Projection-based Block Matching Motion Estimation

Chengjie Tu\textsuperscript{1,2}, Trac D. Tran\textsuperscript{1,2}, Jerry L. Prince\textsuperscript{1}, and Pankaj Topiwala\textsuperscript{2}

\textsuperscript{1}The Johns Hopkins University, ECE Department, Baltimore, MD 21218
\textsuperscript{2}FastVDO Inc., Columbia, MD 21044

ABSTRACT

This paper introduces a fast block-based motion estimation algorithm based on matching projections. The idea is simple: blocks cannot match well if their corresponding 1D projections do not match well. We can take advantage of this observation to translate the expensive 2D block matching problem to a simpler 1D matching one by quickly eliminating a majority of matching candidates. Our novel motion estimation algorithm offers computational scalability through a single parameter and global optimum can still be achieved. Moreover, an efficient implementation to compute projections and to buffer recyclable data is also presented. Experiments show that the proposed algorithm is several times faster than the exhaustive search algorithm with nearly identical prediction performance. With the proposed BME method, high-performance real-time all-software video encoding starts to become practical for reasonable video sizes.

Keywords: Motion Estimation, Fast Block-Matching Algorithm, Projection.

1. INTRODUCTION

Motion estimation/compensation (ME/MC) can effectively eliminate temporal redundancy in a video sequence and is the key to high-quality video coding. Despite its simplicity, block-based or block-matching ME (BME) is a critical component in most state-of-the-art video codecs, including MPEG-2\textsuperscript{,1} H.263+\textsuperscript{,2} and the coming H.26L\textsuperscript{.3}

BME partitions the current frame into non-overlapped blocks. For each block, BME searches all displaced blocks within a search window in the reference frame to find the best matched block. The displacement of the best matched block is called a motion vector. Motion vectors and prediction residuals (instead of the raw data) are coded. The mean of absolute differences (MAD) is the most popular matching criterion because it is computationally inexpensive. The mean square error (MSE) is occasionally used to obtain better objective coding performance. In either case, an expensive 2D block matching process is involved. The simplest search method is the exhaustive search, where full-search 2D matching is applied to all candidate blocks. The exhaustive search certainly achieves global optimum for the given translational motion model. However, the computational complexity for the exhaustive search is tremendous. Many fast algorithms have been developed to reduce the amount of computation.

Early termination is a BME framework that can guarantee optimal matching performance by maintaining the lower bound of the matching errors. The calculation of the matching criterion can be terminated immediately when it is greater than the known lower bound obtained so far. This way, most candidates can be excluded by matching only part of the pixels. The tighter the lower bound and the earlier we can find it, the faster the algorithm. So, the search order is of great importance. The spiral search is a typical example of this philosophy. It starts with zero displacement and moves spirally to candidates with larger displacements. Hence, the spiral search tries to optimize the search order. The combination of early termination and spiral search can significantly speed up BME without losing any prediction performance and are commonly employed in high-performance codecs. The successive elimination algorithm (SEA)\textsuperscript{4,5} is another example of optimal block matching where the authors present a method to obtain much tighter lower bounds.

Most of the popular fast BME algorithms in practical systems sacrifice prediction performance to reduce BME complexity. In fact, the difference between quality levels of various video codecs available in the market today mainly depends on the quality of the BME algorithms implemented inside. Algorithms such as the three-step search,\textsuperscript{6} the logarithmic search,\textsuperscript{7} and the conjugate direction search\textsuperscript{8} all subsample the motion displacement space and thus reduce the number of candidates. They can be extremely fast at the expense of prediction performance. Hierarchical search, which uses reduced resolution for the first one or more stages of the search, has been investigated not only...
in the spatial domain \(^9\) but also in the transform domain. \(^{10}\) Algorithms with this philosophy such as the telescopic search \(^{11}\) and the predictive pattern search \(^{12}\) assume temporal continuity of motion displacements and use prediction to initialize the search and concentrate most of their later searches locally around that point. Needless to say, such algorithms cannot guarantee matching optimality.

The computational complexity of BME is the direct consequence of the expensive 2D block matching process. This paper presents a BME algorithm based on projections (PBME) which can reduce the number of 2D matchings tremendously. The relationship between motion and projections has been established before within the context of global motion estimation for medical imaging applications. \(^{13}\) Our proposed use of this theory is different. Our algorithm uses this theory to eliminate candidates by matching 1D projections. We will also describe the efficient computation of projections as well as the buffering of recyclable data – two important aspects of the proposed algorithm. By controlling the percentage of candidates excluded by 1D matching, the speed of the PBME algorithm is also controllable.

2. PROJECTIONS

2.1. Definition

Suppose the frame size is \(W \times H\), the block size is \(B_h \times B_v\) and the search window size is \(W_h \times W_v\). To predict one block, there are \(W_h \times W_v\) candidates to search.

Let \(B^{x,y}\) denote a 2D block of pixels with the top-left position at \((x,y)\) in a video frame and \(B^{x,y}_{i,j}\), where \(0 \leq i < B_v\) and \(0 \leq j < B_h\), be the pixel at the \(j^{th}\) row and the \(i^{th}\) column of the block. Define the (vertical) projection of \(B^{x,y}\) be \(LB^{x,y}\), a 1D row vector whose \(j^{th}\) value is the sum of the \(i^{th}\) column of \(B^{x,y}\):

\[
LB^x_i = \sum_{j=0}^{B_v-1} B_{i,j}^x, \quad i = 0, 1, \ldots, B_h - 1.
\]

By projection, a \(B_h \times B_v\) 2D block is reduced to a \(B_h\)-component 1D vector and only the DC information of each column is preserved.

We can further define the sum to be:

\[
PB^{x,y} = \sum_{i=0}^{B_h-1} LB^x_i = \sum_{i=0}^{B_h-1} \sum_{j=0}^{B_v-1} B_{i,j}^x.
\]

Hence \(PB^{x,y}\) preserves the DC information of \(B^{x,y}\).

2.2. Fast Projection

In the current frame, there are only \(\frac{WH}{B_hB_v}\) non-overlapped blocks and the computational complexity of the 1D projection above is small \(O(WH)\) operations. In other words, the complexity increases linearly with the video size only.

In the reference frame, there are \(W \times H\) different blocks. \(B^{x,y}\) and its direct right (lower) neighbor \(B^{x+1,y}\) \((B^{x,y+1})\) share all pixels except two columns (or rows). If \(LB^{x,y}\) is known, \(LB^{x+1,y}\) and \(LB^{x,y+1}\) can be updated efficiently:

\[
LB^x_{i+1} = LB^x_i - B^x_{i,0} + B^x_{i,B_v-1}
\]

\[
LB^y_{i} = \begin{cases} 
LB^y_{i+1} & \text{if } i < B_h - 1, \\
\sum_{j=0}^{B_v-1} B^{y+1}_{i,j} & \text{if } i = B_h - 1.
\end{cases}
\]

Starting with \(B^{0,0}\), with proper buffering, only 2 operations are required on average to compute the projection of a block in this updating manner. The cost for projection is therefore only \(O(2WH)\) operations.
2.3. Buffering Scheme
To search for the motion vectors of a strip of blocks with their top corners at the same position \( y \) in the current frame, only the blocks with top corners within \([y - W_h/2, y + W_h/2]\) are involved in the reference frame, which is a \( W \times W_h \) strip. So a \( W \times W_h \) instead of \( W \times H \) (usually \( H >> W_h \)) buffer is sufficient to store all recyclable projections. When moving to the next strip, we need to slide the buffer up by \( B_h \) lines, discard the \( B_h \) lines moving out and update the buffer with \( B_h \) lines moving in using fast projection. An additional \( B_h \)-point buffer is necessary for the current block in the current frame.

3. PROJECTION-BASED BME

3.1. 2D Matching
For a block \( C^{x,y} \) in the current frame, BME searches all displaced blocks \( R^{x+dx,y+dy} \) in the search window in the reference frame for the best matched block. The matching error (MAD) is:

\[
MAD(dx, dy) = \sum_{i=0}^{B_h-1} \sum_{j=0}^{B_v-1} |C_{i,j}^{x,y} - R_{i,j}^{x+dx,y+dy}|.
\]  

(5)

The cost of this expensive 2D block matching is \( O(B_h \times B_v) \) operations.

BME tries to find the minimum value of \( MAD \), labeled \( MAD_{\text{min}} \), and the corresponding displacement \((dx, dy)\) is the optimal motion vector of \( C^{x,y} \):

\[
MAD_{\text{min}} = \arg \min_{dx, dy} \text{MAD}(dx, dy), \text{ subject to } |dx| < \frac{W_h}{2}, |dy| < \frac{W_v}{2}.
\]

(6)

There are \( W_h \times W_v \) candidates to search and we need \( O(B_h \times B_v \times W_h \times W_v) \) operations to find just one motion vector.

3.2. 1D Matching
The matching error (MAD) of the projections of block \( C^{x,y} \) and block \( R^{x+dx,y+dy} \) is

\[
LMAD(dx, dy) = \sum_{i=0}^{B_h-1} |LC_{i}^{x,y} - LR_{i}^{x+dx,y+dy}|.
\]

(7)

This is a low-complexity 1D matching problem and only \( O(B_v) \) operations is involved per given projections, which is only \( \frac{1}{B_v} \) of the 2D block matching cost.

Define \( LMAD_{\text{min}} \) as

\[
LMAD_{\text{min}} = \arg \min_{dx, dy} \text{LMAD}(dx, dy), \text{ subject to } |dx| < \frac{W_h}{2}, |dy| < \frac{W_v}{2}.
\]

(8)

This cost function will serve as a threshold that we can take advantage of in eliminating most of the matching candidates as described in the next section.

3.3. Candidate Exclusion via 1D Matching
The intuition is that the projections of two similar blocks must be quite similar. In other words, two blocks cannot match well if their projections do not. According to the generalized triangular equality, we can obtain

\[
MAD(dx, dy) = \sum_{i=0}^{B_h-1} \sum_{j=0}^{B_v-1} |C_{i,j}^{x,y} - R_{i,j}^{x+dx,y+dy}|
\geq \sum_{i=0}^{B_h-1} | \sum_{j=0}^{B_v-1} C_{i,j}^{x,y} - \sum_{j=0}^{B_v-1} R_{i,j}^{x+dx,y+dy} |
\leq \sum_{i=0}^{B_h-1} |LC_{i}^{x,y} - LR_{i}^{x+dx,y+dy}|
= LMAD(dx, dy).
\]

(9)
Hence, if \( LMAD(dx, dy) > MAD_{\text{min}} \), then \( MAD(dx, dy) > MAD_{\text{min}} \) and the candidate cannot be the best matched block. Usually, the majority of the candidates satisfies this \( LMAD(dx, dy) > MAD_{\text{min}} \) constraint. Therefore they can be excluded from future searches. Then to obtain the optimal motion vector, 2D matching is only necessary for those candidates with \( LMAD(dx, dy) \leq MAD_{\text{min}} \), which is often a very small percentage.

3.4. \( MAD_{\text{min}} \) Estimation

From the previous discussion, we see that the value \( MAD_{\text{min}} \) is necessary in the exclusion of candidates by 1D matching. Unfortunately, we do not know \( MAD_{\text{min}} \) in advance. An estimate \( MAD_{e} \) is used instead. The smaller the \( MAD_{e} \) value, the more candidates we can eliminate from 1D matching, and the faster the algorithm. If \( MAD_{e} \geq MAD_{\text{min}} \), the optimality of the motion vectors is still preserved. However, if \( MAD_{e} < MAD_{\text{min}} \), we might lose optimality, and this is more likely for even smaller \( MAD_{e} \).

A good estimation of \( MAD_{\text{min}} \) is based on the value of \( LMAD_{\text{min}} \):

\[
MAD_{e} = \alpha \times LMAD_{\text{min}},
\]

where \( \alpha \geq 1 \) is a scaling factor. A \( W_{h} \times W_{w} \) buffer is necessary to keep the \( LMAD(dx, dy) \) values for all candidates. It turns out empirically that this estimation method is quite robust because it is partially adaptive, i.e., at least \( LMAD_{\text{min}} \) contains the information about the block.

By changing \( \alpha \), the complexity of the search can be controlled and computational scalability is achievable.

3.5. Overall Algorithm

Let us recapitulate the overall procedure of the proposed projection-based BME algorithm.

1. Compute and buffer necessary projections using the fast projection algorithm.
2. Compute and buffer 1D matching error \( LMAD(dx, dy) \) for all candidates.
3. Estimate \( MAD_{e} \) based on \( LMAD_{\text{min}} \).
4. Eliminate all candidates with \( LMAD(dx, dy) > MAD_{e} \).
5. For the remaining candidates with \( LMAD(dx, dy) \leq MAD_{e} \), perform 2D matching to find the best matched block.

3.6. MSE Extension

If MSE instead of MAD is used as the matching criterion, we can still derive a fast algorithm. In the MSE case, the 2D matching error is now

\[
MSE(dx, dy) = \sum_{i=0}^{B_{x}-1} \sum_{j=0}^{B_{y}-1} (C_{i,j}^{x,y} - R_{i,j}^{x+dx,y+dy})^2. \tag{11}
\]

and the 1D matching error is

\[
LMSE(dx, dy) = \sum_{i=0}^{B_{x}-1} (LC_{i}^{x,y} - LR_{i}^{x+dx,y+dy})^2. \tag{12}
\]

It is easy to see that our proposed algorithm works with MSE since the idea, blocks cannot match well if their corresponding 1D projections do not match well, is still correct. While

\[
MSE(dx, dy) \geq LMSE(dx, dy). \tag{13}
\]

doesn’t always hold, there is no guarantee that the candidate is not the best matched block even if \( LMSE(dx, dy) = MSE_{\text{min}} \).
4. DISCUSSION

In this section, we discuss several implementation issues as well as variations of our proposed PBME algorithm.

1. In the proposed algorithm, the early termination and spiral searching concepts can be applied in both 1D matching and 2D matching to further speed up BME.

2. Only vertical projection is demonstrated in this paper since most videos have more horizontal motions. However, horizontal projection might be better for some video data.

3. A more greedy approach, trying to eliminate candidates by only matching the sums of two blocks using $PB^x_y$ in Eq. (2), does not work well. This is DC matching and we can derive fast methods to compute the DC as well as manage the 1D buffer. Although DC matching is very low cost (only one operation is involved per match operation), the sum contains too little information about a block. Our empirical data indicate that DC matching yields too many mismatches.

4. By using both vertical and horizontal projections, a few more candidates can be eliminated. However, the saving is just enough to compensate the cost of the extra projections. Besides, extra buffers are necessary. Hence, matching two 1D projections is not any better than using just one projection.

5. SIMULATION RESULTS

Popular QCIF ($176 \times 144$) test sequences are used to compare the prediction performance as well as the speed of the proposed algorithm and the exhaustive search algorithm. The block size is $16 \times 16$ pixels and the search window size is $32 \times 32$ pixels. The reference frame is symmetrically extended by 16 pixels to allow motion vectors to point outside the frame. The extended reference frame is bilinearly interpolated to enable half-pixel accuracy BME. To obtain half-pixel motion accuracy, the best matched block with whole-pixel displacement is found first and then only its 8 direct neighbors with half-pixel displacement are checked to get the approximated best matched block with half-pixel accuracy. To find one motion vector, $32 \times 32 + 8 = 972$ 2D matchings are needed for the exhaustive search algorithm and $32 \times 32 = 964$ 1D matching plus at least 9 2D matchings are necessary for the proposed algorithm. Both algorithms employed early termination and spiral searching. To favor the zero motion vector, the corresponding 2D matching error for zero displacement is reduced by a constant amount (100 for MAD and 1000 for MSE).

Three test sequences – Foreman, News and Miss American – are used to benchmark the exhaustive search algorithm and the proposed algorithm with $\alpha = 2, 4, 8$. Fig. 1 - Fig. 3 plot the comparative prediction errors. Four plots nearly overlap indicating that the prediction performance of our algorithm is nearly identical to that of the exhaustive search algorithm. The only video segment with visible quality loss is for frame 250 to frame 350 of the Foreman sequence (where the camera is panning quickly). Table 1 to Table 3 tabulate the overall MAD and MSE. We can observe that using MAD as the matching criterion yields slightly better objective results. Table 1 to Table 3 also summarize the percentage of candidates where 2D matching is necessary. This percentage is always 100% for the exhaustive search algorithm. The smaller the value of $\alpha$, the smaller this 2D percentage is, and thus the faster the PBME algorithm is. The actual running speed in frames per second (fps) based on a DELL Dimension L600r PIII 600Mhz machine with 128M RAM running RedHat Linux 6.1 Workstation is also presented. The proposed algorithm is about 3-5 times faster than the exhaustive search algorithm with almost identical performance. The prediction performance only degrades slightly when $\alpha$ reduces from 8 to 2. However, the PBME algorithm's computational complexity decreases drastically. Computational scalability can be achieved with a small cost.

6. CONCLUSION

We have presented a projection-based fast BME algorithm called PBME in this paper. An efficient method to compute and to buffer projection data was also described. The algorithm greatly reduces the computational complexity of BME while maintaining prediction integrity since most candidates can be quickly eliminated by matching 1D projections, which is much faster than matching 2D blocks. The prediction performance is close to the global optimum and the speed is several times faster than the exhaustive search algorithm. All-software real-time high-performance encoding is certainly practical for QCIF-size videos. Computational scalability, which is often difficult to achieve in other algorithms, can be easily obtained by controlling one single parameter.
### Table 1. Search results for the Foreman sequence.

<table>
<thead>
<tr>
<th></th>
<th>exhaustive search</th>
<th>(\alpha = 8)</th>
<th>(\alpha = 4)</th>
<th>(\alpha = 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAD</td>
<td>MAD percentage (%)</td>
<td>2.9184</td>
<td>2.9185</td>
<td>2.9360</td>
</tr>
<tr>
<td></td>
<td>speed (fps)</td>
<td>100.0</td>
<td>20.2</td>
<td>9.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8.4</td>
<td>18.7</td>
<td>26.7</td>
</tr>
<tr>
<td>MSE</td>
<td>MSE percentage (%)</td>
<td>32.4153</td>
<td>32.4223</td>
<td>32.7915</td>
</tr>
<tr>
<td></td>
<td>speed (fps)</td>
<td>100.0</td>
<td>44.2</td>
<td>24.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8.5</td>
<td>19.9</td>
<td>28.7</td>
</tr>
</tbody>
</table>

### Table 2. Search results for the News sequence.

<table>
<thead>
<tr>
<th></th>
<th>exhaustive search</th>
<th>(\alpha = 8)</th>
<th>(\alpha = 4)</th>
<th>(\alpha = 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAD</td>
<td>MAD percentage (%)</td>
<td>1.1218</td>
<td>1.1218</td>
<td>1.1251</td>
</tr>
<tr>
<td></td>
<td>speed (fps)</td>
<td>100.0</td>
<td>7.2</td>
<td>4.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>15.4</td>
<td>39.0</td>
<td>46.2</td>
</tr>
<tr>
<td>MSE</td>
<td>MSE percentage (%)</td>
<td>22.0718</td>
<td>22.1172</td>
<td>22.2566</td>
</tr>
<tr>
<td></td>
<td>speed (fps)</td>
<td>100.0</td>
<td>7.6</td>
<td>5.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16.3</td>
<td>37.1</td>
<td>40.9</td>
</tr>
</tbody>
</table>

### Table 3. Search results for the Miss America sequence.

<table>
<thead>
<tr>
<th></th>
<th>exhaustive search</th>
<th>(\alpha = 8)</th>
<th>(\alpha = 4)</th>
<th>(\alpha = 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAD</td>
<td>MAD percentage (%)</td>
<td>1.0794</td>
<td>1.0794</td>
<td>1.0794</td>
</tr>
<tr>
<td></td>
<td>speed (fps)</td>
<td>100.0</td>
<td>38.7</td>
<td>24.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7.9</td>
<td>16.1</td>
<td>23.5</td>
</tr>
<tr>
<td>MSE</td>
<td>MSE percentage (%)</td>
<td>3.4014</td>
<td>3.4014</td>
<td>3.4019</td>
</tr>
<tr>
<td></td>
<td>speed (fps)</td>
<td>100.0</td>
<td>24.8</td>
<td>10.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6.2</td>
<td>18.3</td>
<td>28.6</td>
</tr>
</tbody>
</table>
Figure 1. Prediction errors for the Foreman sequence. (a) MAD. (b) MSE.
Figure 2. Prediction errors for the News sequence. (a) MAD. (b) MSE.
Figure 3. Prediction errors for the Miss America sequence. (a) MAD. (b) MSE.
REFERENCES