Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation

Yao Nie, *Student Member, IEEE,* and Kai-Kuang Ma, *Senior Member, IEEE*

***Abstract*—In this paper, we propose a novel and simple fast block-matching algorithm (BMA), called *adaptive rood pattern search* (ARPS), which consists of two sequential search stages: 1) initial search and 2) refined local search. For each macroblock (MB), the initial search is performed only once at the beginning in order to find a good starting point for the follow-up refined local search. By doing so, unnecessary intermediate search and the risk of being trapped into local minimum matching error points could be greatly reduced in long search case. For the initial search stage, an *adaptive rood pattern* (ARP) is proposed, and the ARP's *size* is dynamically determined for each MB, based on the available motion vectors (MVs) of the neighboring MBs. In the refined local search stage, a *unit-size rood pattern* (URP) is exploited repeatedly, and unrestrictedly, until the final MV is found. To further speed up the search, *zero-motion prejudgment* (ZMP) is incorporated in our method, which is particularly beneficial to those video sequences containing small motion contents. Extensive experiments conducted based on the MPEG-4 *Verification Model* (VM) encoding platform show that the search speed of our proposed ARPS–ZMP is about two to three times faster than that of the *diamond search* (DS), and our method even achieves higher *peak signal-to-noise ratio* (PSNR) particularly for those video sequences containing large and/or complex motion contents.**

***Index Terms*—Adaptive search pattern, block-matching algorithm, diamond search, fast motion estimation, rood pattern, spatial correlation, video compression, zero-motion prejudgment.**

I. Introduction

BLOCK-MATCHING algorithm (BMA) for motion estimation (ME) has been widely adopted by current video coding standards such as H.261, H.263, MPEG-1, MPEG-2, MPEG-4, and H.264 due to its effectiveness and simplicity for implementation. The most straightforward BMA is the *full search* (FS), which exhaustively searches for the best matching block within the search window. However, FS yields very high computational complexity and makes ME the main bottleneck in real-time video coding applications. Thus, using a fast BMA is indispensable to reduce the computational cost. In our views, existing fast BMAs can be classified into four categories as follows.

A. Fast BMA Using a Fixed Set of Search Patterns

In this category, the methods are based on the assumption that ME matching error decreases monotonically as the search moves toward the position of the global minimum error [1] and the error surface is uni-modal as shown in Fig. 1(a). The motion vector (MV) of each block is searched independently by using a fixed set of search patterns. Well-known examples are *2-D logarithmic search* (LOGS) [1], *three-step search* (TSS) [2], *four-step search* (4SS) [3], *block-based gradient descent search* (BBGDS) [4] and the latest *diamond search* (DS) [5]. Simplicity and regularity are the most important advantages of these methods, which make them attractive for implementation. However, they have less adaptability and search efficiency in tracking large motions.

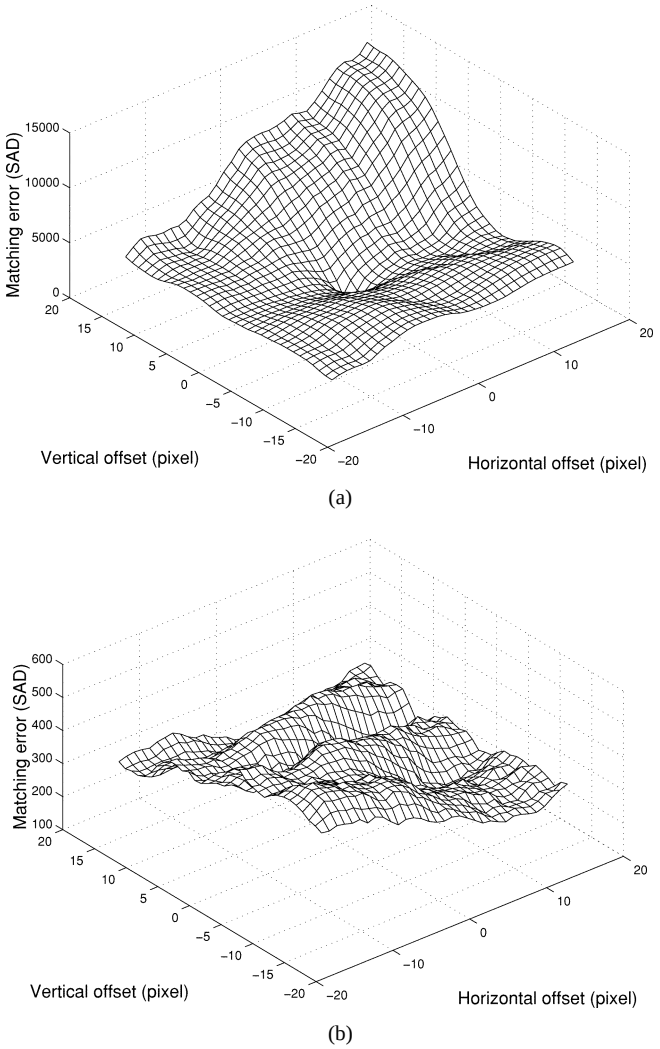

Fig. 1. Examples of matching error surface. (a) Uni-modal error surface with a global minimum error point. (b) Non-uni-modal error surface with multiple local minimum error points.

Manuscript received July 13, 2001; revised September 19, 2002. This work was supported by Nanyang Technological University Scholarship during 1998 to 2000 as part of Y. Nie's M.S. thesis. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Patrick Perez.

Y. Nie is with the Department of Electrical and Computer Engineering University of Delaware, Newark, DE 19716 USA (e-mail: nieyao@hotmail.com).

K.-K. Ma is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (e-mail: ekkma@ntu.edu.sg).

Digital Object Identifier 10.1109/TIP.2002.806251

Among them, DS was accepted in the MPEG-4 Verification Model (VM) [6]. The derivation of DS exploits the characteristic of the center-biased MV distribution typically existed in real-world video sequences and develops two diamond-shaped search patterns: 1) the *large diamond search pattern* (LDSP) with nine search points and 2) the *small diamond search pattern* (SDSP) with five search points. The search process of DS starts from the center of the search window using LDSP, and this pattern will be repeatedly used until the center position of LDSP incurs the *minimal matching error* (MME) in any search iteration. The search pattern is then switched from LDSP to SDSP, which will be used only once to find out the best matching block. DS greatly outperforms other BMAs in terms of search accuracy, efficiency and computational complexity.

However, when the size of the fixed search pattern does not match the magnitude of the actual motion, over search or under search will be incurred which can cause certain search deficiency and inaccuracy. For example, in DS, LDSP will be too large for searching a small MV with the length less than 2 pixels away from the search center, thus causing unnecessary searches (i.e., *over* search). On the other hand, in the case of large and complex motion (e.g., in Foreman sequence), the characteristic of center-biased MV distribution is very weak, and the unimodal error surface assumption is no longer valid [see Fig. 1(b)]. Even LDSP could be too small for searching large MV (i.e., *under* search) and leads to either a long search path (causing unnecessary intermediate searches) or being trapped into a local minimum matching error point (yielding large matching errors and degrading video quality). The above-mentioned observations lay the foundation of developing our adaptive search pattern and search strategy as proposed in this paper for performing fast block-matching ME.

B. Fast BMA Based on Inter-Block Correlation

Instead of using pre-determined search patterns, the methods in this category exploit the correlation between the current block and its neighboring blocks in the spatial and/or temporal domains to predict the target MV. The predicted MV is obtained by calculating the statistical average (such as the mean, the median, the weighted mean/median, etc.) of the neighboring MVs [7], [8] or selecting one of the neighboring MVs according to certain criteria [9]–[12]. After the prediction, the search window's size and the search center are re-defined accordingly, and FS is then performed within this new search window [9]–[12]. In other methods, fast BMAs instead of FS are performed around the predicted MV without re-defining the search window [7], [8], [11]. These methods have achieved quite appreciable performance at the expense of computing the prediction. Note that additional memory for storing the neighboring MVs are needed in these methods.

C. Fast BMA Using Hierarchical or Multiresolution Search Framework

Methods in this category explore the correlation between different levels of representation of the same image. In *hierarchical* methods [15], [16], the image size at different levels are identical to each other, while the block size varies, with lower level having larger block size. It is assumed that larger block's MV in the lower level can be used as a good prediction of the MVs of the smaller blocks (covered by the corresponding larger block) in the higher level [17]. However, this assumption often mismatches the actual situation, and consequently could lead to poor performance [17].

Multiresolution methods use different image resolutions with smaller image size at coarser level. In [18], constant block size is used so that one block in the coarser level covers several corresponding blocks at its next finer level. In [19], variable block sizes (thus constant number of blocks) are employed at each level to maintain an one-to-one correspondence between the blocks in different levels. The methods belonging to this class have relatively good performances in terms of estimation accuracy. However, FS with a reduced search window is employed in most of these methods to search for the final MV, so it still requires considerable amount of computation. Investigations for further improvement can be found in [20], where fast MV estimation algorithm that exploits the spatio–temporal correlations of MVs is incorporated in the multiresolution framework. (In this sense, [20] also belongs to category B.) This method achieves speed-up ratio (with respect to FS) in the range of 150∼310, under the penalty of 2%∼7% mean-square error (MSE) increment [20]. However, its high algorithmic complexity and memory requirement imposed are not desirable for hardware implementation.

D. Fast BMA Using Subsampled Pixels on Matching-Error Computations

The common objective shared by the above-mentioned three groups of BMAs is that they endeavor on reducing the computation of ME by limiting the number of search locations. The methods in this category reduce the number of pixels used in matching error computation to speed up ME [21], [22]. One method [21] uses only a fraction of pixels within a block by performing $2:1$ pixel subsampling in both horizontal and vertical directions. As a result, the total computation can be reduced by a factor of 4. Such computation reduction methodology can be incorporated into other BMAs to achieve higher computational gain.

From our point of view, each class of algorithms achieves different tradeoff between algorithm complexity, search speed and picture quality. Our idea is to combine the pattern-based method with the spatial-correlation based method. Inter-block correlations dynamically determine the size of the search pattern. Simplicity and feasibility for implementation are also our major concerns for the algorithm development.

We compared the proposed algorithm with the first two classes since they are closely related to our work and are at the similar complexity level. Detailed comparison with the state-of-the-art algorithm—DS—is presented through simulation results. Analysis and comparison with typical techniques from class B are described during the development of our method. Algorithms in categories C and D are conceptually different from ours, therefore it is inappropriate to make side-by-side comparison.

In Section II, we describe our ARPS method in detail. The performance of the proposed method is demonstrated by pre-

senting extensive experimental results in Section III. Conclusions are drawn in Section IV.

II. Adaptive Rood Pattern Search (ARPS)

A. Overview of Our Method

As we have observed, a small search pattern made up by compactly spaced search points is more suitable than a large search pattern containing sparsely spaced search points in detecting small motions, because only a small number of positions around the search window center are necessary to be checked. However, when searching for a large MV, the small pattern tends to be trapped into local minimum along the search path and leads to wrong estimation. On the contrary, the large search pattern has the advantage of quickly detecting large motions, but it will incur unnecessary search for small MVs. In summary, the speed and accuracy of pattern-based search algorithms intimately depend on the *size* of the search pattern and the *magnitude* of the target MV. Therefore, it is highly desirable to use different search patterns according to the estimated motion behavior (in terms of the magnitude of motion) for the current block. This boils down to two issues required to be addressed: 1) how to pre-determine the motion behavior of the current block for performing efficient ME? and 2) what are the most suitable size and shape of the search pattern(s)?

Regarding the first issue, in most cases, adjacent MBs belong to the same moving object have the similar motions. Therefore, the current block's motion behavior can be reasonably predicted by referring to its neighboring blocks' MVs in the spatial and/or temporal domains. As for the second issue, we use two types of search patterns. One is the *adaptive rood pattern* (ARP) with adjustable rood arm (thus, the pattern size), which is dynamically determined for each MB according to its predicted motion behavior. Note that ARP will be exploited only *once* at the beginning of each MB search. The objective is to find a good starting point for the remaining local search so as to avoid unnecessary intermediate search and reduce the risk of being trapped into local minimum in the case of long search path. The new starting point identified is hopefully as close to the *global* minimum as possible. If so (in fact, very likely), a small, compact, and fixed-size search pattern would be able to complete the remaining local search quickly. Note also that this small search pattern will be repeatedly and unrestrictedly used in the refined local search until the final MV is found.

In the following, we will be discussing our approaches to effectively address the above-mentioned two issues in detail.

B. Prediction of the Target MV

In order to obtain an accurate MV prediction of the current block, two factors need to be considered: 1) choice of the *region of support* (ROS) that consists of the neighboring blocks whose MVs will be used to calculate the predicted MV, and 2) algorithm used for computing the predicted MV.

In the *temporal* domain, block in the reference frame at the same position as that of the current block is a straightforward choice as a temporal ROS candidate. Furthermore, neighboring blocks from the same reference frame could also provide promising candidates for prediction as well. However, utilizing temporal correlation needs recording the entire previous MV field, which might be undesirable in practical implementation with limited storage space. Therefore, we only exploit the spatial correlation in our method.

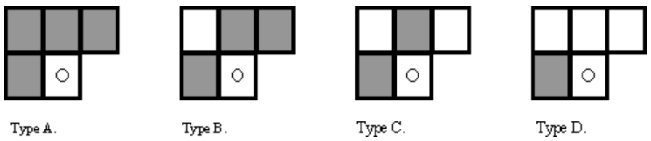

Fig. 2. Four types of ROS, depicted by the shaded blocks. The block marked by "◯" is the current block.

In the *spatial* domain, since all the blocks within a video frame are processed in a raster-scan order, the adjacent blocks whose MVs are available for reference are on the immediate left, above, above-left and above-right to the current block. The blocks in other nearby positions are less correlated to the current block and thus are not reliable for prediction. Since the usage of more blocks will involve higher computational complexity, the spatial ROS is thus limited to the neighboring block(s) with four promising scenarios as shown in Fig. 2. Type A covers all the four neighboring blocks, and +type B is the prediction ROS adopted in some international standards such as H.263 for differential coding of MVs. Type C is composed of two directly adjacent blocks, and type D has only one block that situates at the immediate left to the current block.

Calculating the statistical average of MVs in the ROS is a common practice to obtain the predicted MV. The mean and the median prediction have been tested in our experiments. Others (such as the weighted average) are either too complex in computation or involving undetermined parameters, they are therefore not considered in our work.

Extensive experiments are performed with all four types of ROS and two types of prediction criteria—mean and median. Our experimental results show that these ROSs and prediction criteria yield fairly similar performance in terms of PSNR (with difference within 0.1 dB) and the total number of checking points required (with difference less than 5%). Therefore, we adopt the simplest ROS (i.e., type D) in our method, which has the least memory requirement, because only one neighboring MV needs to be recorded.

C. Selection of Search Patterns

1) Adaptive Pattern—For the Initial Search: The shape of our rood pattern is symmetrical, with four search points locating at the four vertices, as depicted in Fig. 3. The main structure of ARP has a rood shape, and its *size* refers to the distance between any vertex point and the center point. (Since symmetrical, four vertex points have equal distance to the center point.) As a special case, when the *size* of ARP is zero, the ARP will be shrunk from its normal rood shape to the center point itself. The choice of the rood shape is first based on the observation of the motion feature of real-world video sequences. It has been noticed that the MV distribution in horizontal and vertical directions are higher than that in other directions [23], since most of camera movements are in these directions. As the rood shape pattern includes all the horizontal and vertical directions, it can quickly detect such motion, and the search will directly "jump" into the local region of the global minimum. Secondly, any MV

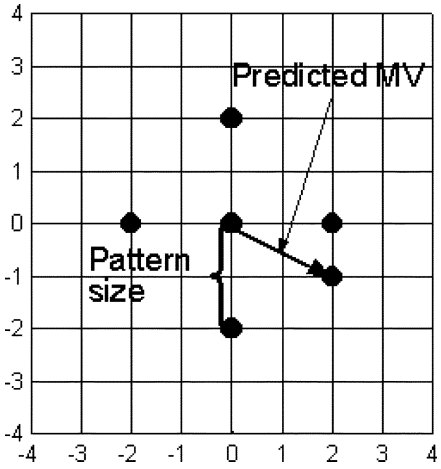

Fig. 3. Adaptive rood pattern (ARP).

can be decomposed into one vertical MV component and one horizontal MV component. For a moving object which may introduce MV in any direction, rood-shaped pattern can at least detect the major trend of the moving object, which is the desired outcome in the initial search stage. Furthermore, ARP's symmetry in shape not only benefits hardware implementation, but also increases robustness. As described in the previous section, although our ROS provides only one reference MV, the resulting performance is as good as those with larger ROS that covers more neighboring blocks. It shows that even if the predicted MV could be inaccurate and its magnitude does not match the true motion very well, the rood-shaped pattern which takes all horizontal and vertical directions into consideration can still track the major direction and favor the follow-up refinement process.

In addition to the four-armed rood pattern, it is desirable to add the predicted MV into our ARP because it is very likely to be similar to our target MV (see Fig. 3). By doing so, the probability of detecting the accurate motion in the initial stage will be increased. Note that when the predicted MV is in the horizontal or vertical direction, it has an overlap over one of the four arms of the rood pattern.

In our method, the four arms of the rood pattern are of equal length. The initial idea in deciding the ARP's size is to make it equal to the length of the predicted MV (i.e., the MV of the immediate left block of the current block). That is, the size of ARP, Γ, is

$$\begin{aligned}\Gamma &= Round\ |\overrightarrow{MV}_{predicted}| \\ &= Round\ \left[\sqrt{MV^2_{predicted}(x) + MV^2_{predicted}(y)}\right] \quad (1)\end{aligned}$$

where $MV_{predicted}(x)$ and $MV_{predicted}(y)$ are the horizontal and vertical components of the predicted MV, respectively. Operator "*Round*" performs rounding operation, which takes the nearest integer value of the argument. Note that parameter Γ defined in (1) involves square and square-root operations; thus, increasing difficulty on hardware implementation. Instead, we use only one of the two components of the predicted MV that has the larger absolute value (or magnitude) to determine the size of our ARP. That is,

$$\Gamma = \mathrm{Max}\{|MV_{predicted}(x)|, |MV_{predicted}(y)|\}. \quad (2)$$

From the mathematical standpoint, the magnitude of MV's component with larger absolute value is fairly close to the length of MV, and thus Equation (2) is a good approximation of measurement about motion magnitude. Experimental results show that the second definition of Γ using (2) is, in fact, slightly superior to the first one using (1) in terms of higher PSNR and less total number of checking points. Hence, we adopt the second method [Equation (2)] for the rest of ARPS development.

Notice that the chosen ROS (type D) is not applicable to all the leftmost blocks in each frame. For those blocks, we do not utilize any neighboring MVs, but adopt a fixed-size arm length of 2 pixels (i.e., $\Gamma = 2$) for the ARP, since this size agrees to that of LDSP which has fairly robust performance as reported in [5]. Also, longer arm lengths are not considered because the boundary MBs in a frame usually belong to static background and are less likely to have very large motion.

In summary, our adaptive pattern consists a rood-shaped pattern (having four vertex points), plus the search point indicated by the predicted MV, as shown in Fig. 3. It is possible that the predicted MV perfectly aligns with one of the four vertices. Hence, our ARP contains either 5 (no overlapping) or 4 (with overlapping) search points required to be searched in the initial search stage when the predicted MV is not zero; otherwise, requiring only one search point.

2) Fixed Pattern—For Refined Local Search: In the initial search using ARP as described earlier, the adaptive rood pattern leads the new search center directly to the most promising area which is around the global minimum; thus, effectively reducing unnecessary intermediate searches along the search path. The assumption of uni-modal error surface formed in this area would be quite valid. Hence, instead of performing FS or other fast BMAs as other methods reported in the literature, we can use a fixed, compact and small search pattern to perform local refined search unrestrictedly for identifying the global minimum. When a fixed pattern is used, the MME point found in the current step will be re-positioned as the new search center of the next search iteration until the MME point is incurred at the center of the fixed pattern. Two types of most compact search patterns have been investigated in our experiments. One is the five-point unit-size (or the smallest) rood pattern (URP) [Fig. 4(a)] which is the same as SDSP used in DS [5], and the other is a 3×3 square pattern [Fig. 4(b)] which is used in [7] and [20]. Our experimental results show that the 3×3 square pattern yields similar PSNR but requires 40%$\sim$80% more checking points when compared with the URP. This demonstrates the efficiency of using URP in local motion detection, and it is therefore adopted in our proposed method.

D. Zero-Motion Prejudgment (ZMP)

In many visual communication applications such as video telephony, there is little motion between the adjacent frames. Hence, a large percentage of zero-motion blocks are encountered in such type of video sequences. Results of some typical test video sequences are documented in Table I. Note that the total number of static blocks per frame could be easily as high as more than 70% except in Foreman and Coastguard. Thus, significant additional reduction in computational cost is possible if we perform a *zero-motion prejudgment* (ZMP) at the

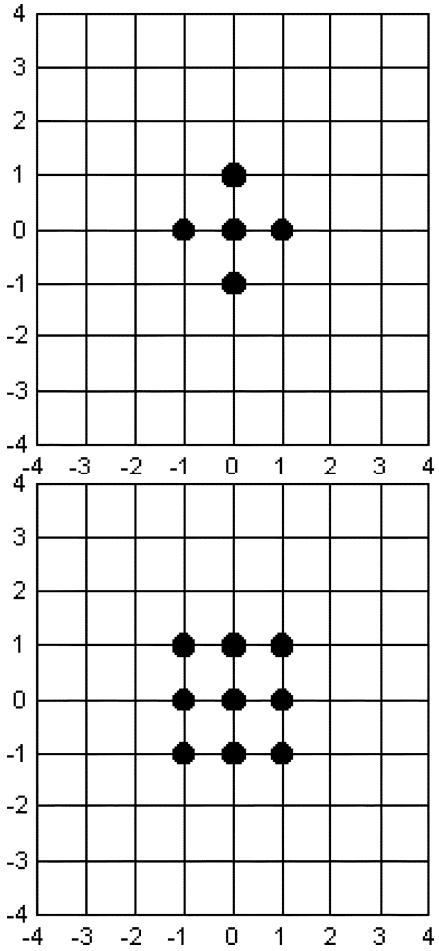

Fig. 4. Two fixed search patterns under consideration.

TABLE I
AVERAGE PROPORTION OF STATIC BLOCKS PER FRAME (THE DATA IS OBTAINED BY PERFORMING FS UNDER THE TEST CONDITIONS LISTED IN TABLE II)

Video and Coding Bit-rate (Kbps)	Average Static Blocks Percentage per Frame
Akiyo(10)	93.36%
Container(10)	90.55%
Hallmonitor(10)	95.18%
Mother&Daughter(24)	80.36%
Silence(24)	78.79%
News(112)	84.61%
Tennis(1024)	70.44%
Coastguard(112)	17.29%
Foreman(1024)	37.41%

beginning of ME. Further investigations indicate that the average matching errors of these static blocks are much smaller than that of moving blocks. So, the prejudgment can be made by first computing the matching errors between the current block and the block at the same location in the reference frame (i.e., the candidate block corresponds to zero-MV) and comparing it with a predetermined threshold T. If the matching error is smaller than T, the current block will be decided to be a static block without performing the remaining search.

Commonly, macroblock (MB) with the size of 16×16 is employed to perform ME, and the sum of absolute difference (SAD) is used as the measurement of matching error. By applying FS on 9 typical video sequences at various coding bit-rates (from 10 to 1024 Kbps, 12 cases in total), we found that the average SAD of the static MBs is within the range of 600$\sim$1300. Since higher threshold will yield larger prediction errors, we conservatively choose $T = 512$, which achieves fairly good speed-up gain without causing noticeable degradation on visual quality.

For those video sequences containing large motion contents (e.g., Foreman and Coastguard), they cannot benefit much from ZMP technique, as the percentages of static blocks in these sequences are usually quite small.

Note that the threshold is certainly adjustable, depending on the application's requirements. For example, if the video quality is not so demanding, this threshold can be increased to a larger value for yielding more speed-up.

E. Summary of Our ARPS–ZMP Method

The left adjacent block of the current block (i.e., type D in Fig. 2) is adopted as our ROS for ARPS–ZMP. For the leftmost boundary blocks in each frame, their ARP's size is set to 2 pixels (i.e., $\Gamma = 2$) as explained previously. The threshold suggested here is set to be $T = 512$ as explained earlier. For each MB, our ARPS–ZMP performs the following steps:

Step 1: Compute the matching error (SAD_{center}) between the current block and the block at the same location in the reference frame (i.e., the center of the current search window).

```
if SAD_center < T
  MV_target = [0 0],
  Stop;
else
  if the current block is a leftmost
  boundary block,
    Γ = 2;
  else
    Γ = Max{|MV_predicted(x)|, |MV_predicted(y)|}.
  Go to Step 2.
```

Step 2: Align the center of ARP with the center point of the search window and check its four search points plus the position of the predicted MV to find out the current MME point.

Step 3: Set the center point of the unit-size rood pattern (URP) at the MME point found in the previous step and check its points. If the new MME point is not incurred at the center of the current URP, repeat this step; otherwise, the MV is found, corresponding to the MME point identified in this step.

Note that in our implementation, a checking bit-map (one bit for denoting the status of each macroblock) has been employed to record whether a search point under checking has already been examined before, so that duplicated checking computation can be avoided.

TABLE II
SOME MPEG-4 TEST CONDITIONS

Video	Format	Bit Rate (Kbps)	Frame Rate (fps)	Search Range (pixels)
Akiyo	QCIF	10	7.5	16
Container	QCIF	10	7.5	16
Hallmonitor	QCIF	10	7.5	16
Mother&Daughter	QCIF	24	10	16
Silence	QCIF	24	10	16
News	CIF	48	7.5	16
News	CIF	112	15	16
Tennis	SIF	1024	30	16
Coastguard	QCIF	48	10	16
Coastguard	CIF	112	10	16
Foreman	CIF	512	15	32
Foreman	CIF	1024	30	32

TABLE III
AVERAGE PSNR (dB) PERFORMANCE OF FS, DS ARPS AND ARPS–ZMP

Video (Kbps)	FS	DS	ARPS	ARPS-ZMP
Akiyo(10)	33.81	33.82	33.77	33.77
Container(10)	29.72	29.84	29.69	29.65
Hallmonitor(10)	30.31	30.28	30.30	30.34
Mother&Daughter(24)	34.82	34.77	34.71	34.71
Silence(24)	30.79	30.79	30.86	30.86
News(48)	31.88	31.88	31.79	31.81
News(112)	33.99	33.99	33.92	33.95
Tennis(1024)	35.95	35.88	35.84	35.83
Coastguard(48)	28.88	28.72	28.77	28.77
Coastguard(112)	27.05	26.44	26.82	26.83
Foreman(512)	35.01	34.26	34.52	34.50
Foreman(1024)	35.70	35.13	35.40	35.37

TABLE IV
AVERAGE NUMBER OF SEARCH POINTS PER MV GENERATION

Video (Kbps)	FS	DS	ARPS	ARPS-ZMP
Akiyo(10)	1024	13.20	5.34	3.86
Container(10)	1024	13.24	5.26	4.54
Hallmonitor(10)	1024	13.23	5.37	4.22
Mother&Daughter(24)	1024	13.84	6.12	4.82
Silence(24)	1024	14.88	7.15	6.99
News(48)	1024	15.06	7.03	5.59
News(112)	1024	14.10	6.18	4.26
Tennis(1024)	1024	14.95	6.86	5.16
Coastguard(48)	1024	17.49	8.78	8.77
Coastguard(112)	1024	20.76	10.85	10.84
Foreman(512)	4096	22.58	11.79	11.08
Foreman(1024)	4096	18.54	9.16	8.22

TABLE V
COMPUTATIONAL GAIN TO FS AND TO DS (IN PARENTHESIS)

Video (Kbps)	DS to FS	ARPS to FS (to DS)	ARPS-ZMP to FS (to DS)
Akiyo(10)	77.59	191.84 (2.47)	265.17 (3.42)
Container(10)	77.35	194.70 (2.52)	225.65 (2.92)
Hallmonitor(10)	77.37	190.58 (2.46)	242.66 (3.14)
Mother&Daughter(24)	73.98	167.30 (2.26)	212.36 (2.87)
Silence(24)	68.82	143.12 (2.08)	146.51 (2.13)
News(48)	68.00	145.60 (2.14)	183.23 (2.69)
News(112)	72.64	165.60 (2.28)	240.32 (3.31)
Tennis(1024)	68.49	149.36 (2.18)	198.39 (2.90)
Coastguard(48)	58.55	116.66 (1.99)	116.80 (1.99)
Coastguard(112)	49.32	94.38 (1.91)	94.43 (1.91)
Foreman(512)	181.43	347.47 (1.92)	369.74 (2.04)
Foreman(1024)	217.38	447.18 (2.06)	498.01 (2.29)

III. EXPERIMENTAL RESULTS

We perform our simulations based on the encoding platform, MoMuSys FCD version 2.0.2, under MPEG-4 test conditions as shown in Table II, where each sequence has 300 frames. These sequences cover a wide range of motion contents and have various formats including QCIF, CIF, and SIF. The original frame-rate is 30 frames per second (or fps). They have been tested at various bit rates (10∼1024 kilo bits per second or Kbps) and sub-sampled frame-rates (7.5∼30 fps). When the coding bit-rate is lower than 512 Kbps, only the first frame in each sequence is coded as I frame, all the remaining frames are coded as P frames. At high bit-rates (512 Kbps and 1024 Kbps), the sequence is coded using IPP…IPP… structure, and the distance between two I frames is set to be 15 frames. Search range p means that the search will be performed within a square region of $[-p, +p]$ around the position of the current block.

For comparison, the performances of FS, DS, ARPS (i.e., without ZMP) and ARPS–ZMP (i.e., with ZMP incorporated) are reported as follows. Average *peak signal-to-noise ratio* (PSNR) per frame of each reconstructed video sequence is computed for quality comparison and documented in Table III. The search speed is measured by the average number of checking points per MV generation as shown in Table IV. The computational gain of our method to FS (or to DS) is defined by the ratio of search speed of FS (or DS) to that of our method, which is shown in Table V.

Compared with FS, ARPS greatly improves the search speed with computational gain in the range of 94∼447. Meanwhile, ARPS maintains similar PSNR performance of FS in most sequences with less than 0.12 dB degradation (except 0.23 dB in Coastguard at 112 Kbps and 0.49 dB in Foreman at 512 Kbps). When compared with DS, ARPS is constantly around 2 times faster with similar PSNR achieved. Even for difficult test sequences such as Foreman and Coastguard where large and/or complex motion contents are involved, ARPS still achieves superior PSNR to that of DS, by 0.27 dB and 0.38 dB, respectively. To gain further insights, we plot the frame-by-frame PSNR of FS, DS, and ARPS for Foreman and Coastguard sequences in Figs. 5 and 6, respectively.

The improvement resulted from the adaptability of our search pattern and, more importantly, avoiding local minimum matching error points by tracking the major trend of the motion at the initial stage. Since complex motions and unevenness of the objects cause large number of local minimums on the matching error surface, checking points in all directions (as being done by LDSP of DS) at the initial step increases the risk of being trapped into the local minima and thus degrades the search accuracy.

By incorporating ZMP into ARPS, ARPS–ZMP achieves higher computational gain especially in small motion video sequences, while incurring very small degradation in PSNR.

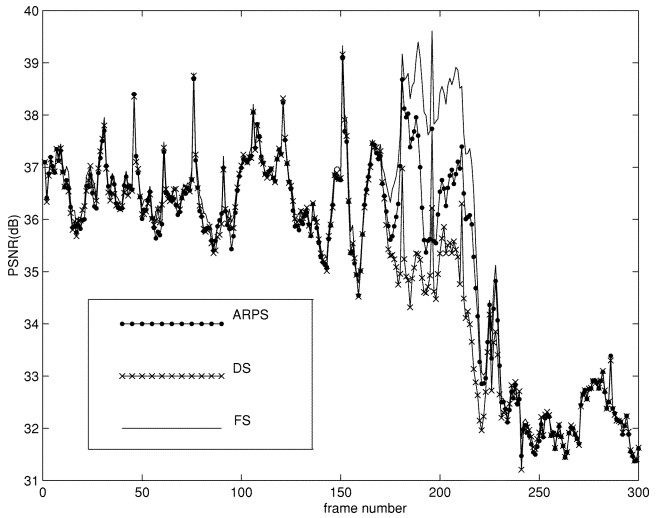

Fig. 5. Frame-based PSNR performance of FS, DS and ARPS in Foreman. Fast camera panning with scene change happens during frames 160∼220.

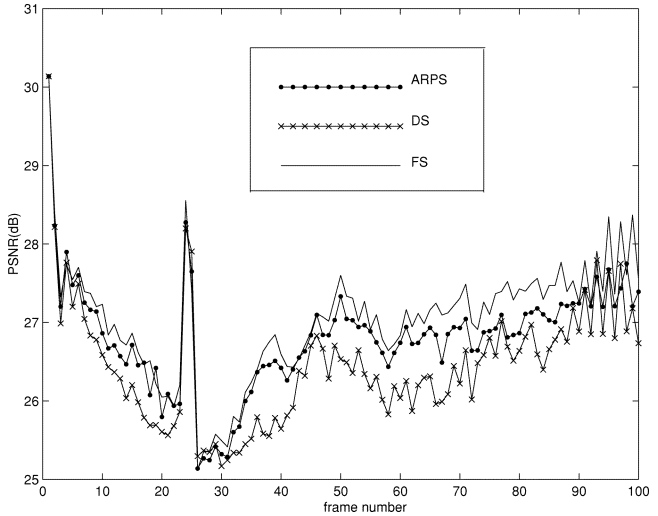

Fig. 6. Frame-based PSNR performance of FS, DS and ARPS in Coastguard. The whole sequence contains large amount of motion content.

IV. Conclusions

In this paper, we have proposed a novel and simple fast block-matching algorithm, called *adaptive rood pattern search* (ARPS). By exploiting higher distribution of MVs in the horizontal and vertical directions and the spatial inter-block correlation, ARP adaptively exploits adjustable rood-shaped search pattern (which is powerful in tracking motion trend), together with the search point indicated by the predicted MV, to match different motion contents of video sequence for each macroblock. In addition, an optional *zero-motion prejudgment* (ZMP) is incorporated into ARPS to further benefit small motion video sequence. When compared with DS, ARPS–ZMP significantly increases the computational gain by the factors ranging from 1.9 to 3.4 with little reduction in average PSNR. In addition, ARPS–ZMP improves average PSNR performance in large motion video sequences (e.g., 0.24 dB higher in Foreman and 0.39 dB higher in Coastguard). Meanwhile, ARPS's simplicity and regularity are very desirable and attractive for hardware implementation. Therefore, ARPS is a very efficient and robust ME algorithm for real-time video coding applications.

Acknowledgment

A patent for this work has been filed and is pending.

References

[1] J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Trans. Commun.*, vol. COM-29, pp. 1799–1808, Dec. 1981.

[2] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," in *Proc. Nat. Telecommunication Conf.*, Nov. 29–Dec. 3, 1981, pp. G5.3.1–G5.3.5.

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

[4] L. K. Liu and E. Feig, "A block-based gradient descent search algorithm for block motion estimation in video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 419–423, Aug. 1996.

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

[6] *Coding of moving pictures and audio*, ISO/IEC JTC1/SC29/WG11 N2932, Oct. 1999.

[7] L.-J. Luo, C. Zou, and X.-Q. Gao, "A new prediction search algorithm for block motion estimation in video coding," *IEEE Trans. Consumer Electron.*, vol. 43, pp. 56–61, Feb. 1997.

[8] J.-B. Xu, L.-M. Po, and C.-K. Cheng, "Adaptive motion tracking block matching algorithms for video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 97, pp. 1025–1029, Oct. 1999.

[9] C.-H. Hsieh, P. C. Lu, J.-S. Shyn, and E.-H. Lu, "Motion estimation algorithm using interblock correlation," *Electron. Lett.*, vol. 26, no. 5, pp. 276–277, Mar. 1, 1990.

[10] J. Chana and P. Agathoklis, "Adaptive motion estimating for efficient video compression," in *Conf. Rec. 29th Asilomar Conf. Signals, Systems and Computers*, vol. 1, 1996, pp. 690–693.

[11] J.-C. Tsai, C.-H. Hsieh, S.-K. Weng, and M.-F. Lai, "Block-matching motion estimating using correlation search algorithm," *Signal Process.: Image Commun.*, vol. 13, pp. 119–133, 1998.

[12] D.-W. Kim, J.-S. Choi, and J.-T. Kim, "Adaptive motion estimation based on spatio–temporal correlation," *Signal Process.: Image Commun.*, vol. 13, pp. 161–170, 1998.

[13] S. Zafar, Y.-Q. Zhang, and J. S. Baras, "Predictive block-matching estimation for TV coding—Part I: Inter-block prediction," *IEEE Trans. Broadcast.*, vol. 37, pp. 97–101, Sept. 1991.

[14] Y.-Q. Zhang and S. Zafar, "Predictive block-matching motion estimation for TV coding—Part II: Inter-frame prediction," *IEEE Trans. Broadcast.*, vol. 37, pp. 102–104, Sept. 1991.

[15] M. Bierling, "Displacement estimation by hierarchical block matching," *Proc. SPIE*, vol. 1001, pp. 942–951, 1988.

[16] F. Dufaux and M. Kunt, "Multigrid block matching motion estimation with an adaptive local mesh refinement," *Proc. SPIE*, vol. 1818, pp. 97–109, 1992.

[17] A. Murat Tekalp, *Digital Video Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1995.

[18] K. M. Uz, M. Vetterli, and D. LeGall, "Interpolative multiresolution coding of advanced television with compatible subchannels," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 1, pp. 86–99, Mar. 1991.

[19] Y.-Q. Zhang and S. Zafar, "Motion-compensated wavelet transform coding for color video compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 2, pp. 285–296, Sept. 1992.

[20] J. Chalidabhongse and C.-C. Jay Kuo, "Fast motion vector estimation using multiresolution-spatio–temporal correlations," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 477–488, 1997.

[21] B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, no. 2, pp. 148–157, 1993.

[22] Y. Yue, Z. Jian, W. Yiliang, L. Fengting, and G. Chenghui, "A fast effective block motion estimation algorithm," in *Proc. 4th Int. Conf. Signal Processing Proceedings (ICSP)*, vol. 1, 1998, pp. 827–830.

[23] S. Zhu and K.-K. Ma, "A new star search algorithm for fast block-matching motion estimation," in *Proc. Workshop on Very Low Bitrate Coding (VLBV)*, Oct. 8–9, 1998, pp. 173–176.

Yao Nie (S'99) received the B.S. degree from Peking University, Beijing, China, and the M.Eng. degree in electrical engineering from Nanyang Technological University, Singapore, in 1998 and 2000, respectively. She is currently pursuing the Ph.D. degree at the University of Delaware, Newark.

She worked on content-based video coding, motion estimation, and bit-rate control for her masters degree. Her current research interests include nonlinear signal processing, statistical signal processing, and image processing.

Kai-Kuang Ma (S'80–M'84–SM'95) received the Ph.D. degree from North Carolina State University, Raleigh, and the M.S. degree from Duke University, Durham, NC, both in electrical engineering, and the B.E. degree from Chung Yuan Christian University, Chung-Li, Taiwan, R.O.C., in electronic engineering.

He is presently an Associate Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. Prior to this, he was with Institute of Microelectronics (IME), National University of Singapore (1992–1995), IBM Corporation at Kingston, NY, and then Research Triangle Park, NC (1984–1992). His research interests are in the areas of multimedia signal processing and communications, including digital image/video coding, joint source and channel coding, content-based image/video indexing and retrieval, video-object segmentation, wavelets and filter banks, nonlinear image processing for denoising, error concealment, and artifact removal. From 1997 to 2001, he served as the Chairman and Head of Delegation for Singapore in MPEG and JPEG. On the MPEG contributions, the proposed fast motion estimation algorithms from his research team have been adopted by the MPEG-4 standards. He was serving as the General and Organizing Chair of ISO/IEC JTC1/SC29 Plenary Meetings and a series of working group meetings in March 2001.

Dr. Ma is serving as Editor of the IEEE TRANSACTIONS ON COMMUNICATIONS and Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA. He is a committee member of the IEEE Communications Society on Multimedia Communications Technical Committee. He is a Technical Program Co-Chair, *IEEE International Conference on Image Processing* (ICIP) 2004. He is also serving as the Chairman of IEEE Signal Processing Singapore Chapter. He is a member of Sigma Xi and Eta Kappa Nu.