USC–SIPI REPORT #289

Fast Motion Vector Estimation Using Multiresolution-Spatio-Temporal Correlations

by

Junavit Chalidabhongse and C.-C. Jay Kuo

October 1995

Signal and Image Processing Institute
UNIVERSITY OF SOUTHERN CALIFORNIA
Department of Electrical Engineering-Systems
3740 McClintock Avenue, Room 400
Los Angeles, CA 90089-2564 U.S.A.

Fast Motion Vector Estimation Using Multiresolution-Spatio-Temporal Correlations*

Junavit Chalidabhongse† and C.-C. Jay Kuo †

October 10, 1995

Abstract

In this paper, we propose a new fast algorithm for block motion vector (MV) estimation based on the correlations of the MVs existing in spatially and temporally adjacent as well as hierarchically related blocks. We first establish a basic framework by introducing new algorithms based on spatial correlation, and then spatio-temporal correlations before integrating them with multiresolution scheme for the ultimate algorithm. The main idea is to effectively exploit the information obtained from the corresponding block at a coarser resolution level and spatio-temporal neighboring blocks at the same level in order to select a good set of initial MV candidates, and then perform further local search to refine the MV result. We show with experimental results that, in comparison with the full search algorithm, the proposed algorithm achieves a speed-up factor ranging from 150 to 310 with only 2-7% MSE increase and a similar rate-distortion performance when applied to typical test video sequences.

1 Introduction

Video image compression plays an important role in transmission and storage of digital video data. The applications include multimedia transmission, teleconferencing, videophone, high-definition television (HDTV), CD-ROM storages, etc. The main idea to achieve compression is to remove temporal and spatial redundancies existing in video sequences. One effective method commonly used in reducing temporal redundancy is motion compensated predictive coding, which is also employed in the MPEG standard [3], [10], [11]. The key ingredient in motion compensated coding is motion vector (MV) estimation. The block matching technique has been widely used for MV estimation due to its simplicity. A straightforward way to obtain MV is to perform the full search block matching algorithm (FBMA) by searching all locations in a given search area and selecting the position where the matching residual error is minimized. However, this procedure requires an

*This work was supported by the National Science Foundation Presidential Faculty Fellow (PFF) Award ASC-9350309.

†The authors are with the Signal and Image Processing Institute and the Department of Electrical Engineering-Systems, University of Southern California, Los Angeles, California 90089-2564. E-mail: jchalida@sipi.usc.edu and cckuo@sipi.usc.edu.

extremely large amount of computation. MV estimation is known to be the main bottleneck in real-time encoding applications, and the search for an effective MV estimation algorithm has been a challenging problem for years.

Fast block matching algorithms have been developed [12] to reduce the computational cost. They can be categorized into different groups as detailed below.

1.1 Fast Block Matching With Unimodal Error Surface Assumption

Most fast block matching algorithms [4], [5], [6], [13], [14], [15] restrict the number of search locations using the unimodal error surface assumption, namely, the matching error increases monotonically as the search moves away from the position of the global minimum error. However, this assumption usually does not hold and, as a result, the search could be trapped to a local minimum with a relatively large matching error. Moreover, these algorithms treat each block independently and tend to result in a noisy motion field and create the blocking effect in reconstructed images. Some well-known algorithms in the class include the three-step search (TSS) [6] and the two-dimensional logarithmic search (TDL) [5].

1.2 Fast Block Matching with Pixel Subsampling

Another interesting technique to reduce the complexity of MV estimation is block matching with pixel subsampling proposed by Liu and Zaccarin [9]. Instead of limiting the number of search locations, the number of pixels used in matching error computation is reduced. The technique is called alternating 4:1 pixel subsampling in [9], since there are four possible 4:1 pixel subsampling patterns to be used alternatively. It was shown that using all four patterns in a specific alternating manner gives a better result than using only one 4:1 subsampling pattern. This technique reduces the number of matching operations by a factor of 4. Furthermore, two other techniques were also presented in [9] to enhance the performance. One is called subsampled motion-field estimation which exploits the idea of block subsampling, and the other is called sub-block motion-field estimation where a smaller block size is used. The first one reduces the number of operations by a factor of 2 while the second one has a reduction factor of 4. Finally, two fast algorithms based on the combination of the first (alternating 4:1 pixel subsampling) and the second (subsampled motion-field) techniques and the combination of the first and the third (sub-block motion-field) techniques were proposed. They reduce the computational complexity by factors of 8 and 16, respectively. The above three techniques provide a set of good tools which can be incorporated in several existing algorithms to obtain an additional amount of computational reduction.

1.3 Fast Block Matching with Spatial/Temporal Correlations

Another direction for fast MV estimation approach is to exploit information from adjacent blocks by using spatial and temporal correlations of MVs [18], [20]. The main idea is to select a set of initial MV candidates from spatially and/or temporally neighboring blocks and choose the best one (according to a certain rule) as the initial estimate for further refinement. Theoretically, the initial estimate can be obtained by using an autoregressive (AR) model [18],[20]. For such a simple model, only one candidate is chosen and used as the initial estimate in experiments. The refinement process involves a full search with a reduced search area. It turns out that the full search procedure still requires a considerable amount of computation despite being performed on the reduced search area.

A hybrid algorithm which use both block-recursive and block-matching methods was proposed in [17]. Although its original motivation did not aim at the use of spatial and temporal correlations, it did provide an interesting way to use both correlations effectively. In the algorithm, the MV candidates were selected from two spatially and one temporally neighboring blocks. In the refinement process, a block recursive idea was explored to compute the gradient direction for MV update. However, the gradient approach does not work well for real-world fast motion image sequences since an oscillation in the search direction may occur in refinement. It is therefore limited to applications with relatively slow motion, e.g. videoconferencing.

1.4 Hierarchical and Multiresolution Fast Block Matching

One family of fast block motion estimation algorithms relies on the idea of predicting an approximate large-scale MV in a coarse-resolution video and refining the predicted MV in a multiresolution fashion to obtain the MV in the finer resolution. They are called the hierarchical [1], [2] or the multiresolution methods [7], [16], [19], [21]. The hierarchical methods [1], [2] use the same image size but different block sizes at each level. The underlying assumption is that the MV obtained from a larger block size provides a good initial estimate for MVs associated with smaller blocks which are contained by the larger block. This assumption is often not true and the estimate can be very poor. Furthermore, a larger block size implies a higher computational cost in performing block matching. The multiresolution methods [7], [16], [19], [21] use different image resolutions with a smaller image size at a coarser level (i.e. of a pyramid form). They can be further divided into two groups: constant block size and variable block size.

In [7], [16], the same block size is used at each level. Thus, a block at the coarser level represents a larger region than that at the finer level so that a smaller search area can be used at coarser levels. If the image size reduced by half as the level becomes coarser, one block at a coarser level covers four corresponding blocks at the next finer level. Then, the MV of the coarser-level block is either directly used as the initial estimate for the four corresponding finer-level blocks [7] or interpolated to obtain four MVs of the finer level [16]. In [19], [21], different block sizes are employed at each level to maintain a one-to-one correspondence between blocks in different levels. Then, the MV of each block is directly used as initial estimate for the corresponding block at the finer level. Methods in this category work relatively well and provide fast computation. However, they only use the information from coarser levels for the MV refinement in finer levels without considering other useful information such as spatial and temporal correlations among MVs at the same level. Furthermore, the refinement process is performed by using a full search with a reduced search area which nevertheless requires a considerable amount of computation.

1.5 Overview of Our Work

Even though many fast MV estimation techniques have been proposed as reviewed before, we feel that the spatial and temporal correlations of MVs have not yet been fully exploited in reducing the search time while maintaining a reasonable rate-distortion trade-off. The use of an AR model to characterize spatio-temporal correlations of the motion field could provide an elegant theoretical result. However, its derivation requires a certain amount of computational complexity and its practical value decreases. Our goal is to develop a sequence of fast MV estimation algorithms which exploit the spatio-temporal correlations of MVs in a computationally simple way and yet works effectively in the sense of producing small residual errors. Furthermore, we incorporate them in a multiresolution framework to improve the overall performance.

This paper is organized as follows. We first propose two new algorithms using only the spatial correlation. They are called S1 and S2 (where S denotes *spatial*) and introduced in Sections 2 and 3, respectively. Algorithm S1 provides the basic framework while algorithm S2 is a modified version. To achieve a better performance, we incorporate the information from the temporal domain and propose two fast algorithms based on spatio-temporal correlations. They are called ST1 and ST2 (where ST stands for *spatial* and *temporal*) and described in Sections 4 and 5, respectively. Then, we integrate the spatio-temporal technique with the multiresolution scheme to obtain the ultimate algorithm called MRST (where MR denotes *multiresolution*) in Section 6. The MVs obtained from all proposed algorithms are compatible with the MPEG standard. The performance of all algorithm is demonstrated via extensive experiments in Section 7. Concluding remarks are given in Section 8.

2 Fast Algorithm Based on Spatial Correlation: S1

The following framework is adopted in our discussion. Each image frame is divided into nonoverlapping square blocks of 16×16 pixels as specified by MPEG. We use $B(i, j, k)$ to represent a block of the kth frame, where i and j are block indices along the row and column directions, respectively. For example, an image of size 352×240 has block indices $i = 0, 1, \ldots, 14$ and $j = 0, 1, \ldots, 21$, and $B(0, j, k)$ and $B(i, 0, k)$ represent the blocks in the first row and first column, respectively. We would like to determine the MV for each block between two consecutive frames with a certain fast MV estimation algorithm. Without loss of generality, when we talk about the MV of block $B(i, j, k)$, it is calculated based on forward prediction, i.e. the MV between frames $k - 1$ and k. However, the same idea can be applied to backward prediction and generalized to bidirectional prediction.

It has been observed that the MV of a certain block is the same or very close to the MVs of its spatially adjacent blocks. To illustrate the spatial correlation of MVs, we compute the MV spatial differentials of consecutive blocks along the horizontal and vertical directions via full search, and plot the histograms of x- and y-components of these differentials in Fig. 1. The high peaks at the zero differential value indicate that the MV field is highly correlated along both horizontal and vertical directions. Even though the specific data are calculated based on the 60th frame of the "football" sequence, the observation is typical for any frame in an image sequence. Such an observation suggests that the MV of a given block can be predicted from its spatially neighboring blocks. This is the main idea behind our algorithm.

Based on this spatial correlation property of MVs, we propose a fast MV estimation algorithm called S1 in this section. It consists of two major building elements: (1) the MV candidate selection and (2) the MV refinement process. Before a detailed discussion on each component, we would like to define the following terms to make the discussion clear.

- *The initial MV candidates* represent a set of MV candidates selected from spatial (and temporal as well as hierarchical in later sections) neighboring blocks with a certain selection rule.
- *The best MV candidate* represents the one MV chosen from the set of initial MV candidates to serve as the starting point for the MV refinement process.
- *The final MV* represents the final MV result obtained.

2.1 MV candidate selection

To begin the MV estimation process, we perform the full search block matching algorithm to determine the MVs of the four blocks at the top left corner, i.e. $B(0,0,k)$, $B(0,1,k)$, $B(1,0,k)$ and $B(1,1,k)$. Since the search cannot go beyond image boundaries, MVs of boundary blocks $B(0,0,k)$, $B(0,1,k)$ and $B(1,0,k)$ are limited to some directions and a full search at block $B(1,1,k)$ will provide more accurate initial MV candidate for the following estimation task. When performing a full search, an alternating 4:1 pixel subsampling technique [9] can be used to reduce the computational cost, i.e. for a block of size 16×16, only the values at 64 pixels are used to compute the MAD (Mean of Absolute Difference). It was shown in [9] that a reasonably good MV estimate can be obtained by using such a pixel subsampling technique. This gives a computational reduction by a factor of 4 in comparison with a straightforward implementation of the MAD computation.

After the initialization step, we proceed to the next block according to a rowwise ordering, i.e. starting from the left to the right for the 1st row, then the 2nd row, and so on, and use the MVs of blocks $B(i,j-1,k)$, $B(i-1,j,k)$, $B(i-1,j-1,k)$ and $B(i-1,j+1,k)$ as the initial MV candidates for block $B(i,j,k)$. Note, however, that blocks along the boundaries have fewer initial candidates than the inner blocks. Among the four initial MV candidates, the one with the smallest MAD (denoted by MAD_0) is chosen as the best MV candidate (V_0) for block $B(i,j,k)$ and used as the starting point for further MV refinement. In other words, let $\mathcal{C} = \{C_1, \ldots, C_M\}$ be the set of initial MV candidates, and $MAD(C_i)$ be the MAD corresponding to the vector C_i. Then, the best MV candidate V_0 can be expressed as

$$V_0 = \arg\min_{C \in \mathcal{C}} MAD(C).$$

2.2 MV refinement process

The refinement process begins with the best MV candidate (V_0) and its corresponding MAD value (MAD_0). A threshold TH_1 is set so that if $MAD_0 \le TH_1$, the best MV candidate V_0 is chosen as the final MV for block $B(i,j,k)$. If $MAD_0 > TH_1$, we use the spatial correlation property and assume that the best MV candidate V_0 is close to the desired final MV. Even though the assumption that the matching error increases monotonically as the searching point moves away from the global minimum (i.e. the unimodal error surface assumption) is generally not true, it seems reasonable to assume that the matching error surface is monotonic in a small neighborhood around the global minimum. This assumption was used to find small motion for low bit rate coding applications in [8]. It implies that if the initial search point is close to the global minimum, there is a high probability to find the global minimum. In the current context, the best candidate V_0 is viewed as a new search center so that a search is performed around its neighborhood. The search starts at the new center and its eight neighboring points. If either the minimum MAD among nine of them occurs in the middle (center), or the new smaller MAD has the value $\leq TH_1$, the procedure stops. Otherwise, it keeps the same search procedure by using the position with the new minimum MAD as the new search center. The process iterates until either the stopping criterion is satisfied or the pre-set maximum number (S) of search steps is reached. We call the above search procedure *local search around*, and depict it in Fig. 2. As shown in the figure, eight surrounding locations are searched at each step.

It is observed that the 4:1 pixel subsampling technique gives an unacceptably coarse matching result in the current context, since we consider nine search points at a time and need an accurate local minimum point for the next step. It is nevertheless possible to reduce computation in the MAD calculation at each location. That is, we consider the checker-board partitioning of 16×16 pixels within each block, and use one half of the pixels beginning at top left pixel for matching. It turns out that this reduction gives sufficiently good results required by a further refinement. Furthermore, even though we have to compute the MAD for nine points at the first step, the MAD has to be computed at only three or five positions in the following steps due to the overlap of the neighborhoods of consecutive centers.

As discussed above, there are three possible conditions for the local search around procedure to stop. First, if it stops because MAD $\leq TH_1$, the position corresponding to this MAD value is chosen as the desired final MV. Second, if the procedure stops because of reaching the maximum step number of iteration, it is likely that the MV of the current block may not be close to MVs of its neighboring blocks. This phenomenon is often resulted from motion discontinuity, occluded regions or a certain type of mixed motion. Thus, we perform a full search with alternating 4:1 pixel subsampling to find the MV for this block. Third, if the search stops because the minimum MAD occurs in the middle among nine points, it seems reasonable to choose that point as the final MV. However, it is possible that the best MV candidate V_0 is too far from the desired final MV so that we actually get trapped to a wrong local minimum. To avoid such a problem, we check the MAD value at that point. If its MAD value is not larger than a threshold value TH_2 (another threshold which is greater than TH_1), then we choose the corresponding position to be the final MV. Otherwise, a full search with alternating 4:1 pixel subsampling is performed to determine the MV for the current block. This completes our refinement process. Let us summarize Algorithm S1 as follows.

Algorithm S1

1. The algorithm is initialized by performing a full search with an alternating 4:1 pixel subsampling technique on four blocks located at the top left corner, i.e., $B(0,0,k)$, $B(0,1,k)$, $B(1,0,k)$ and $B(1,1,k)$. The full search has the maximum displacement $\pm W$ along both horizontal and vertical directions.

2. We proceed the MV search for the block from the top left to the bottom right with a rowwise ordering. For block $B(i,j,k)$, we use the MVs from its four neighboring blocks $B(i,j-1,k), B(i-1,j,k)$, $B(i-1,j-1,k)$ and $B(i-1,j+1,k)$ as possible initial candidates and choose the one with the smallest MAD (MAD_0) as the best MV candidate (V_0) for the block $B(i,j,k)$.

3. If $MAD_0 \leq TH_1$, the best MV candidate V_0 is chosen as the final MV. Then, we move to the next block (returning to Step 2). Otherwise, go to Step 4.

4. If $MAD_0 > TH_1$, perform the local search around procedure until reaching one of the following three stopping criteria:

   (i) the new minimum MAD $\leq TH_1$;

   (ii) the minimum MAD among nine points at one step occurs at the middle (center) location;

   (iii) the search step number reaches the maximum limit (S).

5. If the search stops because of condition (i), the position giving the minimum MAD is chosen as the final MV. If the search stops due to condition (ii), we check the minimum MAD value in the center. If the MAD value $\leq TH_2$ (another threshold greater than TH_1), the position is accepted as the final MV. Otherwise, switch to the full search mode with alternating 4:1 pixel subsampling to determine the MV. If the search stops because of condition (iii), the full search with alternating 4:1 pixel subsampling is performed to obtain the final MV.

6. Proceed to the next block by returning to Step 2. After all the blocks in a frame are processed, go to the next frame with Step 1.

3 Modified Spatial Based Algorithm: S2

The spatial correlation of MVs was exploited to develop a fast MV estimation algorithm known as S1 in Section 2. In this section, we propose another algorithm called S2 with the objective to speed up the search computation while maintaining a similar performance in the resulting MSE (or rate-distortion). The new algorithm is motivated by the observation that, in many cases, the MVs have a long range spatial correlation. That is, the MVs associated with 2:1 horizontally or vertically subsampled blocks still have a strong spatial correlation. Thus, by using block subsampling, we classify blocks into several different types. Then, the MV of a certain type of blocks can have its initial MV candidates coming from four noncausal spatial directions rather than causal directions (i.e. upper and left) only as occurred in S1. The resulting scheme provides a faster search for these blocks since a better set of initial MV candidates is used. Consequently, we obtain an overall performance improvement.

3.1 MV candidate selection

We classify each block into one of the three types $G1$, $G2$ and $G3$ as shown in Fig. 3. We first treat the image as if it consists only of $G1$ blocks, and perform the MV estimation for blocks in the $G1$ group with Algorithm S1. That is, for block $B(i,j,k)$ in $G1$, we use the MVs from blocks $B(i,j-2,k)$, $B(i-2,j,k)$, $B(i-2,j-2,k)$ and $B(i-2,j+2,k)$ to be initial MV candidates as shown in Fig. 4 and the one with the smallest MAD is chosen as the best MV candidate for further MV refinement to obtain the final MV.

Each $G2$ block is surrounded by four $G1$ blocks at its four corner positions. We use the MVs from the four $G1$ blocks as initial MV candidates and choose the one with the smallest MAD as the best MV candidate for further refinement. There are some $G2$ blocks which lie along image boundaries and require some special attention. Following the above rule, its initial MV candidates come from one or two nonboundary blocks. However, the boundary blocks can only have a certain range of MVs so that their MVs may be quite different from those of the inner blocks. Thus, we include the MV from the nearest $G2$ block located on the same boundary which has been obtained earlier as an additional initial MV candidate for boundary $G2$ blocks. Finally, we perform the MV refinement process to obtain the final MV for $G2$ blocks.

The next step is to estimate the MVs for $G3$ blocks. As shown in Fig. 4, we now have the initial MV candidates from $G1$ and $G2$ blocks. To be more precise, for a $G3$ block $B(i,j,k)$, we use the four MVs from blocks $B(i,j-1,k), B(i,j+1,k), B(i-1,j,k)$ and $B(i+1,j,k)$ as its initial MV candidates. These MV candidates come from blocks located at the same row or the same column, and they tend to provide a better candidate set than the ones provided by four corner blocks. Blocks along boundaries can also be handled in a straightforward fashion. The best MV candidate is obtained from the one giving the smallest MAD, and used for the further refinement.

3.2 MV refinement process

The refinement process for blocks in groups $G1$ and $G2$ is performed in the same way as that in S1 (see Section 2.2) with a slight modification for $G3$ blocks. As mentioned above, $G3$ blocks tend to have a better set of initial MV candidates than $G1$ and $G2$ blocks so that the MV of each $G3$ block should be close to one of its surrounding initial MV candidates. Thus, the refinement for $G3$ blocks is performed without switching to the full search mode. In other words, after choosing the best MV candidate (V_0) with the smallest MAD among initial MV candidates, if $MAD_0 \leq TH_1$, the vector V_0 is chosen to be the final MV. Otherwise, we perform the local search around. Whatever condition stops the search, we accept the position with the last minimum MAD as the final MV. Algorithm S2 is summarized as follows.

Algorithm S2

1. Assign each block to one of three types $G1$, $G2$ and $G3$ as shown in Fig. 3
2. Treat the subsampled $G1$ blocks as neighbors, and apply Algorithm S1 to determine the MVs of all $G1$ blocks.
3. After obtaining all MVs of $G1$ blocks, determine the best MV candidate of $G2$ blocks based on the four initial MV candidates from their four corner neighboring $G1$ blocks. For $G2$ boundary blocks, add one more candidate obtained from the nearest $G2$ block along the same boundary. Then, we perform the MV refinement process as used in S1.
4. Determine the MV of each $G3$ block by using the MVs from its four nearest neighboring blocks belonging to $G1$ or $G2$ as initial MV candidates. Choose the one with the smallest MAD (MAD_0) as the best MV candidate, and if $MAD_0 > TH_1$, perform the MV refinement process without the full search mode.
5. After all the blocks in a frame are processed, proceed to the next frame beginning with Step 1.

In the next section, we will show how to incorporate the information from the temporal direction to improve the performance.

4 Fast Algorithm Based on Spatio-Temporal Correlation: ST1

In addition to the spatial correlation, MVs are also highly correlated in the temporal direction. We observe that the histogram of MV temporal differentials is similar to that of MV spatial differentials as shown in Fig. 1 with a high peak appearing at the zero differential value. The temporal correlation property provides additional information for selecting a better set of initial MV candidates and leads to an overall improvement in the search speed and the resulting MSE. In this section, we modify Algorithm S1 by including temporal information. The resulting algorithm is called Algorithm ST1.

A simple way to incorporate the temporal information in Algorithm S1 is to include the MV of the block at the same location from the previous frame (i.e. $B(i,j,k-1)$) in the set of initial MV candidates. Furthermore, we can obtain information from noncausal directions by including two more initial MV candidates obtained from the blocks in the previous frame, i.e. $B(i+1,j,k-1)$ and $B(i,j+1,k-1)$. Finally, it is natural to include MVs of $B(i-1,j,k)$ and $B(i,j-1,k)$. Thus, for block $B(i,j,k)$, there are five initial MV candidates from blocks

$$B(i,j-1,k),\ B(i-1,j,k),\ B(i,j,k-1),\ B(i,j+1,k-1),\ B(i+1,j,k-1).$$

Recall that we initialize the MV estimation process in S1 by performing a full search for the four blocks at the top left corner by exploiting the spatial correlation only. By including temporal information, we can use temporal information to select initial MV candidates for these four blocks without a full search in initialization. After selecting a set of initial MV candidates, we choose the one giving the smallest MAD value (MAD_0) as the best MV candidate (V_0) and perform further refinement to improve the MV result. The refinement process is performed in exactly the same way as stated in Section 2.2. Algorithm ST1 is summarized as follows.

Algorithm ST1

1. We obtain the MV for the block from the top left to the bottom right with a rowwise ordering. For block $B(i,j,k)$, we select its initial MV candidates from $B(i,j-1,k), B(i-1,j,k), B(i,j,k-1), B(i,j+1,k-1)$ and $B(i+1,j,k-1)$ except for the first predicted frame where the temporal information is not available and Algorithm S1 can be performed to find the MVs. Among the initial MV candidates, we choose the one with the smallest MAD (MAD_0) as the best MV candidate (V_0) for the block $B(i,j,k)$.

2. If $MAD_0 \leq TH_1$, the best MV candidate V_0 is chosen to be the final MV. Then, we proceed to the next block by returning to Step 1. Otherwise, go to Step 3.

3. Perform the MV refinement process using the same procedure as stated in Algorithm S1 (Steps 4 and 5).

4. Move to the next block by returning to Step 1. After all the blocks in a frame are processed, go to the next frame.

5 Modified Spatio-Temporal Based Algorithm: ST2

With the framework of the modified algorithm S2, we propose the new spatio-temporal correlation based algorithm ST2 by incorporating the temporal correlation information to improve the speed of Algorithm ST1.

In Algorithm ST2, each block is assigned to one of three groups in the same pattern as in Algorithm S2 (Fig. 3). For $G1$ blocks in Algorithm S2, the only available initial MV candidates come from spatially subsampled blocks in causal (upper left) directions. In ST2, we include candidates from blocks in lower, right and the same locations from the previous frame. Thus, for block $B(i, j, k)$ in the $G1$ group, we select the MVs from two spatially adjacent blocks $B(i, j-2, k)$ and $B(i-2, j, k)$, and three temporally adjacent blocks $B(i, j, k-1)$, $B(i, j+1, k-1)$ and $B(i+1, j, k-1)$ to be initial MV candidates as shown in Fig. 5. The best MV candidate is obtained by choosing the one with the smallest MAD. Then, we perform further refinement.

The $G2$ blocks have more MV information available for its spatially neighboring blocks than $G1$, since each $G2$ block is surrounded by four $G1$ blocks at its four corner positions. We use the MVs from four spatial $G1$ blocks and one temporal $G2$ block on the same location from the previous frame as initial MV candidates as shown in Fig. 5.

Similar to Algorithm S2, $G3$ blocks have initial MV candidates from $G1$ and $G2$ blocks which are located in either the same row or column. Thus, for a $G3$ block $B(i, j, k)$, we use the four MVs from spatial blocks $B(i, j-1, k), B(i, j+1, k), B(i-1, j, k)$ and $B(i+1, j, k)$ and one MV from temporal block $B(i, j, k-1)$ as its initial MV candidates. The selection rule is shown in Fig. 5.

Note that each $G2$ or $G3$ block is surrounded by four spatial candidates with one temporal candidate in the middle location. Such a candidate pattern suggests a simple modification on the decision rule. That is, if all five initial candidates are the same, such a vector can be used as the final MV without any further refinement. This modification saves a certain amount of MAD computation with little sacrifice in the performance. If they are not the same, we employ the original rule where the best MV candidate is set to the candidate with the smallest MAD. Then, we perform the same local refinement as that in Algorithm S2 for each group. Algorithm ST2 is summarized as follows.

Algorithm ST2

1. Assign each block to one of three group types $G1$, $G2$ and $G3$ as shown in Fig. 3.

2. Treat the subsampled $G1$ blocks as spatial neighbors, and select the initial MV candidates from spatio-temporal neighboring blocks $B(i, j-2, k)$, $B(i-2, j, k)$, $B(i, j, k-1)$, $B(i, j+1, k-1)$ and $B(i+1, j, k-1)$ with the exception of the first predicted frame where Algorithm S2 is performed. The best MV candidate is the one giving the smallest MAD. The same refinement process as done in S2 is performed to determine the final MVs of all $G1$ blocks.

3. After obtaining all MVs of $G1$ blocks, determine the MV of each $G2$ block based on four initial MV candidates from its four corner spatially neighboring $G1$ blocks and one initial MV candidate from temporally neighboring block at the same location. If all five candidates give the same vector, it is the final MV. Otherwise, choose the one with the smallest MAD as the best candidate and perform the MV refinement as used in S2.

4. Determine the MV of each $G3$ block by using the MVs from its four nearest neighboring blocks belonging to $G1$ or $G2$ and one temporally neighboring block at the same location as initial MV candidates. The remaining procedure is the same as that in Step 3.

5. After all the blocks in a frame are processed, move to next frame beginning with Step 1.

The proposed MV selection procedure of ST2 provides a scheme to use spatial and temporal information in a complement way. For example, $G1$ blocks do not have as many good spatial candidates as the $G2$ and $G3$ blocks do, therefore, they need more information from the temporal direction. For $G2$ and $G3$ blocks, we give a higher priority to noncausal information from the spatial domain than the temporal domain as reflected from the fact that four (noncausal) spatial neighboring MVs and one temporal MV are used for $G2$ and $G3$ blocks.

We have seen so far that all proposed algorithms S1, S2, ST1 and ST2 require threshold parameters TH_1 and TH_2. These parameters are pre-set as a certain fixed numbers. However, it would be desirable if such parameters can be chosen or adjusted automatically from the algorithm itself. In the next section, we propose a new fast algorithm based on multiresolution-spatio-temporal correlations, which not only solves the threshold problem but also improves the overall performance.

6 Fast MV Estimation Using Multiresolution-Spatio-Temporal Correlations: MRST

The multiresolution approach has been recently studied and applied to motion estimation problem [7], [16], [19], [21]. It provides a relatively fast computational speed while giving a reasonably good prediction. With this approach, an image frame is decomposed into different resolutions. The multiresolution nature provides a hierarchical motion field obtained at different scales, and a smaller search area is used at a coarser level. The MV estimation is first performed on the coarsest resolution and then the MVs of finer resolutions are refined based on the motion information obtained at coarser resolutions. Most existing algorithms use only information from coarser levels to refine the MV in finer levels, and do not exploit spatio-temporal correlations of the MVs at the same level. In the section, we propose a new fast algorithm called MRST by combining the multiresolution, spatial and temporal correlation properties.

In this work, a coarser resolution image is obtained by computing the mean of 2×2 pixels from finer levels to represent a pixel in the next coarser level so that the image size is reduced by half along both horizontal and vertical directions. Note that since we focus on the MV estimation problem without worrying about the residual coding, only a simple averaged mean is used here to obtain coarser resolution images. We also employ different block sizes at different levels as presented in [19], [21], and blocks of smaller sizes are used in the coarser levels so that each level has the same number of blocks. Thus, there is a one-to-one correspondence of blocks between coarser and finer levels. The MVs at different levels are highly correlated since they represent the same motion activities at different scales.

In the proposed algorithm MRST, each image frame is decomposed into 4 resolution levels with a block of size 2×2 at the coarsest level and a block of a size 16×16 at the finest level. The level numbers are ordered from 0 to 3, where levels 0 and 3 represent the coarsest and finest levels, respectively. We begin with the coarsest level by performing a full search to obtain the MV for each block. Due to the coarse scale of the MV, the maximum search displacement at the coarsest level is reduced from $\pm W$ to $\pm W/8$ so that only a small amount of computation is required.

For each of the finer resolution level (level 1 to 3), we adopt the framework of Algorithm ST2 (except for the first predicted frame for which Algorithm S2 is applied) by using the MV information from the coarser level as well as spatially and temporally neighboring blocks at the same level. Most existing algorithms rely only on the initial information from the corresponding coarse-scaled MVs for further refinement. However, the coarse-scaled MVs for some blocks may not be accurate enough and could cause some errors which propagate along the hierarchical structure. Our algorithm exploits the information from both multiresolution and spatio-temporal adjacent blocks to select a number of initial MV candidates. Note that the MV from the coarser level has to be properly scaled to serve as the initial candidate in the finer level. For example, if a block at level l has the motion vector V, then the corresponding block at level $l+1$ uses $2V$ as its initial MV candidate. The set of initial MV candidates can be expressed as:

$$\{C_1, C_2, \ldots, C_M, C_{M+1}\}$$

where $C_1, \ldots, C_M$ are initial MV candidates obtained from the MV selection procedure specified in ST2 and C_{M+1} is the initial candidate from the corresponding coarser level.

The best MV candidate is obtained by using the same rule as that described in ST2. Similar to Algorithm ST2, we also adopt a majority checking rule to select best MV candidate for $G2$ and $G3$ blocks. According to the above selection process, each $G2$ or $G3$ block has a total of 6 candidates (4 from spatial, 1 from temporal and 1 from the coarser level). The new checking rule is that if at least five candidates are the same, this value is the final MV and no further refinement is required. This allow us to save a certain amount of computation with little sacrifice in MSE increase.

Note that when the MV information from the coarse level is used, we can only have the MV of an even number of length as the initial candidate. On the other hand, if only the spatio-temporal information is used as in ST2, there are some blocks that still require a full search (those having the MAD greater than TH_2 or reaching the maximum iteration limit). By incorporating the coarser level information, the MV obtained from the full search in the coarsest level provides a better set of candidates so that the full search mode is not needed at finer levels. The new algorithm is more robust in the sense that the second threshold TH_2 is no longer needed.

One important issue in the refinement process is the selection of threshold TH_1. It is one main factor to determine the computational speed since it indicates whether to choose the best MV candidate as the final MV (according to the criterion $MAD_0 \leq TH_1$). In Algorithms S1, S2, ST1 and ST2, the threshold TH_1 is a fixed number for every frame throughout the entire estimation process. It would be more desirable to find a rule such that TH_1 can be adjusted adaptively for different frames to provide a higher computational speed. Such a rule should be related to the knowledge of matching error (MAD) information. In Algorithm MRST, we obtain the values of minimum MADs via full search at the coarsest level. Such values should give us some rough idea about the range of smallest MADs for each frame. Therefore, we compute the averaged mean of smallest MADs over all blocks at the coarsest level for each frame (denoted as u_k where k is the frame number) and use it to determine the value of TH_1. We also observe that u_k tends to have a small increase as the resolution level goes finer for most frames. Based on the observation, we choose the threshold value TH_1 by adding 0.5 to u_k every time as the level becomes finer, i.e.

$$TH_1(l) = u_k + (0.5 \times l)$$

at level l. Although we know the smallest MAD of each block at the coarsest level, there is no direct relationship among the smallest MADs of blocks at different levels. Therefore, it is difficult to adopt any specific rule for TH_1 value of each block separately since it could give a large error to some blocks. Instead, we use the averaged mean value as described above so that the threshold TH_1 is good enough in the average sense and would not produce a significantly large error. Based on the above rule, threshold TH_1 is automatically chosen from the MAD information of the coarsest level. The new algorithm MRST is summarized as follows.

Algorithm MRST

1. Obtain the coarser resolution images by computing the averaged mean of the nonoverlapping 2×2 pixels from finer-level images for each pixel in coarser-level images until the block size at the coarsest level becomes 2×2. Let the total number of resolution levels be L. The image at resolution l is divided into blocks of size $2^{l-L+1}B \times 2^{l-L+1}B$, where B (= 16) is the block size of the finest resolution $L-1$ and $l = 0, 1, \ldots, L-1$. (L=4 in the current case).

2. At the coarsest level, perform a full search with the maximum search displacement $\pm(W/2^{L-1})$ to obtain the MV for each block.

3. For each finer level, perform Algorithm ST2 (except for the first predicted frame for which S2 is applied) by including one more candidate from the corresponding block in the previous coarser level by multiplying the coarser-scaled MV by a factor of 2. The refinement process is performed in the same way as done in ST2 except the adoption of full search in any case for any block type ($G1, G2$ or $G3$).

4. After the MVs at finest level are obtained, move to the next frame beginning with Step 1.

7 Experimental Results

Experiment results using all proposed algorithms S1, S2, ST1, ST2 and MRST are reported in this section. They are applied to five MPEG test videos: bicycle, cheer (leaders), flower, football, and mobile. Each video contains 150 frames and each frame has a size of 352×240 obtained by subsampling CCIR601 720×480 luminance component only. These sequences provide a variety of motions including still, slow and fast movements, camera panning, and zooming. A 16×16 square block is used as a macroblock for MV estimation as specified by MPEG. Only forward prediction is implemented in the experiments. The maximum horizontal and vertical search displacement is ± 16 ($W = 16$) so that the maximum search locations for one block is 33×33 ($33 = (2 \times 16) + 1$). We use MAD as the error measure in performing block matching, while the MSE per pixel is computed between the original and the resulting motion-compensated frames as the quality measure. For Algorithms S1, S2, ST1 and ST2, the threshold parameters used are $TH_1 = 4$ and $TH_2 = 35$ and the maximum step number for local search around is $S = 10$. Algorithm MRST does not require TH_1 and TH_2, and the maximum search step number for each finer level is 2.

For comparison, we show in Table 1 the results obtained from the full search block matching algorithm (FBMA) and the 2-D logarithmic search. To demonstrate the performance of our algorithms, we present various results with all five proposed algorithms in Tables 2-6. All values in the tables are the averaged values by using data obtained from 149 predicted frames. The meaning of each column is explained below.

- *Speed-up* represents the speed-up factor gained from the algorithms. To be more precise, a speed-up factor $= P$ means that the *matching* operations are reduced by a factor of P from the full search. Equivalently, we can say that it requires $(1/P) \times 100\%$ operations with the total number of operations required by full search normalized to one.

- *MSE (% increase)* represents the increased MSE of reconstructed motion-compensated frames with the MSE obtained via full search as the reference, which is expressed in terms of percentage.

- *Search step* represents the averaged step number in performing the local search around (presented in Tables 2-5). This includes step number 0 with $MAD_0 \leq TH_1$ (i.e. the best MV candidate is used as the final MV without MV refinement).

- *FS bks* shows the averaged number of blocks requiring full search for each frame (presented in Tables 2-5).

- *MV bits* represents the number of bits used in the coding of MVs, where the differential coding scheme adopted by the MPEG standard [10] is applied.

By comparing Algorithms S1 and ST1, and Algorithms S2 and ST2, we can see a significant improvement in the speed-up factor by using both spatial and temporal correlations than using the spatial correlation only. Furthermore, we see from Tables 2-5 that the averaged speed-up factors of ST2 (or S2) are higher than those of ST1 (or S1) in all five sequences. The main reason of gaining a high speed-up factor in ST2 (or S2) is due to computational reduction in $G2$ and $G3$ blocks, especially in $G3$ blocks where the number of blocks counts for one half of the total number of blocks. In Table 6, we see that the speed-up factors of MRST are higher than those of S1, S2, ST1, ST2 and log search. Due to multiresolution and spatio-temporal correlations, we can obtain a better set of initial MV candidates so that there is no need to perform the full search at finer levels (no TH_2). Note, however, that for "mobile" sequence, MRST gives a little lower speed-up factor than ST2 with a similar MSE result. This is because the MVs of "mobile" sequence have very strong spatial and temporal correlations so that very few blocks require full search in ST2 (only the average of 0.7 block in 'FS bks' column). For this case, multiple levels in MRST seem to add a little more computation which causes a lower speed-up factor than that of ST2. For "cheer" and "football" sequences which have a larger number of 'FS bks', we see that Algorithm MRST results in a much higher speed-up factor than ST2.

To gain more insights, we plot the frame-by-frame speed-up factor by using the log search, S2, ST2 and MRST for "football" and "flower" sequences in Figs. 6 (a) and (b), respectively. We see that MRST provides the highest speed-up factor for most frames in both sequences. Besides, we observe from the plots that ST2 has a high fluctuation in the speed-up factor. This is due to the use of a fixed TH_1 for all frames, where the value of TH_1 may be too small or too big for some frames. On the other hand, MRST gives a smaller fluctuation in the plot due to the adaptive nature of TH_1 for each frame.

As a tradeoff of fast MV estimation in MRST, we observe only small MSE increase (between $2-7\%$) in comparison with the full search. It supports our claim that the multiresolution-spatio-temporal correlations can be effectively used to reduce the computational cost without sacrificing much in MSE. For "flower" and "mobile" sequences, the results of ST2 and MRST are similar, while for "bicycle", "cheer" and "football", MRST gives a better result in terms of a higher speed-up factor and a lower MSE value. Note also that the MSE difference between ST1 and ST2 is very small. This justifies the idea of exploiting the block subsampling technique in ST2. The frame-by-frame plots of percentage of MSE increase with the log search, S2, ST2 and MRST algorithms for "football" and "flower" are given in Figs. 7 (a) and (b), respectively. It is clear that the percentages of MSE increase from MRST is much smaller than those from other algorithms. Algorithm MRST not only gives smaller MSE increases in average but also provides smaller deviations in MSE increase. This implies that the quality of the output video obtained from MRST should be better in consecutive display.

In Tables 2-5, the search step number gives us information about how close the best MV candidate is to the final MV. It is also clear that a smaller number of search step implies a higher speed-up factor. Even though the maximum step number S is set to 10, most blocks require only a few steps to reach the final MV. The "flower" and "mobile" sequences have the averaged step number even less than one, since there are many blocks with $MAD_0 \leq TH_1$ that do not require the local search around process.

For the column of 'FS bks', we see that the numbers are relatively small compared to the total of 330 blocks used in the full search. The 'FS bks' number gives us a rough idea of the occurrence of motion discontinuity or regions that cannot be well represented by those in neighboring blocks or frames due to occlusion, zooming, mixed motions, etc. The 'FS bks' and search step numbers are two main factors which determine the speed-up factor. For example, even though the "cheer" sequence has a search step less than that of the "bicycle" sequence, it has a larger number in 'FS bks' and, therefore, a lower speed-up factor.

Next, we examine the column of MV bits. We see from tables that all five proposed algorithms use fewer bits in the coding of MVs than the full search. This is a direct consequence of a smooth motion field by using the spatial correlation for MV estimation. The temporal correlation does not play an important role in MV coding since the coding is based on spatial differentials as specified by MPEG. Both full search and log search do not use the MV information from neighboring blocks. In both methods, each block is treated independently of others. This may cause MVs to jump around in some regions and result in a higher number of MV bits. We also observe that, for "bicycle", "cheer" and "football", the MV bits of MRST are a little larger than those from ST2, which could be resulted from the use of MVs obtained from the full search at the coarsest level.

Finally, we consider the coding of residual errors to obtain the rate-distortion (R-D) plot. We do not go through the whole coding process of MPEG which contains bit stream syntax, layered structure and so on. Instead, we only perform MV differential coding and DCT residual error coding, and add the numbers of bits together to represent the 'rate' with the unit of bpp (bits per pixel). The distortion is represented by PSNR, which is computed via MSE measured from the final reconstructed images (with residual coding). Two R-D plots for "football" and "flower" sequences are demonstrated in Figs. 8 (a) and (b), respectively. In both R-D plots, we see a small deterioration from the results of our algorithms in comparison with the full search while the results of the log search are much worse. The results of MRST and ST2 are very similar, and they are both better than that from S2. For the "football" sequence, the R-D plot of ST2 is in fact a little better than that of MRST despite a larger MSE. The reason could be that in some regions the ST2 gives higher correlated residual errors so that fewer bits are required by DCT. For the "flower" sequence, the results of ST2 and MRST are so similar that they appear to coincide with each other. To conclude, the proposed algorithm MRST provides a very fast MV estimation procedure while maintaining a good performance in terms of MSE as well as the rate-distortion tradeoff.

8 Conclusions

In this work, we first introduced two fast MV estimation algorithms S1 and S2 based on spatial correlation of MVs between adjacent blocks and then incorporated the temporal information to obtain Algorithms ST1 and ST2. We finally proposed the ultimate algorithm MRST by combining the multiresolution scheme with spatio-temporal correlations. The initial MV candidates of MRST are selected from the corresponding coarser-level block as well as spatially and temporally neighboring blocks at the same level. We showed with experimental results that the proposed algorithm MRST has a speed-up factor ranging from 150 to 310 with only 2-7% MSE increase and a similar rate-distortion performance in comparison with the full search. Therefore, this algorithm can be very useful for real-time video encoding applications.

References

[1] M. Bierling, "Displacement estimation by hierarchical block matching," in *Proc. SPIE Visual Commun. and Image Processing '88*, vol. 1001, pp. 942–951, 1988.

[2] F. Dufaux and M. Kunt, "Multigrid block matching motion estimation with an adaptive local mesh refinement," in *Proc. SPIE Visual Commun. and Image Processing '92*, vol. 1818, pp. 97–109, 1992.

[3] D. L. Gall, "MPEG: a video compression standard for multimedia applications," *Communications of the ACM*, Vol. 34, pp. 46–58, Apr. 1991.

[4] M. Ghanbari, "The cross-search algorithm for motion estimation," *IEEE Trans. on Communications*, Vol. 38, pp. 950–953, July 1990.

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

[6] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," in *Proc. Nat. Telecommunication Conf.*, pp. G5.3.1–5.3.5, Nov. 29 - Dec. 3 1981.

[7] J. Li, X. Lin, and Y. Wu, "Multiresolution tree architecture with its application in video sequence coding: A new result," in *Proc. SPIE Visual Commun. and Image Processing '93*, vol. 2094, pp. 730–741, 1993.

[8] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. on Circuits and Systems for Video Tech.*, Vol. 4, pp. 438–442, Aug. 1994.

[9] B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," *IEEE Trans. on Circuits and Systems for Video Tech.*, Vol. 3, pp. 148–157, Apr. 1993.

[10] MPEG1, "Coding of moving pictures and associated audio," tech. rep., Committee Draft of Standard ISO 11172, 1990.

[11] MPEG2, "Information technology - generic coding of moving pictures and associated audio," tech. rep., ISO/IEC 13818-2, Committee Draft, Mar. 1994.

[12] H. G. Musmann, P. Pirsch, and H.-J. Grallert, "Advances in picture coding," *Proceedings of the IEEE*, Vol. 73, pp. 523–548, Apr. 1985.

[13] M. T. Orchard, "A comparison of techniques for estimating block motion in image sequence coding," in *Proc. SPIE Visual Commun. and Image Processing '89*, vol. 1199, pp. 248–258, 1989.

[14] A. Puri, H.-M. Hang, and D. L. Schilling, "An efficient block matching algorithm for motion-compensated coding," in *Proc. IEEE ICASSP '87*, pp. 25.4.1–25.4.4, 1987.

[15] R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," *IEEE Trans. on Communications*, Vol. COM-33, pp. 888–896, Aug. 1985.

[16] K. M. Uz, M. Vetterli, and D. LeGall, "Interpolative multiresolution coding of advanced television with compatible subchannels," *IEEE Trans. on Circuits and Systems for Video Tech.*, Vol. 1, pp. 86–99, Mar. 1991.

[17] K. Xie, L. V. Eycken, and A. Oosterlinck, "A new block-based motion estimation algorithm," *Signal Processing: Image Communication*, Vol. 4, pp. 507–517, 1992.

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

[19] S. Zafar, Y. Q. Zhang, and B. Jabbari, "Multiscale video representation using multiresolution motion compensation and wavelet decomposition," *IEEE Journal on Selected Areas in Communications*, Vol. 11, pp. 24–35, Jan. 1993.

[20] Y.-Q. Zhang and S. Zafar, "Predictive block-matching motion estimation for TV coding – Part II: Inter-frame prediction," *IEEE Trans. on Broadcasting*, Vol. 37, pp. 102–105, Sept. 1991.

[21] Y. Q. Zhang and S. Zafar, "Motion-compensated wavelet transform coding for color video compression," *IEEE Trans. on Circuits and Systems for Video Tech.*, Vol. 2, pp. 285–296, Sept. 1992.

Table Captions

Table 1: Experimental results of five test video sequences with the log search and the full search.

Table 2: Experimental results of five test video sequences with Algorithm S1.

Table 3: Experimental results of five test video sequences with Algorithm S2.

Table 4: Experimental results of five test video sequences with Algorithm ST1.

Table 5: Experimental results of five test video sequences with Algorithm ST2.

Table 6: Experimental results of five test video sequences with Algorithm MRST.

Figure Captions

Figure 1: Histograms of MV spatial differentials of x- and y- components along the (a) horizontal and (b) vertical directions for the 60th frame of the "football" sequence (ft60)

Figure 2: Local search around procedure: the search starts at the new center (best MV candidate) V_0 and its eight neighboring points, and moves from V_0 to V_1 and to V_2 since the minimum points in the first two steps are not in the center and the new MADs are still greater than TH_1. The search stops after searching around V_2 since the minimum is located at the center V_2.

Figure 3: Block pattern employed in Algorithm S2 (and ST2 in later section) where blocks are divided into three groups: $G1$, $G2$ and $G3$ denoted by numbers 1, 2 and 3 in the figure, respectively.

Figure 4: The MV candidate selection procedure for Algorithm S2.

Figure 5: The MV candidate selection procedure for Algorithm ST2.

Figure 6: The plot of the speed-up factor as a function of the frame number with log search, S2, ST2 and MRST: (a) football and (b) flower.

Figure 7: The plot of percentage of MSE increase as a function of the frame number with log search, S2, ST2 and MRST: (a) football and (b) flower.

Figure 8: Comparison of the rate-distortion performance with full search, log search, S2, ST2 and MRST, where the rate is represented by bits per pixel (bpp) while distortion is represented by PSNR(dB): (a) football and (b) flower.

Video	**Log search**			**FBMA**	
	Speed-up	MSE (% increase)	MV bits	MSE	MV bits
Bicycle	48.6	535.9 (27.7%)	2695	419.5	2405
Cheer	56.4	542.6 (16.9%)	1856	464.3	2121
Flower	54.0	250.1 (13.2%)	1297	221.0	1080
Football	45.6	231.3 (32.5%)	3149	174.6	2450
Mobile	62.6	349.0 (7.0%)	893	326.3	840

Table 1: Experimental results of five test video sequences with the log search and the full search.

Video	**Proposed algorithm S1**				
	Speed-up	MSE (% increase)	Search step	FS bks	MV bits
Bicycle	74.3	458.8 (9.4%)	1.81	11.9	1946
Cheer	55.1	495.6 (6.7%)	1.57	19.4	1904
Flower	110.2	226.7 (2.6%)	0.87	7.3	957
Football	83.1	205.3 (17.6%)	1.51	10.8	1991
Mobile	128.4	330.6 (1.3%)	0.87	5.6	807

Table 2: Experimental results of five test video sequences with Algorithm S1.

Video	**Proposed algorithm S2**				
	Speed-up	MSE (% increase)	Search step	FS bks	MV bits
Bicycle	85.8	463.6 (10.5%)	1.81	8.2	1934
Cheer	70.6	512.4 (10.3%)	1.57	13.0	1863
Flower	128.9	228.6 (3.4%)	0.85	5.5	981
Football	95.2	207.9 (19.1%)	1.50	8.4	2039
Mobile	138.0	331.5 (1.6%)	0.86	4.8	802

Table 3: Experimental results of five test video sequences with Algorithm S2.

Video	**Proposed algorithm ST1**				
	Speed-up	MSE (% increase)	Search step	FS bks	MV bits
Bicycle	110.7	450.1 (7.3%)	1.56	5.7	1997
Cheer	84.4	489.9 (5.5%)	1.31	11.4	1911
Flower	178.1	224.6 (1.7%)	0.75	3.0	954
Football	151.5	195.2 (11.8%)	1.24	3.7	2034
Mobile	217.5	330.4 (1.3%)	0.80	1.2	808

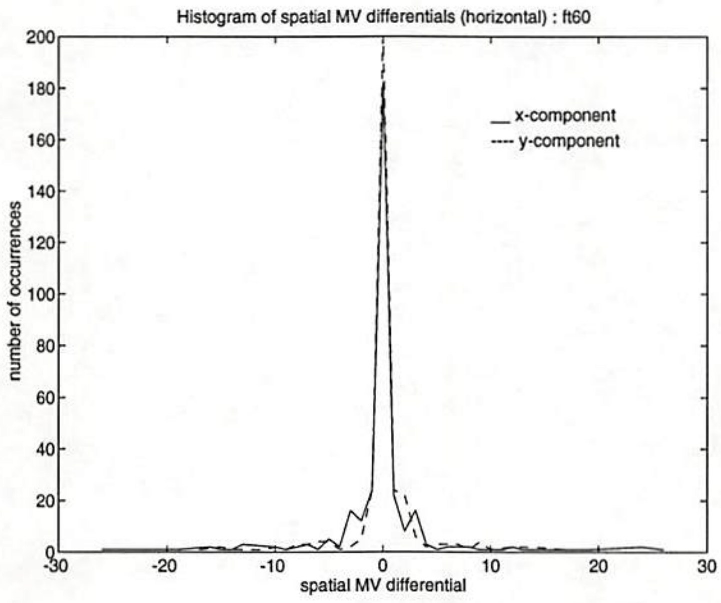
Table 4: Experimental results of five test video sequences with Algorithm ST1.

Video	**Proposed algorithm ST2**				
	Speed-up	MSE (% increase)	Search step	FS bks	MV bits
Bicycle	131.8	455.2 (8.5%)	1.54	3.1	1960
Cheer	119.6	505.7 (8.9%)	1.23	6.4	1860
Flower	271.5	225.5 (2.0%)	0.57	1.4	951
Football	177.3	198.3 (13.6%)	1.23	2.1	2035
Mobile	356.7	331.8 (1.7%)	0.55	0.7	793

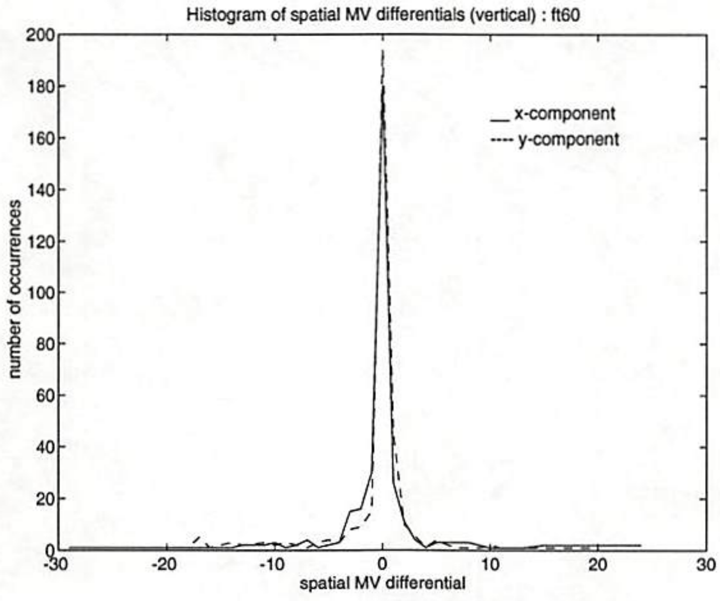
Table 5: Experimental results of five test video sequences with Algorithm ST2.

Video	**Proposed algorithm MRST**		
	Speed-up	MSE (% increase)	MV bits
Bicycle	150.4	448.7 (7.0%)	2107
Cheer	231.5	492.8 (6.1%)	1957
Flower	287.8	224.6 (1.6%)	958
Football	208.4	186.7 (6.9%)	2219
Mobile	314.5	332.6 (1.9%)	786

Table 6: Experimental results of five test video sequences with Algorithm MRST.



(a)



(b)

Figure 1: Histograms of MV spatial differentials of x- and y- components along the (a) horizontal and (b) vertical directions for the 60th frame of the "football" sequence (ft60)

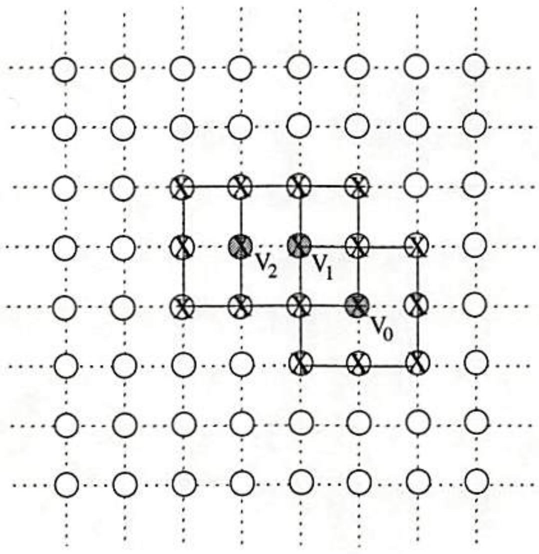


Figure 2: Local search around procedure: the search starts at the new center (best MV candidate) V_0 and its eight neighboring points, and moves from V_0 to V_1 and to V_2 since the minimum points in the first two steps are not in the center and the new MADs are still greater than TH_1. The search stops after searching around V_2 since the minimum is located at the center V_2.

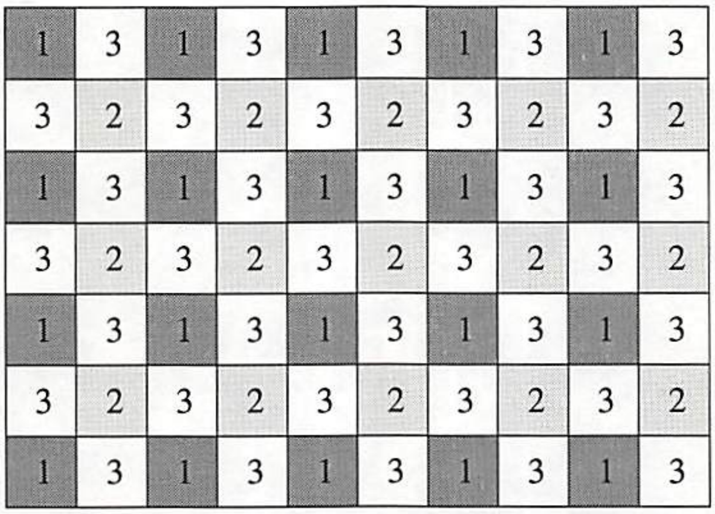


Figure 3: Block pattern employed in Algorithm S2 (and ST2 in later section) where blocks are divided into three groups: $G1$, $G2$ and $G3$ denoted by numbers 1, 2 and 3 in the figure, respectively

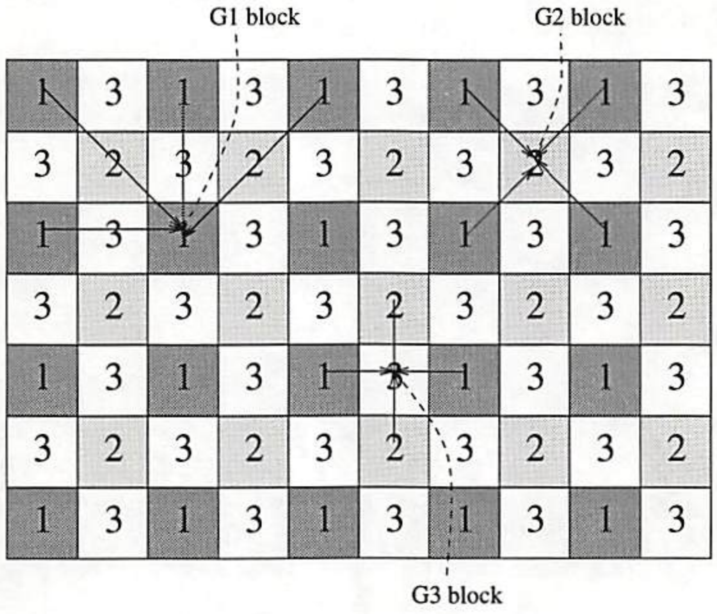


Figure 4: The MV candidate selection procedure for Algorithm S2

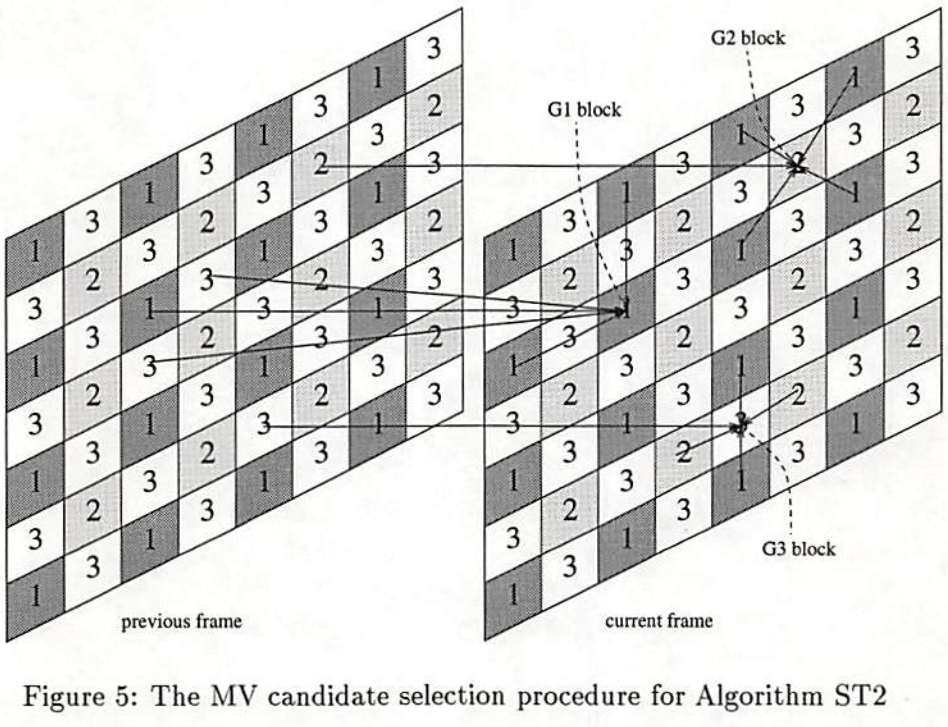
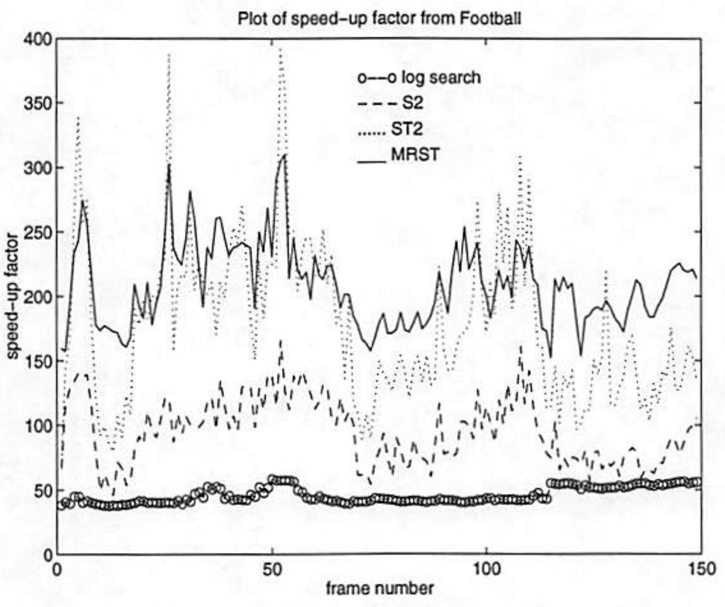
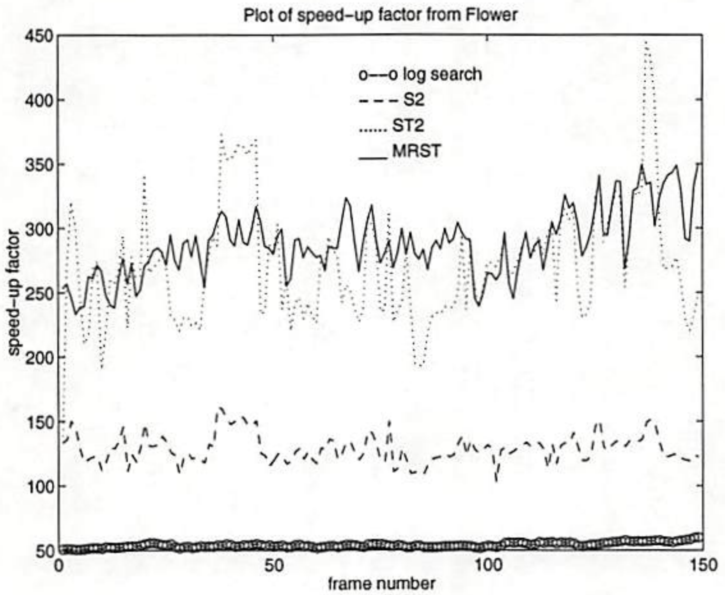


Figure 5: The MV candidate selection procedure for Algorithm ST2

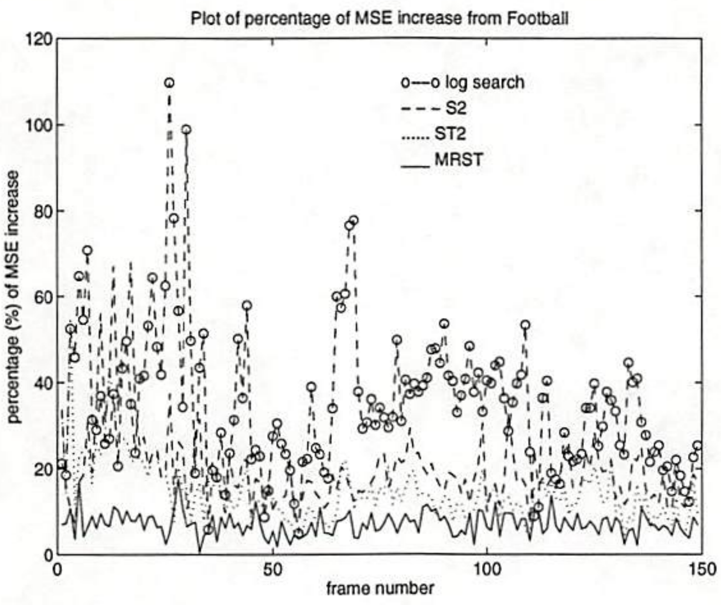


(a)

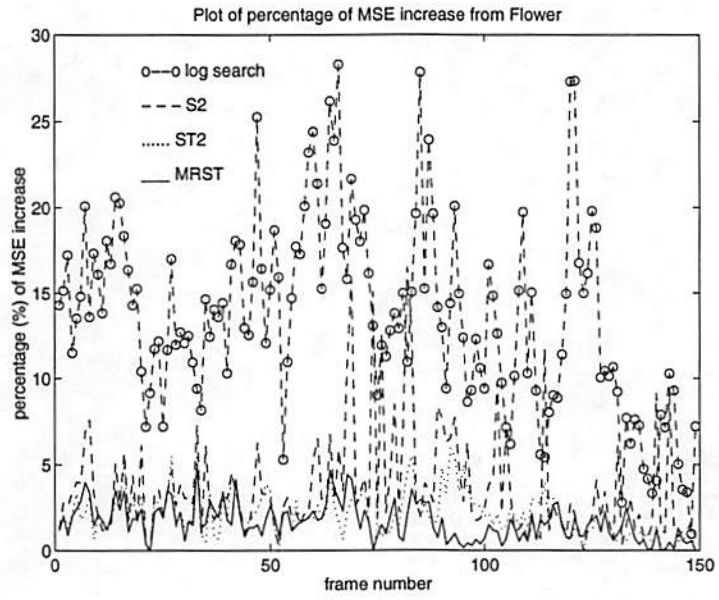


(b)

Figure 6: The plot of the speed-up factor as a function of the frame number with log search, S2, ST2 and MRST: (a) football and (b) flower.

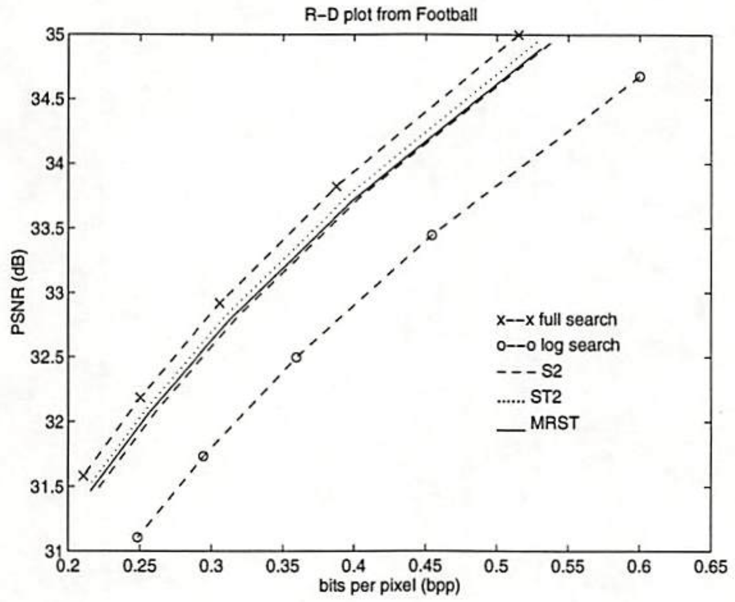


(a)

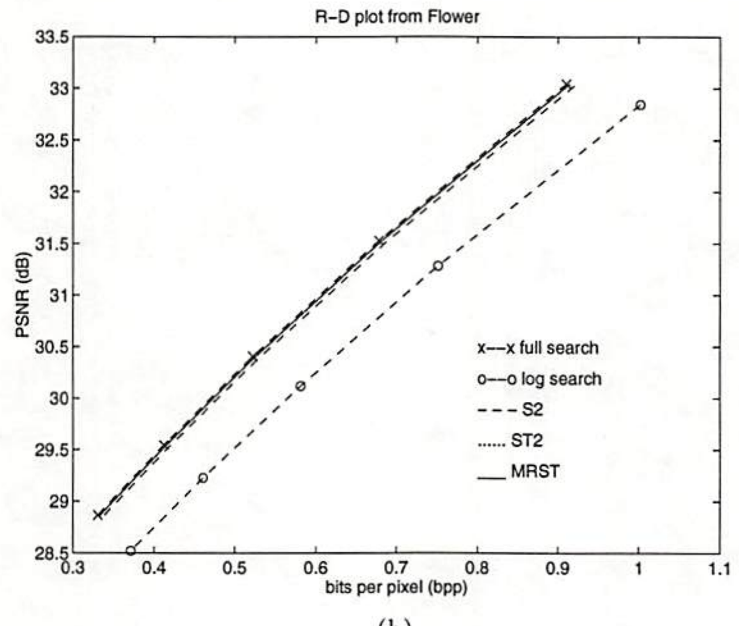


(b)

Figure 7: The plot of percentage of MSE increase as a function of the frame number with log search, S2, ST2 and MRST: (a) football and (b) flower.



(a)



(b)

Figure 8: Comparison of the rate-distortion performance with full search, log search, S2, ST2 and MRST, where the rate is represented by bits per pixel (bpp) while distortion is represented by PSNR(dB): (a) football and (b) flower.