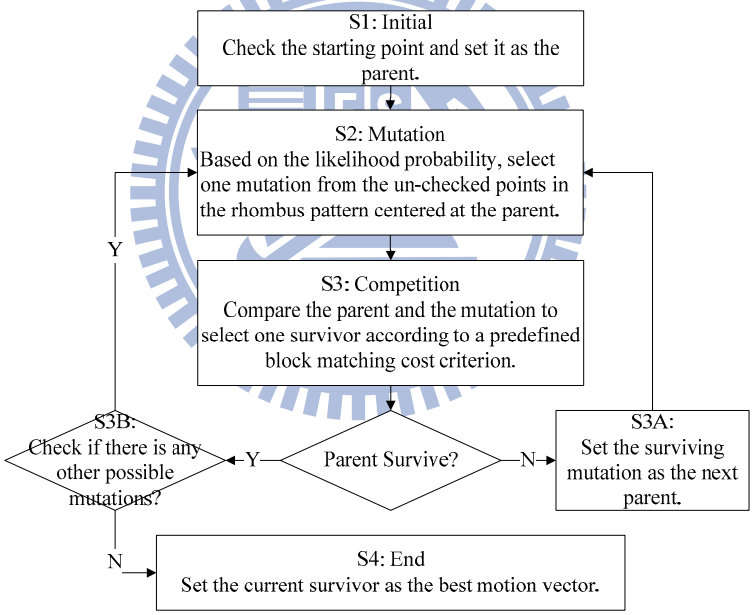


Fig. 5-9 The flow chart of MD-GRPS

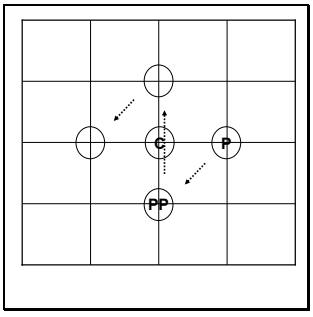


Fig. 5-10 The search priority of all candidate mutations in MD-GRPS.

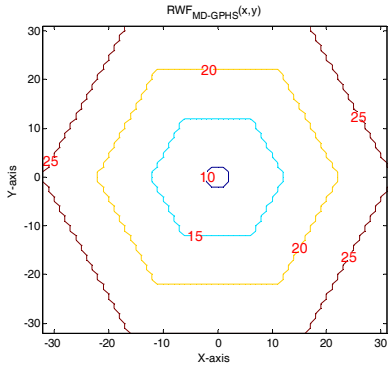


Fig. 5-11 The RWF of MD-GPHS

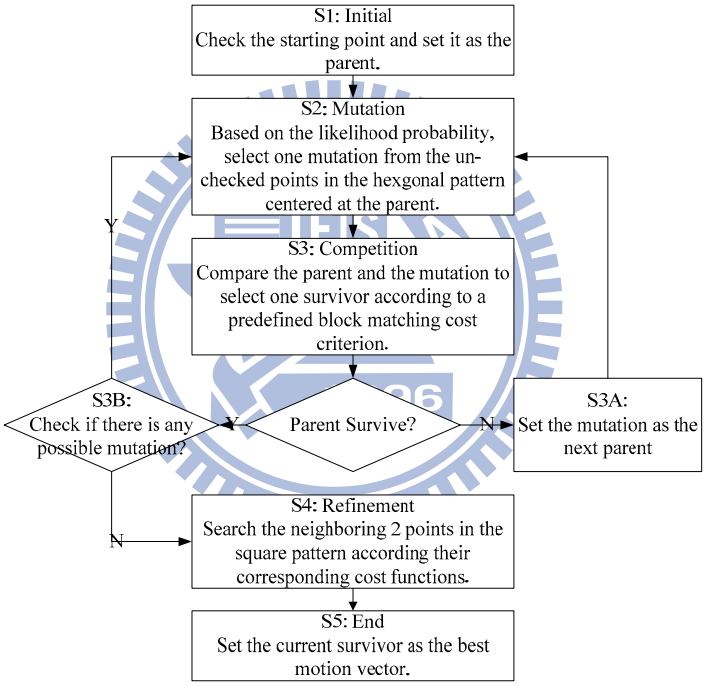


Fig. 5-12 The flow chart of MD-GPHS

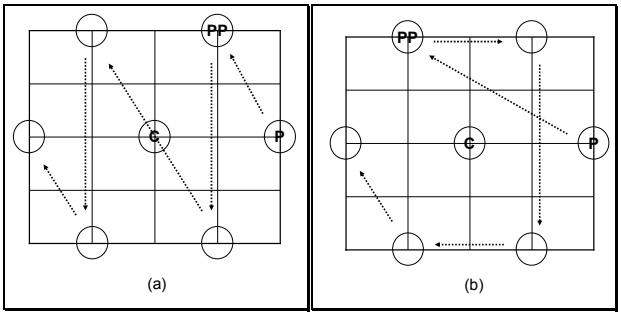


Fig. 5-13 The search priority of all candidate mutations in MD-GPHS.

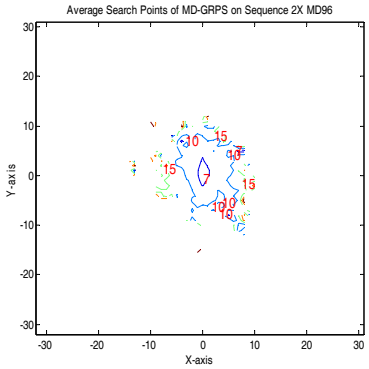


Fig. 5-14 The real average search points of MD-GRPS when it is applied on the sequence '2X MD96'.

Fig. 5-14 shows the real average search points of MD-GRPS when it is applied to the sequence '2X MD96'. The outer ring of **Fig. 5-14** is empty because these points never become the best matching points. When we compare **Fig. 5-14** with **Fig. 5-8**, we can find that the shapes of their inner contours are similar. Thus, RWF characterizes well the real average search points of MD-GRPS.

5.2.1 Performance of Momentum-Directed Genetic Pattern Searches

To test the proposed algorithm, ten sequences (denoted as '1X') with different MV variances are tested under the parameter settings given in **Table 3-1**. Moreover, to test the extreme cases, we generate ten new test sequences by skipping the even frames of these sequences, and these new sequences are denoted as '2X'. They are roughly the two times fast forward playback of the originals. These 20 test sequences are coded by an MPEG-4 SP@L3 encoder. The other simulation settings are the same as described in Section 3.1.

In selecting the simulation platform, our focus is whether it provides a fair and direct comparison among different ME algorithms. The H.264 scheme is a newer and very sophisticated platform. It contains many tools that affect the choice of motion vectors, such as multiple (block) mode decision and rate distortion optimization. For example, at different bit rates, the same mode

decision tool can select different motion vectors. Thus, the PSNR impact due to the use of different ME algorithms may become hidden or be blurred. Thus, we adopt a simpler MPEG-4 platform on which the impact of different motion estimation algorithms can be observed more clearly.

The average number of search points (ASP) and the peak signal to noise ratio (PSNR) for various sequences and search algorithms are listed in **Table 5-2** and **Table 5-3**, respectively. The predicted MV (*PMV*, defined by (2.2)) is used as the search starting point in all cases. FS denotes the full search, ERPS is proposed by [30] but we replace the MV predictors in [30] by the PMV, and PHS is proposed by [33].

The pair-wise performance comparisons in ASP and PSNR between MD-GRPS and some selected popular algorithms are given in **Fig. 5-15** and **Fig. 5-16**. The pair-wise performance comparisons in ASP and PSNR between MD-GPHS and some selected popular algorithms are given in **Fig. 5-17** and **Fig. 5-18**. In **Fig. 5-15** and **Fig. 5-17**, the computing gain (CG) is defined as the ASP ratio between the original and the chosen algorithm minus one. In **Fig. 5-16** and **Fig. 5-18**, the quality gain (QG) is the PSNR difference. The CG of MD-GRPS and MD-GPHS substantially outperforms the other popular algorithms, while their average QG is near 0.

MD-GRPS can be up to 18% faster than GRPS for very fast sequences (2X FB1024), and their PSNR values are about the same. On the average, comparing their ASP values, MD-GRPS is 7% faster than GRPS, 35% faster than ERPS, 1.39 times faster than DS, 1.76 times faster than FSS and 143 times faster than FS. And the PSNR of MD-GRPS is about the same as that of all the other search algorithms (+0.06dB ~ -0.06dB).

Similarly, MD-GPHS can be up to 13% faster than GPHS for very fast sequences (2X FB1024) and its PSNR quality is roughly at the same level. On the average, MD-GPHS is 5% faster than GPHS, 12% faster than PHS, 69% faster than DS, 96% faster than FSS, and 101 times faster than FS. And the PSNR of MD-GPHS is about the same as those of GPHS and the

non-genetic version (PHS) (+0.02dB ~ -0.05dB). When being compared to the conventional pattern search algorithms, all these three algorithms (MD-GPHS, GPHS, and PHS) have slightly PSNR drop (-0.12dB ~ -0.17dB).

Generally MD-GRPS is significantly better in speed than MD-GPHS for most test sequences. However, MD-GPHS outperforms MD-GRPS by 3% for very fast sequences (2X FB1024). As expected in comparing **Fig. 5-8** with **Fig. 5-11**, $RWF_{MD\text{-}GPHS}$ has smaller values than $RWF_{MD\text{-}GRPS}$ near the outer border of the search area. In short, one algorithm beats the other in certain scenarios but none is the best for all cases. Thus, a good adaptive pattern scheme that dynamically selects the most appropriate pattern search algorithms further reduces the computational complexity.

The computation overheads of MD-GRPS and MD-GPHS are negligible. For all possible pairs of the last and the second-to-the-last successful mutation directions, we generate the corresponding search priority tables in advance. In execution of a momentum-directed algorithm, we record the last and the second-to-the-last successful mutation directions, use this direction pair to choose the search priority table and decide the search priority accordingly. A few memory access and comparisons can do all the works.

Note that, in **Table 5-2**, the PSNR of both '1X' and '2X' HL40 acquired by FS are lower than those acquired by other algorithms. Because HL40 has slight noise textures, the motion vector field produced by FS is much noiser (with larger magnitude) than those produced by the other algorithms. This phenomenon influences the matching error little but the size of motion vectors a lot, therefore, has a more significant influence on the coded picture quality, particularly for low bitrate sequences.

Table 5-2 ASP (Average Number of Search Points).

Type	Sequence	MD-GRPS	GRPS	ERPS	MD-GPHS	GPHS	PHS	DS	FSS	FS
1X	CT256	5.28	5.36	5.75	9.19	9.37	9.52	13.81	17.53	1024
	CT40	5.85	5.98	7.04	9.51	9.88	10.31	15.03	18.38	1024

	HL40	6.23	6.35	7.33	9.60	9.68	10.10	15.38	18.72	1024
	MD96	5.94	5.98	6.83	9.58	9.65	10.02	14.85	18.37	1024
	CG112	5.75	6.08	7.63	9.29	9.76	10.25	15.09	18.25	1024
	FM512	6.80	7.13	8.65	9.80	10.00	10.57	16.17	19.03	1024
	FM1024	6.64	6.94	8.32	9.67	9.85	10.35	15.76	18.71	1024
	FB1024	10.35	11.89	16.36	11.36	12.75	14.18	22.36	22.70	1024
	FG768	6.06	6.38	7.57	9.72	9.95	10.34	15.30	18.73	1024
	ST1024	7.24	7.65	9.95	9.90	10.56	11.40	16.96	19.47	1024
2X	CT256	5.43	5.62	6.35	9.26	9.51	9.74	14.15	17.72	1024
	CT40	6.40	6.60	8.15	9.82	10.34	10.89	16.05	19.11	1024
	HL40	6.37	6.51	7.57	9.66	9.74	10.22	15.62	18.88	1024
	MD96	6.29	6.40	7.56	9.77	9.85	10.38	15.44	18.76	1024
	CG112	6.73	7.36	9.54	9.74	10.64	11.48	17.04	19.57	1024
	FM512	8.25	9.07	11.70	10.52	11.01	12.02	18.72	20.67	1024
	FM1024	7.98	8.85	11.36	10.35	10.79	11.75	18.26	20.28	1024
	FB1024	13.27	15.75	22.32	12.94	14.62	17.15	27.39	26.22	1024
	FG768	6.55	7.01	8.69	9.88	10.35	10.83	16.30	19.29	1024
	ST1024	8.61	9.28	12.45	10.72	11.73	13.00	19.49	21.26	1024
	Average	7.10	7.61	9.56	10.01	10.50	11.23	16.96	19.58	1024.00

Table 5-3 PSNR (Peak Signal to Noise Ratio).

Type	Sequence	MD-GRPS	GRPS	ERPS	MD-GPHS	GPHS	PHS	DS	FSS	FS
1X	CT256	39.48	39.49	39.50	39.47	39.43	39.44	39.51	39.49	39.56
	CT40	31.99	32.21	32.08	31.28	31.24	31.47	31.92	31.69	32.04
	HL40	34.41	34.49	34.60	34.15	34.14	34.22	34.25	34.17	33.55
	MD96	40.05	40.08	40.09	39.78	39.79	39.85	39.99	39.93	39.80
	CG112	29.13	29.14	29.16	29.06	29.03	29.06	29.14	29.13	29.08
	FM512	34.04	34.05	34.10	33.86	33.89	33.92	34.06	34.02	34.06
	FM1024	36.55	36.52	36.61	36.49	36.46	36.44	36.59	36.48	36.56
	FB1024	34.92	34.87	34.88	34.85	34.73	34.87	34.93	34.94	35.28
	FG768	26.18	26.17	26.19	26.14	26.15	26.17	26.18	26.16	26.20
	ST1024	29.16	29.39	29.31	29.31	29.42	29.33	29.44	29.35	29.48
2X	CT256	38.63	38.65	38.68	38.52	38.52	38.51	38.60	38.72	38.95
	CT40	30.15	30.28	30.22	29.30	29.22	29.54	29.94	29.73	29.81
	HL40	33.25	33.31	33.38	32.91	32.95	33.02	33.07	32.93	32.33
	MD96	38.66	38.66	38.66	38.39	38.37	38.44	38.60	38.57	38.41
	CG112	27.43	27.43	27.53	27.33	27.23	27.34	27.50	27.45	27.37
	FM512	32.36	32.34	32.45	32.16	32.19	32.23	32.38	32.35	32.42
	FM1024	35.23	35.25	35.29	35.14	35.12	35.21	35.24	35.17	35.28
	FB1024	33.26	33.22	33.24	33.21	33.12	33.22	33.28	33.30	33.44

FG768	25.52	25.51	25.53	25.46	25.42	25.48	25.53	25.49	25.51
ST1024	27.86	27.99	27.93	27.87	27.87	27.88	27.97	27.93	28.10
Average	32.91	32.95	32.97	32.73	32.71	32.78	32.91	32.85	32.86

Fig. 5-15 Performance comparisons in ASP between MD-GRPS and some popular algorithms.

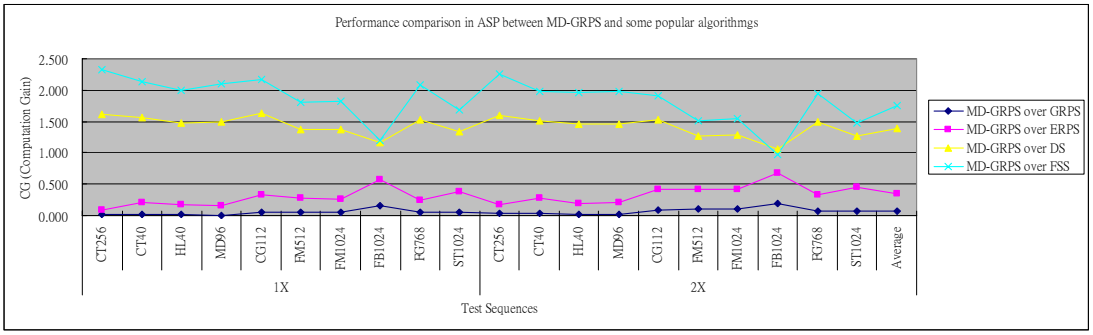


Fig. 5-16 Performance comparisons in PSNR between MD-GRPS and some popular algorithms.

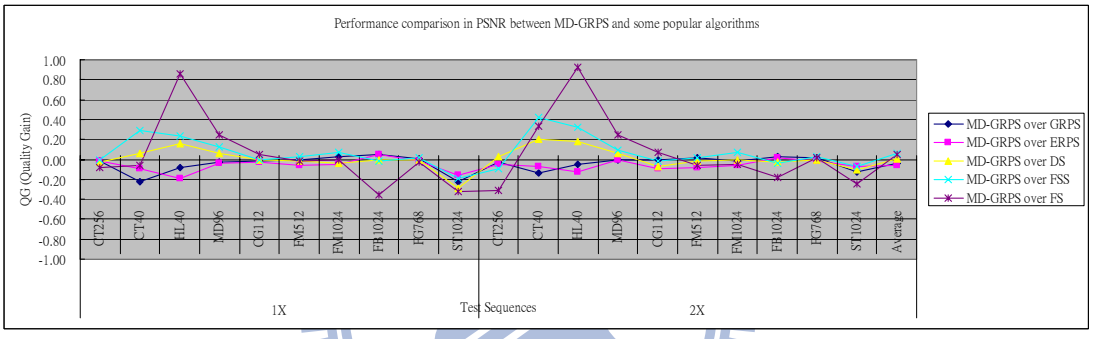


Fig. 5-17 Performance comparisons in ASP between MD-GPHS and some popular algorithms.

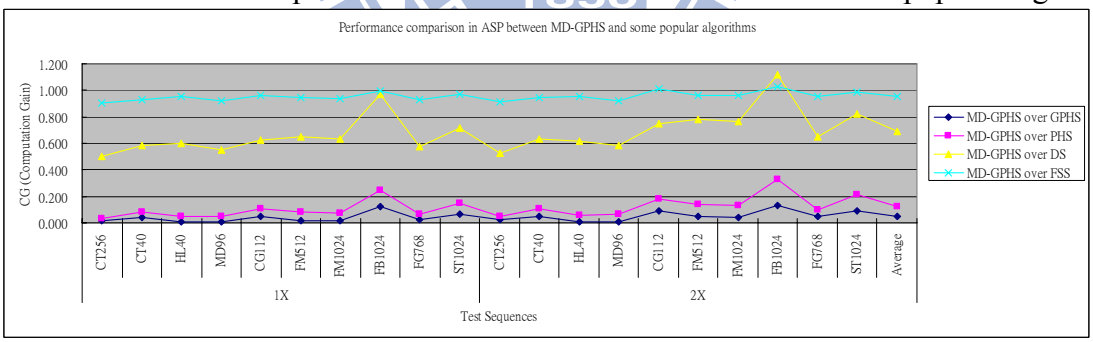
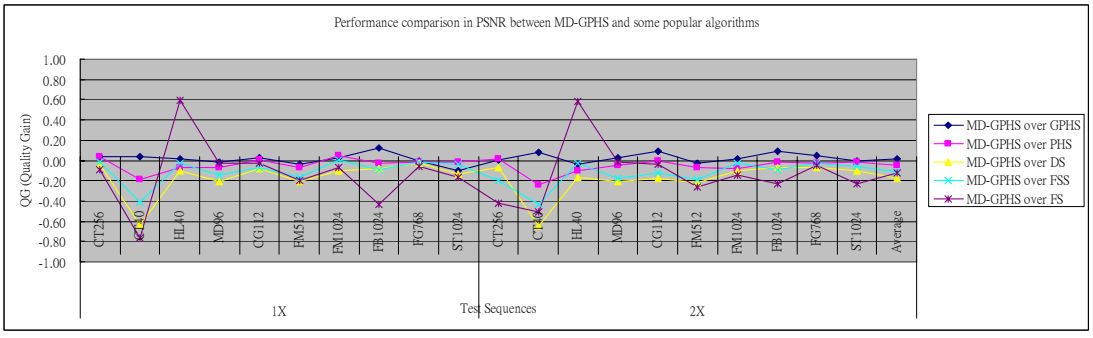


Fig. 5-18 Performance comparisons in PSNR between MD-GPHS and some popular algorithms.



Section 5.3 Refined Analytic Model for PBME and Its Accuracy

In Chapter 3 [54], we propose a mathematical model (expressed by (3.18)) that can predict the average number of search points (ASP) produced by a PBME. In Chapter 4 [55], we demonstrate the construction of a new PBME by using this model. With RWF, the original model is enhanced. The refined analytical model is introduced in this section.

Refined Analytical Model

The *refined weighting function* [53], $RWF_{SA}(x,y)$, is defined to be the *average number* of search points produced by a search algorithm when the best matching point is located at (*x*,*y*). We are able to calculate the RWF associated with a search algorithm when the matching error (distortion) surface is *unimodal* and *monotonic*. For the deterministic search algorithms (conventional PBME), their WF and RWF are the same. For the probabilistic search algorithms (for example, genetic algorithms), the WF expression is not an accurate representation. In comparison, RWF better portrays the behavior of a search algorithm.

Fig. 5-20 shows the RWF contour plots of 4 popular pattern search algorithms, FSS [26], DS [27][28], PHS [33] and ERPS. The ERPS algorithm adopted here is the adaptive rood pattern search in [31] but with a single starting point - *PMV*. The value marked on a contour represents the *average search points* required for a search algorithm to move from the origin to a point (location) on the contour.

Because WF does not well convey the randomness nature of the genetic pattern searches, the RWF replaces WF in (3.18). Thus, (3.18) becomes (5.5). We use (5.5) as the *refined* model to characterize the behavior of a pattern search algorithm.

$$ASP = C_1 \times \sum_{x,y \in A} S_{FS}(x,y) \times RWF_{SA}(x,y) + C_2 \tag{5.5}$$

Training Methods

Similar to the training methods of the original model [54] using WF, there are two methods to decide the C_1 and C_2 in the refined model using RWF. In the first method, we apply a fixed SA to a set of training sequences to compute C_1 and C_2 by the regression method. Our objective is that the refined model with trained C_1 and C_2 can predict the *ASP* of a *new* sequence accurately. In the second method, we apply a few search algorithms (the training algorithms) to a specific sequence, and then calculate C_1 and C_2 based on the acquired data. In this case, the goal is that the refined model with trained C_1 and C_2 can predict the *ASP* values produced by a *new* search algorithm on the same sequence.

Prediction Accuracy

When we use the second method to predict the ASP of new (genetic) search algorithms, **Fig. 5-19** show the comparison between the actual ASP and the predicted ASP for the 1X (left part) and 2X sequences (right part). The blue dash straight line is obtained by applying the regression method to the purple cross data points that are generated by FSS, DS, ERPS and PHS. Therefore, it shows the perfect prediction case (actual ASPs equal predicted ASP). The red open diamond is the prediction data point of GRPS using WF and the green solid diamond uses RWF. The red open square is the prediction data point of GPHS using WF and the green solid square uses RWF. It is quite obvious that the green solid symbols (the refined model with RWF) are closer to the blue dash line (perfect prediction) than the red open symbols (the original model with WF) in most cases. Quantitatively, **Table 5-4** shows the average absolute difference between the actual ASP and predicted ASP when either WF or RWF is in use. Herein, the parameters C_1 and C_2 in the predictive ASP model are trained by using FSS, DS, ERPS and PHS. When we replace WF by RWF, the average prediction error for GRPS is reduced from 2.76 to 0.74, and that for GPHS is reduced from 2.8 to 1.5. Clearly, the refined model with RWF is more accurate.

On the other hand, when we use the first method to predict the ASP of a new sequence, the prediction differences owing to the adaptation of WF and RWF in the model are about the same. It is because the inaccuracy of WF is compensated by adjusting parameters C_1 and C_2 in its overall prediction model.

Table 5-4 The average absolute difference between predicted ASP and actual ASP for all 1X and 2X sequences.

The average absolute difference between predicted ASP and actual ASP	WF	RWF
GRPS	2.76	0.74
GPHS	2.80	1.50

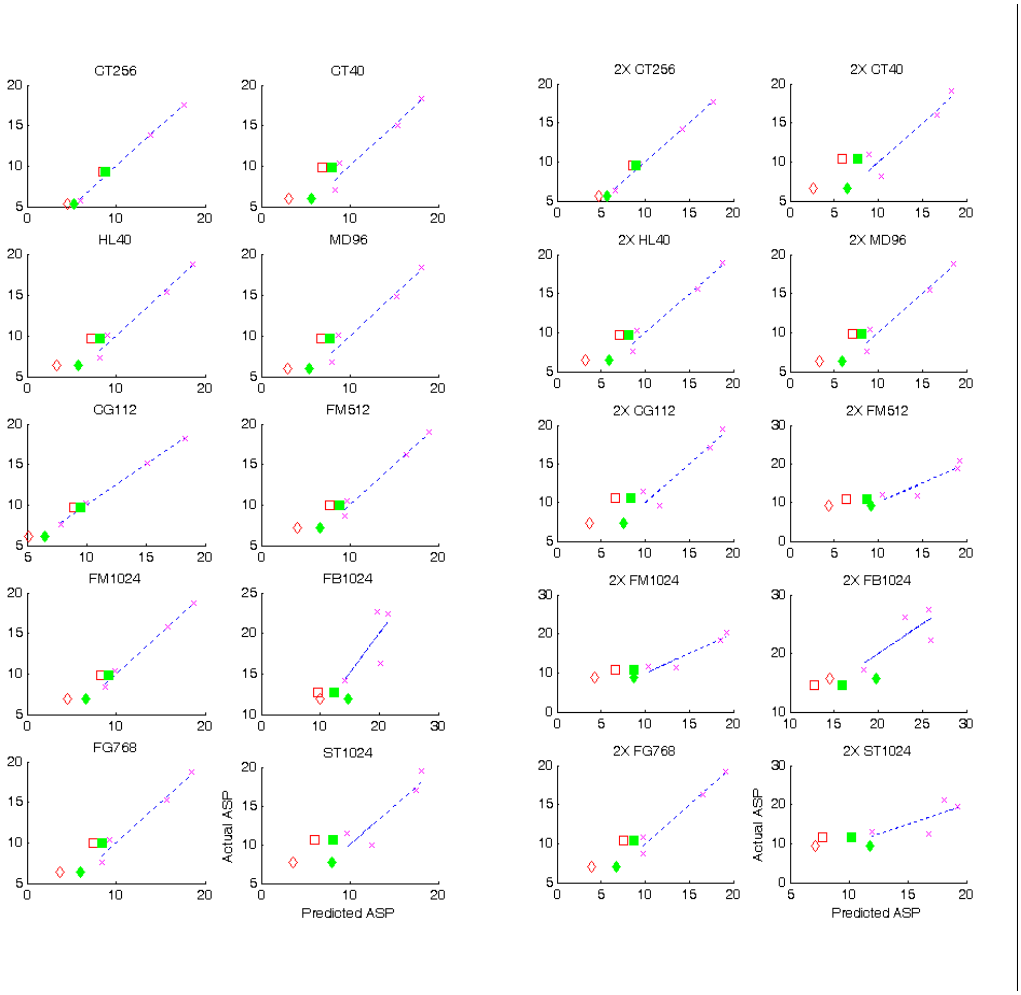


Fig. 5-19 The relationships between the actual ASP and the predicted ASP for the 1X and 2X sequences.

Section 5.4 Refined Model and Coding Tool Design

Typically there are three major coding tools in an adaptive pattern search algorithm – the constituent pattern searches, the pattern switching strategy, and the starting point set.

In constructing the adaptive pattern searches, we first select the constituent pattern searches. An effective pattern search should have a small RWF at all locations in a search window. However, it is difficult to devise such a pattern search; therefore, two complementary pattern searches, one is good at small motion vectors and the other is good at large motion vector, are selected as the pattern search set in an adaptive pattern switching scheme.

From the RWF profiles given in **Fig. 5-20**, we conclude that 1) DS outperforms FSS for all possible MVs, 2) ERPS uses the least number of search points when the motion vectors are located near the PMV, and 3) PHS uses the least number of search points for the motion vectors located far from PMV. Therefore, two conventional pattern searches, ERPS and PHS, are selected as the traditional pattern search set.

Fig. 5-21 shows the RWF of the genetic pattern searches, GRPS (the genetic-based ERPS) and GPHS (the genetic-based PHS). When comparing **Fig. 5-20** to **Fig. 5-21**, we choose GRPS and GPHS as the genetic pattern search set, because the RWF of GRPS has the smallest values for small motion vectors and that of GPHS has the smallest values for large motion vectors. The details of GRPS and GPHS are described in Subsection 4.2.1 [54][55].

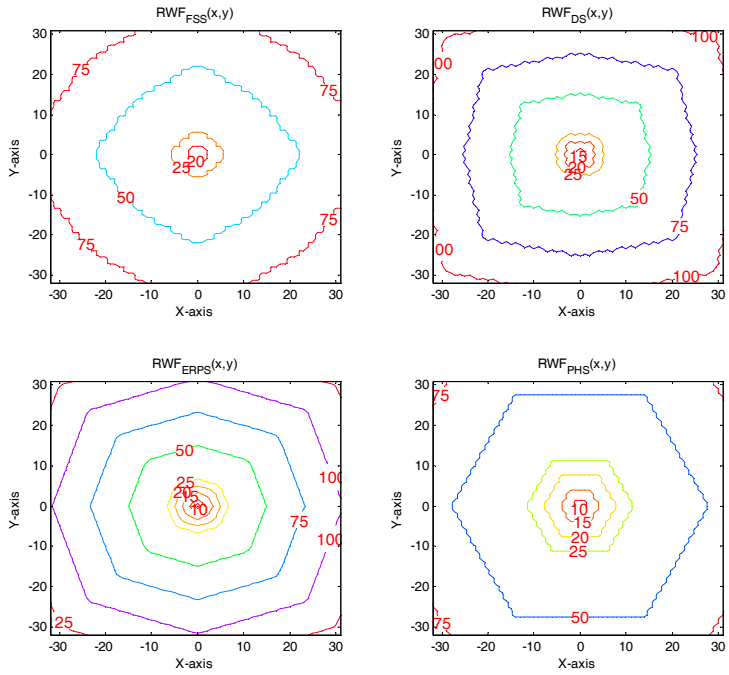


Fig. 5-20 Contour plots of the RWF for FSS, DS, ERPS and PHS.

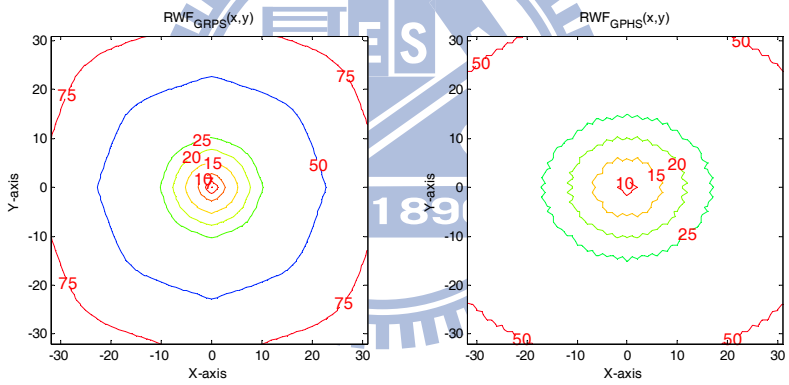


Fig. 5-21 Contour plots of the RWF for GRPS and GPHS.

5.4.1 Pattern Switching Strategy

The second component in an adaptive pattern search scheme is the pattern switching strategy. Similar to Subsection 4.2.2, we design one single level and one double level pattern switching strategies for each of the two selected pattern search sets. Totally four schemes are proposed. Each pattern search set comprises two pattern searches. One set is the traditional pattern searches {ERPS, PHS}, and the other set is the genetic pattern searches {GRPS, GPHS}. In Subsection 4.2.2, the original model with WF is used. In this subsection, we use the refined model with RWF

to decide a proper switching threshold and the pattern switching strategy for each pattern search set is described by a flowchart.

The key element in the pattern switching scheme is an adequate threshold. It is used to decide which pattern search is to be used. The target is to lower the computational complexity. That is, when search algorithm 1 (SA1) is in use, its average search points should be fewer than that produced by using search algorithm 2 (SA2). According to our refined average search point equation (eq. (5.5)), their difference in ASP is in (5.6).

$$\Delta_{ASP} = C_1 \times \sum_{x,y \in A} S_{FS}(x,y) \times (RWF_{SA1}(x,y) - RWF_{SA2}(x,y)) . \tag{5.6}$$

Note that both RWF_{SA1} and RWF_{SA2} depend on search algorithms only. Because S_{FS} is a function of the MV variance and Δ_{ASP} is thus picture-dependent, the parameter C_1 is fixed for a video sequence. Dividing Δ_{ASP} by C_1, we obtain the *refined switching index* (J_{ASP}) in (5.7).

$$J_{ASP} = \Delta_{ASP} / C_1 \tag{5.7}$$

Given a set of search patterns (SA1 and SA2), their J_{ASP} for a video sequence can be calculated. When $J_{ASP} > 0$, SA2 should be used; otherwise, SA1 should be selected. In principle, by using plural thresholds in a cascaded architecture, we may choose the best-performed pattern searches and expand the number of component pattern searches from two to many. Yet, it is impractical to identify a large set of pattern searches of which each pattern individually produces the least search points for a portion of image sequence. Therefore, only the 2-pattern search schemes are designed in the following examples.

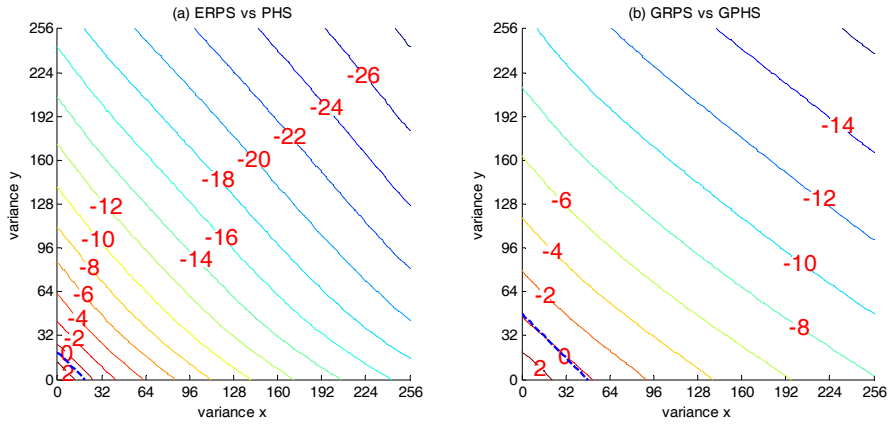


Fig. 5-22 The J_{ASP} between ERPS and PHS w.r.t. MV variance (a) and that between GRPS and GPHS w.r.t. MV variance (b).

Note that when the parameters of our model, such as (ζ_x, ζ_y) in $S_{FS}(x,y)$, are calculated based on the data of a picture (frame), J_{ASP} is a function of the MV variance measured for one frame. The J_{ASP} between ERPS and PHS, drawn against MV variance is shown in **Fig. 5-22**(a). In **Fig. 5-22** (a), the X-axis is the MV variance of the horizontal component and the Y-axis is that of the vertical component. When $J_{ASP} > 0$, ERPS outperforms PHS in terms of *ASP*, and when $J_{ASP} < 0$, PHS is better. Therefore, the switching criterion can be the MV variance values, at which J_{ASP} equals zero. For the case of ERPS and PHS pair, the threshold, J_{ASP}=0, is approximately a straight line below in the MV *variance* coordinates.

$$P \cdot VAR_X + Q \cdot VAR_Y = R\,. \tag{5.8}$$

Thus, this dash straight line in **Fig. 5-22** (a) acts as the pattern switching threshold. That is, Eq.(5.8) is used to decide the pattern search in use, wherein P, Q, and R are determined by the numerical methods. In our experiments, $P = 1$, $Q = 1$ and $R = 20$.

When GRPS and GPHS are the two component pattern searches, their J_{ASP} is shown in **Fig. 5-22** (b). Similarly, J_{ASP}=0 can be approximated by a straight line in the MV variance coordinates. This dash straight line in **Fig. 5-22** (b) serves as the pattern switching threshold. That is, Eq. (5.8) is again used to decide the pattern search and P=1, Q= 1 and R = 48 . Thus, similar to Subsection

4.2.2, single level pattern-switching strategy and double level pattern-switching strategy are developed.

Single Level Strategies

An adaptive pattern switching strategy is thus developed based on the threshold equation defined by (5.8). To ease the following discussions, the adaptive algorithm using ERPS and PHS is called *adaptive pattern search* (APS). Its flow chart is shown in **Fig. 5-23**. A similar algorithm is developed for the GRPS and GPHS pair and is called *adaptive genetic pattern search* (AGPS), which has a similar procedure but uses different parameters in Step S3 in **Fig. 5-23** and, of course, ERPS and PHS are replaced by GRPS and GPHS, respectively.

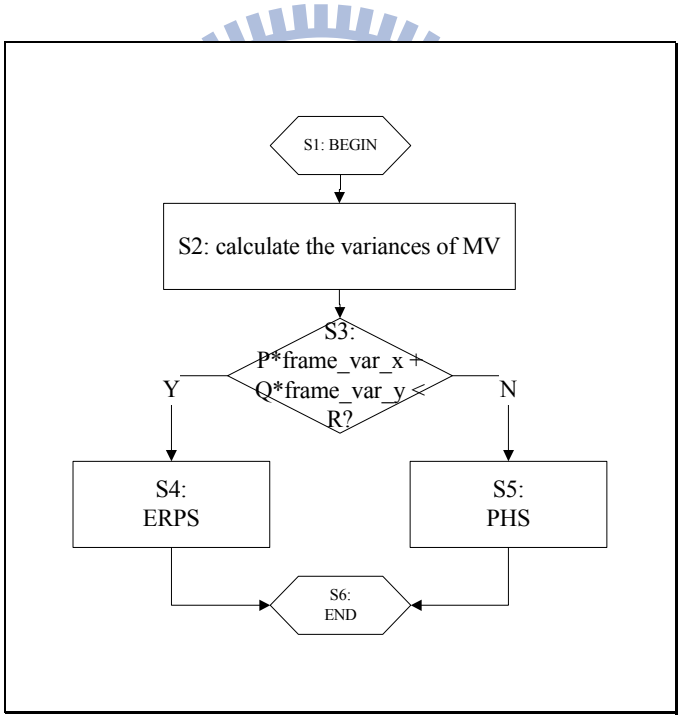


Fig. 5-23 The flow chart of APS.

Moreover, when comparing **Fig. 5-22** (a) to **Fig. 5-22** (b), we find that the optimal threshold for APS is smaller than that for AGPS. It means the search capability difference between the constituent pattern searches in APS locates in the relative low motion part but that in AGPS locates in the relative high motion part. This is consistent with the characteristics of the constituent pattern searches.

Double Level Strategies

Because the MV characteristics vary at different parts of a frame, using one single search pattern for the entire frame is a rough strategy. To refine this strategy, we also switch the search pattern for each image block inside a frame. Because the MV characteristics in the nearby spatial/temporal areas tend to be similar, three neighboring blocks in the current and previous frame are used in calculating the MV variance as defined by (4.8).

The so-called *double level pattern switching strategy* for APS (abbr. DL APS) constitutes the frame-level switching, which is similar to the single-level strategy, and the block-level switching, described by the last paragraph. Its flow chart is shown by **Fig. 5-24**. If the previous frame has small MV variances, we incline towards using ERPS as the search pattern with the exception that the MV variances derived from the nearby blocks are very large. On the other hand, if the previous frame has large MV variances, PHS is often chosen unless the MV variances derived from the neighboring blocks are very small. The parameter values of P, Q, R_{frame}, R_{block1}, and R_{block2} are derived from data by using the numerical method. In our experiments, $P = 1$, $Q = 1$, $R_{frame} = 20$, $R_{block1} = 3$, and $R_{block2} = 37$.

Likewise, the flow chart of the double level pattern switching strategy for AGPS (abbr. DL AGPS) is similar but the corresponding parameters in S3, S4, and S5 are $P = 1$, $Q = 1$, $R_{frame} = 48$, $R_{block1} = 6$, and $R_{block2} = 90$. Also, ERPS and PHS are replaced by GRPS and GPHS, respectively.

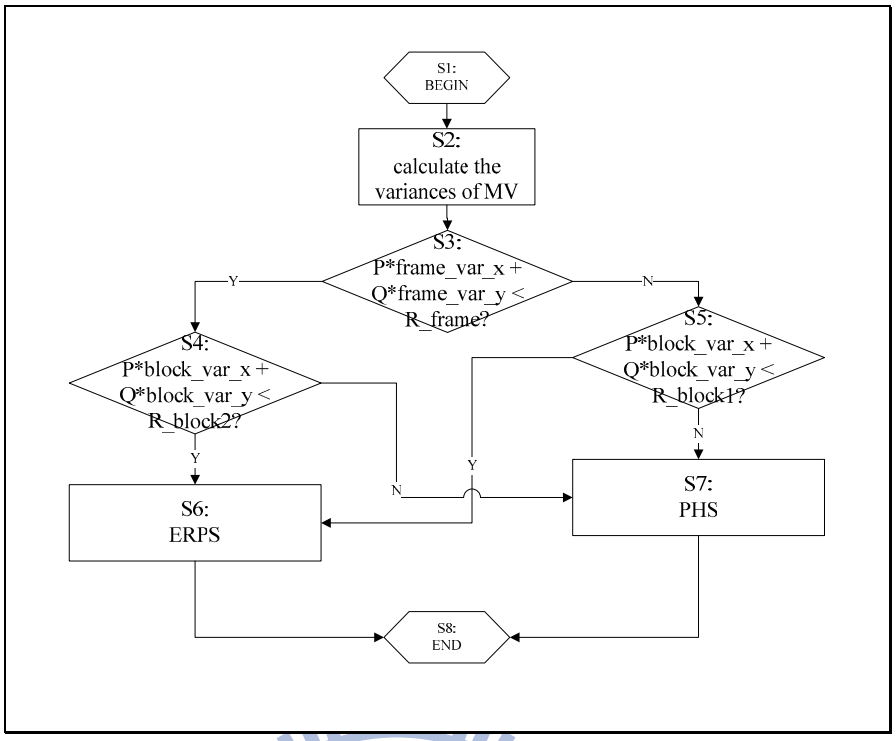


Fig. 5-24 Flow chart of the *double level adaptive pattern search* (DL APS).

5.4.2 Starting Point Selection

The third component in an adaptive pattern search scheme is the starting point or initial point in a search. Often, the starting point is predicted by using a combination of the MVs of a few neighboring blocks. The most probable MV *predictor* is used as the starting point for PBME algorithms. We like to design a criterion that evaluates the effectiveness of MV predictors and propose a systematic approach that constructs the optimal *Starting Point Set* (SPS). The DL AGPS and DL APS discussed in the previous section are the search algorithms used to test our SPS in this section.

We assume that the refined PBME model is valid for different starting points. Then, because the MV field acquired by FS is fixed for a given video sequence, a different starting point only does a translational shift on the motion vector distribution. Given two starting points, SP1 and SP2, their difference in *ASP* can be represented by (5.9).

$$\varepsilon_{ASP} = C_1 \times \sum_{x,y \in A} ((S_{FS\_SP1}(x,y) - S_{FS\_SP2}(x,y)) \times RWF_{SA}(x,y)) \tag{5.9}$$

Let SP2 be a fixed starting point for comparison purpose; (5.9) thus becomes (5.10), in which η is a constant.

$$\varepsilon_{ASP} = C_1 \times \sum_{x,y \in A} (S_{FS\_SP1}(x,y) \times RWF_{SA}(x,y)) - \eta \tag{5.10}$$

$$H_{ASP} = (\varepsilon_{ASP} + \eta)/C_1 = \sum_{x,y \in A} (S_{FS\_SP1}(x,y) \times RWF_{SA}(x,y)) \tag{5.11}$$

Rearrange (5.10), we obtain H_{ASP} defined by (5.11), which is a function of ASP using SP1. Thus, it is used as the performance assessment criterion for starting point evaluation. Because $RWF_{SA}(x,y)$ is fixed for a specific algorithm and only $S_{FS\_SP1}(x,y)$ may vary, H_{ASP} is a function of MV characteristics. Herein, the MV characteristics are the MV variances calculated from the MV w.r.t. a specific starting point (SP1). And the MVs are acquired by using FS on the selected sequences.

Similar to Section 4.3, we consider the MV candidates in **Fig. 4-12** in the starting point selection and investigate the representative MV predictors in Eq.(4.14)-(4.25). We find the MV predictors with the smallest average H_{ASP} and form the candidate set. We choose one or several starting points from the candidate set to form the *starting point set* (SPS) by using the progressive SPS construction in **Fig. 4-13**. Accordingly, we can get the constructed SPS. Like Section 4.3, we obtain similar SPS for our proposed search schemes.

5.4.3 Coding Performance

Test Image Sequences and Platform

Table 5-5 The test sequences and their parameters.

Abbreviation	Sequence	Bit rate (K bps)	Frame rate (fps)	Number of frames	PSNR
CR2048	crew	2048	60	600	35
CR1024	crew	1024	60	600	32
SC3072	soccer	3072	60	600	35

IC1536	ice	1536	60	480	38
MB768	mobile	768	30	300	25
FM512	foreman	512	30	300	34
FM1024	foreman	1024	30	300	36
FB1024	football	1024	30	90	35
FG768	flower garden	768	30	250	26
ST1024	Steven	1024	30	300	29

Table 5-5 lists the test image sequences (denoted as the '1X' sequences), their target coding bit rates (which are chosen to produce acceptable image quality), peak signal noise ratio (PSNR), and the other parameters. To test the extreme cases, we enlarge the extent of motion by generating some new sequences consisting of the odd frames of the '1X' sequences (denoted as the '2X' sequences) and one quarter frames of the original (denoted as the '4X' sequences). All the sequences are in the CIF (352X288) resolution. The video coding platform in our experiments is an MPEG-4 (SP@L3) encoder. Only the first frame is coded as I frame, and all the remaining frames are coded as P frames. The motion vector search range is 16, the initial quantization step size is set to 15, and the block size is 16x16. The quantization step is adjusted to achieve the desired bit rate. The frame skip and the block skip (macroblock not coded) modes are not in use.

Performance of Pattern Switching Strategy

Fig. 5-25 and **Fig. 5-26** show the performances of ERPS, PHS, APS and DL APS, and **Fig. 5-27** and **Fig. 5-28** show the performances of GRPS, GPHS, AGPS and DL AGPS, when they are tested on the '1X', '2X' and '4X' sequences under the settings given in **Table 5-5**. In these figures, 'ASP' is the average number of search points per block, and 'PSNR' is the average frame PSNR of a sequence.

Regarding the ASP performances of the conventional pattern searches in **Fig. 5-25**, PHS outperforms ERPS in 1 out of ten 1X sequences, 3 out of ten 2X sequences and 7 out of ten 4X sequences. And the computational complexity of our proposed APS and DL APS are usually

below the lower one of ERPS and PHS. In most cases, DL APS has the lowest computation complexity, and APS takes the second place. On average, APS outperforms EPRS by 4.1%, 8.9% and 15.6% for the 1X, 2X and 4X sequences, and outperforms PHS by 20.2%, 11.7% and 4.7%. And DL APS outperforms ERPS by 4.3%, 9.3% and 16.2%, and outperforms PHS by 20.5%, 12.1% and 5.2%. In terms of PSNR in **Fig. 5-26**, the performances of both APS and DL APS are very close to those of FS in all our test sequences. Specifically, their PSNR performances usually are between those of the constituent pattern searches.

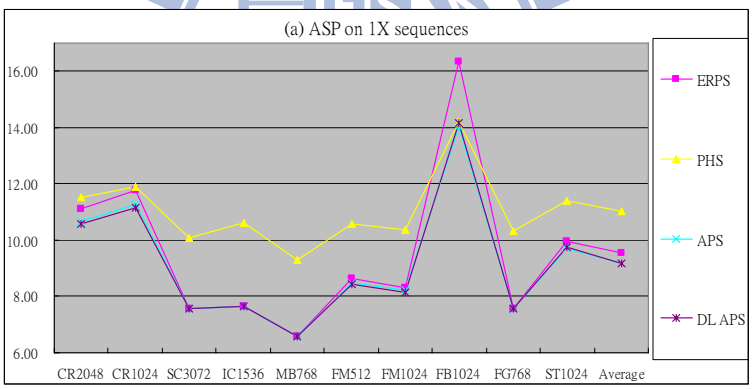
Regarding the ASP performances of the genetic pattern searches in **Fig. 5-27**, GPHS never outperforms GRPS in 1X sequences. Yet, GPHS outperforms GRPS in 1 out of ten 2X sequences and 2 out of ten 4X sequences. And the computational complexity of our proposed AGPS and DL AGPS are usually below the lower one of GRPS and GPHS. In most cases, DL AGPS has the lowest computation complexity, and AGPS takes the second place. On average, AGPS outperforms GPRS by 0.6%, 2.5% and 4.5% for the 1X, 2X and 4X sequences, and outperforms GPHS by 38.0%, 26.1% and 14.5%. And DL AGPS outperforms GRPS by 0.9%, 2.5% and 4.9%, and outperforms GPHS by 38.3%, 26.1% and 14.9%. In terms of PSNR in **Fig. 5-28**, the performances of AGPS and DL AGPS are very near to those of FS in all our test sequences. Likewise, their PSNR performances usually are between those of the constituent pattern searches.

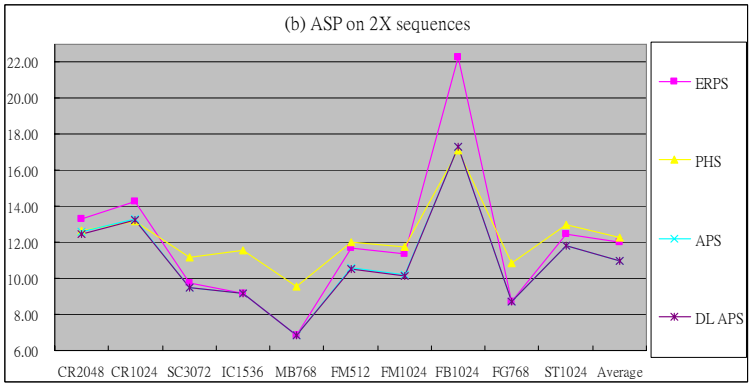
Clearly, the adaptive pattern switching strategy is robust. It does not hurt the low motion variance sequences but effectively reduces the computational complexity on the high motion variance sequences. The proposed algorithms outperform their constituent pattern search algorithms in ASP, and their PSNR qualities typically are in-between those of their constituent algorithms.

When we compare the conventional adaptive schemes with the genetic adaptive schemes, the genetic versions are better than their corresponding non-genetic versions in computational complexity. For example, GRPS is better than ERPS, and GPHS is better than PHS. GRPS is an

efficient search algorithm for almost all image sequences. Therefore, the advantage offered by the adaptive switching mechanism is relatively small for the genetic searches. In contrast, the adaptive pattern switching mechanism helps the conventional searches more. Though marginally, the double level strategy further improves in both PSNR and speed.

Note that the sequences with high but regular motions, like 'flower garden' (FG768), are considered as moderate motion sequences because we use a very good MV predictor. In our pattern switching schemes, *the MV difference to its predictor* decides which pattern search to be used. We do not compare our pattern switching algorithms, DL APS or DL AGPS, with the other pattern switching algorithms because our selected constituent pattern searches differ from those used by the other existing pattern switching algorithms. Moreover, the performances of our constituent pattern searches already exceed those of many known pattern switching algorithms.





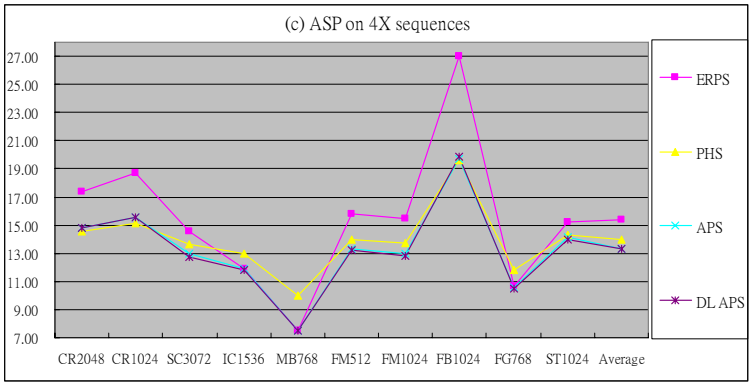


Fig. 5-25 The ASP values of applying ERPS, PHS, APS and DL APS on the 1X, 2X and 4X sequences.

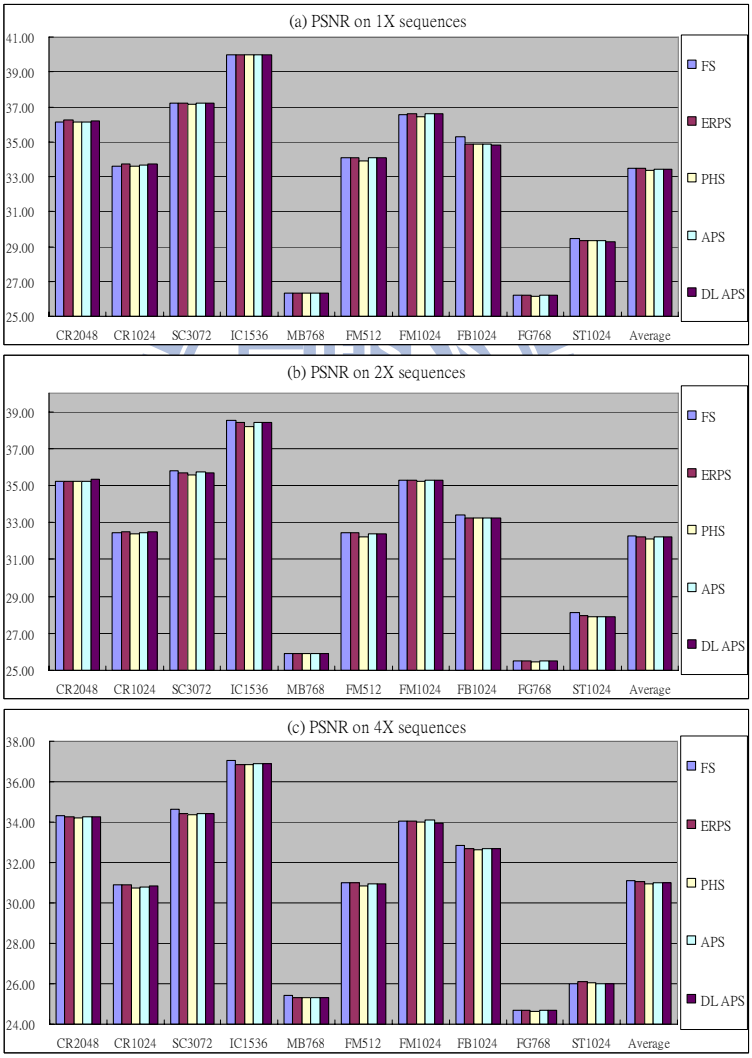


Fig. 5-26 The PSNR values of applying FS, ERPS, PHS, APS and DL APS on the 1X, 2X and 4X sequences.

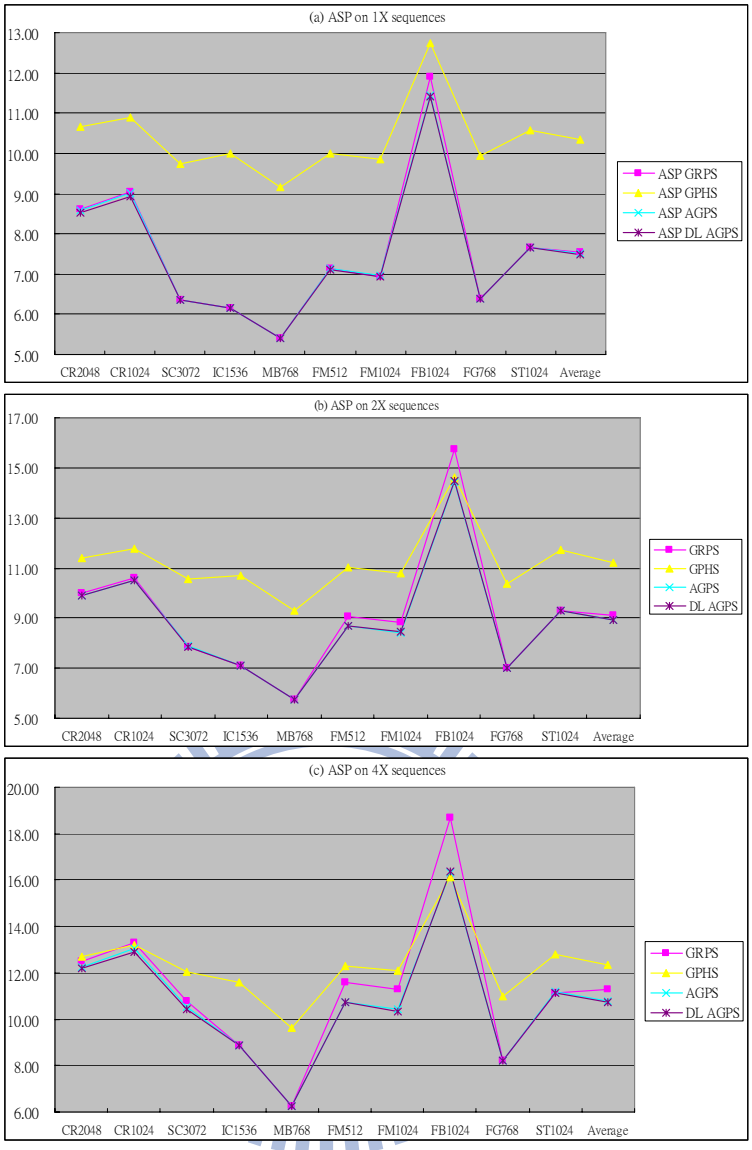
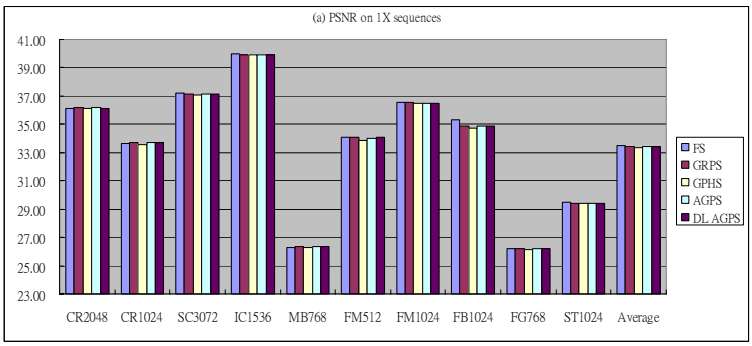


Fig. 5-27 The ASP values of applying GRPS, GPHS, AGPS and DL AGPS on the 1X, 2X and 4X sequences.



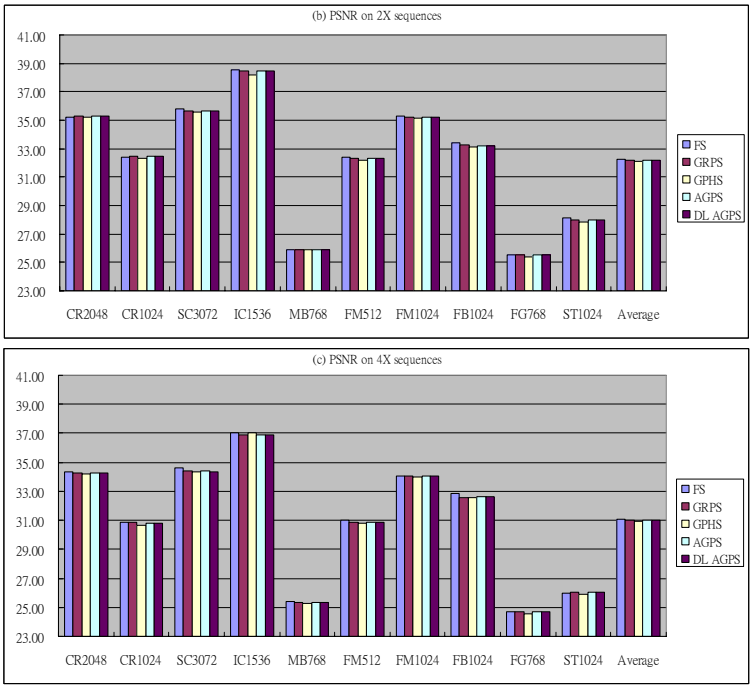


Fig. 5-28 The PSNR values of applying FS, GRPS, GPHS, AGPS and DL AGPS on the 1X, 2X and 4X sequences.

Discussions

To examine the correctness of the switching strategy, **Fig. 5-29** shows the frequency (in percentage) that PHS is chosen when the adaptive pattern schemes, APS and DL APS, are applied to the 1X, 2X and 4X sequences. **Fig. 5-30** shows the percentage that GPHS is chosen when the adaptive genetic pattern schemes, AGPS and DL AGPS, are applied to those sequences. In **Fig. 5-29**, the percentages of using PHS on the 4X sequences are higher than those on the 2X sequences, and in turn, the percentages on the 2X sequences are higher than those on the 1X sequences in both APS and DL APS. This is consistent with our earlier projection that the adaptive algorithms show advantages on fast moving sequences. Similar conclusion applies to the use of GPHS in both AGPS and DL AGPS.

Fig. 5-31 and **Fig. 5-32** display the pattern switching thresholds (represented by the dash straight line) and the *refined switching index* J_{ASP}. In these figures, a yellow dot denotes the J_{ASP} of an image frame (equivalently, an MV variance pair) and a cross denotes the J_{ASP} of an entire sequence. In **Fig. 5-31**, the dots in the higher-right part of J_{ASP}=0 are increasing as the sequences

are getting faster. Similar situation happens in **Fig. 5-32**. In **Fig. 5-31**, the MV variance pairs are evenly distributed on the two sides of the pattern switching threshold between ERPS and PHS. In contrast, in **Fig. 5-32**, most MV variance pairs are in the lower-left side of the pattern switching threshold designed for GRPS and GPHS. Consequently, the percentages of using PHS in **Fig. 5-29** are higher than that of using GPHS in **Fig. 5-30**.

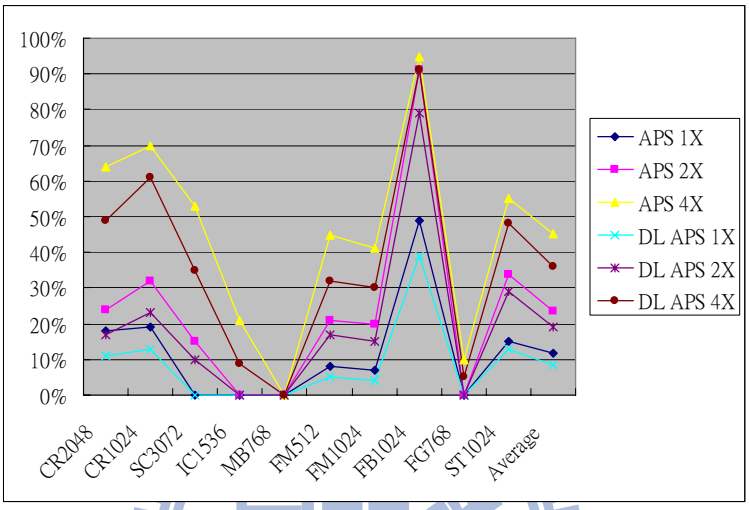


Fig. 5-29 The frequency (in percentage) that PHS is chosen when the adaptive pattern schemes, APS and DL APS, are applied to the 1X, 2X and 4X sequences.

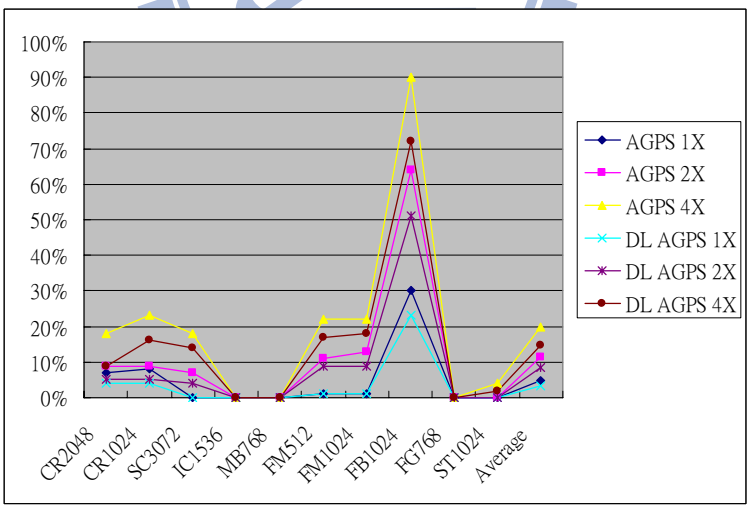


Fig. 5-30 The frequency (in percentage) that GPHS is chosen when the adaptive genetic pattern schemes, AGPS and DL AGPS, are applied to the 1X, 2X and 4X sequences.

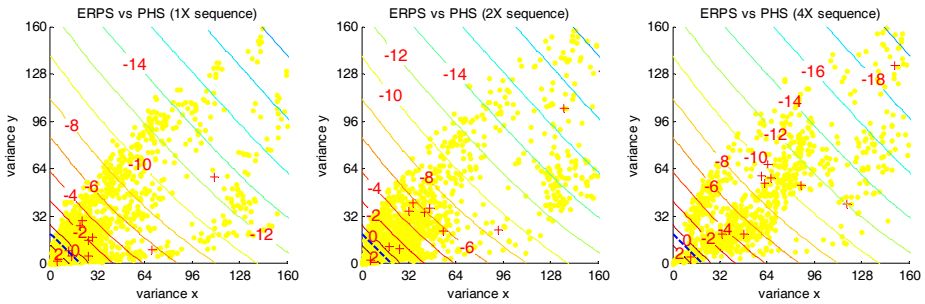


Fig. 5-31 Pattern switching threshold (dash line), J_{ASP} (solid line) and the frame MV variance of the 1X, 2X and 4X sequences when the constituent searches are ERPS and PHS.

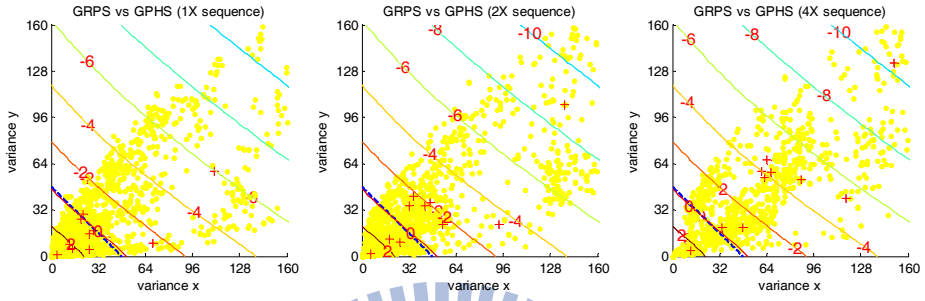


Fig. 5-32 Pattern switching threshold (dash line), J_{ASP} (solid line) and the frame MV variance of the 1X, 2X and 4X sequences when the constituent searches are GRPS and GPHS.

For our selected image sequences, the adaptive switching methods offer reasonable computation reduction and ensure robust performance in the occasional high motion cases. It provides nearly the best ASP with negligible PSNR degradation. Overall, our design methodology produces a stable and efficient fast MV search scheme that can be used for all types of motion sequences. Indeed, based on the refined ASP prediction model, we can systematically choose the nearly optimal decision threshold. In practical implementation, the non-linear ideal threshold function is approximated by a liner equation.

Section 5.5 Chapter Summary

This study tries to improve the modeling accuracy of the pattern-based motion vector search algorithms. Specifically, we propose the *refined weighting function*, which is defined as the *average number* of search points needed to find the best matched point. Based on the *QMSB matching error surface assumption*, we can analytically calculate the RWF for the search algorithms of our interests. RWF is a better replacement for our previously proposed *weighting function*. When it is used to predict the ASP performance of a new search algorithm, it reduces the

average prediction errors significantly.

In the model improving process, we also find clues to further speed up the previously proposed genetic pattern search algorithms. The basic idea is that the search direction used in the previous search steps hints us in finding the next matching point. By properly prioritizing the candidate search order, we lower the average computations. Two momentum-directed genetic pattern search algorithms are thus devised. Simulation results show that the modified algorithms offer 5% to 7% average computation reduction when compared with their corresponding genetic pattern searches.

In addition, this study provides a methodology to design a robust and high performance adaptive pattern search algorithms. Our refined analytical model for pattern-based block motion estimations (PBME) serves as the foundation of the entire design process. The refined model can accurately predict the average *number of search points* (ASP) of a single pattern search. By using the refined model, we re-examine the critical coding tools in the adaptive pattern searches, the decision threshold and the starting point set.

Our proposed design methodology leads to a systematic procedure in choosing the appropriate threshold for selecting the pattern search. Based on the characteristics of the constituent searches, the motion vector variance is chosen as the decision metric. And the threshold function is well approximated by a linear equation to reduce computation. Accordingly, the frame-level switching strategy and the block-level switching strategy are constructed. With different constituent pattern searches, two examples of adaptive pattern search design are presented. One uses the conventional pattern searches and the other uses genetic pattern searches. A most distinct advantage of the adaptive schemes is their robust performance (in both computation and quality) on both slow and fast motion sequences.

Chapter 6 Conclusions

A systematic approach is taken in this dissertation to construct a mathematical model for the pattern based block motion estimation (PBME) algorithms. The complete PBME model includes two elements: the statistical probability distribution function (PDF) of motion vectors derived from a video sequence and the *weighting function* (WF) derived from a search algorithm. With the proposed model, we can predict the performance of a new search pattern without actually applying the search algorithm to a video sequence. Thus, it helps us in constructing new patterns searches. Two application examples are given. One is the design of a genetic pattern search, and the other is the performance prediction of a PBME algorithm.

Based on the analytical model, three important techniques have been investigated for reducing complexity of PBME. They are adaptive pattern switch, starting point selection and early termination. The contribution of this study is to examine these techniques using a systematic approach. Optimal or nearly optimal solutions are thus identified. A PBME algorithm combining all the above features is constructed and the simulations show that the search speed of the proposed algorithm is much faster than many previous search algorithms and its coding quality is kept at about the same PSNR level.

We further improve the modeling accuracy of the pattern-based motion vector search algorithms. Specifically, we propose the *refined weighting function* (RWF) on the quadrant monotonic function with smooth quadrant border (*QMSB*) *matching error surface assumption.* RWF is a better replacement for WF. When it is used to predict the ASP performance of a new search algorithm, it reduces the average prediction errors significantly. In the model improving process, we also find clues to further speed up the previously proposed genetic pattern search algorithms. Two momentum-directed genetic pattern search algorithms are thus devised. We also use our refined analytical model for PBME as the foundation of the PBME design process. In

re-examining the major coding tools in the adaptive pattern search algorithm, we suggest a better threshold for selecting the pattern search.

In summary, we build an analytical model for PBME and demonstrate how we use the model to develop new pattern searches for video coding applications. In addition, we further refine the model and apply it for developing better search schemes.

Section 6.1 Future Works

This study may be further extended in two directions as described in this section. One is regarding the theoretical robustness of our proposed model and the other, new video coding applications.

Theory

In developing the analytical model, the motion vectors are obtained by applying the full search (FS) to video sequences of CIF size (352X288) in integer pixel precision and the block size in the block matching process is fixed to 16X16. Yet, in current standardization process of next generation video coding standard (high efficiency video coding, HEVC), video sequences may alter from Sub-QCIF (128X96) to 16HD (7680X4320) in size [48], the block size may vary from 2X2 to 64X64 [49] and the pixel precision may range from 1/4 pixel to 1 pixel. With the variations in these three factors, the accuracy of proposed analytical model may need adjustment. We should examine the influences of image size and block size. If a mismatch occurs, additional parameters may be introduced into this model. Providing that additional parameters are introduced to the model, we may develop new fast algorithms for mode selection and sub-pixel motion estimation.

Application

The computation time of PBME varies drastically due to the nature of PBME. This high variation is not welcomed by the practical system, whose computing power (due to its hardware or software platform) is usually limited and fixed. The frame-to-frame and block-to-block

computation variation may cause various synchronization problems or, otherwise, may waste computation power. Thus, the computation-aware motion estimation is desirable. How can we achieve the best coded image quality under the given or restricted computation resources? Can our proposed analytical model help in designing the computation aware motion estimation schemes? These issues are worth exploration.

Chapter 7 References

[1] B.G. Haskell and J.O. Limb, "Predictive video encoding using measured subjective velocity", U.S. Patent No. 3, 632, 865, Jan. 1972.

[2] F. Rocca, "Television bandwidth compression utilizing frame-to-frame correlation and movement compensation", In Proc. *1969 Symp. Picture Bandwidth Compression*, T.S. Huang and O.J. Tretiak (Eds.), Gordon and Breach, New York, NY, 1972.

[3] H.-M. Hang, Y.-M. Chou and S.-C. Cheng "Motion estimation for video coding standards", *Journal of VLSI Signal Processing*, Vol. 17, pp. 113-136, 1997.

[4] A. H. Sadka, *Compressed Video Communications*, John Wiley and Sons Ltd., 2002.

[5] T. Wiegand, et al., "Overview of the H.264/AVC video coding standard", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 13, pp. 560–576, July 2003.

[6] ISO/IEC 14496–10, *Information technology - Coding of Audiovisual Objects—Part 10: Advanced Video Coding,* 2003, also ITU-T Recommendation H.264, *Advanced Video Coding for Generic Audiovisual Services.*

[7] ISO/IEC 14496-2, *Information Technology – Coding of Audio-Visual objects – Part2: Visual*, MPEG-4, Dec, 2001.

[8] C. Zhu, W. S. Qi and W. Ser, "Predictive fine granularity successive elimination for fast optimal block-matching motion estimation", *IEEE Trans. Image Processing,* vol. 14, no. 2, pp. 213- 221, Feb. 2005.

[9] P. I. Hosur and K.-K. Ma, "Motion vector field adaptive fast motion estimation", presented at the *2nd Int. Conf. Information, Communications, and Signal Processing (ICICS)*, Singapore, Dec. 1999, CDROM.

[10] S.D. Silvey, *Statistical Inference*, London, England, Chapman Hall, 1975.

[11] W. J. Conover, *Practical Nonparametric Statistics*, John Wiley & Sons, 1980.

[12] R. Reininger and J. Gibson, **"**Distributions of the two-dimensional DCT coefficients for images", *IEEE Trans. Communication*, vol. 31, no. 6, Jun. 1983.

[13] J. Peacock, "Two-dimensional goodness-of-fit testing in astronomy", *Monthly Notices of the Royal Astronomical Society*, vol. 202, pp.615, 1983.

[14] A.M. Tourapis, O.C. Au, and M.L. Liou, "Predictive motion vector field adaptive search technique (PMVFAST) - enhancing block based motion estimation" , Proc. *Visual Communications and Image Processing (VCIP'01)*, pp.883-892, Jan. 2001.

[15] A.M. Tourapis, O.C. Au, and M.L. Liou, “New results on zonal based motion estimation algorithms-advanced predictive diamond zonal search”, Proc. *IEEE Int. Symp. Circuits and Systems (ISCAS’01)*, vol. 5, May 2001.

[16] A. M. Tourapis, “Enhanced predictive zonal search for single and multiple frame motion estimation”, Proc. *Visual Communications* and *Image Processing (VCIP’02)*, Jan. 2002.

[17] T. Koga, K. Iinuama, A. Hirano, Y. Iijima and T. Ishiguro, "Motion compensated inter frame coding for video conferencing", in 1981 Proc. *Nat. Telecommunication Conf.*, Nov. 1981.

[18] M. Ghanbari, "The cross-search algorithm for motion estimation", *IEEE Trans. Communications*, vol.38, no.7, pp.950-953, Jul 1990.

[19] C. H. Cheung and L.M. Po, “A novel cross-diamond search algorithm for fast block motion estimation”, *IEEE Trans. Circuits and Systems for Video Technology*, vol. 12, no. 12, pp.1168-1177, Dec. 2002.

[20] C. H. Cheung and L.M. Po, “A new cross-diamond search algorithm for fast block matching motion estimation”, in 2003 Proc. *Int. Conf. Neural Networks and Signal Processing*, vol. 2, p.p.1262-1265, Dec. 2003.

[21] C. H. Cheung and L.-M. Po, “Novel cross-diamond-hexagonal search algorithms for fast block motion estimation”, *IEEE Trans. Multimedia*, vol. 7, no. 1, pp.16-22, Feb. 2005.

[22] K.H.-K. Chow and M.L. Liou, “Genetic motion search algorithm for video compression”, *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 3, pp. 440 – 445, Dec. 1993.

[23] C.-H. Lin and J.-L. Wu, “A lightweight genetic block-matching algorithm for video coding”, *IEEE Trans. Circuits and Systems for Video Technology*, vol. 8, no. 4, pp.386-392, Aug. 1998.

[24] M. F. So and A. Wu, “Four-step genetic search for block motion estimation” , IEEE Proc. *IEEE Int. Conf. Acoustic, Speech, and Signal Processing (ICASSP’98)*, vol. 3, pp. 1393-1396, May 1998.

[25] M.Z. Coban and R. M. Mersereau, “A fast exhaustive search algorithm for rate-constrained motion estimation”, *IEEE Trans. Image Processing*, vol. 7, no. 5, pp. 769–773, May 1998.

[26] L. M. Po and W. C. Ma, "A novel four-step search algorithm for fast block motion estimation", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 6, pp. 313-317, June 1996.

[27] S. Zhu and K.-K. Ma, "A new diamond search algorithm for fast block-matching motion estimation", Proc. *Int. Conf. Information, Communications and Signal Processing (ICICS’97)*, vol. 1, pp. 292-296, Sep. 9-12, 1997.

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

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

[30] Y. Nie and K.-K. Ma, "Adaptive irregular pattern search with matching prejudgment for fast block-matching motion estimation", *IEEE Trans. Circuits and Systems for Video Technology*, vol.15, no.6, pp. 789- 794, June 2005.

[31] Y. Nie and K.-K. Ma, "Adaptive rood pattern search for fast block-matching motion estimation", *IEEE Trans. Image Processing*, vol. 11, no. 12, pp. 1442-1449, Dec. 2002.

[32] C. Zhu, X. Lin, L.P. Chau, and L.M. Po, "Enhanced hexagonal search for fast block motion estimation", *IEEE Trans. Circuits Systems for Video Technology*, vol. 14, no. 10, pp. 1210-1214, Oct. 2004.

[33] L.-M. Po, et al., "Novel Point-Oriented Inner Searches for Fast Block Motion Estimation", *IEEE Trans. Multimedia*, vol.9, no.1, pp.9-15, Jan. 2007.

[34] C.-M. Kuo et al., "A novel prediction-based directional asymmetric search algorithm for fast block-matching motion estimation", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 19, no.6, pp.893-899, June 2009.

[35] I. Gonzalez-Diaz, et al., "Adaptive Multi-Pattern Fast Block-Matching Algorithm Based on Motion Classification Techniques," Proc. *IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP'07)*, Hawaii, U.S.A., Apr. 2007, vol.1, pp.I-1177-I-1180.

[36] I. Gonzalez-Diaz, and F. Diaz-de-Maria, "Improved Motion Classification Techniques for Adaptive Multi-Pattern Fast Block-Matching Algorithm", Proc. *IEEE Int. Conf. Image Processing (ICIP '07)*, Texas, U.S.A., Oct. 2007.

[37] I. Gonzalez-Diaz, and F. Diaz-de-Maria, "Adaptive multipattern fast block-Matching algorithm based on motion classification techniques", *IEEE Trans. Circuits and Systems for Video Technology*, vol.18, no.10, pp. 1369-1382, Oct. 2008.

[38] K.-H. Ng et al., "A search patterns switching algorithm for block motion estimation", IEEE *Trans. Circuits and Systems for Video Technology*, vol.19, no.5, pp. 753-759, May 2009.

[39] S.-Y. Huang, C.-Y. Cho, and J.-S. Wang, "Adaptive fast block-matching algorithm by switching search patterns for sequences with wide-range motion content", IEEE *Trans. Circuits and Systems for Video Technology*, vol. 15, no. 11, pp.1373-1384, Nov. 2005.

[40] S.-F. Lin, M.-T. Lu, C.-H. Pan, and H.H. Chen, "Fast multi-frame motion estimation for

H.264 and its applications to complexity-aware streaming", Proc. *IEEE Int. Symp. Circuits and Systems (ISCAS'05)*, Kobe, Japan, May 2005, vol.2, pp. 1505-1508.

[41] I. Ahmad, et al., "A fast adaptive motion estimation algorithm", *IEEE Trans. Circuits and Systems for Video Technology*, vol.16, no.3, pp.420-438, Mar. 2006.

[42] M.-C. Chang and J.-S. Chien, "An adaptive search algorithm based on block classification for fast block motion estimation", Proc. *IEEE Int. Symp. Circuits and Systems (ISCAS'06)*, 21-24 May 2006, pp. 3982-3985.

[43] Y. Liu and S. Oraintara, "Complexity comparison of fast block-matching motion estimation algorithms", *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP'04)*, May 2004, vol. 3, pp. ii.i-341-4.

[44] J. Jain and A. Jain, "Displacement measurement and its application in interframe image coding", *IEEE Trans. Communications*, vol.29, no.12, pp. 1799-1808, Dec. 1981.

[45] A. Puri, H.-M. Hang, and D. Schilling., "An efficient block-matching algorithm for motion-compensated coding", Proc. *IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP '87)*, vol.12, pp. 1063-1066, Apr. 1987.

[46] J. Luo et al., "Motion estimation for content adaptive video compression", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 18, no.7, pp.900-909, July 2008.

[47] L.-M. Po et al., "Novel directional gradient descent searches for fast block motion estimation", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 19, no.8, pp.1189-1195, Aug. 2009.

[48] S. Sakaida et al., "7680X4320 format test sequences for JCT-VC", Contribution ISO/IEC JTC1/SC29/WG11 JCT-VC/A023, Apr. 2010.

[49] K. Panusopone et al., "AHG Report – Large block structures", Contribution ISO/IEC JTC1/SC29/WG11 JCT-VC/B008, Jul. 2010.

[50] J.-J. Tsai and H.-C. Chen, "Predictive block-matching discrepancy based rhombus pattern search for block motion estimation", Proc. *IEEE Int. Conf. Image Processing (ICIP'05)*, Genova, Italy, Sep. 2005, pp. I-1073-6.

[51] J.-J. Tsai and H.-M. Hang, "A genetic rhombus pattern search for block motion estimation", Proc. *IEEE Int. Symp. Circuits and Systems (ISCAS'07)*, New Orleans, U.S.A., May. 2007, pp. 3655-3658.

[52] J.-J. Tsai and H.-M. Hang, "On adaptive pattern selection for block motion estimation algorithms", Proc. *IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP'07)*, Hawaii, U.S.A., Apr. 2007, vol. 1, pp. I-1173-I-1176.

[53] J.-J. Tsai and H.-M. Hang, "On Modeling genetic pattern search for block motion estimation", *Proc. IEEE Int. Conf. Image Processing (ICIP'08)*, Oct. 2008, pp.1980-1983.

[54] J.-J. Tsai and H.-M. Hang, "Modeling of pattern-based block motion estimation and its application", IEEE *Trans. Circuits and Systems for Video Technology*, vol.19, no.1, pp. 108-113, Jan. 2009.

[55] J.-J. Tsai and H.-M. Hang, "On the Design of Pattern-Based Block Motion Estimation Algorithms", IEEE *Trans. Circuits and Systems for Video Technology*, vol.20, no.1, pp. 136-143, Jan. 2010.

[56] J.-J. Tsai and H.-M. Hang, "On modeling genetic pattern-based block motion estimation", submitted for publication, Aug. 2010.

[57] J.-J. Tsai and H.-M. Hang, "Analysis and design of adaptive pattern searches for block motion estimation", submitted for publication, Aug. 2010.

[58] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves", in Document VCEG-M33, VCEG 13th, Austin, Texas, U.S.A., Apr. 2001.

[59] G. Bjontegaard, "An excel add-in for computing Bjontegaard metric and its evolution", in Document VCEG-AE07, VCEG 31st, Marrakech, MA, 15-16 January, 2007.

Vita

Jang-Jer Tsai was born in Taiwan, R.O.C. He received B.S. degree from Department of Electrical Engineering, National Tsing-Hua University, Hsin-Chu, Taiwan, in 1997, M.S. and Ph.D. degree from Department of Electronics Engineering and Institute of Electronics, National Chiao-Tung University, Hsin-Chu, Taiwan, in 1999 and 2010, respectively.

From 1999 to 2004, he was with Computer and Communication Laboratories, Industrial Technology Research Institute, wherein he was engaged in the embedded system design, the multimedia set top boxes and the advanced reconfigurable processor. From 2004 to 2009, he was with Department of Algorithm Design, Pixart Imaging Inc. In this period of time, he had been involved in the development of video/audio compression systems and human-input devices. He had participated in the MPEG-4 and H.264 codec design for mobile phones. Particularly, he specialized in optimizing the video encoder.

He has published 5 journal papers and 7 conference papers related to image compression, video codec design and embedded systems. Currently he holds 9 patents (R.O.C, China, and U.S.) in these areas. He has been an IEEE reviewer for IEEE Trans. Circuits and Systems for Video Technology and IEEE Trans. Image Processing since 2007.