Block Matching Algorithm Based on Particle Swarm Optimization

for Motion Estimation

Xuedong Yuan*
College of Computer Science
Sichuan University
Chengdu, Sichuan, 610064, P.R.China
yxdongdong@163.com

Xiaojing Shen
College of Mathematics
Sichuan University
Chengdu, Sichuan, 610064, P.R.China
Xiao23332@163.com

Abstract

In this paper, based on Particle Swarm Optimization (PSO), we propose a fast block matching algorithm for motion estimation (ME) and compare the algorithm with other popular fast block-matching algorithms for ME. A real-world example shows that the block matching algorithm based on PSO for ME is more feasible than others. Moreover, the initial values of parameters in PSO are empirically discussed, since they directly affect the computational complexity. Thus, an improved PSO algorithm for ME is empirically given to reduce computational complexity.

1. Introduction

Today, ME plays a much more important role in real-time video coding and processing systems. And popular video coding standards such as MPEG-1, MPEG-2, MPEG-4, H.263, H.264, usually use block-matching algorithm for ME. In the way, the first, a current frame is partitioned into small blocks called micro block. Second, search an area in the previous frame to find a 'matching' region for each block of the current frame. This is carried out by comparing the block in the current frame with some or all of the regions in the search area called search window which is usually a region centered on the current block position and finding the region that gives the 'best' match. There are various matching criterions, a more popular and less computationally expensive is Mean Absolute Difference (MAD) given by equation (1). Another matching criterion is Mean Squared Error (MSE) given by equation (2) [1].

\[
MAD(i, j) = \frac{1}{MN} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} |S_c(I+n, J+m) - S_p(I+n+i, J+m+j)|
\]  

(1)

\[
MSE(i, j) = \frac{1}{MN} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} [S_c(I+n, J+m) - S_p(I+n+i, J+m+j)]^2
\]  

(2)

where \(S_c(I+n, J+m)\) and \(S_p(I+n+i, J+m+j)\) are the pixels value in the current and previous frames, \(M \times N\) is the size of block, \((I, J)\) represents the coordinates of the upper left corner pixel of the current block and \((i, j)\) is the displacement that is relative to current block located at \((I, J)\).

The most straightforward block matching algorithm is the exhaustive search (ES), which exhaustively searches for the best matching block within the search window. However, ES yields very high computational complexity and makes ME the main bottleneck in real-time video coding applications.

In fact, the ME could be very computational intensive and consume up to about 80% of computational power of the video coding and processing systems [2], [3]. During the past two decades, many fast block-matching algorithms have been proposed to reduce the computation time by searching only a subset of the eligible candidate blocks, These fast block ME algorithms [4] include Three Step Search (TSS), New Three Step Search (NTSS), Simple and Efficient Search (SES), Four Step Search (4SS), Diamond Search (DS), Adaptive Rood Pattern Search (ARPS) [5].

In this paper, we propose a fast block matching algorithm based on PSO for ME and compare the
algorithm with other popular fast block matching algorithms for ME. Moreover we focus on addressing the initial values of parameters in PSO, since they directly affect the computational complexity. Thus, an improved PSO algorithm for ME is empirically given to reduce computational complexity.

The rest of this paper is organized as follows: Section 2 provides a preliminary knowledge for particle swarm optimization. Section 3 presents a fast block matching algorithm based on PSO, and compares the computation times per block and the MAD per pixel of the algorithm with other popular fast block-matching algorithms. In Section 4, the initial values of parameters in PSO are discussed and an improved PSO algorithm is presented.

2. Particle swarm optimization

Particle swarm optimization (PSO) is a population-based, self-adaptive search optimization technique first introduced by Kennedy and Eberhart [6] in 1995. It is unlike the most of other population-based evolutionary algorithms; however, the PSO is motivated by the simulation of social behavior instead of survival of the fittest. It is a stochastic optimization technique that can be likened to the behavior of a flock of birds or the sociological behavior of a group. Instead of using evolutionary operators to manipulate the individuals, like in other evolutionary computational algorithms, each individual in PSO flies in the search space with a velocity which is dynamically adjusted according to its own flying experience and its companions’ flying experience. And each individual is treated as a volume-less particle (a point) in the D-dimensional search space. The ith particle is represented as \( S_i = (S_{i1}, S_{i2}, \ldots, S_{id}) \). The best previous position of the ith particle is recorded and represented as \( P_i = (P_{i1}, P_{i2}, \ldots, P_{id}) \), also represented as symbol \( P_{best} \). The index of the best particle among all the particles in the population is represented by the symbol \( g \) and the best previous position among all the particles in the population is represented by the symbol \( P_g \). The rate of the position change (velocity) for particle \( i \) is represented as \( V_i = (V_{i1}, V_{i2}, \ldots, V_{id}) \). The ith particle is manipulated according to the following equations:

\[
V_{id} = w \times V_{id} + C_1 \times \text{Rand}(i) \times (P_{id} - X_{id}) \\
\quad + C_2 \times \text{Rand}(i) \times (P_g - X_{id}), \quad (3)
\]

\[
X_{id} = X_{id} + V_{id}, \quad (4)
\]

where \( C_1 \) and \( C_2 \) are two positive constants, and \( \text{Rand}(i) \) and \( \text{Rand}(i) \) are two random functions in the range [0,1]. Variable \( w \) is the inertia weight.

The original PSO does not have an inertia weight \( w \). Shi and Eberhart [7] introduced the concept of inertia weight to the original version of PSO, in order to balance the local and global search during the optimization process. Its value is typically setup to vary linearly from 1 to near 0 during the course of a training run.

Acceleration coefficients \( C_1 \) and \( C_2 \) also control how far a particle will move in a single iteration. Typically, these are both set to a value of 2.0 [8].

The first part of equation (3) represents the previous velocity, which provides the necessary momentum for particles to roam across the search area. The second part, known as the “cognitive” component, represents the personal thinking of each particle. The cognitive component encourages the particles to move toward their own best positions found so far. The third part is known as the “social” component, which represents the collaborative effect of the particles in finding the global optimal solution. The social component always pulls the particles toward the global best particle found so far.

The flow of the PSO algorithm is shown as follows.

**Step 1**: Initialize a swarm with random position and velocity vectors.

**Step 2**: Evaluate the fitness of each particle.

**Step 3**: For each particle, compare its fitness with the best previous position \( P_{best} \) which has flown. If better then change the \( P_{best} \).

**Step 4**: For each particle, compare its fitness with the best global previous position, if better then change the \( P_g \).

**Step 5**: The new velocities and the positions of the particles for the next fitness evaluation are calculated using equations (3), (4).

**Step 6**: Do not reach the stop criteria, and then go against to the Step 2.

3. Block Matching Algorithm Based on PSO for ME

3.1 Applying the PSO algorithm to ME

The processing of block matching is looking for the best position within the search window, in which a point of the minimum of MAD needs to be found. In order to reaching a better MAD, the more positions within the search window will be matched; however, the more computation times will be spent on searching. A better matching algorithm should spend less computation time on searching and obtain the better position. In this paper, the aim of the application of the PSO algorithm to ME is to accelerate matching search and reach a better ME.
The block matching algorithm based on PSO for ME is summarized as follows.

**Step 1**: A population of particles is generated with random positions within the searching window in the previous frame, the search area called search window which is usually a region centered on the current block position; and then random velocities are assigned to each particle.

**Step 2**: The fitness of each particle is then evaluated according to the objective function. In the processing of block matching, we choose the MAD as the matching criterion. In the PSO algorithm for ME, evaluating the fitness of each particle is calculating the block’s MAD.

**Step 3**: Each time if a particle finds a better position than the previous found which location was stored in memory. Then update the $P_{best}$.

**Step 4**: Each time if a particle finds a position is better than the global position, then update it.

**Step 5**: At each generation, the velocity of each particle is calculated according to (3) and the position is updated according to (4). Generally, a maximum velocity ($V_{max}$) for each modulus of the velocity vector of the particles $V_{id}$ is defined in order to controlling excessive roaming of particles outside the searching window. Whenever a $V_{id}$ exceeds the defined limit, its velocity is set to $V_{max}$, and the position is beyond the searching window, its position is set to the border of the searching window.

**Step 6**: Termination criteria. If the number of iteration equals to the maximum($I_{max}$), or MAD of the block less than a given small number $\varepsilon$, then iteration terminate; Otherwise go back to step 2.

### 3.2 Example

To illustrate the performance and feasibility of our algorithm, we consider an example of video sequence (PAL 25 frame/second 720x560) (see Fig.1). Fig. 2 gives the motion vector which ME was computed based on PSO by Matlab software (Block size: 16x16; Search window size:30). We also compare computation times per block and MAD per pixel of the block matching algorithm based on PSO with those of the popular fast block ME algorithms for the same video sequence. The results are proposed in Table 1. From Table 1, we can see, the Exhaustive Search (ES) has the best MAD, but the most accuracy comes at the highest computational complexity. The table also shows that the block matching algorithm based on PSO is a fast and efficient algorithm.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>ES</th>
<th>TSS</th>
<th>NTSS</th>
<th>SES</th>
<th>FSS</th>
<th>DS</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>computations</td>
<td>214.46</td>
<td>24.12</td>
<td>23.62</td>
<td>16.18</td>
<td>20.21</td>
<td>20.05</td>
<td>13.94</td>
</tr>
<tr>
<td>MAD</td>
<td>10.81</td>
<td>10.90</td>
<td>13.21</td>
<td>11.23</td>
<td>11.00</td>
<td>10.99</td>
<td>11.92</td>
</tr>
</tbody>
</table>
4. Block Matching Algorithm based on Improved PSO for ME

In this section, we can improve the block matching algorithm based on PSO for ME from the following two aspects.

4.1. Discussion of the parameter of inertia weight

The PSO algorithm is a simple but effective algorithm. In order to improving the result of the PSO, we can adjust those parameters in PSO.

Acceleration coefficients $C_1$ and $C_2$ can also control how far a particle will move in a single iteration. Typically, both of them are initialized as 2.0 [8].

The inertia weight $w$ is used to balance the local and global search during the optimization process. Its value is typically setup to vary linearly from 1 to near 0 during the course of a training run. A large inertia weight facilitates a global search while a small inertia weight facilitates a local search. By changing the inertia weight dynamically, the search ability is dynamically adjusted [9]-[11]. In order to reaching global optimization, we give a large value to the inertia weight $w$.

4.2. Discussion of the initial positions and velocity of particles in PSO

In Step 1 of the block matching algorithm based on PSO for ME, a population of particles is generated with random positions. In the block-matching algorithm, considering video sequence with center-biased and spatial correlation feature, the searching window is centered on the current block position. Center-biased feature denotes that match point may existed within a small zone around block center. if we place the particles around the optimal position, that can speed up convergence of the algorithm. So we initialize the particle with position around the center.

In most cases, adjacent blocks have the similar motions, which belong to the same moving object. Accordingly, the current block’s motion behavior can be predicted by referring to its neighboring blocks’ motion vector [5]. So we can initialize the velocity of particle of current block with the motion vector of previous adjacent block. This way uses the motion vector of the previous block (it is immediate left to the current block) to predict its own motion vector.

Table 2 gives the computation times per block and the MAD per pixel of the PSO and the Improved PSO. Improved PSO has a better result.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>PSO</th>
<th>Improved PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>computations</td>
<td>13.94</td>
<td>13.34</td>
</tr>
<tr>
<td>MAD</td>
<td>11.92</td>
<td>11.32</td>
</tr>
</tbody>
</table>

5. Conclusions

We have presented the fast block matching algorithm based on PSO for ME. The comparison results of the example in Section 3 show that the algorithm is more feasible than other popular algorithms. In fact, it has the advantage of balancing global search and local search, and avoiding trapping into local minima. In addition, initial values of parameters in PSO are discussed, then we obtain the faster block matching algorithm based on improved PSO for ME which not only succeeds the advantages of the fast block matching algorithm based on PSO for ME, but also uses the information of the adjacent blocks, which yields the fast block matching algorithm based on improved PSO for ME with less computational complexity. Thus, it is probably the algorithm of choice for practical applications.

Acknowledgment

This work is supported by younger science foundation of College of computer Science, Sichuan University.

References


