A Novel Block Matching Algorithm Based on Cat Swarm Optimization for Efficient Motion Estimation

Prof. Dr. Israa Hadi, Mustafa Sabah

Abstract

Motion estimation is regarded as vital importance and one of the major problems in developing video coding applications. Among all motion estimation approaches, Block matching (BM) algorithms are the most popular methods due to their effectiveness and simplicity for both software and hardware implementations. A BM approach assumes that the movement of pixels within a defined region of the current frame can be modeled as a translation of pixels contained in the previous frame. When designing search methods for motion estimation, both the estimation accuracy and computational complexity are the main considerations. The existing fast search methods can provide satisfactory result at a very low computational cost and none of these methods can effectively jump out of the local optimum when processing large motion sequences.

To improve the performance of motion estimation in video coding, a novel block matching algorithm will be proposed, which utilize the global search ability of Cat swarm optimization (CSO) that is a new evolutionary soft computing tool has been proposed to reduce the number of search locations in the BM process and serves as an unbiased guide for solving various optimization problems. Since the proposed algorithm does not consider any fixed search pattern or other different assumption, a high probability for finding the true minimum (accurate motion vector) is expected without the degradation of the image quality. In comparison to other fast BM algorithms, the proposed method deploys more accurate motion vectors yet delivering competitive time rates and the computational complexity of the proposed algorithm has negligible increase.

Keywords: Optimization, Block matching algorithms, Cat swarm optimization, motion estimation.

1. Introduction

The moving object tracking in video pictures has attracted a great deal of interest in computer vision. The aim of object tracking and detection is to establish a correspondence between objects or object parts in consecutive frames and to extract temporal information about objects such as trajectory, posture, speed and direction. Tracking detected objects frame by frame in video is a significant and difficult task [1].

Motion Estimation (ME) is an important part of video tracking system, since it can achieve significant compression by exploiting the temporal redundancy that commonly exists in a video sequence. Several ME methods have been studied seeking for a complexity reduction at video coding such as block matching (BM) algorithms, parametric-based models [2], optical flow [3] and percussive techniques [4]. Among such methods, BM seems to be the most popular technique due to its effectiveness and simplicity for both software and hardware implementations [5]. In order to reduce the computational complexity in ME, many BM algorithms have been proposed and employed at implementations for several video compression standards such as MPEG-4 [6] and H.264 [7].

In BM algorithms, the video frames are partitioned into non overlapping blocks of pixels. Each block is predicted from a block of equal size in the previous frame. In particular, for each block at the current frame, the algorithm aims for the best matching block within a search window from the previous frame, while minimizing a certain matching metric such as sum of absolute differences (SAD), Mean Absolute Difference (MAD) and Mean Squared Error (MSE) which is given by equation (1).
where $Sc(I+n, J+m)$ and $Sp(I+n+i, J+m+j)$ are the pixel values 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 best matching block thus represents the predicted block, whose displacement from the previous block is represented by a transitional motion vector (MV) as seen in figure 1.

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

Figure 1. Block matching concept

In order to decrease the computational complexity of the BM process, several BM algorithms have been proposed considering the following three techniques:

1. Using a fixed pattern: the search operation is conducted over a fixed subset of the total search window. The Three Step Search (TSS) [8], the New Three Step Search (NTSS) [9], the Simple and Efficient TSS (SES) [10], the Four Step Search (4SS) [11] and the Diamond Search (DS) [12], all represent some of its well-known examples. Although such approaches have been algorithmically considered as the fastest, they are not able to eventually match the dynamic motion-content, sometimes delivering false motion vectors (image distortions).

2. Reducing the search points: the algorithm chooses as search points only those locations that iteratively minimize the error-function (SAD values). This category includes the Adaptive Rood Pattern Search (ARPS) [13], the Fast Block Matching Using Prediction (FBMAUPR) [14], the Block-based Gradient Descent Search (BBGD) [15] and the Neighborhood Elimination algorithm (NE) [16]. Such approaches assume that the error-function behaves monotonically, holding well for slow-moving sequences but failing for other kind of movements in video sequences [17], making the algorithm prone to get trapped into local minima.

3. Decreasing the computational overhead for every search point: the matching cost (MSE operation) is replaced by a partial or a simplified version that features less complexity. The New pixel-Decimation (ND) [18], the Efficient Block Matching Using Multilevel Intra, the Inter-Sub-blocks [9] and the Successive Elimination Algorithm [19], all assume that all pixels within each block, move by the same finite distance and a good estimate of the motion can be obtained through only a fraction of the pixel pool. However, since only a fraction of pixels enters into the matching computation, the use of such regular sub-sampling techniques can seriously affect the accuracy of the detection of motion vectors due to noise or illumination changes.

Another popular group of BM algorithms employ spatiotemporal correlation by using neighboring blocks in the spatial and temporal domain in order to predict MVs. The main advantage of such algorithms is that they alleviate the local minimum problem to some extent as the new initial or predicted search center is usually closer to the global minimum and therefore the chance of getting trapped in a local minimum decreases. This idea has been incorporated by many fast-block motion.
estimation algorithms such as the Enhanced Predictive Zonal Search (EPZS) [20]. However, the information delivered by the neighboring blocks occasionally conduces to false initial search points producing distorted motion vectors. Such problem is typically caused by the movement of very small objects contained in the image sequences [21].

Therefore, BM is essentially an optimization problem whose goal is to find the best matching block within a search space, evolutionary approaches such as genetic algorithms (GA) [22] and particle swarm optimization (PSO) [23] are well known for delivering the location of the global optimum in complex optimization problems. Despite of such fact, only few evolutionary approaches have specifically addressed the problem of BM, such as the Light-weight Genetic Block Matching (LWG) [24], the Genetic Four-step Search (GFSS) [25] and the PSO-BM [26].

In this paper, a new algorithm based on CSO is proposed to reduce the number of search locations in the BM process. The cat swarm optimization (CSO) algorithm is relatively recent addition to the family of algorithms known as swarm intelligence and is based on the common behavior of cats. The algorithm uses a simple fitness calculation which is (MSE operation) for several candidate solutions (search locations). The proposed method achieves the best balance over other fast BM algorithms, in terms of both estimation accuracy and computational complexity. The overall paper is organized as follows: Section 2 summarizes studies that are related to the proposed algorithm. In Section 3 holds a brief description about the CSO algorithm. Section 4 provides backgrounds about CSO movement while Section 5 exposes the final BM algorithm as a combination of CSO. Section 6 demonstrates experimental results for the proposed approach over tested sequences and some conclusions are drawn in Section 7.

2. Related work

Enhancement the Block matching technique in Tracking is an important topic in computer vision and it has been studied for several decades. In this section some studies that related to proposed algorithm have been summarized below:

Zhang Ping, Wei Ping, Yu Hongyang [27] proposed a novel block matching algorithm, which utilize the global search ability of particle swarm optimization (PSO) with mutation operator and the local search ability of simplex method (SM), In order to accelerate the convergence of PSO and improve the accuracy of local search, mutation operator and simplex method are used. Meanwhile, based on the feature of static macro blocks, the proposed algorithm intelligently uses early termination strategies. Experimental results demonstrate that the proposed algorithm has better PSNR values than conventional fast block matching algorithms.

Erik Cuevas, Daniel Zaldívar, Marco Pérez-Cisneros and Diego Oliva [28] proposed a new algorithm based on Differential Evolution (DE) to reduce the number of search locations in the BM process. In order to avoid computing several search locations, the algorithm estimates the SAD values (fitness) for some locations using the SAD values of previously calculated neighboring positions. Since the proposed algorithm does not consider any fixed search pattern or other different assumption, a high probability for finding the true minimum (accurate motion vector) is expected.

L.Koteswara Rao, Dr.D.Venkata Rao [29] studies the correlation between local statistical characteristics, scene duration and scene change. Based on this analysis, have been proposed a scene change algorithm for H.264 codec, and an automated, dynamic threshold model with fast motion estimation algorithm having low complexity which can efficiently trace out scene changes. Chunfei, Ping Zhang, Jian Li [30] proposed a new search algorithm based on artificial fish-swarm algorithm (AFSA) which is a new efficient optimizing method. Firstly, some characteristics of AFSA are modified, such as visual, step, moving direction and initial fish positions in order to better adapt to motion estimation. Secondly, an adaptive search strategy based on modified AFSA Algorithm is presented to further reduce computational complexity, including double search mode, dynamic search range and early termination strategy.

Erik Cuevasa, Daniel Zaldivara, Marco Pérez-Cisnerosa, Humberto Sossab and Valentin Osuna [31] Proposed a new algorithm based on Artificial Bee Colony (ABC) optimization to reduce the number of
search locations in the BM process. In this study, the computation of search locations is drastically reduced by considering a fitness calculation strategy which indicates when it is feasible to calculate or only estimate new search locations.

Improved particle filter based on genetic algorithm (GA) is proposed in [32]. This study introduces genetic Monte Carlo sampling method, and then uses it in the re-sampling step of particle filter with the basic idea of solving particle degeneration; also combining particle filter with ant colony optimization is used in video object tracking, which reduces the size of sample set and effects of the problems of PF.

3. Cat Swarm Optimization (CSO)

Swarm Intelligence (SI) is a novel artificial intelligence approach inspired by the swarming behaviors of groups of organisms such as ants, termites, bees, birds, fishes in foraging and sharing the information with each other. SI focuses on the collective intelligence of a decentralized system consisting of a group of organisms interacting with each other and their environment. So, by means of their collective intelligence swarms are able to effectively use their environment and resources. SI is also a mechanism that enables individuals to overcome their cognitive limitations and solve problems which are difficult for individuals to resolve alone. Swarm intelligence algorithms are essentially stochastic search and optimization techniques and were developed by simulating the intelligent behavior of these organisms. These algorithms are known to be efficient, adaptive, robust, and produce near optimal solutions and utilize implicit parallelism approaches [33].

One of the more recent optimization algorithm based on swarm intelligence is the Cat Swarm Optimization (CSO) algorithm. The CSO algorithm was developed based on the common behavior of cats. It has been found that cats spend most of their time resting and observing their environment rather that running after things as this leads to excessive use of energy resources. To reflect these two important behavioral characteristics of the cats, the algorithm is divided into two sub-modes and CSO refers to these behavioral characteristics as —seeking model and —tracing model, which represent two different procedures in the algorithm. Tracing mode models the behavior of the cats when running after a target while the seeking mode models the behavior of the cats when resting and observing their environment [34].

Furthermore, previous researches have shown that the CSO algorithm has a better performance in function minimization problems compared to the other similar optimization algorithms like Particle Swarm Optimization (PSO) and weighted-PSO [35]. Cat Swarm Optimization algorithm has two modes in order to solve the problems which are described below:

3.1. Seeking Mode: Resting and Observing

For modeling the behavior of cats in resting time and being-alert, the seeking mode will be used. This mode is a time for thinking and deciding about next move. This mode has four main parameters which are mentioned as follow:

Seeking memory pool (SMP), seeking range of the selected dimension (SRD), counts of dimension to change (CDC) and self-position consideration (SPC)[36]. The process of seeking mode is described as follow:

**Step1:** Make j copies of the present position of cat, where j = SMP. If the value of SPC is true, let j = (SMP-1), then retain the present position as one of the candidates.

**Step2:** For each copy, according to CDC, randomly plus or minus SRD percent the present values and replace the old ones.

**Step3:** Calculate the fitness values (FS) of all candidate points.

**Step4:** If all FS are not exactly equal, calculate the selecting probability of each candidate point by (2); otherwise set all the selecting probability of each candidate point is 1.

**Step5:** Randomly pick the point to move to from the candidate points, and replace the position of cat.
If the goal of the fitness function is to find the minimum solution, $FS_b = FS_{\text{max}}$, otherwise $FS_b = FS_{\text{min}}$.

### 3.2. Tracing Mode: Running After a Target

Tracing mode is the second mode of the algorithm. In this mode, cats desire to trace targets and foods. The process of tracing mode can be described as follows:

**Step 1:** Update the velocities for every dimension according to (3).

$$V_{k,d} = V_{k,d} + r_1 c_1 (X_{\text{best},d} - X_{k,d}) \tag{3}$$

**Step 2:** Check if the velocities are in the range of maximum velocity. In case the new velocity is over-range, it is set equal to the limit.

**Step 3:** Update the position of cat $k$ according to (4).

$$X_{k,d} = X_{k,d} + V_{k,d} \tag{4}$$

$X_{\text{best},d}$ is the position of the cat who has the best fitness value, $X_{k,d}$ is the position of cat $k$, $c_1$ is an acceleration coefficient for extending the velocity of the cat to move in the solution space and usually is equal to 2.05 and $r_1$ is a random value uniformly generated in the range of $[0,1]$.

### 4. CSO Movement = Seeking Mode + Tracing Mode

When applying the CSO algorithm to solve optimization problems, the initial step is to make a decision on the number of individuals or cats to use. Each cat in the population has the following attributes:

a) A position made up of $M$ dimensions;
b) Velocities for each dimension in the position;
c) A fitness value of the cat according to the fitness function; and
d) A flag to indicate whether the cat is in seeking mode or tracing mode.

The CSO algorithm keeps the best solution after each cycle and when the termination condition is satisfied, the final solution is the best position of one of the cats in the population. CSO has two sub-modes, namely seeking mode and tracing mode and the mixture ratio $MR$ dictates the joining of seeking mode with tracing mode. To ensure that the cats spend most of their time resting and observing their environment, the $MR$ is initialized with a small value.

The CSO algorithm can be described in 6 steps as presented in [38]:

**Step 1:** Create $N$ cats in the process.

**Step 2:** Randomly sprinkle the cats into the $M$-dimensional solution space and randomly give values, which are in-range of the maximum velocity, to the velocities of every cat. Then haphazardly pick number of cats and set them into tracing mode according to $MR$, and the others set into seeking mode.

**Step 3:** Evaluate the fitness value of each cat by applying the positions of cats into the fitness function, which represents the criteria of our goal, and keep the best cat into memory. Note that the position of the best cat ($x_{\text{best}}$) will be remembered because it represents the best solution so far.

**Step 4:** Move the cats according to their flags, if cats is in seeking mode, apply the cat to the seeking mode process, otherwise apply it to the tracing mode process.

**Step 5:** Re-pick number of cats and set them into tracing mode according to $MR$, then set the other cats into seeking mode.

**Step 6:** Check the termination condition, if satisfied, terminate the program, and otherwise repeat Step 3 to Step 5.
5. Block Matching Algorithm Based on CSO for ME

The processing of block matching is looking for the best position within the search window, in which point of the minimum of MSE needs to be found. In order to reaching a better MSE, 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 CSO algorithm to ME is to accelerate matching search and reach a better ME.

The block matching algorithm based on CSO for ME is summarized as follows:

5.1. Initialization

A given frame is divided into 16x16 macro blocks. Then create \( N \) virtual cats and randomly spread them into the \( M \)-dimensional solution space for each MB and randomly give values, which are in-range of the maximum velocity, to the velocities of every cat. Then haphazardly pick number of cats and set them into tracing mode according to MR=20%, and the others (80%) set into seeking mode. Each cat of a given MB represents a matching MB within the search window in the reference frame. Using the CSO iterations, the positions of the cat is continuously updated until the global minimum of the Mean Squared Error (MSE) function is reached.

In the standard CSO algorithm, the initial population is randomly selected, which brings high computational complexity to the motion search since the iterations are starting from random points which might be far from the global minimum. However, if the initial points are chosen to be close to the optimum, then faster convergence can be achieved. Since motion vectors have a high temporal correlation feature, initialization 9 cats of each MB to the MVs of the collocated MB in the previous frame as well as its 8 adjacent neighbors has been done. Also initialization one of the cat to the (0, 0) MV to account for static blocks has been assumed. The rest of the N cats are randomly generated. Notice that at this point, the MVs of the adjacent blocks in the same frame cannot be used since these
MVs are not calculated yet and the only a priori information is the motion of the MBs of the previous frame will be had.

5.2. Evaluation

After initialization, the swarms of cats of all MBs are allowed to run for a predefined K number of iterations in parallel. During each iteration, each MB with index j adjusts the positions and velocities of its cats, independently from other MBs, evaluates the fitness function at the new positions, then it updates the values of Cat_{ij} and Cat_{gj} which are the position of the best fitness attained so far for cat i and the global best position for MB_j respectively.

Calculate the fitness value of each virtual cat by applying the positions of cats into the fitness function (which represents the criteria of this study) and check whether current coordinate present better fitness value then its memorized best solution. The memorized best solution should be updated, accordingly, when the current fitness value is better than the memorized one. Note that only need to remember the position of the best cat (Cat_best) due to it represents the best solution so far. In the processing of block matching, the MSE as the matching criterion will be chosen. In the CSO algorithm for ME, evaluating the fitness of each cat is calculating the block’s MSE.

5.3. Movement

Move the virtual cats in the solution space with the seeking/tracing mode according to the decision made in the initialization step A. The seeking mode and the tracing mode process are described below.

5.3.1. The seeking mode

Seeking mode is intended to look for points in an area around the initial points that chosen in initialization steep which have possibilities resulting a more optimal fitness value. Seeking mode starts with making SMP copy of the present position. Then define j value, where j value represents how many copy of selection point i that will experience mutation. If the value of SPC = 1, let j = (SMP – 1) then retain the present position as one of the candidates.

The next step will be calculating the mutative value, which is:

\[(SRD \times \text{selected point}) \pm SRD\] percent of current value on the selected dimensions:

\[
\text{cat: } \text{cat} (\text{it}) = \text{cat} (\text{it}) - SRD, \text{cat} (\text{it}) = \text{cat} (\text{it}) + SRD
\]

This step will give \((SMP \times k)\) candidates of points as the output. For every point candidates determine the fitness function. After the MSE value has been got, calculate the selecting probability of each candidate point based on their MSE value.

Pick the new point from the candidate points by using Roulette Wheel Selection method. Candidate with the biggest cat value will have the biggest opportunity to be chosen.

5.3.2. The tracing mode

The tracing mode allows the virtual cats imitating the movement of tracing the prey. Tracing mode is intended to shift the point so it will be concentrated to a better position with a more optimal fitness value. The process of the tracing mode can be disassembled in 2 steps:

1. For \(i = 1\) to \(k\), do
   Update velocity \((\text{i})\)
   Update position \((\text{i})\), get the new point \((\text{i})\)
2. Calculate MSE

Generally, a maximum velocity \(V_{\text{max}}\) for each modulus of the velocity vector of the cats \(V_{\text{id}}\) is defined in order to controlling excessive roaming of cats outside the searching window. Whenever a \(V_{\text{id}}\) exceeds the defined limit, its velocity is set to \(V_{\text{max}}\), and the position is beyond the searching window, its position is set to the border of the searching window.
5.4. Permutation

Re pick (N X MR) virtual cats pending to be processed in the tracing mode in the next iteration.

5.5. MB Synchronization

After the K iterations are completed by all MBs of the frame, a synchronization step is performed to refine the MVs found so far in the CSO process. This is done by exploiting the high spatial correlation existing between MVs of neighboring blocks. To do that, each MB \( j \) sorts its N cats in a decreasing order according to their cat\(_{ij} \) values. Then the last 8 cats which have the worst cat\(_{ij} \) values are eliminated and replaced by 8 new cats which are initialized to the Pg values of its 8 neighboring MBs. In this synchronization step, neighboring MBs are allowed to refine their motion search process using information from neighboring blocks. Weak cats having the worst fitness values are replaced with strong cats which are located closer to the global optimum. This process is expected to speed up the convergence of the CSO algorithm.

Communication between neighboring MBs is required in this step where each MB will broadcast to its 8 neighbors the value of its global best location cat\(_g \) found so far in the motion search process.

5.6. Termination criteria

After each set of K iterations, a synchronization step is performed between neighboring MBs. The whole process is terminated when a predefined number of total iterations satisfies the termination condition (If the number of iteration equals to the maximum I\(_{\text{max}} \), or MAD of the block less than a given small number \( \varepsilon \), then iteration terminate). If the process is terminated, output the coordinate representing the near best solution and stop the program. Otherwise, go back to evaluation step. Note that the process, however, is allowed to terminate early for a given MB\(_j \) whenever cost function value of its best position falls below the threshold Tth.

6. Simulation Results

To illustrate the performance and feasibility of proposed algorithm, an example of video sequence (AVI 25 frame/second 720x576) can be considered (see Fig.3). Fig.4 gives the selected motion object which ME was computed based on CSO by Matlab software (Block size: 16x16; Search window size: 30). Also the compare computation times per block and MSE per pixel of the block matching algorithm based on CSO with PSO for the same video sequence have been implemented. The results are proposed in Table1. The table shows that the block matching algorithm based on CSO is a fast and efficient algorithm.

Figure 3. Tested sequence
Table 1. Computation times per pixels and the MSE per pixel

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>PSO</th>
<th>CSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computations</td>
<td>13.94</td>
<td>12.92</td>
</tr>
<tr>
<td>MSE</td>
<td>11.92</td>
<td>11.12</td>
</tr>
</tbody>
</table>

6.1. Discussion of the initial positions and velocity of cats in CSO

In the first step of the block matching algorithm based on CSO for ME, a population of cats 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 place the cats around the optimal position, that can speed up convergence of the algorithm. So initialization the cat with position around the center has been done.

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. So the velocity of cat of current block with the motion vector of previous adjacent block can be initialization. This way uses the motion vector of the previous block (it is immediate left to the current block) to predict its own motion vector.

6.2. Average searching points

The average searching points of CSO algorithms are much less than that of FS, and a little more than that of TSS and DS. The reason is that CSO algorithm takes more times for each iteration while dramatically decrease the overall cycles.

6.3. Analysis of the Computational Complexity and Speedup of the Proposed Scheme

The computational complexity of this ME approach depends on the number of fitness function evaluations performed. This is directly related to the population size $M$ and the maximum number of total iteration $N_{\text{max}}$ allowed. Theoretically, a maximum of $M \times N_{\text{max}}$ cost function evaluations required for each MB will be had. The value of $M$ is at least equal to 10 and $N_{\text{max}}$ should be large enough to guarantee good estimation accuracy. Nevertheless, because this approach exploits the spatial and temporal correlations of the motion vectors, continuously refines the motion search process through cat mutation, and allows an early termination condition for the MBs, it is estimated that the CSO algorithm will converge to the global optimum before $N_{\text{max}}$ is reached. Moreover,
6.4. Parameter Settings for CSO

The parameter settings for CSO parameters are listed in Table 2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value or Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMP</td>
<td>5</td>
</tr>
<tr>
<td>SRD</td>
<td>20%</td>
</tr>
<tr>
<td>CDC</td>
<td>80%</td>
</tr>
<tr>
<td>MR</td>
<td>20%</td>
</tr>
<tr>
<td>C</td>
<td>2.0</td>
</tr>
<tr>
<td>R</td>
<td>[0,1]</td>
</tr>
</tbody>
</table>

7. Conclusion

Block-matching algorithm is very popular for video coding and the motion estimation method has a critical impact on the efficiency of block-matching algorithm. Thus, in this paper, a novel adaptive block-matching algorithm based on Cat Swarm Optimization (CSO) is proposed to reduce the number of search locations in the BM process without the degradation of the image quality. Since the proposed algorithm does not consider any fixed search pattern or any other movement assumption, a high probability for finding the true minimum (accurate motion vector) is expected regardless of the movement complexity contained in the sequence, yet the CSO approach is capable of achieving high accuracy in block matching. Therefore, the chance of being trapped into a local minimum is reduced in comparison to other BM algorithms. The CSO is a new soft computing tool with faster convergence by marinating the good quality of solution and promising improvement in terms of accuracy, while drastically reducing the computational complexity. Experimental results demonstrate the high performance of the proposed method in terms of computational complexity, finding the global best solution, faster convergence and estimation accuracy. This work can be further extended by using dynamic search window adjustment in order to reduce the computational complexity and also decrease the complexity in cat swarm optimization.

8. References


[30] Chun Fei, Ping Zhang, Jianping Li,“ Motion Estimation based on Artificial Fish-Swarm in H.264/AVC Coding”, WSEAS TRANSACTIONS on SIGNAL PROCESSING, Volume 10, 2014.


