Modified Cuckoo Search Algorithm for Motion Vector Estimation

Suvojit Acharjee
Electronics and Telecommunication Department, Jadavpur University, India
acharjeesuvo@gmail.com

Sheli Sinha Chaudhuri
Electronics and Telecommunication Department, Jadavpur University, India
shelism@rediffmaill.com

Abstract: *Motion estimation and motion compensation are the accepted process in H.264 and H.265 video coding standard to reduce temporal redundancy. Several fast block matching algorithms have been developed to reduce the calculation cost in the motion estimation process. But quick block matching algorithms often lead to a local minimum. Several researchers used different population-based nature-inspired algorithms to perform block matching. Algorithms like genetic algorithm, differential evolution, particle swarm optimization were used in numerous motion estimation algorithms. Different algorithms used a fitness approximation strategy to reduce computation cost. Jaya algorithm-based block matching is the most efficient block matching algorithm in the available literature. Jaya algorithm is free from algorithmic specific parameter which speeds up the process. This article proposes a few modifications to the traditional cuckoo search algorithm and then, a block matching algorithm was proposed based on the modified cuckoo search algorithm. Fitness approximation, adaptive termination, and zero motion prejudgment modules were used with the modified cuckoo search algorithm to reduce the number of redundant calculations. The performance of the proposed algorithm was compared with the exhaustive search algorithm and other benchmarking algorithms in terms of Peak Signal to Noise Ratio (PSNR), Structure Similarity Index (SSIM), and average search point required to calculate a motion vector for a block. The proposed algorithm delivers better performance compared to the benchmarking algorithms.*

Keywords: *Block matching algorithm, motion estimation, cuckoo search, video compression, metaheuristic*.

Received July 2, 2021; accepted September 20, 2022
https://doi.org/10.34028/iajit/20/3/6

1. Introduction

During the last few years, the use of video content witnessed a huge leap compared to other multimedia materials due to better connectivity and advancements of social media. Different video coding standards have been developed to store or transmit video contents efficiently. All international video coding standards accepted Block Matching Algorithm (BMA) [1] for the motion estimation process to reduce temporal redundancy in video sequences. Other motion estimation technique like pel-recursive algorithm [25], optical flow [4] etc., are more computationally expensive than BMA. Furthermore, the use of motion estimation is not limited to reducing temporal redundancy; it is also used in other applications such as action recognition [27, 30], autonomous driving [26], and so on.

In BMA, a frame divided into non-overlapping equal-sized blocks. Then the motion of the blocks in the current frame is calculated by comparing the position of the block at search area defined in the reference frame at a distance of +/- d pixels around the location of the block (as shown in Figure 1). Comparison between the blocks of the present frame and reference frame is performed using any of the metric [2] defined in Equations (1), (2), and (3): Sum of Absolute Difference (SAD), Mean Square Error (MSE) and Mean of Absolute Difference (MAD).

The type of encoding employed for the frame influences the choice of reference frame. Before the beginning of video encoding, every video sequence is divided into multiple Groups Of Pictures (GOP). The first frame of every GOP is the intra coded frame or I-frame which only use intra prediction to reduce the spatial redundancy inside the frame. Therefore, the motion estimation processes are not available for I-frames. The predicted frames or P-frames use inter and intra coding to reduce the temporal and spatial redundancy. The first P-frame inside a GOP uses the I-frame as its reference frame in motion estimation process. Other P-frames inside the GOP use the previous P-frames as their reference frame. The Bi-directional frames or B-frames in the GOP use two reference frames for motion estimation. Previous I-frame or P-frame and post P-frames are used as two reference frames by B-frame.

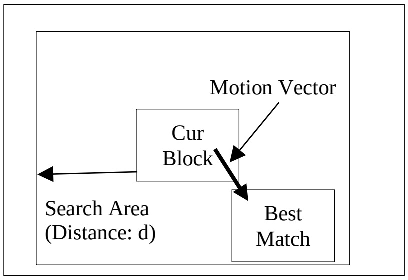


Figure 1. Representation of search window and motion vector in the reference frame.

$$SAD = \sum_{i=1}^{N}\sum_{j=1}^{N}|curblock(i,j) - refblock(i,j)| \quad (1)$$

$$MSE = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}[curblock(i,j) - refblock(i,j)]^2 \quad (2)$$

$$MAD = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}|curblock(i,j) - refblock(i,j)| \quad (3)$$

Exhaustive Search (ES) algorithm [3] produces the best motion vector among all block matching algorithm. ES compares all the possible blocks in the search area for the best match. This brute force search has a very high computational complexity and is not applicable for real-time video coding. Jain and Jain [12] first developed a fast block matching algorithm which used a rectangular pattern to pick a subset of search points from the search area. Few other algorithms like Adaptive Rood Pattern Search (ARPS) [16], block-based gradient descent (BBGD) [15], etc., reduces the search points to increase the computation speed. All these fast block matching algorithms developed on the assumption that error monotonically decreases as the search point moves towards global minima.

But Chow and Liou [6] proved that the fast block matching algorithm often ends up in local minima. Enhanced Predictive Zonal Search (EPZS) [23] exploited the Spatio-temporal relation between frames. The surrounding blocks of interframe or intraframe predicted the initial search points in the algorithm. This prediction helps the algorithm to avoid the local minima problem as most of the cases; the predicted point is nearer to global minima. But sometimes, the estimate leads to a false search point and distorted motion vector.

Population-based Evolutionary Algorithms (EA) are an alternate approach to find global minima. Population-based algorithms distribute the initial search points uniformly across the search space. This strategy helps the EA to avoid local minima trapping. But search problems with smaller search space (i.e., motion estimation) are not suitable for evolutionary algorithms as these algorithms often have a premature convergence in smaller search space. The most computationally expensive steps in an EA are

a) Random number generation.
b) Fitness calculation of new solutions in successive iteration.

The absence of an adaptive early termination strategy in traditional EA also increases the burden of unnecessary computation.

Chow and Liou [6] first proposed a motion search process based on the genetic algorithm. In Genetic Motion Search (GMS) chow and Liou [6] introduced a dynamic population control scheme to avoid the premature convergence of solution. Lightweight Genetic Motion Search (LGMS) [14] aims to reduce the computational overhead of genetic algorithm by replacing the random number generation with a deterministic approach and altogether abandoning crossover operation. Cuevas *et al*. [7, 8] proposed a fitness estimation strategy through search history preservation to reduce the fitness computation in successive iterations. Pandian *et al*. [18] defined a new adaptive termination strategy to avoid redundant calculations. Dash *et al*. [9] proposed the most state-of-the-art block matching process based on Jaya algorithm [20] which did not employ any algorithmic specific parameter to reduce overall computation time. This algorithm uses only the general parameters of evolutionary algorithms like the size of the initial population and maximum iteration number. Dash *et al*. [9] also employed other strategies like fitness approximation, early termination, and zero motion prejudgment [16] to reduce the calculation cost of Jaya algorithm further. Jaya algorithm moves the new solution closer towards the best solution and far from the worst solution. But significant drawback of Jaya algorithm-based block matching process is the high use of random numbers in the process of new solution generation as described in Equation (4).

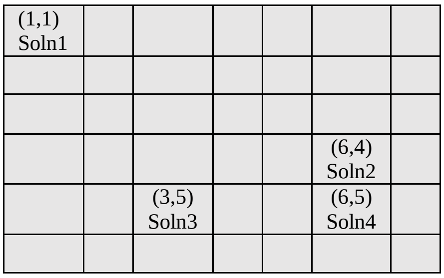
$$S'_{l,m,n} = S_{l,m,n} + rand()(S_{l,best,n} - S_{l,m,n}) - rand()(S_{l,worst,n} - S_{l,m,n}) \quad (4)$$

$S_{I,m,n}$ represents the lth element of kth solution at nth iteration and $S_{I,best,n}$ and $S_{I,worst,n}$ represents the best and worst solution of nth iteration.

Jaya algorithm-based block matching process sometimes observes a deadlock situation when it comes across the best solution of a particular iteration at the edge of the search area. In the next iteration, equation four will push the new solution out of the search area while the algorithm is calculating the replacement of best solution (explained in Figure 2). Video with complex motion often encounters such a deadlock situation. Dash *et al*. [9] have not discussed any steps to mitigate such a deadlock scenario.

Cuckoo search algorithm [29] was earlier used by Bhattacharjee and Kumar [5] for block matching process where the worst search points replaced by new ones were generated using Levy flight [24] operation. In this paper, a modified hybrid cuckoo search algorithm was proposed, which removes the use of algorithmic specific parameter (discovery probability of cuckoo's egg by host bird) with a deterministic evolution factor. Based on the modified cuckoo search algorithm, a block matching algorithm is introduced in this article.

Proposed BM algorithm halves the use of random numbers to reduce the computational burden of traditional cuckoo search. Moreover, the fitness approximation strategy, adaptation of early termination, and zero motion prejudgment are used together to lessen the computation cost further. An overview of the traditional cuckoo search algorithm, fitness approximation strategy, zero motion prejudgment, and adaptive termination strategy have been presented in next section. Part 3 will outline the modified cuckoo search algorithm and the block matching process based on a modified cuckoo search. Section four will provide a detailed performance analysis of the proposed algorithm and comparison with most state-of-the-art algorithms. The last segment will conclude the overall finding from the article and also outline the future challenges and research opportunities related to the algorithm.



Let us assume the position of solution at any iteration is (1,1), (6.4), (3,5), (6,5)
Position of Best Solution Soln1 is (1,1)
Position of Worst Solution Soln3 is (3,5)
So first new solution according to Jaya Algorithm
newsoln1=soln1-rand()(soln3-soln1)+ rand()(soln1-soln1)
= soln1-rand()(soln3-soln1)
Position of newsoln1=[(1-2*rand()),(1-4*rand())]=(x,y) where x,y<1

Figure 2. Deadlock situation in Jaya algorithm.

2. Overview

2.1. Cuckoo Search via Levy Flight

Yang and Deb [29] developed the cuckoo search algorithm, which simulates the unusual breeding behaviour of cuckoo. Cuckoo often lays eggs on another host bird's nest. Cuckoo removes the eggs of the host bird to reduce the competition and increases the survival probability of Cuckoo's egg. There is a certain probability that the host bird might recognize the alien eggs and destroy them. Based on this observation, the cuckoo search algorithm developed with three simple rules.

1. Each cuckoo can lay a single egg in any randomly chosen nest at any point of time. There are a fixed number of host nests from which cuckoo can select randomly.
2. The best nests will survive for the next generation.
3. There is a probability P [0-1] that host bird might find out about the alien eggs. Then the host bird completely abandons the nest and moves to a new one.
4. Egg represents a solution. For simplicity, it was assumed that each nest could have a single egg. One levy flight operation performed to generate a new solution V^{t+1} from the older one V^t. Levy flight is more efficient than the Markov chain in exploring the search space.

$$V^{t+1} = V^t + Levy\ Flight$$

Algorithm 1: Pseudocode for Cuckoo search via Levy Flight

```
begin
  Generate initial solution randomly.
  Calculate the fitness of the solution.
  Arrange the solution based on their fitness.
  while (iteration<maxIteration or stopping criteria)
      Get a cuckoo (i) using Levy Flight
      Evaluate the fitness of the solution (Costi)
      Choose a random nest (j) among all nests.
      if (Cost_i < Cost_j)
          Replace j by new solution.
      end if
      Keep best solutions
      Replace fraction of worse solutions with new one.
      Rank the solution and find the current best.
      Increase iteration by one
  end While
  Publish the last best solution
end
```

2.2. Fitness Approximation

One of the most computationally expensive steps in the evolutionary algorithm is fitness calculation at successive iterations. Fitness Approximation is a strategy to reduce the computation cost associated with fitness calculation by approximating the fitness value. Several fitness approximation technique [17, 22] were developed to speed up the evolutionary algorithms. The performance mostly depends on the quality and size of the dataset used to train the approximation models.

Cuevas *et al*. [7, 8] proposed a simple local approximation strategy based on Nearest-Neighbour Interpolation (NNI). According to the technique, fitness will be calculated for a few solutions, and rest will be approximated. Three simple rules were defined to evaluate or approximate the fitness (illustrated in Figure 3).

1. Fitness will be evaluated based on fitness function if the New Solution (NS) is closer than a specified distance D from the Best Solution (BS) among all other solutions stored in the table. (New Solution 1 in Figure 3)
2. Fitness will be evaluated based on actual fitness function if a NS has a distance more than D from its Closest Solution (CS) stored in the table. (NS 2 in Figure 3)
3. Fitness will be approximated using NNI if NS has distance smaller than D from its CS stored in the

table, and the CS is not the BS. (NS 3 in Figure 3) NNI will assign the fitness of the closest solution as the fitness of the new solution

$$(cost_{new-solution} = cost_{closest-solution}) \tag{5}$$

2.3. Zero Motion Prejudgment

Video content with smaller and moderate motion often observes area with zero action in successive frames. The calculation of Motion Vectors (MVs) for this type of block is unnecessary. A simple comparison based on SAD between the block in the present frame and reference frame can reveal such stationary blocks which are having zero MV. Motion search algorithm will exclude these fixed blocks. Video sequence like "Akiyo" and "Container" have more than 90% stationary blocks (as in Table 7). Excluding the stationary blocks from motion search, is known as Zero Motion Prejudgment (ZMP), a term first coined by Nie and Ma [16]. They projected a conservative SAD threshold (Tzmp) 512 to separate between regular block and stationary block.

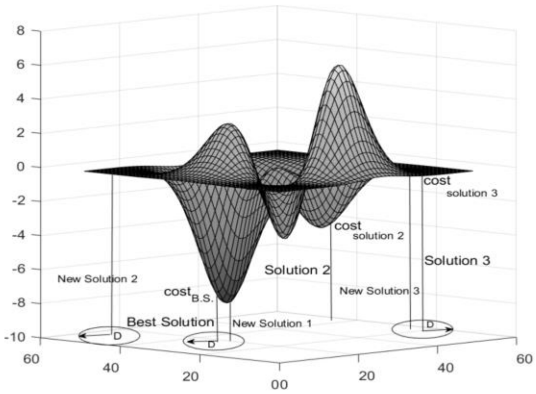


Figure 3. Fitness approximation strategy.

2.4. Adaptive Termination Strategy

Evolutionary algorithm terminates only when the error is stabilized or maximum iteration is reached. The adaptive termination strategy ends the algorithm based on problem-specific criteria which are defined carefully. Pandian *et al*. [18] proposed to terminate the algorithm if the global best solution is the same as the current block position. In this paper, another termination strategy is proposed, which will compare the SAD between the current block and global best solution with a predefined threshold (T_s). The predefined threshold (T_s) is one-third of the proposed T_{zmp} by Nie and Ma [16].

3. Proposed Algorithm

3.1. Cuckoo Breeding Behaviour

Cuckoo has a parasitic reproduction strategy where female cuckoo lays eggs on the nest of other host species. The eggs should look precisely similar to the host bird's egg in terms of colour, shape, and pattern. Potential host birds evolve quickly to stop the parasitic incident. An evolutionary race continues between host bird and cuckoo. As time progresses, cuckoo's egg becomes more similar to the host birds' egg. [11] Different cuckoos can lay eggs of different shapes, colours, and pattern. But individual cuckoo specializes on a single host species.

3.2. Modified Cuckoo Search

Proposed modified cuckoo search algorithm will follow these four rules:

- The number of cuckoos is same as the number of nests.
- In every iteration, each cuckoo will target to replace the host egg of a specific nest. The fittest egg between host and cuckoo will survive for the next iteration.
- As each cuckoo targets a single nest, the eggs of the cuckoo will have an uncanny resemblance to the host species. There is almost zero probability that host species will be able to discover alien eggs and destroy them.
- The evolutionary race between host species and cuckoo is represented using an evolutionary factor (α) which can be calculated by the ratio between the fitness of i^{th} solution ${V_i}^t$ and worst solution ${V_{worst}}^t$ in i^{th} iteration. α_i will have a smaller value if ${V_i}^t$ has better fitness, and i^{th} solution needs smaller evolution. The evolution factor in the proposed modified cuckoo search algorithm pushes the low-quality solution far away from the worst solution. But it helps the better solutions to look for more quality result around. Thus, evolution factor strikes a balance between exploration and local search.

During the generation of new solution ${V^{t+1}}_i$ for $(t+1)_{th}$ iteration, ${V_i}^t$ will be pushed away from the worst-performing solution ${V^t}_{worst}$ by a factor α_i and a random element will be added using Levy Flight. The deadlock situation faced in Jaya Algorithm (as explained in Figure 2) is avoided by adding the random element in the new solution generation in the proposed algorithm. Random element also adds a variation in local search.

$$V_i^{t+1} = V_i^t - \alpha_i(V_i^t - V^t{}_{worst}) + Levy\ Flight \tag{6}$$

$$\alpha_i = \frac{fitness_i{}^t}{fitness_{worst}{}^t} \tag{7}$$

The modified cuckoo search algorithm is summarized in the pseudo-code shown in Algorithm (2). The differences between the cuckoo search and modified cuckoo search algorithm illustrated in Table 1.

Algorithm 2: Pseudocode for Modified Cuckoo search

begin

Generate n number of initial solution using random/biased distribution.

Calculate the fitness of the solutions.

Arrange the solution based on their fitness.
while (iteration< maxIteration or stopping criteria)
Get n number of cuckoos (V_i^{t+1}) using equation 6.
Evaluate the fitness of n cuckoos
for i=1: n
Compare i_{th} cuckoo with i_{th} nest
if ($V_i^{t+1} < V_i^t$)
Replace V_i^t with V_i^{t+1}.
else
Discard V_i^{t+1}
end if
end for
Rank the solution and find the current best and worst.
Increase iteration by one
end while
Publish the last best solution
end

Table 1. Differences between cuckoo search and modified cuckoo search algorithm.

	Cuckoo Search [29]	Proposed Modified Cuckoo Search
Generation of new solution	One levy flight around V_i^t.	V_i^t will be pushed away from the worst-performing solution $V^t{}_{worst}$ by a factor α_i and a random element added using Levy Flight
Evolutionary factor (α)	No evolutionary factor. A scaling factor taken to add up with Levy Flight, which varies between [0-1]. For most applications, this scaling factor is one.	The evolutionary race between host species and cuckoo will be represented using an evolutionary factor (α). It is the ratio between the fitness of V_i^t and $V_{worst}{}^t$.
Algorithmic specific parameter	Probability of discovery (P) that host bird might find out about the alien eggs and destroy them. P varies between [0-1]. The value P controls the exploration rate of the algorithm.	No algorithmic specific parameter. As each cuckoo targets only one nest, the eggs of the cuckoo will have an uncanny resemblance to the host species. There is almost zero probability that host species will be able to discover alien eggs and destroy them. Evolution factor strikes a balance between exploration and local search.

3.3. BM with Modified Cuckoo Search

3.3.1. Zero Motion Prejudgment

SAD between the block in the present frame and reference frame is compared with a predefined threshold to decide whether the block is stationary or not. The threshold for zero motion prediction is taken as 500 for the proposed algorithm, though a limit of 1000 for zero motion prejudgment has a minimal impact on the output quality of the algorithm.

3.3.2. Generation of Initial Population

In traditional cuckoo search, the initial population were selected randomly. This process helps to distribute the solution uniformly over the search space. However, Li and Xiao [13] and Xiao [28] proved that incorporating domain knowledge to generate initial population results quick convergence of the algorithm. A survey by Cuevas *et al*. [7, 8] discovered that video with complex motion having half MVs nearer to the origin. Therefore, in this paper, the initial solutions are generated from any of the fixed pattern shown in Figure 4. The initial populations are allocated in each direction to preserve the equal opportunity of finding the motion vector from any direction.

3.3.3. Fitness Calculation and Ranking

For the initial population, fitness will be calculated using the original fitness function. But from the next iteration, eligibility will be calculated according to the fitness approximation strategy. Arrange the solutions based on their fitness and decide about the best and worst solution.

3.3.4. Generation of New Solution in Iterations

Random number generation added extra computation burden in BM. Production of new solution using equation 6 requires two random number generation as MV has vertical and horizontal component. Instead of generating two random number, the algorithm proposed to select a solution randomly (V_n) from the search history of previous generations {V}. The random selection of solution will reduce the use of the random numbers. At the same time, it will maintain the balance between exploration and localization.

V_n = randomly selected from $\{V\}$ and $V_n \neq V_{worst}$ and $V_n, V_i, V_{worst} \in \{V\}$

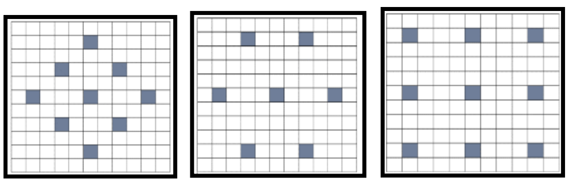
$$V_i^{t+1} = V_n - \alpha_i\,(V_i^t - V_i^{worst}) \tag{8}$$

3.3.5. Early Termination

The above algorithm will terminate if one of the following conditions is satisfied.

1. The algorithm reached the maximum iteration.
2. The algorithm is already converged.
3. The fitness of the best solution has a value less than the predefined threshold (T_s).
4. Global best solution is the same as the current block position.

Pseudocode for BM with modified cuckoo search is explained in Algorithm (3).



a) Diamond. b) Hexagonal. c) Square.

Figure 4. Search patterns.

Algorithm 3. BM with Modified Cuckoo search

Begin
if (SAD (refBlock, curBlock) <T_{zmp})
Set mv as {0,0}
else
Generate initial solution from any of the fixed pattern.
Calculate the fitness of the solutions normally

Arrange the solution based on their fitness.
while (stopping criteria matched)
Get n number of solution (V_i^{t+1}) *using equation 8.*
Evaluate the fitness of solutions using NNI.
for i=1: n
if ($V_i^{t+1} < V_i^{t}$)
Replace V_i^{t} *with* V_i^{t+1}.
else
Discard V_i^{t+1}
end if
end for
Rank the solution and find the current best and worst.
Increase iteration by one
end While
Set the motion vector as (position of block-best solution)
end

4. Result and Discussion

4.1. Dataset

Experimental analysis of the algorithm performance was evaluated based on standard video dataset available for testing video coding standards. Video sequence with different motion characteristics is used to test the robustness of the proposed algorithm. The motion vector is calculated based on only the luminance part of the test video sequences. Motion characteristics and resolution details of few video sequences briefed in Table 2.

Table 2. Details of used video sequences.

Name of Video Sequence	Format	Frame Size	Total Frames	Motion Details
Container	QCIF	176X144	300	Low
Carphone	QCIF	176X144	381	Moderate
Foreman	QCIF	176X144	398	Moderate
Akiyo	QCIF	176 X 144	300	Moderate
Football	QCIF	352 X 240	300	High
Tennis	SIF	352 X 240	150	High
Crowd-Run	HD	1280 X 720	500	High
Old-Town Cross	HD	1280 X 720	500	High

4.2. Benchmarking Algorithm

ES [3] is the default motion estimation algorithm in H.265 reference software. As previously stated, this brute force technique consistently gives the best motion estimate despite demanding the most computational effort. Every rapid motion estimation technique strives to achieve motion estimation similar to ES with less computation effort than ES. Therefore, performance of the proposed algorithm is compared with the performance of ES.

Furthermore, one earlier important study [7] and two current important research work [9, 10] on optimized block matching algorithms were used to evaluate the performance of the proposed approach. The differential evolution-based block matching method [7] was one of the first to include metaheuristic optimisation into motion estimation. For H.264 and H.265 [1] video coding standards, the Jaya Algorithm-based BMA [9] outperforms existing fast BMA. Another Differential Evolution-Based Block Matching (DEBM) [10] method achieved encouraging results and was employed as a benchmark in this work.

4.3. Initial Parameters

For the experiments, the block size is 16x16 for the proposed algorithm as well as other benchmarking algorithms. This is the standard macroblock size in block matching algorithm and used by several other authors [7, 8, 21]. The survey on distribution of motion vector at various distances from the search window centre by Cuevas *et al*. [7] found that the majority of motion vectors had a value less than or equal to 7. Therefore, the search window's border is set to be 7 pixels away from the macroblock's boundary. Maximum iteration size for the proposed algorithm and other benchmarking algorithms is 50.

4.4. Performance Analysis Based on Image Quality

Performance of the proposed algorithm is measured based on five parameters, i.e., Peak Signal to Noise Ratio (PSNR), Structure Similarity Index (SSIM), Average Search Count, Degradation in PSNR quality (DP) and Speed Improvement Ratio (SIR). PSNR and SSIM both measure the closeness of the reconstructed frame with the actual frame. Peak Signal to noise ratio compares the luminance difference between the reconstructed frame and the actual frame where SSIM finds the structural similarity between two frames. Identical two frames have a unit SSIM and infinite PSNR. PSNR and SSIM can be calculated using Equations (9) and (10).

$$PSNR = 20\log\left(\frac{255}{\sqrt{\left||Image1 - Image2|\right|^2}}\right) \tag{9}$$

$$SSIM = \frac{(2u_1u_2 + C1)(2O_{12} + C2)}{(u_1^2 + u_2^2 + C1)(O_1^2 + O_2^2 + C2)} \tag{10}$$

u1 and *u2* are the mean intensity, and *O1* and *O2* are the variance of image 1 and 2. *C1* and *C2* are constants. Degradation in PSNR and Degradation in SSIM observes the deterioration in output quality of another algorithm compared to the exhaustive search algorithm. It is the degree of difference between the output frame of ES and alternate algorithm.

$$D_P = -\frac{(PSNR_{ES} - PSNR_{OtherAlgo})}{PSNR_{ES}} \times 100 \tag{11}$$

$$D_S = -\frac{(SSIM_{ES} - SSIM_{OtherAlgo})}{SSIM_{ES}} \times 100 \tag{12}$$

From Table 3, it is clear that the output of the proposed algorithm deviates very little from the ES algorithm.

Small degradation in PSNR (DP) value (2%-3%) for video with complex motion like tennis and foreman establishes robustness of the algorithm. Table 4 provides a comparison of SSIM performance between the proposed algorithm and other algorithms. The proposed algorithm performed better than any different meta-heuristic algorithm-based block matching process with a minuscule (<2%) degradation in SSIM (DS) value compared to the ES algorithm. Jaya Algorithm based BM algorithm has a better efficiency for HD video sequence compared to test zone search [19] algorithms. The proposed algorithm has shown a better or almost similar output for HD video sequence in terms of PSNR and SSIM compared with Jaya algorithm-based BM (Shown in Table 6).

Table 3. Comparison of PSNR and DP between the proposed algorithm and other algorithms.

	Akiyo		Carphone		Container		Foreman		Tennis		Football	
	PSNR	DP (%)	PSNR	DP (%)	PSNR	DP (%)	PSNR	DP (%)	PSNR	DP (%)	PSNR	DP (%)
ES [3]	45.25	0.00	32.2	0	42.5	0	32.3	0	29.5	0	29.1	0
DEBM [10]	44.27	-2.16	31	-3.7	42.37	-0.3	31.13	-3.7	28.67	-2.8	28.4	-2.4
DE [7]	44.67	-1.29	30.1	-6.7	42.5	-0	30.4	-5.7	27.8	-5.5	28.6	-1.6
Jaya-Square [9]	43.75	-3.33	29.2	-9.5	42.5	-0	29.4	-8.8	27.9	-5.2	28.7	-1.5
Jaya-Diamond [9]	43.75	-3.33	29.2	-9.3	42.5	-0	29.7	-7.8	28.4	-3.6	28.7	-1.6
Modified Cuckoo (Square)	45	-0.47	30.8	-4.3	42.5	-0	31.5	-2.3	28.7	-2.5	28.9	-0.6
Modified Cuckoo (Diamond)	45	-0.48	31.1	-3.6	42.5	-0	31.6	-2.1	28.6	-2.9	28.9	-0.8
Modified Cuckoo (Hexagonal)	44.9	-0.73	31.1	-3.5	42.5	-0	31.6	-2.2	28.6	-3	28.8	-1.2

Table 4. Comparison of SSIM and Ds between proposed algorithm and other algorithms.

	Akiyo		Carphone		Container		Foreman		Tennis		Football	
	SSIM	Ds (%)	SSIM	Ds (%)	SSIM	Ds (%)	SSIM	Ds (%)	SSIM	Ds (%)	SSIM	Ds (%)
ES [3]	0.99	0.00	0.96	0	0.99	0	0.95	0	0.92	0	0.88	0
DEBM [10]	0.99	0.00	0.94	-1.43	0.99	0	0.9	-5.2	0.9	-1.7	0.87	-1.1
DE [7]	0.99	0.00	0.93	-2.41	0.99	-0	0.93	-1.78	0.91	-0.84	0.88	-0.63
Jaya-Square [9]	0.99	-0.03	0.92	-3.95	0.99	-0	0.92	-3.15	0.91	-0.76	0.88	-0.98
Jaya-Diamond [9]	0.99	-0.03	0.92	-3.85	0.99	-0	0.92	-2.67	0.92	-0.33	0.88	-0.95
Modified Cuckoo (Square)	0.99	0.00	0.94	-1.43	0.99	-0	0.95	-0.32	0.92	-0.217	0.88	-0.19
Modified Cuckoo (Diamond)	0.99	0.00	0.95	-1.29	0.99	-0	0.95	-0.23	0.92	-0.214	0.88	-0.37
Modified Cuckoo (Hexagonal)	0.99	0.00	0.95	-1.34	0.99	-0	0.95	-0.44	0.92	-0.002	0.88	-0.61

4.5. Performance Analysis based on Search Speed

The speed of execution for a BM algorithm calculated by the number of search points required to calculate the best motion vector of a block. ES search algorithm has the best motion vector and slowest execution speed among all block matching algorithms. In Table 5, the proposed algorithm was compared with the ES algorithm and other benchmarking algorithms in terms of the average search point required for a block. Search speed with proposed algorithm for video with small motion has witnessed an improvement of 99% to ES. Test video sequences having a moderate or complex movement (like Carphone and tennis) also observed an increase in search speed by 95% using the proposed algorithm.

$$SIR = \frac{SC_{ES} - SC_{Other\ Algorithm}}{SC_{ES}} \times 100 \quad (13)$$

Improvement in search speed is observed because of zero motion prejudgment and early termination strategies. Percentage of stationary blocks in every video sequence, which were skipped from the calculation of MVs, is detailed in Table 7. Video with slow and moderate motion has the highest percentage of stationary blocks, which results in high SIR.

Table 5. Comparison of average search point and speed improvement ratio between the proposed algorithm and other algorithms.

	Akiyo		Carphone		Container		Foreman		Tennis		Football	
	SC	SIR (%)	SC	SIR (%)	SC	SIR (%)	SC	SIR (%)	SC	SIR (%)	SC	SIR (%)
ES [3]	245	0	234	0	234	0	234	0	244	0	234	0
DE [7]	3.7	98.5	8.68	96	8.07	97	8.46	96.4	8.95	96	8.35	96
Jaya-Square [9]	3.4	98.6	8.52	96	7.34	97	8.36	96.4	8.99	96	8.17	97
Jaya-Diamond [9]	3.4	98.6	8.4	96	7.34	97	8.26	96.5	8.83	96	8.01	97
Modified Cuckoo (Square)	0.1	99.9	10.2	96	1.4	99	10.5	95.5	13.9	94	11.1	95
Modified Cuckoo (Diamond)	0.1	99.9	8.11	97	1.12	100	8.23	96.5	11.3	95	8.84	96
Modified Cuckoo (Hexagonal)	0.1	99.9	9.79	96	1.38	99	9.89	95.8	13.6	94	10.4	96

Table 6. Comparison between Jaya Algorithm and proposed algorithm-based BM for HD video sequence.

	Old_town_cross		Crowd_run	
	PSNR	**SSIM**	**PSNR**	**SSIM**
Jaya-Square [9]	26.22	0.72	22.16	0.71
Jaya- Diamond [9]	26.42	0.73	22.55	0.73
Modified Cuckoo (Square)	26.26	0.72	22.13	0.71
Modified Cuckoo (Diamond)	26.46	0.72	22.26	0.71
Modified Cuckoo (Hexagonal)	26.29	0.72	22.07	0.71

Table 7. Percentage of stationary blocks in every video sequence.

Video Name	$\frac{\boldsymbol{SkippedBlock}}{\boldsymbol{TotalBlock}} \times \mathbf{100}\,\%$
Akiyo	90.33838
Carphone	34.97276
Container	91.33333
Foreman	17.50281
Tennis	27.99192
Football	23.09764

5. Conclusions and Future Scope

In this article, a modified cuckoo search algorithm was proposed which is free from any algorithmic specific parameter (like discovery probability in traditional cuckoo search). The proposed algorithm only depends on general parameters of population-based search strategy (like maximum iteration and population size). Modified cuckoo search pushes the new solutions away from the worst solution of the present generation with an added random factor calculated using levy flight. BM algorithm based on modified cuckoo search showed a significant improvement in performance over the benchmarking algorithms. The proposed algorithm performed with high efficiency for different motion variation.

Though the proposed algorithm has a significant improvement over the benchmarking algorithms, there are several scopes of development in the proposed algorithm. The algorithm can be tested with adaptive window pattern and size for initial population generation. Initial population by prediction in the proposed algorithm may result a quick convergence. The modified cuckoo search algorithm can be applied in several other engineering problems to assess the efficiency of the modifications over traditional algorithm.

References

[1] Abramowski A., "Towards H. 265 Video Coding Standard," *in Proceedings of the Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments*, Poland, pp. 387-393, 2011.

[2] Acharjee S., Chakraborty S., Karaa W., Azar A., and Dey N., "Performance Evaluation of Different Cost Functions in Motion Vector Estimation," *International Journal of Service Science, Management, Engineering, and Technology*, vol. 5, no. 1, pp. 45-65, 2014.

[3] Barjatya A., "Block Matching Algorithms for Motion Estimation," *IEEE Transactions Evolution Computation*, vol. 8, no. 3, pp. 225-239, 2004.

[4] Barron J., Fleet D., and Beauchemin S., "Performance of Optical Flow Techniques," *International Journal of Computer Vision*, vol. 12 no. 1, pp. 43-77, 1994.

[5] Bhattacharjee K. and Kumar S., "A Novel Block Matching Algorithm Based on Cuckoo Search," *in Proceedings of the 2nd International Conference on Telecommunication and Networks*, Noida, pp. 1-5, 2017.

[6] Chow K. and Liou M., "Genetic Motion Search Algorithm for Video Compression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 3, no. 6, pp. 440-445, 1993.

[7] Cuevas E., Zaldivar D., Pérez-Cisneros M, and Oliva D., "Block-matching Algorithm Based on Differential Evolution for Motion Estimation," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 1, pp. 488-498, 2013.

[8] Cuevas E., Zaldívar D., Pérez-Cisneros M., Sossa H., and Osuna V., "Block Matching Algorithm for Motion Estimation Based on Artificial Bee Colony (ABC)," *Applied Soft Computing*, vol. 13, no. 6, pp. 3047-3059, 2013.

[9] Dash B., Rup S., Mohanty F., and Swamy M., "A hybrid Block-Based Motion Estimation Algorithm Using JAYA for Video Coding Techniques," *Digital Signal Processing*, vol. 88, pp. 160-171, 2019.

[10] Dixit A., Mani A., and Bansal R., *Intelligence Enabled Research*, Springer Singapore, 2021.

[11] Fossøy F., Sorenson M., Liang W., Ekrem T., Moksnes A., Møller A., Rutila J., Røskaft E., Takasu F., Yang C., and Stokke B., "Ancient Origin and Maternal Inheritance of Blue Cuckoo Eggs," *Nature Communications*, vol. 7, no. 1, pp. 10272, pp. 1-6, 2016.

[12] Jain J. and Jain A., "Displacement Measurement and its Application in Interframe Image Coding," *IEEE Transactions on Communications*, vol. 29, no. 12, pp. 1799-1808, 1981.

[13] Li X., Xiao N., Claramunt C., and Lin H., "Initialization Strategies to Enhancing the Performance of Genetic Algorithms for the P-Median Problem," *Computers and Industrial Engineering*, vol. 61 no. 4, pp. 1024-1034, 2011.

[14] Lin C., and Wu J., "A Lightweight Genetic Block-Matching Algorithm for Video Coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 4, pp. 386-392, 1998.

[15] Liu L. and Feig E., "A block-based Gradient Descent Search Algorithm for Block Motion Estimation in Video Coding," *IEEE Transactions*

on Circuits and Systems for Video Technology, vol. 6, no. 4, pp. 419-422, 1996.

[16] Nie Y. and Ma K., "Adaptive Rood Pattern Search for Fast Block-matching Motion Estimation," *IEEE Transactions on Image Processing*, vol. 11, no. 12, pp. 1442-1449, 2002.

[17] Ong Y., Nair P., and Keane A., "Evolutionary Optimization of Computationally Expensive Problems via Surrogate Modeling," *AIAA Journal*, vol. 41, no. 4, pp. 687-696, 2003.

[18] Pandian S., Bala G., and Anitha J., "A Pattern Based PSO Approach for Block Matching in Motion Estimation," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 8, pp. 1811-1817, 2013.

[19] Parmar N. and Sunwoo M., "Enhanced Test Zone Search Motion Estimation Algorithm for HEVC," *in Proceedings of the International Soc Design Conference*, Jeju, pp. 260-261, 2014.

[20] Rao R., "Jaya: A Simple and New Optimization Algorithm for Solving Constrained and Unconstrained Optimization Problems," *International Journal of Industrial Engineering Computations*, vol. 7, no. 1, pp. 19-34, 2016.

[21] Salih Y. and George L., "Improved Hybrid Block-based Motion Estimation for Inter-frame Coding," *Circuits, Systems, and Signal Processing*, vol. 40, no. 7, pp. 3500-3522, 2021.

[22] Shi L. and Rasheed K., *Computational Intelligence in Expensive Optimization Problems*, Springer Berlin, pp. 3-28. 2010

[23] Tourapis A., "Enhanced Predictive Zonal Search for Single and Multiple Frame Motion Estimation," *in Proceedings of the Visual Communications and Image Processing*, San Jose, 4671, pp. 1069-1079, 2002.

[24] Viswanathan G., Afanasyev V., Buldyrev S., Havlin Sh., da Luz M., Raposo E., and Eugene Stanley H., "Lévy Flights in Random Searches," *Physica A: Statistical Mechanics and its Applications*, vol. 282, no. 1-2, pp. 1-12, 2000.

[25] Walker D. and Rao K., "Improved Pel-recursive Motion Compensation," *IEEE Transactions on Communications*, vol. 32, no. 10, pp. 1128-1134, 1984.

[26] Wang J., Liu T., Yuan Z., and Shang Y., "A Robust Saliency Integrated Method for Monocular Motion Estimation," *in Proceedings of the International Conference on Machine Vision and Applications*, Aichi, pp. 49-54, 2021.

[27] Wu L., Yang Z., Jian M., Shen J., Yang Y., and Lang X., "Global Motion Estimation with Iterative Optimization-Based Independent Univariate Model for Action Recognition," *Pattern Recognition*, vol. 116, pp. 107925, 2021.

[28] Xiao N., "A Unified Conceptual Framework for Geographical Optimization Using Evolutionary Algorithms," *Annals of the Association of American Geographers*, vol. 98, no. 4, pp. 795-817, 2008.

[29] Yang X. and Deb S., "Cuckoo Search via Lévy Flights," *in Proceedings of the World Congress on Nature and Biologically Inspired Computing*, Coimbatore, 2009.

[30] Zebhi S., Almodarresi S., and Abootalebi V., "Human Activity Recognition Based on Transfer Learning with Spatio-Temporal Representations," *The International Arab Journal of Information Technology*, vol. 18, no. 6, pp. 839-845, 2021.

Suvojit Acharjee is a Ph.D. scholar at Jadavpur University. He completed his M.Tech in Intelligent Automation and Robotics from Jadavpur University in 2012. He worked as Assistant Professor in National Institute of Technology, Agartala. He has a research interest on soft computing, video coding and signal compression. He has also worked on the application of non-volatile memory in neuro-computing. He has published several research articles in various reputed journals.

Sheli Sinha Chaudhuri has been serving the ETCE Department of Jadavpur University for the last 23 years. She currently holds the position of professor. She is a member of IEEE and IEI. She is involved in research related to the field of Machine Learning and Computer Vision. Her research efforts have resulted in the supervision of several PhD and Master's dissertation. Her research has been published in high-quality journals, book chapters, and conference proceedings.