A FAST MOTION ESTIMATION ALGORITHM EQUIVALENT TO EXHAUSTIVE SEARCH

Mohammad Gharavi-Alkhansari

Tarbiat Modares University,
Intersection of Jalal-Al-Ahmad St. and Chamran Highway, Tehran, Iran

Email: gharavi@modares.ac.ir

ABSTRACT

In this paper, a fast algorithm is proposed for block motion estimation for video sequences. The proposed algorithm is proven to be equivalent to exhaustive search.

In a multiresolution approach, it uses mathematically derived threshold to prune search candidates whose low-resolution versions are too far from the low resolution version of the block for which a best match is sought.

Experimental results show that speed ups of around 36, compared to full search, may be achieved, for some typical test video sequences. This is the fastest full-search-equivalent motion estimation reported in the literature to date, and has speed ups comparable to inexact fast motion estimation methods.

1. INTRODUCTION

Block motion estimation is the most common method of estimation of motion in video sequences [1]. In this method, each frame in a video sequence is partitioned into small non overlapping blocks. For each of these blocks, a search is conducted in its neighborhood in the previous decoded frame, to find the best match based on a mean absolute difference (MAD) or a mean squared error (MSE) criteria.

Block motion estimation is the essential part of all video compression standards so far. It is also the most time-consuming part of most video compression algorithms. Several methods have been developed to reduce this computational burden, most of which achieve this computation reduction at the expense of lower precision, i.e., the lower complexity search does not necessarily find the best match that would be found by a full search. This results in a higher residual error and a larger distortion. In this paper, we propose a fast search method that finds the absolute best match at a significantly reduced computational cost compared to a full search.

To simplify the notation, in what follows, we will assume the signals discussed are all one-dimensional. The two-dimensional case may easily be derived in a similar fashion.

Let us denote the block for which the best match is sought by $\mathbf{x}$, and denote the frame in which a search is conducted by $\mathbf{z}$. Vectors $\mathbf{x}$ and $\mathbf{z}$ may be represented by

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \text{ and } \mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_C \end{bmatrix},$$

where N and C are the number of elements (pixels) in $\mathbf{x}$ and $\mathbf{z}$, respectively. A pool of N-dimensional blocks is made from a search window in $\mathbf{z}$, against whose members $\mathbf{x}$ will be matched. A total of $B = C - N + 1$ of N-dimensional vectors $\mathbf{y}_i, i = 1, 2, \ldots, B$ may be extracted from $\mathbf{z}$ by

$$\mathbf{y}_i = \begin{bmatrix} y_{i,1} \\ y_{i,2} \\ \vdots \\ y_{i,N} \end{bmatrix} = \begin{bmatrix} z_i \\ z_{i+1} \\ \vdots \\ z_{i+N-1} \end{bmatrix}, \qquad i = 1, 2, \ldots, B.$$

Typically, in video coding, only a portion of these B vectors are searched. Let us assume that the range of search in $\mathbf{z}$ includes only the K vectors $\mathbf{y}_i$, $i = G, G+1, \ldots, G+K-1$, where G is the index of the first block in the search window.

In an exhaustive search, the distance between $\mathbf{x}$ and $\mathbf{y}_i$, $i = G, G+1, \ldots, G+K-1$ is computed, and $\mathbf{y}_i$ with the smallest distance is selected:

for $i = G$ to $G + K - 1$
$\quad d_i := \|\mathbf{x} - \mathbf{y}_i\|_p^p = \sum_{j=1}^{N} |x_i - y_{i,j}|^p$
end
$I := \arg\min_i d_i$

where $\|\cdot\|_p$ represents the p-norm for vectors [2]. Normally, $p = 1$ or $p = 2$ are used for norms, where $p = 1$ corresponds to computing mean absolute difference (MAD), and $p = 2$ corresponds to computing mean squared error (MSE).

For any given value of p, in line 2 of the above algorithm, we call the computations required for one term in the summation, as one *basic operation*. Hence, the exhaustive search requires KN basic operations.

In the next section, we will present an algorithm that, for typical cases, finds the index I of the best match using far less computations.

2. THE PROPOSED METHOD

In its single-resolution form, the proposed method consists of three steps:

1. Using a bounded operator $\mathbf{A}$, compute transformed versions of $\mathbf{x}$ and $\mathbf{y}_i$'s:
$$\begin{aligned}\bar{\mathbf{x}} &= \mathbf{A}\mathbf{x}, \\ \bar{\mathbf{y}}_i &= \mathbf{A}\mathbf{y}_i.\end{aligned}$$
2. Compute $\bar{d}_i = \|\bar{\mathbf{x}} - \bar{\mathbf{y}}_i\|_p^p$ for $G \le i \le G + K - 1$.
3. Prune any candidate $\mathbf{y}_i$ for which $\bar{d}_i > D$, where D is a theoretically determined threshold.
4. Compute $d_i = \|\mathbf{x} - \mathbf{y}_i\|_p^p$ for the remaining candidates, and find the candidate with smallest d_i.

Step 4 is just similar to a full search, except that it is only conducted for candidates that survive the pruning conducted in step 3.

For the proposed algorithm to be effective, steps 1 and 2 should require significantly less computations compared to a full search, and the number of pruned candidates should be large.

In what follows, for any given bounded operator $\mathbf{A}$, we introduce the threshold D. Then, we focus our attention on a particular form of $\mathbf{A}$ which results in effective pruning, and for which steps 1 and 2 may be done with relatively small number of operations.

2.1. The Pruning Threshold

Consider a linear operator $\mathbf{A} : \Re^N \longmapsto \Re^{N'}$. Norm of $\mathbf{A}$, denoted by $\|\mathbf{A}\|$, is defined [2, 3] as
$$\|\mathbf{A}\| = \sup_{\mathbf{x}\neq\mathbf{0}} \frac{\|\mathbf{A}\mathbf{x}\|}{\|\mathbf{x}\|} = \sup_{\mathbf{x}\neq\mathbf{0}} \frac{\|\bar{\mathbf{x}}\|}{\|\mathbf{x}\|}.$$
Therefore, for any $\mathbf{x} \in \Re^N$, we have
$$\|\mathbf{A}\mathbf{x}\| \le \|\mathbf{A}\|\|\mathbf{x}\|. \quad (1)$$
Now, as before, let us denote the index of the best match by I. This means that for $G \le i \le G + K - 1$, we have
$$\|\mathbf{x} - \mathbf{y}_I\| \le \|\mathbf{x} - \mathbf{y}_i\|. \quad (2)$$
Then,
$$\begin{aligned}\|\bar{\mathbf{x}} - \bar{\mathbf{y}}_I\| = \|\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{y}_I\| &\le \|\mathbf{A}\|\,\|\mathbf{x} - \mathbf{y}_I\| \\ &\le \|\mathbf{A}\|\,\|\mathbf{x} - \mathbf{y}_i\|, \quad (3)\end{aligned}$$
where the first inequality is due to (1), and the second inequality is due to (2). Note that (3) provides us with the thresholds
$$D_i = \|\mathbf{A}\|\|\mathbf{x} - \mathbf{y}_i\|, \quad i = G, G+1, \ldots, G+K-1, \quad (4)$$
and the best match $\mathbf{y}_I$ should satisfy
$$\|\bar{\mathbf{x}} - \bar{\mathbf{y}}_I\| \le D_i, \quad i = G, G+1, \ldots, G+K-1.$$
This means that any vector $\mathbf{y}_j$ that does not satisfy $\|\bar{\mathbf{x}} - \bar{\mathbf{y}}_j\| \le D_i$ for any i, cannot be the best match, and may be removed from further consideration. Even though any value of i may be used to compute D_i, to achieve the greatest pruning, we would like D_i to be as small as possible. Let us define
$$J = \arg\min_i \bar{d}_i.$$
For the type of operators that we will use, $\bar{d}_i$ closely follows $\|\mathbf{A}\|d_i$ for different i's, and therefore, a small $\bar{d}_i$ usually indicates a small $\|\mathbf{A}\|d_i$, and therefore, D_J is a reasonable threshold. Also, if J' is the index of the image block at zero displacement at the previous frame, we usually expect $D_{J'}$ to be small. In a multiresolution algorithm that will be introduced later, we will use both of these thresholds.

2.2. The Operator

Now, consider the linear operator $\mathbf{A} : \Re^N \longrightarrow \Re^{\frac{N}{m}}$, represented by the matrix
$$\begin{aligned}\mathbf{A}_m &= \begin{bmatrix} \mathbf{1}_{1\times m} & \mathbf{0}_{1\times m} & \cdots & \mathbf{0}_{1\times m} \\ \mathbf{0}_{1\times m} & \mathbf{1}_{1\times m} & \cdots & \mathbf{0}_{1\times m} \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{0}_{1\times m} & \mathbf{0}_{1\times m} & \cdots & \mathbf{1}_{1\times m} \end{bmatrix}_{\frac{N}{m}\times N} \\ &= \mathbf{I}_{\frac{N}{m}\times\frac{N}{m}} \otimes \mathbf{1}_{1\times m},\end{aligned}$$
where $\mathbf{1}$ and $\mathbf{0}$ represent matrices with all elements equal to 1 and 0, respectively, and $\otimes$ represents the Kronecker product. This operator, replaces every m elements in any vector in $\Re^N$ with a single element equal to the sum of the original m elements. This operation is in fact a decimation operation [4] and generates a lower-resolution vector from any N-dimensional vector. Jensen's inequality [5] may be used to show that for the above operator,
$$\|\mathbf{A}_m\|_p = m^{\frac{p-1}{p}}.$$
For this operator, step 1 of the proposed algorithm, may be done very efficiently by noting that $\mathbf{y}_i$'s have significant overlap, and one does not need to apply summation for each element of every $\mathbf{y}_i$ separately. Instead, one may apply a low-pass filtering on $\mathbf{z}$, to obtain its low-passed version $\bar{\mathbf{z}}$, by:
$$\bar{z}_i = \sum_{j=0}^{m-1} z_{i+j},$$
and then, obtain $\bar{\mathbf{y}}_i$'s by appropriate sampling from $\bar{\mathbf{z}}$. Computation of $\bar{\mathbf{z}}$ usually needs much less operations than step 2.

For step 2, we note that because $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}_i$'s have length N/m, therefore, computing $\bar{d}_i = \|\bar{\mathbf{x}} - \bar{\mathbf{y}}_i\|_p^p$ requires only $1/m$ the number of basic operations required for computing $d_i = \|\mathbf{x} - \mathbf{y}_i\|_p^p$.

Note that the above operator $\mathbf{A}_m$, when m is a power of 2, is in fact an operator that computes the N/m low frequency coefficients of the Haar transform of a vector [6]. Similar operators, for other types of transformations, may also be used. These transformations, for example, may be other types of wavelet transform, with better energy compaction properties. However, they usually require more overhead computations.

3. THE MULTIRESOLUTION APPROACH

In the proposed approach, if we choose a smaller m, computing $\bar{d}_i$ requires more computation, but finding J corresponds to a higher-resolution search. So, we may adopt a multi-resolution approach, where we begin with a large m, for example $m = N$, and we conduct a low resolution pruning with little computation. Then increase m, and conduct the pruning only on the remaining candidates. This approach may be done at different values of m, until the final stage, at which a full-resolution search is conducted.

As mentioned earlier, the threshold D_i may be computed for any i in (4). In experiments for motion estimation, it was found that at the lowest resolution, i.e., first stage of pruning, it was more effective to use $i = J'$. In the following stages, $i = J$ are used for computing D_i.

The proposed multiresolution motion estimation method algorithm is shown is Table 1. Several additional techniques have been

Table 1. The proposed fast motion estimation algorithm.

FastMotionEstimation($\mathbf{z}_n, \mathbf{z}_{n-1}, N, W, p$)

$C := \text{length}(\mathbf{z}_n)$
$M := 2$
$T := \log_M N$
$m_0 = 1$
$\bar{\mathbf{z}}_0 := \mathbf{z}_{n-1}$
for $t := 1$ to T
  $m_t := M^t$
  $\bar{\mathbf{z}}_t := \mathbf{P}_{m_t}\ \bar{\mathbf{z}}_{t-1}$
end
$\mathbf{J} := [\ ]$
for $j := 1$ to $C - N + 1$ in steps of N
  $J_T := j$
  $L_T := \{\min(j - W, 1), \ldots, \max(j + W, C)\}$
  $\mathbf{x} := \mathbf{F}_j\ \mathbf{z}_n$
  $\bar{\mathbf{x}}_0 := \mathbf{x}$
  for $t := 1$ to T
    $\bar{\mathbf{x}}_t := \mathbf{Q}_{m_t}\ \bar{\mathbf{x}}_{t-1}$
  end
  $E_T := \|\mathbf{x} - \mathbf{y}_{J_T}\|_p^p$
  $D := \left(\|\mathbf{A}_N\|_p^p\ E_T\right)^{\frac{1}{p}}$
  for all $i \in L_T$
    $\bar{y}_{i,1} := \mathbf{H}_{i,N}\ \bar{\mathbf{z}}_T$
    $\bar{d}_i := |\bar{x}_{T,1} - \bar{y}_{i,1}|$
    if $\bar{d}_i > D$ then $L_{T-1} := L_T \setminus \{i\}$
  end
  for $t := T - 1$ downto 0
    for all $i \in L_t$
      $\bar{\mathbf{y}}_{i,t} := \mathbf{H}_{i,m_t}\ \bar{\mathbf{z}}_t$
      $\bar{d}_i := \|\bar{\mathbf{x}}_t - \bar{\mathbf{y}}_{i,t}\|_p^p$
    end
    $J_t := \underset{i \in L_t}{\text{argmin}}\ \bar{d}_i$
    if $t > 0$
      if $J_t \neq J_{t+1}$
        $\mathbf{y}_{J_t} := \mathbf{F}_{J_t}\ \mathbf{z}$
        $E_t := \|\mathbf{x} - \mathbf{y}_{J_t}\|_p^p$
        if $E_t > E_{t+1}$
          $E_t := E_{t+1}$
          $J_t := J_{t+1}$
        end
      else
        $E_t := E_{t+1}$
      end
      $D := \|\mathbf{A}_{m_t}\|_p^p\ E_t$
      $L_{t-1} := L_t \setminus \{i\ :\ \bar{d}_i > D\}$
      if $|L_{t-1}| = 1$ or $D = 0$ then break
    end
  end
  $\mathbf{J} := [\mathbf{J}\ \ J_t]$
end
return $\mathbf{J}$

used in this algorithm to reduce computations and avoid repeated computation of identical values. The inputs to this algorithm are: the current frame $\mathbf{z}_n$, the previous frame $\mathbf{z}_{n-1}$, block size N, offset of the search window W, and the parameter p for the norm being used (e.g. 1 for MAD or 2 for MSE).

In this algorithm, lowpass filtered versions of $\mathbf{z}$ for different resolutions are obtained by a recursive formula that computes $\bar{\mathbf{z}}_t$ from $\bar{\mathbf{z}}_{t-1}$ using the matrix

$$\mathbf{P}_m = [p_{i,j}]_{(C-m+1)\times(C-\frac{m}{M}+1)}$$

where

$$p_{i,j} = \begin{cases} 1 & \text{if } j = i,\ i + \frac{m}{M},\ i + \frac{2m}{M}, \ldots,\ i + \frac{(M-1)m}{M} \\ 0 & \text{otherwise} \end{cases}.$$

Similarly, $\bar{\mathbf{x}}_t$ is computed from $\bar{\mathbf{x}}_{t-1}$ using matrix

$$\mathbf{Q}_m = \mathbf{I}_{\frac{N}{m}\times\frac{N}{m}} \otimes \mathbf{1}_{1\times M}.$$

Matrix $\mathbf{F}_i$ is a fetch operator that constructs $\mathbf{x}$ from $\mathbf{z}_n$:

$$\mathbf{F}_i = \left[\ \mathbf{0}_{N\times(i-1)}\quad \mathbf{I}_{N\times N}\quad \mathbf{0}_{N\times(C-N-i+1)}\ \right]_{N\times C}.$$

Matrix $\mathbf{H}_{i,m_t}$ is the operator that constructs $\bar{\mathbf{y}}_{i,t}$ by appropriate sampling of $\bar{\mathbf{z}}_t$, and is defined by

$$\mathbf{H}_{i,m} = \left[\ \mathbf{0}_{\frac{N}{m}\times(i-1)}\quad \mathbf{S}_{\frac{N}{m}\times N}\quad \mathbf{0}_{\frac{N}{m}\times(C-N-i+1)}\ \right]_{\frac{N}{m}\times C},$$

where

$$\mathbf{S} = \mathbf{I}_{\frac{N}{m}\times\frac{N}{m}} \otimes \left[\ 1\quad 0\quad 0\quad \cdots\quad 0\ \right]_{1\times m}.$$

In this algorithm, for $t = T$, m_t equals N, and $\bar{\mathbf{x}}_T$ and $\bar{\mathbf{y}}_{i,T}$ each have only one element, and

$$\bar{d}_i = \|\bar{\mathbf{x}}_T - \bar{\mathbf{y}}_{i,T}\|_p^p = |\bar{x}_1 - \bar{y}_{i,1}|^p$$

needs to be compared with D_i. The power p may be removed by noticing that at this resolution instead of comparing $\|\bar{\mathbf{x}} - \bar{\mathbf{y}}_i\|_p^p = |\bar{x}_1 - \bar{y}_{i,1}|^p$ with $D_{J'}$, we may compare $|\bar{x}_1 - \bar{y}_{i,1}|$ with $D_{J'}^{1/p}$.

Also, when $p > 1$, we still may avoid the power p computation in all resolutions, if we note that, again, using Jensen's inequality, it may be shown that

$$\forall \mathbf{v} \in \Re^N, \quad \|\mathbf{v}\|_1 \leq \alpha \|\mathbf{v}\|_p \tag{5}$$

where

$$\alpha = N^{\frac{p-1}{p}}.$$

Then, we have

$$\begin{aligned} \|\bar{\mathbf{x}} - \bar{\mathbf{y}}_I\|_1 &\leq \alpha \|\bar{\mathbf{x}} - \bar{\mathbf{y}}_I\|_p & (6) \\ &\leq \alpha \|\mathbf{A}_m\|_p \|\mathbf{x} - \mathbf{y}_I\|_p & (7) \end{aligned}$$

where (6) is due to (5), and (7) is due to (1). This gives the threshold $D_i' = \alpha D_i$ for $\|\bar{\mathbf{x}} - \bar{\mathbf{y}}_I\|_1$. Obviously,

$$\|\bar{\mathbf{x}} - \bar{\mathbf{y}}_I\|_1 \leq D_i' \tag{8}$$

is a looser bound compared to $\|\bar{\mathbf{x}} - \bar{\mathbf{y}}_I\|_p \leq D_i$, hence resulting in less effective pruning. However, conducting search using (8) requires less computations.

Table 2. Number of candidates tested at each resolution, total number of basic operations, and ratio of total number of basic operations to that of exhaustive search, for the proposed fast algorithm and exhaustive search.

t	Block Size	Number of Candidates Searched	
		Proposed Algorithm	Exhaustive Search
4	1×1	11,441,196	0
3	2×2	3,912,397	0
2	4×4	920,769	0
1	8×8	239,933	0
0	16×16	91,246	11,441,196
Total Number of Basic Ops		8.05×10^7	2.93×10^9
Ratio of # of Basic Ops		0.0275	1.0000

4. EXPERIMENTAL RESULTS

The two-dimensional version of algorithm of Table 1 was used for finding the best match for each block in each frame of video sequences, from a search conducted in its neighborhood in the previous frame.

For the first 30 frames of the gray-scale, 8 bit-per-pixel, 360×288, "salesman" video sequence, with blocks of size 16×16, and search area of 33×33 (W=16), and $p = 2$, the proposed method gives an speed up of more than 36 compared to a full search. Table 2 shows the number of low-resolution searches conducted at each resolution, along with the total number of basic operations.

Note that in this experiment, no coding was conducted on the motion compensation residuals, and for each block, the search was conducted on the previous original frame.

It is interesting to mention that if at the top level with $t = T$, we used D_J rather than $D_{J'}$ for finding the pruning threshold, the obtained speed up would be approximately 26 instead of 36.

5. COMPARISON WITH OTHER METHODS

In this section, we compare the proposed method with the fast method reported by Li and Salari [7]. In their method, which is similar to a fast method for vector quantization, also known as Triangular Inequality Elimination [8, 9], the inequalities

$$\begin{aligned} \|\mathbf{x}\| - \|\mathbf{y}_I\| &\leq \|\mathbf{x} - \mathbf{y}_I\| \\ \|\mathbf{y}_I\| - \|\mathbf{x}\| &\leq \|\mathbf{x} - \mathbf{y}_I\| \\ \|\mathbf{x} - \mathbf{y}_I\| &\leq \|\mathbf{x} - \mathbf{y}_i\| \end{aligned}$$

are combined to yield

$$\|\mathbf{x}\| - \|\mathbf{x} - \mathbf{y}_i\| \leq \|\mathbf{y}_I\| \leq \|\mathbf{x}\| + \|\mathbf{x} - \mathbf{y}_i\|.$$

First, $\|\mathbf{y}_j\|$ is computed for $j = 1, 2, \ldots, K$, as an overhead. Then, any j for which $\|\mathbf{y}_j\|$ does not satisfy the following inequalities is rejected

$$\|\mathbf{x}\| - \|\mathbf{x} - \mathbf{y}_i\| \leq \|\mathbf{y}_j\| \leq \|\mathbf{x}\| + \|\mathbf{x} - \mathbf{y}_i\|,$$

where $\mathbf{y}_i$ is any previously tested candidate.

In this approach, assuming all pixels have positive values, the overhead is equivalent to the overhead of finding $\bar{\mathbf{z}}_T$ in our approach. In one experiment, for the "salesman" sequence, an speed up of approximately 7.6, compared to full search, is reported in [7]. This is in contrast to the 36 times speed up of the method proposed in this paper.

6. CONCLUSIONS

This paper proposes the fastest full-search-equivalent motion estimation method reported in the literature for MAD and MSE criteria. The proposed algorithm is theoretically shown to be equivalent to full search. It uses the information from low resolution matching to prune the number of candidates needed to be searched at full resolution. For a typical case, the proposed method has an speed ratio of approximately 36 to 1 compared to a full search. This speed up is roughly around the same order of magnitude obtained by inexact motion estimation methods.

7. REFERENCES

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

[2] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD: The Johns Hopkins University Press, 3rd ed., 1996.

[3] C.-T. Chen, *Linear System Theory and Design*. New York: Holt, Rinehart and Winston, Inc., 1984.

[4] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1989.

[5] T. Cover and J. Thomas, *Elements of Information Theory*. New York: John Wiley and Sons, Inc., 1991.

[6] A. K. Jain, *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ: Printice Hall, Inc., 1989.

[7] W. Li and E. Salari, "Successive elimination algorithm for motion estimation," *IEEE Transactions on Image Processing*, vol. 4, pp. 105–107, Jan. 1995.

[8] W. Li and E. Salari, "A fast vector quantization encoding method for image compression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, pp. 119–123, Apr. 1995.

[9] S. H. Huang and S. H. Chen, "Fast encoding algorithm for VQ-based image coding," *Electronics Letters*, vol. 26, pp. 1618–1619, Sept. 1990.