

# Parallel Bacterial Foraging Optimization for Video Compression

K. M. Bakwad<sup>1</sup>, Dr. S.S. Pattnaik<sup>1</sup>, Dr. B. S. Sohi<sup>2</sup>, Dr. S. Devi<sup>1</sup>, M.R. Lohakare<sup>1</sup>

<sup>1</sup>National Institute of Technical Teachers' Training and Research, Chandigarh-160019, India

Email: km\_malikaforu@yahoo.co.in

<sup>2</sup>UIET, Punjab University, Chandigarh-160019, India

Email: shyampattnaik@yahoo.com

**Abstract**— Parallel Bacterial Foraging Optimization (PBFO), a new evolutionary soft computing tool, has been proposed for real time video compression. The convergence of BFO is very slow and its performance heavily degraded for real time processing. In BFO, best optimum value is updated after each fitness function evaluations but in PBFO best optimum value is updated after fitness function evaluations all bacteria. The PBFO is used to reduce computational time of motion estimation in video compression. In the proposed technique, the parallel computation of fitness function ensures the convergence of the optimum solution is very fast. The adaptive step size with prediction, zero motion vector and Von Neumann neighborhood topology implemented in PBFO ensures the best matching block in current frame computationally very fast. The chemotaxis and swimming step can be varied to save computational time and to enhance peak signal to noise ratio. The presented method saves computational time up to 96.59 % when compared with other published methods. The proposed method is new and computationally efficient and suitable for any real time applications.

**Index Terms**— Parallel Bacterial Foraging Optimization (PBFO), Motion Estimation, Peak Signal to Noise Ratio (PSNR), Computational Time, Convergence

## I. INTRODUCTION

Research in the optimization field is very active and new optimization methods are being developed continuously. Since 1960, Genetic Algorithm (GA) has been playing a dominant role in the optimization methods [1]. The Particle Swarm Optimizer (PSO) is a population-based optimization method developed by Dr. Eberhart and Dr. Kennedy in 1995. PSO is inspired by social behavior of bird flocking or fish schooling. It can handle efficiently arbitrary optimization problems [1]. Compared to GA, PSO is very easy to implement, converges fast and uses few parameters to adjust [1]. However, unlike GA, PSO has no evolution operators such as crossover and mutation. PSO has been successfully applied in many areas like function optimization, artificial neural network, fuzzy system control, image denoising etc [1][2]. Bacterial Foraging Optimization (BFO)[3][4] is another optimization technique proposed by K. M. Passino in 2002. BFO mimics foraging strategies of the E. coli bacterium cells. The accuracy of BFO is high as

compared to PSO and GA because of more number of parameters used for optimization but its speed is low.

Motion estimation is an important research topic in video processing as well as computer vision. Motion estimation is computationally expensive and resource hungry operation in video compression as it needs a large amount of computational resources [5]. Researchers have proposed several block-matching algorithms [5]-[11]. The exhaustive search (ES) or full search algorithm gives the highest peak signal to noise ratio amongst all block-matching algorithm but requires more computational time [5]. To reduce this computational time, many other methods are proposed i.e. Simple and Efficient Search (SES)[5], Three Step Search (TSS)[6], New Three Step Search (NTSS)[6], Four step Search (4SS)[7], Diamond Search (DS)[8], Adaptive Road Pattern Search (ARPS)[9] and Fast Motion Estimation for H.264/AVC in Walsh Hadmard Domain [10]. Further genetic algorithm has also been used for motion estimation [11]. However GA suffers from high computational time, more operators and the possibility of getting trapped in local minima [1]. In this paper, Parallel Bacterial Foraging Optimization is proposed for fast motion estimation in video sequence. The proposed PBFO technique is discussed in section II. The motion estimation using PBFO is explained in section III. Section IV provides experimental results comparing PBFO with other methods for motion estimation. The conclusion is given in section V.

## II PROPOSED PARALLEL BACTERIAL FORAGING OPTIMIZATION

The Bacterial Foraging optimization [3] is based on foraging (i.e. searching food) strategy of E. coli bacteria. In BFO, the optimization follows chemotaxis, swarming, reproduction, elimination and dispersal step to reach global minima until computational limitations are exceeded. BFO was used to calculate resonant frequency of rectangular microstrip antenna [4]. The BFO algorithm implemented on single processor is given in [3].

In BFO, best optimum value is updated asynchronously i.e. after each fitness evaluations. In PBFO, best optimum value is updated synchronously i.e. fitness function evaluations of all bacteria. During chemotaxis step fitness function evaluations are

performed parallel or serial. Serial fitness function evaluations require only one processor. Parallel fitness function evaluations requires multiprocessor nodes i.e. one master and some slaves. The number of slave node is equal to number of bacterium. Synchronization is required between master and slave nodes for evaluating fitness value and for updating the positions before starting the next iteration. Slave nodes should report fitness evaluations to the master node before starting next chemotaxis steps. The performances of the PBFO on multiprocessor or single processor will not lose convergence to the best solution but the computational cost will more on single processor. In PBFO, the optimization follows chemotaxis, swimming, tumbling, and reproduction steps until computational limitations are exceeded. The PBFO performs a local search through chemotaxis, swimming and tumbling and global search by step size equation introduced in chemotaxis steps. The step size equation (in step 5 of PBFO algorithm) consists of two terms. The first term is previous position and second term is the social component for global search. The difference of  $J_{best}(j,k)$  and  $J_{best}(j+1,k)$  is positive, if the previous fitness value is best compared to fitness value in the next step otherwise the difference is negative. The positive difference change the position of bacteria in swimming direction otherwise it will tumble. 'k'(chemotaxis step counter) in the step size equation is used to accelerate the speed of bacteria towards global best value. During reproduction of PBFO, the least healthy bacteria die and others split into two, which are placed in the same location. Jglobal is updated in each reproduction steps. The algorithm for PBFO is presented below.

*PBFO algorithm*

*Step1:* Initialize Parameters  $p, S, Nc, Ns, Nre, C(i), i=1,2,\dots,\dots,\dots, S$

where,

- $p$  = Dimension of search space
- $S$  = Number of bacteria in the population
- $Nc$  = Number of chemotaxis steps
- $Ns$  = Number of swimming steps
- $Nre$  = Number of reproduction steps
- $C(i)$  = Step size taken in the random direction specified by the tumble.
- $J(i, j, k)$  = Fitness value or cost of the i-th bacteria in the j-th chemotaxis and k-th reproduction steps.
- $\theta(i, j, k)$  = Position vector of the i-th bacterium in j-th chemotaxis step and k-th reproduction steps.
- $J_{best}(j, k)$  = Fitness value or cost of best position in the j-th chemotaxis and k-th reproduction steps.
- $J_{global}$  = Fitness value or cost of the global best position in the entire search space.

*Step 2:* Update the following parameters.

$$J(i, j, k)$$

$$J_{best}(j, k)$$

$$J_{global} = J_{best}(j, k)$$

*Step 3:* Reproduction Loop:  $k = k+1$

*Step 4:* Chemotaxis loop:  $j = j+1$

- a) Compute fitness function  $J(i, j, k)$  for  $i = 1, 2, 3, \dots, S$ .
- b) Update  $J_{best}(j, k)$ .
- c) *Tumble:* Generate a random vector  $\Delta(i) \in \mathbb{R}^p$  with each element  $\Delta_m(i)$   $m = 1, 2, \dots, p$ , a random number on  $[-1, 1]$
- d) Compute  $\theta$  for  $i = 1, 2, \dots, S$

$$\theta(i, j+1, k) = \theta(i, j, k) + C(i) \frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$$

e) *Swim*

- i) Let  $m = 0$  (counter for swim length)
- ii) While  $m < Ns$  (if have not climbed down too long.
  - o Let  $m = m+1$
  - o Compute fitness function  $J(i, j+1, k)$  for  $i = 1, 2, 3, \dots, S$
  - o Update  $J_{best}(j+1, k)$
  - o If  $J_{best}(j+1, k) < J_{best}(j, k)$  (if doing better),  $J_{best}(j, k) = J_{best}(j+1, k)$ . Compute  $\theta$  for  $i = 1, 2 \dots S$

$$\theta(i, j+1, k) = \theta(i, j, k) + C(i) \frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$$

Use this  $\theta(i, j+1, k)$  to compute the new  $j(i, j+1, k)$ .

- o Else, let  $m = Ns$ . This is the end of the while statement

*Step 5:* Change the position of bacteria by adding social component (*Step Size Equation*)

$$\theta(i, j+1, k) = \theta(i, j, k) + k * rand * (J_{best}(j, k) - J_{best}(j+1, k))$$

*Step 6:* If  $j < Nc$ , go to step 4. In this case, continue chemotaxis, since the life of bacteria is not over.

*Step 7:* The  $Sr=S/2$  bacteria with the highest cost function values die and other  $Sr=S/2$  bacteria with the best values split.

*Step 8:* Update  $J_{global}$  from  $J_{best}(j, k)$ .

*Step 9:* If  $k < Nre$ , go to step3 otherwise end.

III PBFO FOR MOTION ESTIMATION

Five bacteria are used in PBFO for motion estimation. The Von Neumann neighborhood topology shown in Fig. 1 is used in the search space of PBFO because the current macro block is closer to previous macro block in temporal and spatial domain. Due to neighborhood topology the convergence is very fast. In the proposed method, macro block is considered as a bacterium. The initial position of block to be searched in reference frame is same as block in current frame for which motion vector is found. In PBFO, the chemotaxis step and swimming

count is chosen one to save maximum computational time for motion estimation. In chemotaxis step the  $J_{best}$  is updated among five bacteria. The horizontal and vertical position (Motion Vector) of  $J_{best}$  block is added with step size to predict the best matching block in current frame. In swimming iteration  $J_{best}(j+1,k)$  is updated among five bacterium.  $J_{best}(j+1,k)$  and  $J_{best}(j,k)$  are compared and  $J_{best}(j,k)$  is updated. The best matching block in search space is  $J_{best}(j,k)$ . The cost required for finding best matching block or  $J_{best}$  in the reference frame is maximum of 10 bacterium or blocks, which is less than existing methods. The flowchart of algorithm used for motion estimation using PBFO is given in Fig.2. In PBFO, the optimization follows chemotaxis, swimming, tumbling, and elimination step to reach global minima until computational limitations are exceeded. The authors proposes an adaptive step size equation, which is used to predict best matching block in the reference frame with respect to macro block in the current frame for which motion vector is found. Due to adaptive step equation of PBFO, the next block search will start from nearer to best matching block in previous step. In PBFO, the adaptive step size equation is,

$$Step\ size = J_{best} [Mx + My]$$

$$\theta(i, j, k) = \theta(i, j, k) + C(i) \frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}} + Stepsize \quad (1)$$

where,  $\theta$  = Position vector of the bacterium,  
 $Mx$  = Horizontal position of motion vector of previous block and  $My$  = Vertical position of motion vector of previous block.  
 The mean absolute difference is taken as objective or cost function, which is expressed as,

$$MeanAbsoluteDifference = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N |CurrentBlock(i, j) - ReferenceBlock(i, j)| \quad (2)$$

where,  $M$  = Number of Rows in the frame and  $N$  = Number of Columns in the frame

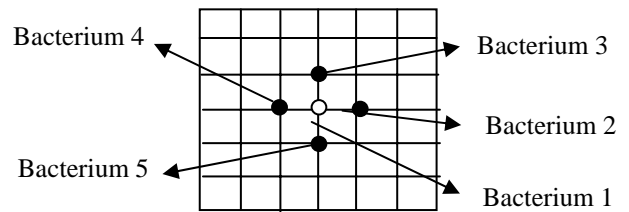


Fig. 1.Von Neumann Topology

The performance of the proposed method is evaluated by peak signal to noise ratio, which is given by following equation.

$$PSNR = 10 \log_{10} \frac{255^2}{\frac{1}{MN} \sum_{i,j=1}^{M,N} (OriginalFrame(i, j) - CompensatedFrame(i, j))^2} \quad (3)$$

IV. EXPERIMENTAL RESULTS

The proposed method (PBFO) has been tested on standard video i.e. caltrain and lecture based video sequences. Video sequences with a distance of two frames between current frame and reference frame were used to generate frame-by-frame results of the proposed algorithm. To test efficiency of the proposed algorithm with other existing algorithms, the algorithms are executed on single machine i.e. HP workstation CPU 3.0 GHz and 2GB RAM with MATLAB 7.0. The performance of PBFO is compared with other methods [5][6][7][8][9] and is presented in Table 1 and Table 2. Fig.3 and Fig.4 respectively show comparison of PSNR and computational time of PBFO with published methods for caltrain video sequence. The fig.5 and fig.6 respectively shows the comparison of PSNR and computational time of proposed method with other existing methods for lecture based video sequence. The speed of PBFO is faster than other methods and PSNR is close to published methods as shown in Table 3. The PBFO saves computational time from 96.60% to 25.00 % with PSNR gain ranging from of -0.09 to +2.01. In the suggested method zero motion is stored directly. The step size used for motion estimation and zero motion vectors implemented PBFO gives promising results over the published methods.

TABLE 1

COMPARISON OF MEAN PSNR (IN DB) OF THE PROPOSED METHOD WITH OTHER METHODS

Sr. No.	Type of Video Sequence	No. of Frames	ES	TSS	SESTSS	NTSS	4SS	DS	ARPS	Proposed Method
1	Caltrain	30	28.63	26.94	26.53	27.37	28.34	28.29	28.29	28.54
2	Lecturer Based	24	33.79	33.67	33.75	33.75	33.64	33.35	33.64	33.76

TABLE 2

COMPARISON OF MEAN COMPUTATIONAL TIME OF THE PROPOSED METHOD WITH OTHER METHODS

Sr. No.	Type of Video Sequence	No. of Frames	ES	TSS	SESTSS	NTSS	4SS	DS	ARPS	Proposed Method
1	Caltrain	30	5.56	0.67	0.51	0.74	0.64	0.66	0.40	0.30
2	Lecturer Based	24	32.96	4.13	3.29	3.42	3.22	3.06	1.94	1.12

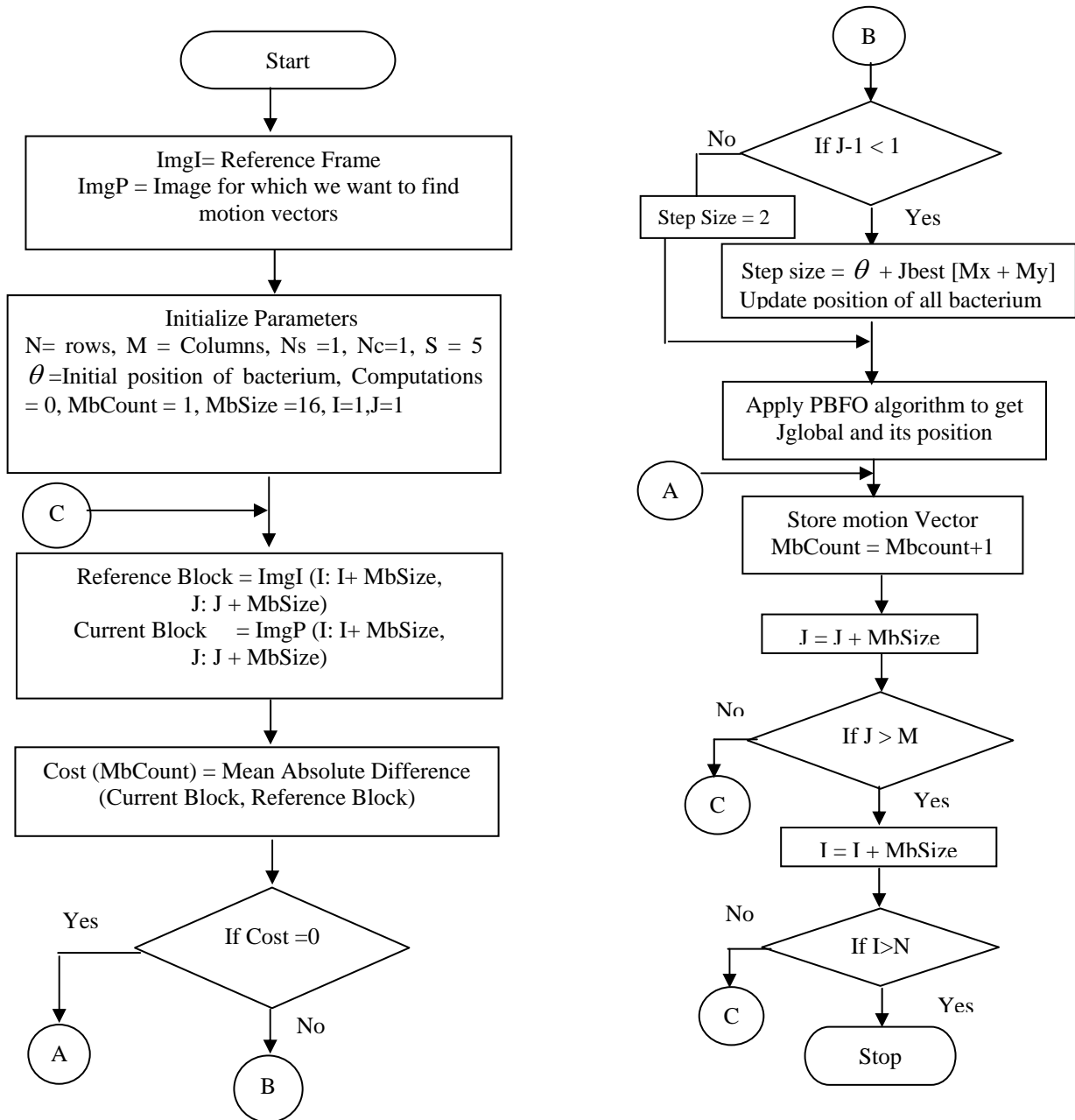


Fig.2. Flowchart of PBFO used for Motion Estimation

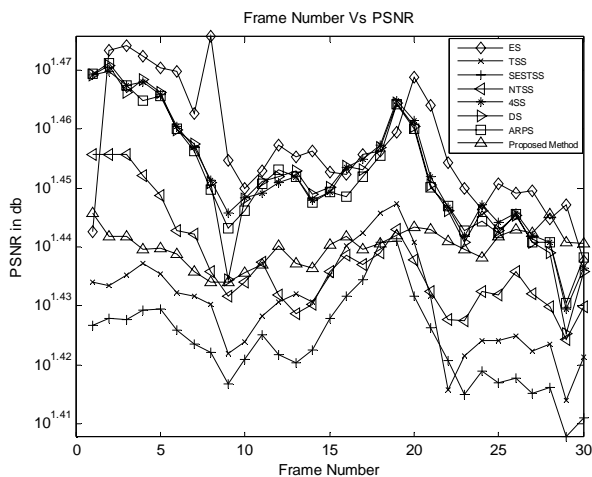


Fig.3. Comparison of PSNR for Caltrain Video

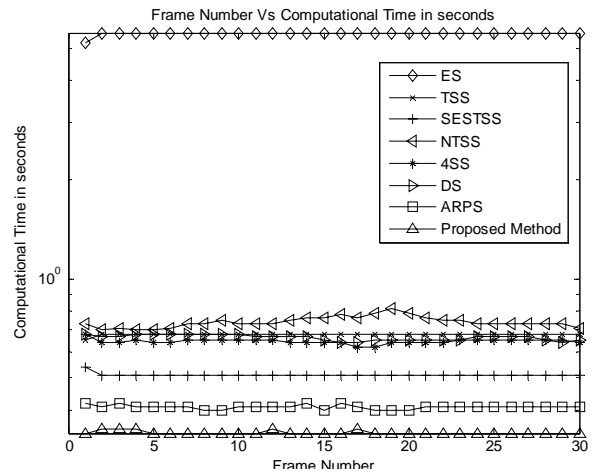


Fig 4. Comparison of Computational Time for Caltrain Video

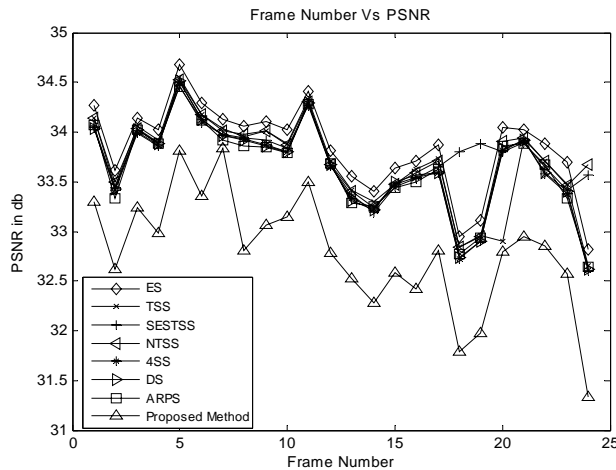


Fig 4. Comparison of PSNR for Lecture Based Video.

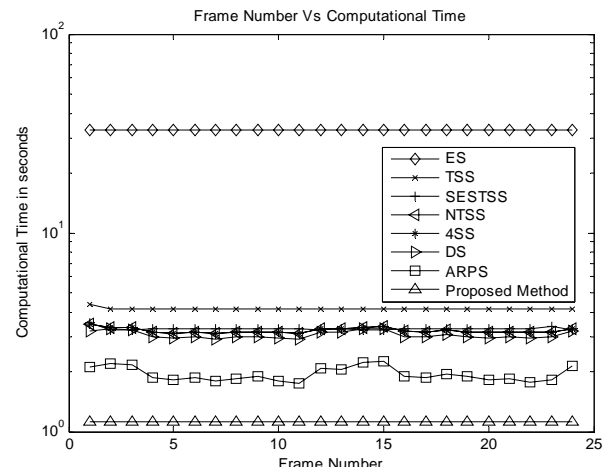


Fig 5. Comparison of Computational Time for Lecture Based Video.

TABLE 3

COMPUTATIONAL TIME SAVING AND PSNR GAIN BY PBFO OVER EXISTING METHODS

Sr. No	Proposed Method	Type of Video Sequence	Existing Block Matching Method						
			ES	TSS	SESTSS	NTSS	4SS	DS	ARPS
1	Computational Time save by PBFO (In percentage)	Caltrain	94.60	55.22	41.17	59.45	53.12	54.54	25.00
		Lecture Based	96.60	72.88	65.95	67.25	65.21	63.39	42.26
2	PSNR gain by PBFO (in db)	Caltrain	-0.09	+1.6	+2.01	+1.17	+0.2	+0.25	+0.25
		Lecture Based	-0.03	+0.09	+0.01	+0.01	+0.12	+0.41	+0.12

V. CONCLUSION

The paper presents a new optimization technique i.e. PBFO with its application to motion estimation. The presented technique is computationally faster than other methods. The results show promising improvement in terms of accuracy (PSNR), while drastically reducing the computational time. The PBFO saves maximum computational time 96.60% with PSNR degradation of -0.09. In the suggested method zero motion is stored directly. The PBFO is new soft computing tool with faster convergence by maintaining the good quality of solution. The BFO is not used for real time processing which helps us to develop new computationally efficient technique, which is suitable for any application.

REFERENCES

[1] R.C. Eberhart, Y. Shi, "Comparison between genetic algorithm and particle swarm optimization", in *proc. IEEE Int. Conf. Comput. Anchorage, AK*, pp. 611-616, May 1998.

[2] Vidya Sagar Chintakindi, Shyam Sundar Pattnaik, O. P. Bajpai, Malya Datta, S. Devi, G. V. R. S. Sastry, P. K. Patra, "Particle Swarm Optimization Technique a Soft Computing Tool for Denoising Images", *Proceedings of 10th International Conference on Information Science*, Salt Lake City, Utah, USA, pp. 995-1001, 2007.

[3] Liu, K.M. Passino, M.A. Simaan, "Biomimicry of Social Foraging Bacteria for distributed Optimization: Models, Principles, and Emergent Behaviors", *Journal of Optimization Theory and Application*, vol. 115, no.3, pp 603-628, December 2002.

[4] Sastry V.R.S.Gollapudi, Dr. Shyam S.Pattnaik, Dr.O.P.Bajpai, Swapna Devi, Ch.Vidya Sagar, Patra K.Pradyumna, K.M.Bakwad, "Bacterial Foraging Optimization technique to calculate resonant frequency of rectangular microstrip antenna", *International Journal of RF and Microwave Computer-Aided Engineering*, Volume 18, Issue 4, pp. 383-388, July 2008.

[5] Jianhua Lu., Ming L. Liou, "A simple and Efficient Search Algorithm for Block Matching Motion Estimation", *IEEE Trans. Circuits and Systems for Video Technology*, vol.7, no.2, pp. 429-433, April 1997.

[6] Renxiang Li, Bing Zeng, Ming L. Liou, "A New Three- Step Search Algorithm for Block Motion Estimation", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 4, no. 4, pp. 438-442, August 1994.

[7] Lai-Man Po., Wing -Chung Ma, "A Novel Four- Step Search Algorithm for Fast Block Motion Estimation", *IEEE Trans. Circuits and Systems for Video Technology*, vol.6, no. 3, pp. 313-317, June 1996.

[8] Shan Zhu, Kai-Khuang Ma, "A New Diamond Search Algorithm for Fast Matching Motion Estimation", *IEEE Trans. Image Processing*, vol.9, no.2, pp.287-290, February 2000.

[9] Yao Nie, Kai-Khuang Ma, "Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation", *IEEE Trans. Image Processing*, vol.11, no.12, pp. 1442-1448, December 2002.

[10] Chun-Man Mak, Chi keung Fong, and Wai Khen Chan, "Fast Motion Estimation For H.264/AVC in Walsh Hadamard Domain", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no.8, June 2008.

[11] Shen Li., Weipu Xu, Nanning Zheng, Hui Wang, "A Novel Fast Motion Estimation Method Based on Genetic Algorithm", *ACTA ELECTRONICA SINICA*, vol.28, no.6, pp.114-117, June 2000.

are requesting that you follow these guidelines as closely as possible.