An Improved Block Matching Algorithm for Motion Estimation in Video Sequences and Application in Robotics

Kamanasish Bhattacharjee[a], Sushil Kumar[b], Hari Mohan Pandey[c*], Millie Pant[d], David Windridge[c], Ankit Chaudhary[e]

[a] *Department of Computer Science & Engineering, Amity University, Noida, Uttar Pradesh, India*
[b]*School of Computing Science and Engineering, Galgotias University, Greater Noida, India*
[*c] *Department of Computer Science, School of Technology, Middlesex University, London, UK*
[d]*Department of Applied Science and Engineering, IIT Roorkee, India*
[e] *Data Science Division, Department of Computer Science, Northwest Missouri State University, USA*

ARTICLE INFO

Article history:
Received
Received in revised form
Accepted
Available online

Keywords:
Block Matching
Differential Evolution
Harmony Search
Robotics
Motion Estimation
Video Compression

ABSTRACT

Block Matching is one of the most efficient techniques for motion estimation for video sequences. Metaheuristic algorithms have been used effectively for motion estimation. In this paper, we propose two hybrid algorithms: Artificial Bee Colony with Differential Evolution and Harmony Search with Differential Evolution based motion estimation algorithms. Extensive experiments are conducted using four standard video sequences. The video sequences utilized for experimentation have all essential features such as different formats, resolutions and number of frames which are generally required in input video sequences. We compare the performance of the proposed algorithms with other algorithms considering various parameters such as Structural Similarity, Peak Signal to Noise Ratio, Average Number of Search Points etc. The comparative results demonstrate that the proposed algorithms outperformed other algorithms.

1. Introduction

BM is important for motion estimation in video compression where frames of a video sequence are divided into macro blocks. For each block in the current frame, the best matching block is identified in the search space of the previous frame to minimize the MAD or MSE or SAD between blocks. The key challenge is the evaluation of SAD/MAD/MSE as it is highly time consuming. Hence, BM for motion estimation is considered as an optimization problem and it has an objective to search the best matching block for a target block. There exist various approaches that were introduced to speed up BM through a fixed subset of the search area at the cost of deficient accuracy. Some of the approaches are: 3SS [2], SESTSS [4], NTSS [3], 4SS [5], DS [6], ARPS [7]. These approaches were found effective, but they failed to establish a trade-off between accuracy and speed.

Lin et al. [8] proposed a BM algorithm using GA. It was an extension of 3SS. The experimental results demonstrated that LGA performed better than ES or FSA, 3SS and M3SS. So et al. [9] proposed 4GS by combining GA and 4SS. It requires less number of search points than the LGA, but more number of search points than 4SS. It takes approximately 14% of search points compared to FSA. Li et al. [10] suggested a BM algorithm based on an improved GA, where an objective search and random search derived from genetic mutation are utilized to search the global optimum and a threshold selection operator is applied to speed up the estimation. Li et al. [10] utilized GA to reduce the high computational complexity.

A fair amount of research has been conducted on BM algorithm utilizing PSO. Du et al. [11] proposed a BM algorithm which was based on PSO and it operates faster than the GA. Ren et al. [12] presented a PSO-ZMP algorithm. It consists of ZMP, predictive image coding and PSO matching routine. Though it produces positive results in terms of computational complexity as compared to the DS and ARPS, at the same time it generated negative trends in terms of quality. Yuan et al [13] utilized an improved PSO for BM through a centre-biased particle initialization and neighbor based velocity initialization. Bakwad et al. [15] implemented a BM algorithm on a SPMPPSO, it was computationally faster. SPSO for BM was proposed by Zhang et al. [14]. It combines the high accurate local search ability of SPSO with the powerful global search ability of the PSO. It demonstrated the ability to avoid the local minima sticking problem. Cai et al. [16] proposed a fast and accurate BM algorithm was based on PSO using time variant acceleration coefficients. The time variant acceleration coefficient helps in exploration in the early stage and converges to a good solution. Jalloul et al. [17] suggested a BM algorithm using an improved parallel PSO. The improved parallel PSO incorporates synchronization that helps the neighboring macro-blocks of frame to exchange information about the motion vectors. This process allows exploiting the spatial correlation between adjacent blocks and it speed up the convergence. Liu et al. [18] formulated a technique for BM through a PSO, was based on a *Good-Point* set theory to reduce the deviation of the two random numbers selected in velocity updating formula. Good-point set theory helps in the selection of the better points than the random

selection, which accelerates the convergence. Britto et al. [19] applied a combination of a PSO and AMEA to reduce the computational complexity and search points. A cooperative motion estimation algorithm based on multi-warm PSO was proposed by Jalloul et al. [20]. In this method, information exchange about the motion vectors was found effective in exploiting spatial correlation, refining the motion search and, therefore, leads to a faster convergence and demonstrated improvement in the resulting motion vectors. Cuevas et al. [21] implemented ABC, DE and HS respectively along with a fitness estimation strategy for BM. These approaches substantially reduce the number of search points while preserving good search capabilities of the meta-heuristic methods. These algorithms maintain a good balance between coding efficiency and computational complexity.

From the above discussion, we noticed that nature-inspired algorithms have demonstrated a good trade-off between accuracy and speed. Researchers have utilized GA, PSO, ABC, DE and HS for motion estimation – a key feature used in vision and robotic application. Empirical studies were also conducted that showed the ability of hybridization of these meta-heuristic algorithms. In this paper, we implement two hybrid algorithms: HS-DE and ABC-DE for BM. We have customized both algorithms (HS-DE and ABC-DE) to suit the problem and implemented them to improve the BM algorithm. Hybrid version of ABC-DE and HS-DE gives good results when it compared with other algorithms. Both the algorithms are novel as they have not been implemented for BM. We take four standard video sequences for simulation. The performance comparison of our proposed two hybrid algorithms: ABC-DE and HS-DE is done considering the parameters: SSIM, PSNR, Average number of Search Points which directly corresponds to computational complexity, Computational Gain of HS-DE and ABC-DE over other algorithms.

The rest of the paper is organized as follows: Section 2 discusses three meta-heuristic algorithms are implemented for motion estimation and BM. Section 3 presents two hybrid algorithms: ABC-DE and HS-DE are proposed for video sequences motion estimation. The experimental setup, results and discussion are given in Section 4. A brief discussion on motion estimation in robotics is given in Section 5. Section 6 concludes the paper and gives suggestions for future work.

2. Previous approaches for BM

In this section, we present three different meta-heuristic algorithms utilized for BM.

2.1 BM uses Differential Evolution

DE algorithm for BM was proposed to reduce search location. DE algorithm tries to improve the solution vector iteratively and optimize the problem by initializing a large population and then through mutation, crossover and selection operations. The steps applied to optimize the problem through DE algorithm are given below:

Step-1: Population generation

2 dimensional NP blocks $B_i (i \in 1 \text{ to } NP)$, each of size 16x16 pixels, are generated using a fixed pattern from the search space of $(2*W+1)\times(2*W+1)$ blocks.

Step-2: Mutation

DE/best/1 strategy is used, where the block with minimum SAD value *Bbest* is mutated by adding the scaled difference of two randomly selected blocks B_{n1} and B_{n2} from the current population. The two random blocks are chosen in such way that their indices should not be equal to each other and to the iteration number.

$$V = B_{best} + F*(B_{n1} - B_{n2}) \quad (1)$$

Where F and V respectively represent mutation probability and mutant vector.

Step-3: Crossover

Uniform crossover between parent block (B_i) and mutant block (V) is applied to generate a utility block (U) with CP as the Crossover probability. If the value of rand (0, 1) is less than CP then attribute value is chosen from mutant block, otherwise from parent block.

$$U = \begin{cases} V_{j,i} & if\ rand(0,1) \le CP\ or\ j = j_{rand} \\ B_{j,i} & otherwise \end{cases} \quad (2)$$

Step-4: Selection

SAD value is calculated through objective function for each parent block-utility block pair. If a utility block is superior to corresponding parent block, then it replaces the parent in the population otherwise parent remains same. Through this operation population is generated for the next generation.

$$B_i = \begin{cases} U_i & if\ f(U_i) \le f(B_i), \\ B_i & otherwise \end{cases} \quad (3)$$

2.2 BM uses Artificial Bee Colony

Cuevas et al. [21] proposed the ABC algorithm for BM to reduce search place in BM. Figure 1 presents the block diagram is divided into four steps (each step is discussed in detail) for ABC algorithm used for BM.

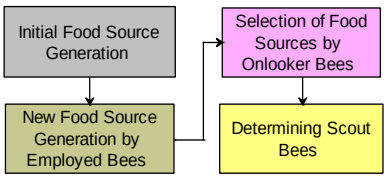


Figure 1. Block diagram for ABC algorithm used for BM.

Step-1: Initial food source generation

Generate 2 dimensional NP blocks B_i ($i \in 1$ to NP), each of size 16x16 pixels, using a fixed pattern from the search space of $(2*W+1)\times(2*W+1)$ blocks. The fitness function value for each block is calculated using equation (4).

$$fitness_i = \begin{cases} 1/(1+f(B_i)) & if\ f(B_i) \ge 0 \\ 1+abs(f(B_i)) & otherwise \end{cases} \quad (4)$$

Where, $f(\cdot)$ represents an objective function.

Step-2: New food source generation.

Each bee generates new food source (block) V_i in the neighborhood of each block B_i using equation (5).

$$V_{j,i} = B_{j,i} + r*(B_{j,i} - B_{j,k}) \quad (5)$$

Where, r is a random number in range of $[-1,1]$ and i, j and k are indexed parameters with a constraint $i \neq k$. The

fitness function value is calculated, which is then compared with the fitness function value of the corresponding initial block.

Step-3: Selection of food sources by onlooker bees

The probability of a food source (block) is calculated using equation (6).

$$\Pr ob_i = \frac{fitness}{\sum_{i=1}^{NP} fitness_i} \tag{6}$$

Onlooker bees utilize probability to select the food sources. A new candidate food source (block) is generated and if it is found better than the old one, it replaces the old food source (block).

Step-4: Determine scout bees

After step-3, if no improvement in the fitness function value is seen, then in such situation onlooker bee becomes scout bee. These scout bees generate new food source (block) and repeat the steps 1-3.

2.3 BM uses Harmony Search

HS algorithm was utilized for BM. The HS algorithm was applied to reduce the number of search locations. Figure 2 depicts a block diagram of HS used for BM.

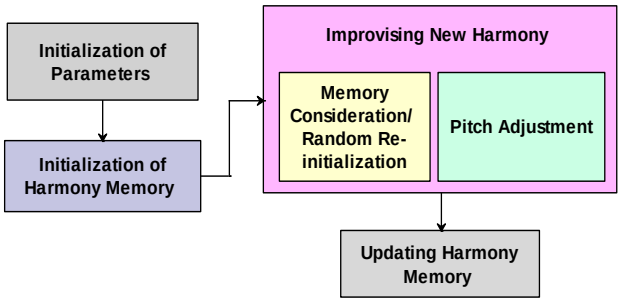


Figure 2. Block diagram for HS algorithm used for BM.

Below, we discuss the steps shown in Figure 2 for HS algorithm used for BM.

Step-1: Initialization of the problem and the parameters

The problem is to minimize the SAD value. The main algorithm parameters to be initialized are: HMS, HMCR [0 ≤ HMCR ≤ 1], PAR [0 ≤ PAR ≤ 1], BW and NI.

Step-2: Initialization of Harmony Memory

Equation (7) is used to initialized HM considering the HMS blocks Bi (i ϵ 1 to HMS) with 2 dimensions are generated using a fixed pattern from the search space of $(2*W+1)\times(2*W+1)$ blocks.

$$HM = \begin{bmatrix} B_1 \\ B_2 \\ . \\ . \\ . \\ B_{HMS} \end{bmatrix} \tag{7}$$

Step-3: Initialization of Harmony Memory

Improvisation of HM is done by generating a New Harmony vector or block B_{new} as shown in equation (8).

$$B_{new}(j) = \begin{cases} B_i(j) \in \{B_1(j), B_2(j), \mathrm{K}, B_{HMS}(j)\} & if \quad rand(0,1) < HMCR \\ Randomly \quad generate \quad new \quad block \quad from \quad search \quad space & otherwise \end{cases} \tag{8}$$

Every component generated through equation (8) is pitch-adjusted using equation (9).

$$B_{new}(j) = \begin{cases} B_{new}(j) \pm rand(0,1) \cdot BW & if \quad rand(0,1) < PAR \\ B_{new}(j) & otherwise \end{cases} \tag{9}$$

PAR assigns the frequency of the adjustment and BW controls the local search around the selected elements of HM. Pitch adjustment generates new potential harmonies by modifying the original variable positions, which is similar to the mutation operation in EAs. Hence, each dimension of the vector is either perturbed by a random number between 0 and BW or left unchanged.

Step-4: Updating Harmony Memory

The decision of updating the HM is depends upon the criteria: "*whether the new block* B_{new} *replaces the worst block* B_{worst}". Equation (10) is used to update the HM.

$$B_{worst} = \begin{cases} B_{new} & if \quad f(B_{new}) < f(B_{worst}) \\ B_{worst} & otherwise \end{cases} \tag{10}$$

3. Proposed approaches

In this section, we discuss two hybrid algorithms are implemented for BM.

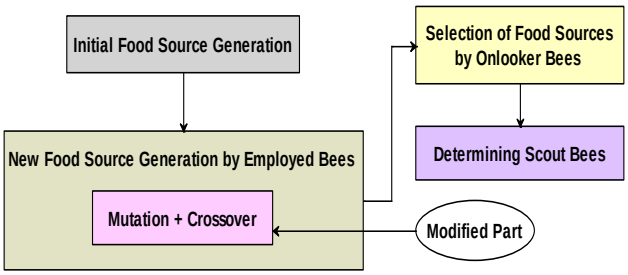


Figure 3. Block diagram for ABC-DE algorithm used for BM.

3.1 Hybrid ABC-DE based BM algorithm

Hybridization of DE-ABC was proposed previously [22]. We propose a customized version of hybrid ABC-DE algorithm to fit in the goal: "*to minimize the number of SAD evaluations with an acceptable solution for BM*". In our approach, the food source generation operations of ABC (bee phase and onlooker bee phase) is replaced by mutation and crossover operation of the DE algorithm as shown in Figure 3. Algorithm-1 presents the customized version of hybrid ABC-DE based BM algorithm.

***Algorithm-1**: Hybrid ABC-DE based BM Algorithm*

1. *Initialize the parameters F_employed = 0.25, F_onlooker = 0.25, CP = 0.5 for DE and limit = 10 for ABC. Dimension D = 2. Search Parameter W = 8 or 16. Block Size is 16x16 pixels.*
2. *Initialize the population of NP=5 individuals with D dimensions using the fixed pattern from the search space of*

*(2*W+1) X (2*W+1) blocks. Initialize counter for each individual C_i=0 (i ϵ 1 to NP).*

3. *Calculate the SAD between current block and each block of NP (B_i where i ϵ 1 to NP).*
4. *Calculate the fitness value for each individual of NP.*
5. *While the terminating criteria is not satisfied do*
6. *New population of NP blocks is generated using mutation and crossover.*
7. *For i = 1 to NP*
8. *Select three blocks B_p, B_q and B_r from population where p ≠ q ≠ r ≠ i*
9. $V_i = B_p + F\_employed * (B_q - B_r)$
10. *For j = 1 to D*
11. *If rand(0,1) ≤ CP or j = j_{rand} Then*
12. *Trial vector $U_{j,i}$ = $V_{j,i}$*
13. *Else*
14. *Trial vector $U_{j,i}$ = $B_{j,i}$*
15. *End if*
16. *End for*
17. *End for*
18. *Applying Fitness Approximation method to calculate SAD value of each newly generated food source (V_i) followed by calculating fitness value.*
19. *If fitness(U_i) > fitness(B_i) Then*
20. *B_i = U_i*
21. *Else*
22. *C_i=C_i+1*
23. *End if*
24. *Calculate probability of each selected food source.*
25. *For i = 1 to NP*
26. $\text{Prob}_i = fitness_i / \sum_{i=1}^{NP} fitness_i$
27. *r = rand(0,1)*
28. *If (r < $Prob_i$)*
29. *Follow step 6 for this food source using F_onlooker*
30. *If C_i > limit*
31. *Block is abandoned and a new block is randomly selected.*
32. *End if*
33. *End for*
34. *End while*
35. *Select the block with highest fitness value for Motion Vector calculation*

3.1.1 Advantage of the hybrid approach (ABC-DE)

The proposed algorithm is more powerful as it utilizes the search space exploration ability of DE algorithm, which is combined with the solution's exploitation ability of the ABC algorithm. Exploration and exploitation is the key to the success of any search and optimization algorithm. The ABC algorithm performs the exploration in two steps:

a) When new food sources are generated in the neighborhood of the initial population and

b) When a new food source is generated in the neighborhood of the food source with the highest probability.

In both these cases, the proposed ABC-DE algorithm uses mutation and crossover operation of the DE algorithm. Mutation and crossover operation have shown tendency to explore the new search space more effectively. Then applying the operators of ABC algorithm will exploit the population. Hence, the proposed ABC-DE has ability to explore and exploit the search space adequately. It also addresses the issue (exploitation) of the DE algorithm.

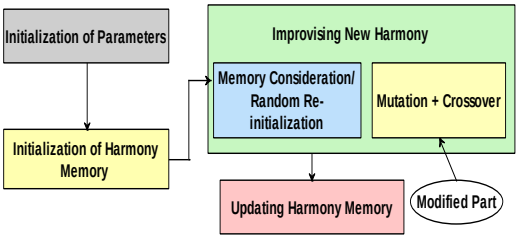


Figure 4. Block diagram for HS-DE algorithm used for BM.

3.2 Hybrid HS-DE based BM algorithm

Hybridization of DE-HS was proposed in [23]. Chakraborty et al. [23] used mutation operator of DE algorithm to perturb the target vector instead of pith adjustment. We propose a hybrid version of HS-DE algorithm, where the crossover operator of DE algorithm is utilized (as shown in Figure 4) to increase the diversity of the perturbed vector. Algorithm-2 presents the working of the hybrid HS-DE based BM algorithm.

***Algorithm-2**: Hybrid HS-DE based BM Algorithm*

1. *Set the parameters. HMS = 5, HMCR = 0.7, PAR = 0.3, BW = 8 for HS and F = 0.25, CP = 0.8 for DE. Dimension D = 2. Search Parameter W = 8 or 16. Block Size is 16x16 pixels.*
2. *Initialize the population of HMS blocks with D dimensions using the fixed pattern from the search space of (2*W+1) X (2*W+1) blocks.*
3. *Calculate the SAD values between current block and each block of Harmony Memory (B_i where i ϵ 1 to HMS)*
4. *While the terminating criteria is not satisfied do*
5. *Determine the block B_{worst} with worst SAD value, i.e. the highest SAD value*
6. *Improvise new block B_{new}*
7. *For j = 1 to D*
8. *If (rand(0,1) < HMCR)*
9. $B_{new}(j) = B_i(j)$ *where i = 1, 2, …, HMS*
10. *Else*
11. $B_{new}(j) = 1 + round(r * W)$ *where r ∈ rand(-1, 1)*
12. *If (B_{new}(j) < l(j))*
13. *B_{new}(j) = l(j)*
14. *End if*
15. *if (B_{new}(j) > u(j))*
16. *B_{new}(j) = u(j)*
17. *End if*
18. *End if*
19. *End for*
20. *Select two blocks from population B_p and B_q where p≠q*

21. $V = B_{new} + F*(B_p - B_q)$
22. *For j = 1to D*
23. *If rand(0,1) ≤ CP or j = j_{rand} Then*
24. *Trial block U_j = V_j*
25. *Else*
26. *Trial block U_j = B_j*
27. *End if*
28. *End for*
29. *Applying Fitness Approximation method calculates SAD value of B_{new}*
30. *B_{worst}=B_{new} if SAD(B_{new}) < SAD(B_{worst})*
31. *End while*
32. *Select the block with minimum SAD value for Motion Vector calculation*

3.2.1 Advantage of the hybrid approach (HS-DE)

The HS algorithm suffers with premature or false convergence. In the proposed hybrid HS-DE algorithm for BM pitch adjustment of HS algorithm is performed through crossover and mutation operations of the DE algorithm. It alleviates premature convergence of the HS algorithm. In turn, it solves the drawback of DE algorithm. The DE algorithm updates the current individuals based on only the differences among certain randomly selected individual, whilst HS algorithm uses the combination of all the individuals which increases the diversity of individuals.

4. Simulation model

Extensive experiments have been conducted on MATLAB 8.5 on an Intel Core i3 2.5 GHz PC with 4GB of memory and 64-bit Windows 10 Operating System. Luminance component of video sequences (more noticeable to human eyes) as luminance of videos or images have been used during simulation.

Table I.

Test Video Sequences

Sequence	Format	Resolution	Number of Frames
Container	QCIF	176x144	300
Carphone	QCIF	176x144	382
Akiyo	CIF	352x288	300
Foreman	CIF	352x288	300

Four standard video sequences are considered for the simulation as shown in Table I and one of the frames of each video sequence is depicted in Figure 5. These video sequences have different formats, resolutions and number of frames with sufficient complexity involved to conduct the experiments. Previously, Cuevas et al. [21] compared the performance of the ABC, DE and HS based BM approach with other algorithms such as ES [1], 3SS [2], NTSS [3], SESTSS [4], 4SS [5], BBGD [24], DS [6], NE [25], ND [26], LWG [8], 4GS [9] and PSO-BM [13]. The comparative results demonstrated the superiority of the meta-heuristic algorithms based BM algorithms over the others. But, these algorithms have not considered ARPS, which gives better results in case of non-metaheuristic algorithms.

In this research, our objective is to present an improved BM algorithm for motion estimation in video sequence and compare the performance of the proposed algorithms with ARPS, ABC, DE and HS based BM algorithms. We have considered ES, 3SS, SESTSS, NTSS, 4SS, DS, ARPS, ABC-BM, DE-BM and HS-BM for comparison. We have determined SSIM, PSNR, Average Number of Search Points (directly corresponding to the computational complexity), Computation Gain and Quality of Loss for each algorithm. The term quality of loss is similar to the PSNR degradation ratio. Quality Loss corresponds to the percentage by which the PSNR has been reduced with respect to a specific algorithm while PSNR degradation ratio corresponds to the percentage by which the PSNR has been reduced with respect to Exhaustive Search. Hence the PSNR degradation ratio is not presented in the paper as it would be redundant.

Computational Gain: By what percentage the computation has been reduced with respect to a specific algorithm.

SP_{HSDE} = Average Search Points for HSDE

SP = Average Search Points for any other Algorithm

$$\text{Computational Gain (HS-DE)} = -\left(\frac{SP_{HSDE} - SP}{SP} \times 100\right) \quad (1)$$

SP_{ABCDE} = Average Search Points for ABCDE

SP = Average Search Points for any other Algorithm

$$\text{Computational Gain (ABC-DE)} = -\left(\frac{SP_{ABCDE} - SP}{SP} \times 100\right) \quad (2)$$

Quality Loss: By what percentage, the PSNR has been reduced with respect to a specific algorithm.

$PSNR_{HSDE}$ = Average PSNR for HSDE

PSNR = Average PSNR for any other Algorithm

$$\text{Quality Loss (HS-DE)} = \left(\frac{PSNR - PSNR_{HSDE}}{PSNR} \times 100\right) \quad (3)$$

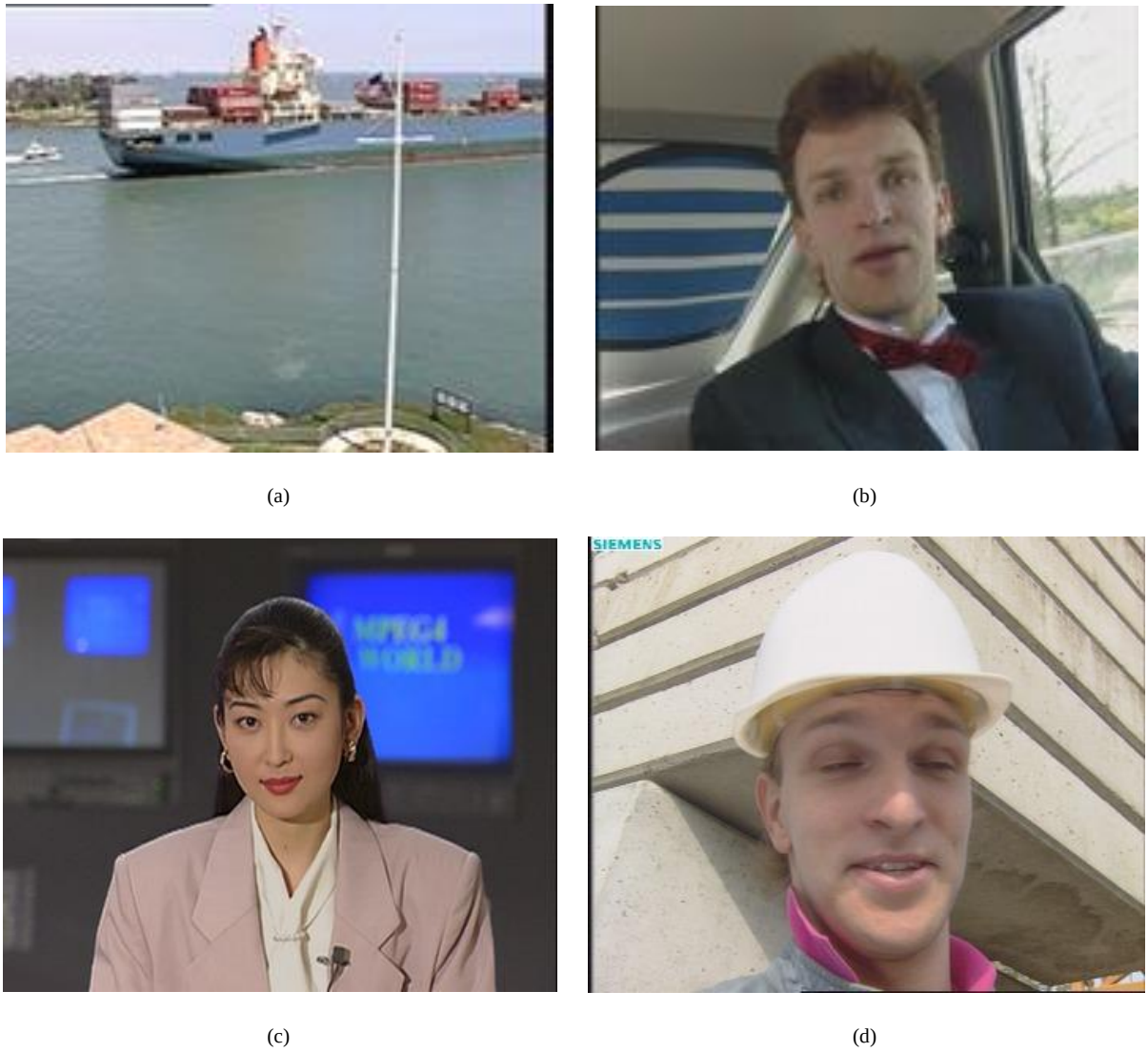
$PSNR_{ABCDE}$ = Average PSNR for ABCDE

PSNR = Average PSNR for any other Algorithm

$$\text{Quality Loss (ABC-DE)} = \left(\frac{PSNR - PSNR_{ABCDE}}{PSNR} \times 100\right) \quad (4)$$

Table II, III, IV, and V presents the comparative results of various BM algorithms considering the test video sequences as given in Table I for Container, Carphone, Akiyo and Foreman sequences respectively.

Figure 6, 8, 10 and 12 depicts frame wise PSNR comparison chart, whereas Figure 7, 9, 11 and 13 show comparative charts for frame wise search points with respect to different BM algorithms for test video sequences (Table I). These results revealed that the proposed hybrid algorithms (ABC-DE and HS-DE) showed significantly better performance in terms of computational complexity is concerned. The best value of the average number of search points is marked bold in Table II, III, IV, and V. We have noticed that HS-DE revealed the best value of computational complexity for Container and Foreman video sequences whilst ABC-DE has shown the best response for Carphone and Akiyo sequences. We also noticed that computational gain of the proposed ABC-DE and HSE-DE is significantly high as compared to other algorithms.



(a) (b)

(c) (d)

Figure 5. Test video sequence. (a) Container, (b) Carphone, (c) Akiyo and (d) Foreman

Table II.
Comparison of various algorithms for *Container* sequence

BM Algorithm	Avg. SSIM	Avg. PSNR	Avg. Search Points	Computational Gain (HS-DE) %	Quality Loss (HS-DE) %	Computational Gain (ABC-DE) %	Quality Loss (ABC-DE) %
ES	0.9926	44.1108	236.6364	97.9522	0.3414	98.0300	0.3634
3SS	0.9925	44.0624	21.4876	77.4483	0.2319	78.3051	0.2539
SESTSS	0.9925	44.0584	16.198	70.0839	0.2228	71.2205	0.2449
NTSS	0.9925	44.0624	14.7209	67.0821	0.2319	68.3327	0.2539
4SS	0.9925	44.0448	14.6852	67.0021	0.1920	68.2557	0.2141
DS	0.9925	44.0439	11.4667	57.7402	0.1900	59.3457	0.2120
ARPS	0.9925	44.0198	4.9085	1.2773	0.1353	5.0280	0.1574
DE	0.9924	43.9806	9.2312	47.5062	0.0463	49.5006	0.0684
HS	0.9924	43.9797	5.3911	10.1148	0.0443	13.5297	0.0663
HSDE	0.9924	43.9602	4.8458	-	-	3.7991	0.0220
ABC	0.9924	43.9781	7.4532	34.9836	0.0407	37.4537	0.0627
ABCDE	0.9924	43.9505	**4.6617**	-3.9492	-0.0220	-	-

Table III.
Comparison of various algorithms for *Carphone* sequence

BM Algorithm	Avg. SSIM	Avg. PSNR	Avg. Search Points	Computational Gain (HS-DE) %	Quality Loss (HS-DE) %	Computational Gain (ABC-DE) %	Quality Loss (ABC-DE) %
ES	0.9372	32.7196	236.6364	97.9192	2.7231	97.7705	2.7368
3SS	0.9339	32.4837	21.6199	77.2256	2.0167	75.5979	2.0305
SESTSS	0.9299	32.2893	15.8705	68.9751	1.4267	66.7578	1.4407
NTSS	0.9347	32.5627	16.9685	70.9827	2.2544	68.9088	2.2682
4SS	0.9336	32.4554	15.6924	68.6230	1.9312	66.3805	1.9451
DS	0.9342	32.5153	13.1586	62.5811	2.1119	59.9068	2.1257
ARPS	0.9331	32.4357	7.0025	29.6851	1.8717	24.6597	1.8855
DE	0.9035	30.7807	9.0372	45.5163	-3.4044	41.6224	-3.3897
HS	0.9036	30.7822	5.3785	8.4540	-3.3993	1.9113	-3.3847
HSDE	0.9218	31.8286	**4.9238**	-	-	-7.1469	0.0141
ABC	0.9034	30.774	7.2376	31.9691	-3.4269	27.1070	-3.4122
ABCDE	0.9218	31.8241	5.2757	6.6702	-0.0141	-	-

Table IV.
Comparison of various algorithms for *Akiyo* sequence

BM Algorithm	Avg. SSIM	Avg. PSNR	Avg. Search Points	Computational Gain (HS-DE) %	Quality Loss (HS-DE) %	Computational Gain (ABC-DE) %	Quality Loss (ABC-DE) %
ES	0.9931	44.1053	262.1717	98.0532	0.6366	98.1242	0.6289
3SS	0.993	43.9835	23.2121	78.0127	0.3614	78.8136	0.3537
SESTSS	0.9928	43.8795	17.0745	70.1092	0.1253	71.1979	0.1175
NTSS	0.9931	44.0984	15.9253	67.9522	0.6211	69.1195	0.6134
4SS	0.993	44.0211	15.8453	67.7904	0.4466	68.9636	0.4388
DS	0.9931	44.0903	12.2746	58.4206	0.6028	59.9351	0.5951
ARPS	0.9931	44.0725	5.0498	-1.0673	0.5627	2.6139	0.5549
DE	0.9894	42.0536	9.0158	43.3916	-4.2110	45.4535	-4.2191
HS	0.9894	42.0541	5.4606	6.5359	-4.2098	9.9402	-4.2179
HSDE	0.9927	43.8245	5.1037	-	-	3.6424	-0.0077
ABC	0.9984	42.05	8.0112	36.2929	-4.2199	38.6134	-4.2280
ABCDE	0.9927	43.8279	**4.9178**	-3.7801	0.0077	-	-

Table V.
Comparison of various algorithms for *Foreman* sequence

BM Algorithm	Avg. SSIM	Avg. PSNR	Avg. Search Points	Computational Gain (HS-DE) %	Quality Loss (HS-DE) %	Computational Gain (ABC-DE) %	Quality Loss (ABC-DE) %
ES	0.9201	32.6896	262.1717	98.0242	10.1821	97.9049	10.4017
3SS	0.8976	32.009	23.3295	77.7967	8.2723	76.4564	8.4966
SESTSS	0.8866	31.5079	15.9777	67.5804	6.8135	65.6233	7.0414
NTSS	0.9019	32.2292	21.2373	75.6094	8.8990	74.1370	9.1218
4SS	0.8989	32.053	18.8784	72.5617	8.3982	70.9053	8.6222
DS	0.9028	32.2209	17.6867	70.7130	8.8756	68.9450	9.0984
ARPS	0.9086	32.3647	8.9747	42.2833	9.2804	38.7990	9.5023
DE	0.8034	28.1001	8.6891	40.3862	-4.4875	36.7874	-4.2320
HS	0.8034	28.094	5.454	5.0256	-4.5102	-0.7077	-4.2546
HSDE	0.8367	29.3611	**5.1799**	-	-	-6.0367	0.2445
ABC	0.8024	28.0362	7.67	32.4654	-4.7256	28.3885	-4.4695
ABCDE	0.8356	29.2893	5.4926	5.6931	-0.2451	-	-

Table. VI
Quality comparison of different initialization patterns of *Carphone* sequence

BM Algorithms	Avg. SSIM		Avg. PSNR	
	Without Centre-biased pattern	With Centre-biased pattern	Without Centre-biased pattern	With Centre-biased pattern
DE	0.9035	0.9216	30.7807	31.8147
HS	0.9036	0.9216	30.7822	31.8077
ABC	0.9034	0.9215	30.774	31.8073

Table VII
Quality comparison of different initialization patterns of *Akiyo* sequence

BM Algorithms	Avg. SSIM		Avg. PSNR	
	Without Centre-biased pattern	With Centre-biased pattern	Without Centre-biased pattern	With Centre-biased pattern
DE	0.9894	0.9927	42.0536	43.8220
HS	0.9894	0.9927	42.0541	43.8220
ABC	0.9984	0.9927	42.05	43.8220

Table VIII
Comparison between different numbers of iterations for *Carphone* sequence

BM Algorithms	Number of iterations							
	1		2		3		4	
	Avg. PSNR	Avg. Search Points	Avg. PSNR	Avg. Search Points	Avg. PSNR	Avg. Search Points	Avg. PSNR	Avg. Search Points
HS-DE	31.8286	4.9238	31.8452	5.2614	31.8637	5.6	31.8746	5.9438
ABC-DE	31.8241	5.2757	31.8316	5.9421	31.8444	6.5809	31.8526	7.1832

Table IX
Comparison between different population sizes for *Carphone* sequence

BM Algorithms	Population Size			
	5		9	
	Avg. PSNR	Avg. Search Points	Avg. PSNR	Avg. Search Points
HS-DE	31.8286	4.9238	32.1419	7.6869
ABC-DE	31.8241	5.2757	32.1538	8.3483

In addition, ABC-DE and HSE-DE algorithms have shown very low quality loss. From the results presented in Table II, III, IV and V, we can see that the algorithms: ABC-BM, DE-BM and HS-BM have shown higher loss in quality. It has happened due to the initialization patterns that are used by these algorithms (ABC-BM, DE-BM and HS-BM). The quality of ABC-BM, DE-BM and HS-BM algorithms can be enhanced by changing the initialization patters to center-biased as presented in Table VI and VII respectively for Caphone and Akiyo video sequences.

Previous scientific researches on BM algorithms utilizing nature inspired algorithms [21] have chosen average number of search points as one of the measure of computational complexity. In this paper, we have also used average number of search points as a measure of computational complexity. In addition, we noticed that the metaheuristic algorithms have shown better results in terms of computational complexity on distributed systems and parallelization of operations of metaheuristics algorithms can be achieved easily. Hence, comparing them with the classical BM algorithms on a non-distributed environment will not be an effective thought.

The computational time might be higher in case of the proposed algorithms with respect to some classical BM algorithms, but the main aim of this research is to present hybridization of the metaheuristic algorithms for motion estimation in video sequences. The results showed that the proposed hybrid algorithms have outperformed other algorithms. The experimental results revealed that the computational time of HSDE has outperformed both HS and DE, whilst ABCDE has outperformed both ABC and DE.

The main advantage of utilizing metaheuristic algorithms for BM is it has tendency to maintain a good balance between quality and computational complexity. Extensive experiments have been conducted over five blocks and based on the results following observations have been made: " *any increase in both number of block in the population and number of generation increases the computational complexity, but decreases the quality of loss*". Table VIII presents the results after increasing the number of iterations/generations for carphone video sequence. On the other hand, Table IX shows the results of different population sizes for foreman video sequence.

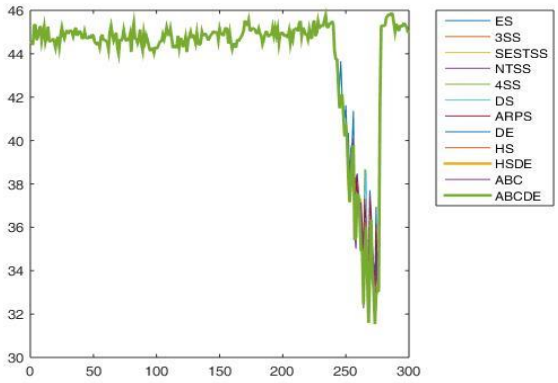


Figure 6. Frame wise PSNR performance for Container sequence.

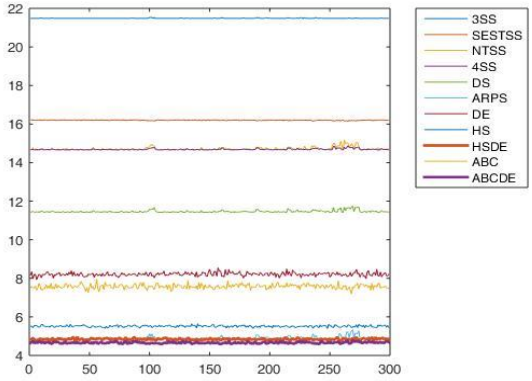


Figure 7. Frame wise Search Points for *Container* sequence.

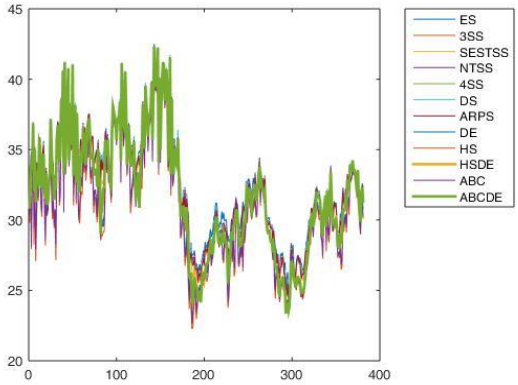


Figure 8.. Frame wise PSNR performance for Carphone sequence

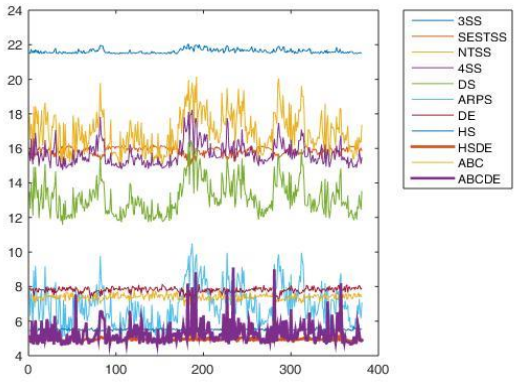


Figure 9. Frame wise Search Points for Carphone sequence

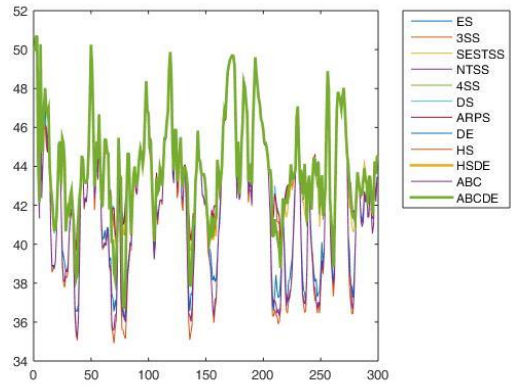


Figure 10. Frame wise PSNR performance for Akiyo sequence

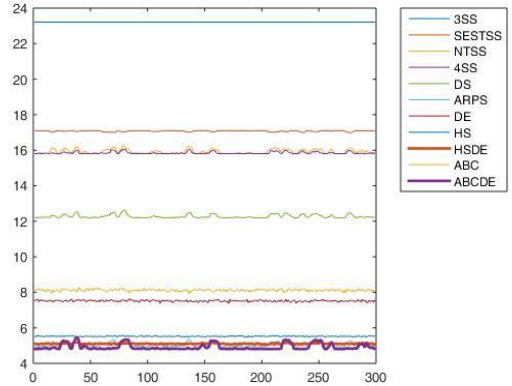


Figure 11. Frame wise Search Points for Akiyo sequence

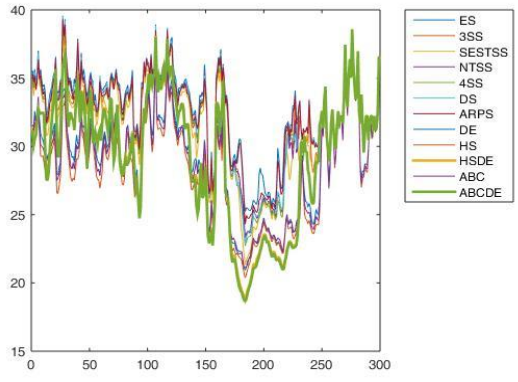


Figure 12. Frame wise PSNR performance of *Foreman* sequence

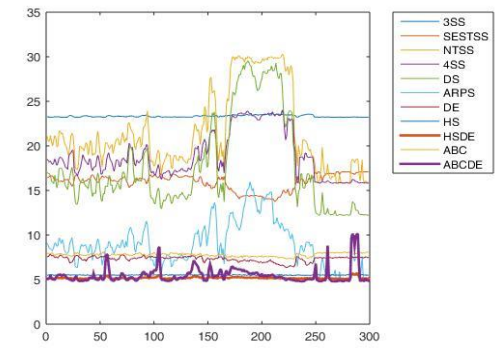


Figure 13. Frame wise Search Points for *Foreman* sequence

Table X.
Computational Time (Sec.) of various BM algorithms for different video sequences.

BM Algorithm	Video Sequences			
	Container Sequence	Carphone Sequence	Akiyo Sequence	Foreman Sequence
ES	679.222	812.322	2581.164	2662.610
3SS	63.830	76.574	233.668	254.007
SESTSS	45.417	57.118	170.910	172.607
NTSS	41.137	58.530	160.318	217.748
4SS	41.335	55.544	158.224	194.540
DS	35.384	52.302	135.960	204.570
ARPS	18.162	31.835	67.499	115.610
DE	78.495	99.689	309.724	312.490
HS	52.358	68.607	235.522	235.025
HSDE	51.509	66.178	235.478	231.039
ABC	71.218	91.635	305.870	312.423
ABCDE	66.305	86.427	287.981	293.141

We have also utilized diamond pattern to analyze the effect of increased population size.

The computational time of various BM algorithms implemented on four video sequences are presented in Table X. This results show that the proposed hybrid algorithms (HS-DE and ABC-DE) consumes moderate computational time, but both the algorithms show better results on other factors (computational gain and quality of loss).

5. Motion estimation in robots

In this paper, we have presented two hybrid algorithms (ABC-DE and HS-DE) for motion estimation in video sequence. Motion estimation has vital applications in robotics. In this section, we are highlighting some of exiting work on motion estimation had been used in robotics.

Booij et al. [27] proposed an estimation method to determine the full likelihood in the space of all possible planar relative space. The standard Bayesian method was used to learn likelihood function from the existing data. The result of this approach was impressive as it was efficient to estimate the likelihood of new pose effectively. In addition, this approach was capable to create and estimate new poses. Though, this approach was successfully implemented for planer robot, but it was limited to pair of images only. Spacek and Burbridge [28] suggested two related methods (localization by trilateration and inter-frame motion estimation) for autonomous visual guidance of robots. These methods were based on co-axial omni-directional range, which returns guiding points detected in the images. It was also limited to images only. Gonzalez and Gutierrez [29] estimated the motion parameters of a mobile robot equipped with a radial laser rangefinder. This method was based on the spatial and temporal linearization of range function. The experiments were conducted on a computer simulation which later on downloaded to a real robot. Ferreira et al. [30] presented a comprehensive survey on real-time motion estimation techniques for underground robots. The above discussion indicates that motion estimation is important in the field of robotic applications. The approaches suggested in existing scientific literatures have their own strengths and weaknesses. In this paper, we have presented two hybrid algorithms using metaheuristic algorithm for motion estimation with believes that we will extend these algorithms purely for robotics in the near future.

6. Conclusion

In this paper, we presented and evaluated two hybrid algorithms: *Artificial Bee Colony – Differential Evolution* and *Harmony Search – Differential Evolution* for motion estimation in video sequences. Extensive experiments have been conducted on four standard video sequences to evaluate the performance of the proposed algorithms. We have compared the proposed algorithms with other nine algorithms: *Three Step Search*, *Simple and Efficient Three Step Search*, *New Three Step Search*, *Four Step Search*, *Diamond Search*, *Adaptive Road Pattern Search*, *Differential Evolution*, *Harmony Search* and *Artificial Bee Colony*. The computational results have revealed that the proposed hybrid algorithms can reduce computational complexity significantly and improve overall performance. We noticed that computational gain of proposed hybrid algorithms is significantly high with very low quality loss as compared to other algorithms. The results reported in Table X indicate that the computation time of the proposed hybrid algorithms is significantly better than *Harmony Search*, *Differential Evolution* and *Artificial Bee Colony Algorithms*. Further, we have found that the hybrid algorithms consume little high computational time as compared to other six algorithm (*Three Step Search*, *Simple and Efficient Three Step Search*, *New Three Step Search*, *Four Step Search*, *Diamond Search*, *Adaptive Rood Pattern Search*), but both hybrid algorithms show better results on other factors: computational gain and quality loss. So, the proposed algorithms improve the performance of Block Matching algorithm for motion estimation in video sequences.

A mobile robot must perceive the motions of an external object to perform a certain tasks successfully. The proposed algorithms have ability to perform both motion estimation and video compression successfully. We have shown the application of motion estimation in robots. Hence, to deal with motion estimation in mobile robot utilizing the proposed algorithms is an immediate future work.

7. References

1. J. Jain and A. Jain. "*Displacement measurement and its application in interframe image coding.*" *IEEE Transactions on communications* 29.12 (1981), pp. 1799-1808.
2. T. Koga. "*Motion compensated inter-frame coding for video conferencing.*" *Proc. Nat. Telecommunications Conf., New Orleans, LA, Nov. 1981*. 1981.
3. R. Li BZeng and M.L.Liou. "*A new three-step search algorithm for block motion estimation.*" IEEE transactions on circuits and systems for video technology, 4(4), 1994, pp. 438-442.
4. J. Lu and M. L. Liou. "*A simple and efficient search algorithm for block-matching motion estimation.*" IEEE Transactions on Circuits and Systems for Video Technology, 7(2), 1997, pp. 429-433.
5. L. M. Po and W. C. Ma. "*A novel four-step search algorithm for fast block motion estimation.*" IEEE transactions on circuits and systems for video technology, 6(3), 1996, pp. 313-317.
6. S. Zhu and K.K. Ma. "*A new diamond search algorithm for fast block matching motion estimation.*" In Information, Communications and Signal Processing, 1997. ICICS, Proceedings of 1997 International Conference on (Vol. 1, pp. 292-296). IEEE.
7. Y. Nie and K.K. Ma. "*Adaptive rood pattern search for fast block-matching motion estimation*." IEEE Transactions on image processing, 11(12), 2002, pp. 1442-1449.
8. C. I. Lin and J.L. Wu. "*A lightweight genetic block-matching algorithm for video coding*." IEEE Transactions on Circuits and Systems for Video Technology, 8(4), 1998, pp. 386-392.
9. M. F. So and A. Wu. "*Four-step genetic search for block motion estimation.*" In Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on Vol. 3, pp. 1393-1396.
10. S. Li, W. Xu, H. Wang and N. N. Zheng. "*A novel fast motion estimation method based on genetic algorithm.*" In Image Processing,

1999. ICIP 99. Proceedings. 1999 International Conference on Vol. 1, pp. 66-69.
11. G. Y. Du, T. S. Huang, L. X. Song and B. J. Zhao. "*A novel fast motion estimation method based on particle swarm optimization.*" In Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on Vol. 8, pp. 5038-5042.
12. R. Ren, Y. Shi and B. Zheng. "*A Fast Block Matching Algorithm for Video Motion Estimation Based on Particle Swarm Optimization and Motion Prejudgment*". arXiv preprint cs/0609131, 2006.
13. X. Yuan and X. Shen. "*Block matching algorithm based on particle swarm optimization for motion estimation.*" In Embedded Software and Systems, 2008. ICESS'08. International Conference on pp. 191-195.
14. P. Zhang, P. Wei, H. Yu and Z. Wang. "*Simplex particle swarm optimization for block matching algorithm.*" In Intelligent Signal Processing and Communication Systems (ISPACS), 2010 International Symposium on pp. 1-4.
15. K. M. Bakwad, S.S. Pattnaik, B. S. Sohi, S. Devi, S. V. Gollapudi, C. V. Sagar and P.K. Patra. "*Fast motion estimation using small population-based modified parallel particle swarm optimisation.*" International Journal of Parallel, Emergent and Distributed Systems, 26(6), 2011, pp. 457-476.
16. J. Cai and W. D. Pan. "*On fast and accurate block-based motion estimation algorithms using particle swarm optimization.*" Information Sciences, 197, 2012, pp. 53-64.
17. M. K. Jalloul and M. A. Al-Alaoui. "*A novel parallel motion estimation algorithm based on particle swarm optimization.*" In Signals, Circuits and Systems (ISSCS), 2013 International Symposium on pp. 1 - 4.
18. X. Liu, S. Xuan and F. Liu. "*An advanced particle swarm optimization based on good-point set and application to motion estimation.*" In *International Conference on Intelligent Computing* (pp. 494-502). Springer, Berlin, Heidelberg.
19. J. J. Britto and K. S. Chandran. "*A predictive and pattern based PSO approach for motion estimation in video coding.*" In Communications and Signal Processing (ICCSP), 2014 International Conference on pp. 1572-1576.
20. M.K. Jalloul and M.A. Al-Alaoui. "*A novel Cooperative Motion Estimation Algorithm based on Particle Swarm Optimization and its multicore implementation.*" Signal Processing: Image Communication, 39, 2015, pp. 121-140.
21. E. Cuevas, D. ZaldíVar, M. PéRez-Cisneros, H. Sossa and V. Osuna. "*Block matching algorithm for motion estimation based on Artificial Bee Colony (ABC)*". Applied Soft Computing, 13(6), 2013, pp. 3047-3059.
22. C. Worasucheep. "*A Hybrid Artificial Bee Colony with Differential Evolution.*" International Journal of Machine Learning and Computing, 5(3), 2015, 179.
23. P. Chakraborty, G. Roy, S. Das, D. Jain and A. Abraham. "*An improved harmony search algorithm with differential mutation operator.*" Fundamenta Informaticae, 95(4), 2009, pp. 401-426.
24. L. K. Liu and E. Feig. "*A block-based gradient descent search algorithm for block motion estimation in video coding*". IEEE Transactions on circuits and systems for Video Technology, 6(4), 1996, pp. 419-422.
25. A. Saha, J. Mukherjee and S. Sural. "*A neighborhood elimination approach for block matching in motion estimation.*" Signal Processing: Image Communication, 26(8), 2011, pp. 438-454.
26. A. Saha, J. Mukherjee and S. Sural. "*New pixel-decimation patterns for block matching in motion estimation.*" Signal Processing: Image Communication, 23(10), 2008, pp. 725-738.
27. O. Booij, B. Kröse and Z. "*Efficient probabilistic planar robot motion estimation given pairs of images.*" (2010).
28. L. Spacek and C. Burbridge. "*Instantaneous robot self-localization and motion estimation with omni directional vision.*" *Robotics and Autonomous Systems* 55.9 (2007): 667-674.
29. J. Gonzalez and R. Gutierrez. "Mobile robot motion estimation from a range scan sequence." *Robotics and Automation, 1997. Proceedings. 1997 IEEE International Conference on*. Vol. 2. IEEE, 1997.
30. F. Ferreira, G. Veruggio, M. Caccia and G. Bruzzone "*A survey on real-time motion estimation techniques for underwater robots*." Journal of Real-Time Image Processing 11.4 (2016): 693-711.

Kamanasish Bhattacharjee received his B.E (Hons.) degree in Electrical and Electronics Engineering from Birla Institute of Technology and Science, Pilani, India and M.Tech in Computer Science and Engineering from Amity University Uttar Pradesh, Noida, India. He is currently pursuing his Ph.D. from Indian Institute of Technology, Roorkee, India. His research interests include metaheuristic in solving complex engineering problems.

Sushil Kumar is major in computer science and engineering. He has completed B.Tech (CSE), M.Tech (CSE) and Ph.D. in CSE from IIT Roorkee. His area of research includes soft computing, metaheuristic algorithms to solve complex engineering problems. He is associated with many journals as reviewer and participated in International conferences.

Hari Mohan Pandey is major in computer science and engineering, completed his Ph.D. in Grammatical Inference using Evolutionary Algorithms and presently working on DREA4CAR project. He is working with Middlesex University, London U.K. He has authored over 40 research papers and associated with many International Journals as a reviewer, editorial board member and served as a leading guest editor.

Millie Pant is working as an associate professor in the Indian Institute of Technology Roorkee, India. Her areas of expertise include numerical optimization, evolutionary algorithms and their application to real life problems. She has over 100 publications in journals and conferences of national and international repute.

David Windridge is Associate Professor in Computer Science at Middlesex University and visiting Sr. Lecturer at the University of Surrey, U.K. His research interests centre on machine learning, cognitive systems and computer vision, with a former research interest in astronomy/astrophysics. He has authored more than 100 academic publications and played a leading role on a number of large-scale machine-learning projects.

Ankit Chaudhary is Assistant Professor at Department of Computer Science, Northwest Missouri State University, USA. He received his Ph.D. in Computer Engineering and his areas of research interest are computer vision, artificial intelligence and graph algorithms. He has authored 75 research papers and an Associate Editor of Computer & Electrical Engineering. Journal, Elsevier.

Abbreviations

BM	Block Matching
MAD	Mean Absolute Difference
MSE	Mean Squared Error
SAD	Sum of Absolute Differences
3SS	Three Step Search
SESTSS	Simple and Efficient Three Step Search
NTSS	New Three Step Search
4SS	Four Step Search
DS	Diamond Search
ARPS	Adaptive Rood Pattern Search
GA	Genetic Algorithm
LGA	Lightweight Genetic Algorithm
ES	Exhaustive Search
FSA	Full Search Algorithm
M3SS	Multicandidate Three Step Search
4GS	Four-step Genetic Search
PSO	Particle Swarm Optimization
PSO-ZMP	Particle Swarm Optimization- Zero Motion Prejudgment
ZMP	Zero Motion Prejudgment
SPMPPSO	Small Population Based Modified Parallel Particle Swarm Optimization
SPSO	Simplex Particle Swarm Optimization
AMEA	Adaptive Motion Estimation Algorithm
MA	Memetic Algorithm
CSO	Cat Swarm Optimization
AFSA	Artificial Fish Swarm Algorithm
ABC	Artificial Bee Colony
DE	Differential Evolution
HS	Harmony Search
SSIM	Structural Similarity
PSNR	Peak Signal to Noise Ratio
NP	Number of Population
B	Parent Block
W	Search Parameter
B_{best}	Best Block
V	Mutation Vector
F	Mutation Probability
U	Utility block
CP	Crossover Probability

r	Random
Prob	Probability
HMS	Harmony Memory Size
HMCR	Harmony Memory Consideration Rate
PAR	Pitch Adjustment Rate
BW	Distance Bandwidth
NI	Number of Improvisations
EA	Evolutionary Algorithm
B_{new}	New Block
B_{worst}	Worst Block
F_employed	Mutation Probability used in the Employed bee phase of hybrid ABCDE
F_onlooker	Mutation Probability used in the Onlooker bee phase of hybrid ABCDE
D	Dimension
C	Counter
QCIF	Quarter Common Intermediate Format
CIF	Common Intermediate Format
BBGD	Block Based Gradient Descent Search
NE	Neighborhood Elimination
ND	New Pixel-Decimation
LWG	Light Weight Genetic Search