Speed up of Reindexing in Adaptive Particle Swarm Optimization

Niraimathi Ponnusamy and Bhoopathy Krishnaswamy
Department of Electronics Engineering, Anna University, India

Abstract: *Palette re-ordering is a class of pre-processing method with the objective to manipulate the palette index such that the adjacent symbols are assigned close indices in the symbol space, thus enhancing the compressibility of the image with many lossless compressors. Finding an exact reordered palette would certainly be exhaustive and computationally complex. A solution to this NP hard problem is presented by using an Adaptive Particle Swarm Optimization (APSO) to achieve fast global convergence by maximizing the co-occurrences. A new algorithm with improved inertia factor is presented here to accelerate the convergence speed of the reindexing scheme. In this algorithm, the key parameter inertia weight is formulated as a factor of gradient based rate of particle convergence. Experimental results assert that the proposed modification helps in improving APSO performance in terms of solution quality and convergence to global optima.*

Key words: *Reindexing, palette-indexed image, cross entropy, rate of particle convergence (k), improved inertia weight adaptive particle swarm optimization.*

Received April 3, 2013; accepted November10, 2014; published online March 19, 2015

1. Introduction

A colour-mapped (pseudo-colour) image is composed of colour information contained in a look-up table and pixel values that are indices, which point to colour values in the look-up table. In most computer applications, images are used to help stimulate human visual perception. A true color image is a matrix of pixels, each consisting of red, green and blue color triplets. Each component (R_i, G_i, B_i) triplet has a range between 0 and 255 and is represented with a byte. Since, a color-mapped image utilises approximately one third of memory space of its corresponding true-colour representation, colour-mapped images are used as user interface elements of most windowing operating systems.

A color-quantized image is generally represented with a color index map each element of which serves as an index to select a color from a predefined set of colors to represent the color of a pixel in the image. The predefined set of colors is called a palette. To reduce the size of a color-indexed image further, lossless compression techniques are generally used because the index used to pick a particular palette color must be exact in decoding. A minor difference between two index values may result in a serious color shift.

In a standard image coding scenario, pixel-to-pixel correlation nearly always exists in the data, especially if the image is a natural scene. This correlation is what allows predictive coding schemes (e.g., DPCM) to perform efficient compression. In a color mapped image, the values stored in the pixel array are no longer directly related to the pixel intensity. For each pixel in the image, only the index of the corresponding color needs to be stored. Two color indices which are numerically adjacent (close) may point to two very different colors. The correlation still exists, but only via the color map. The efficiency of a lossless compression algorithm for indexed images may greatly depend on the assignment of indexes in the relative lookup table [2].

Palette reordering is a well-known and very effective approach for improving the compression of color-indexed images. Highly compressed palettized images are needed in many applications such as game cartridges, computer graphics and World Wide Web (WWW) on-line services.

The bottleneck of this solution is the intrinsic inefficiency to numerically optimize the palette re-indexing. If the optimal palette configuration is sought, the computational complexity involved would be high. As a matter of fact, a table of M colors corresponds to M! Configurations [26]. Clearly, this exhaustive search is impractical and thus transforms to an NP hard problem.

In typical applications where the search space is large and multidimensional, prior information about the function is not available and traditional mathematical techniques are not applicable. Global optimization is a NP complete problem and heuristic approaches like Genetic Algorithms (GAs) and Simulated Annealing (SA) have been used historically to find near optimal solutions. The Particle Swarm Optimization (PSO) algorithm is a new sociologically inspired stochastic optimization algorithm introduced by Kennedy and Eberhart [14].

1.1. Related Works

The existing solutions to the re-indexing problem can be classified into two groups, according to the strategy adopted. The first group of solutions performs the re-indexing of colour indexes according to perceptive similarity between different colours.

The second group of algorithms relies only on the statistical information conveyed by the index image to perform the reordering operation, guided by both information theory and local adaptive considerations. Menon and Venkateswaran [17] formulated the problem of palette reordering within the framework of linear predictive coding well modeled by a laplacian distribution formulated as the optimization version of the linear ordering problem. Pinho and Neves [23] had reported the survey on palette reordering methods providing details about main strategy underlying most effective algorithms.

The palette re-indexing method proposed by Pinho and Neves [22] is based on one-step look ahead greedy approach. The algorithm starts by finding the index that is most frequently located adjacent to other (different) indexes and the index that is most frequently found adjacent to it. This pair of indexes is the starting base for an ordered set, S, that will be constructed, one index at a time, during the operation of the re-indexing algorithm. New indexes can only be attached to the left or to the right extremity of the ordered set.

Battiato *et al.* [3] proposed a greedy strategy based on sequentially selecting the best edge still not processed, i.e., the one with the largest weight. The methodology tends to smooth the relative transitions in the indexed image, solving in an approximate way a related optimization problem over a weighted graph.

A new colour image compression system dedicated to HSI colour space was proposed. The proposed algorithm was based on encoding each channel using a suitable encoding algorithm which tends to be very lossy compression technique with subjective quality preservation even with highly compressed gray channel [19].

Recent works in this field have concentrated on the application of soft computing algorithms to the reindexing problem. Pei *et al.* [20] proved that it is possible to achieve high compression with acceptable image quality using the topology-preserving property of self-organizing kohonen feature map which considers "1D string neural structure" wherein, the neuron closest to each fed training vector, called the "winning neuron", will update itself while the neighboring neurons will update according to the neighboring function and gain function. The training vectors are extracted from the image using the butterfly-jumping sequence which leads to fast-converging training.

Battiato *et al.* [4] suggested a motor map neural network based re-indexing that uses an unsupervised, application independent, highly adaptive learning algorithm called "winner-take-all" learning driven by the reward function. Perhaps the most recent work was the application of PSO by using a static inertia factor proposed by Hook *et al.* [10] that showed very slow convergence with the risk of getting trapped to the sub-tours.

1.2. Contribution

In this paper, we give a framework of our proposed algorithm ordered as follows:

1. Adaptive swarm based solution to the problem of reindexing is proposed aiming to relatively reduce the computational complexity in previous algorithms to improve the performance of compression on palette images [10]. Palette reordering is formulated as a graph optimization problem, by calculating the degree of occurrence of a pair of symbols in the index image using Cross Entropy (CE) method. This along with the palette index forms the input to the Adaptive Particle Swarm Optimization (APSO) which returns the ordering of palette indices that would maximize the edge weights of the hamiltonian path [18].
2. The crucial parameter inertia weight is formulated as a function of gradient based rate of particle convergence. The proposed new inertia weight improves diversity of the swarm.
3. The stagnation phenomenon thus is avoided resulting in speeding convergence to global optima.

Experimental results assert that the proposed modification helps in improving APSO performance in terms of solution quality and convergence to global optima.

1.3. Organization

The paper is structured as follows: Section 2 describes the reindexing problem. Section 3 introduces the outline of PSO with the basic variants in PSO. Section 4 describes the new algorithm with improved inertia weight factor to yield better convergence speed for reindexing scheme. Experimental results and discussion are presented in section 5. In section 6, conclusions are drawn.

2. Problem Formulation

The re-indexing problem can be formulated as follows: Let I be an image of $m \times n$ pixels with distinct colors in which $I(x, y)$ denotes the color at pixel location (x, y) of I. Consider V to be the color palette consisting of N colors with typical values of 16, 64 or 256, represented by $V=\{v_1, v_2, \ldots, v_3\}$ If I' is an $m \times n$ matrix of indexes and $I'(x, y)$ point to colors in the color palette V, then

the relationship between the image I and indexed image $V(I')$ is expressed by the following equation:

$$I(x, y) = V(I'(x, y)) \quad (1)$$

Thus, palette reordering is used to find a new palette $P=\{p_1, p_2,\ldots, p_3\}$ obtained from V through permutation such that the corresponding $m \times n$ matrix may be compressed efficiently [24]. The principle behind palette reordering is that, the frequently occurring colors should have close indexes. Therefore, based on this principle, the assignment of the indexes is usually guided by the function $c(i, j)$, measuring the number of occurrences corresponding to pixels with index that are spatially adjacent to pixels with index j, according to some predefined neighborhood.

3. Particle Swarm Optimization

Biologically inspired algorithms have been gaining popularity in recent decades and beyond. These methods are based on biological metaphor, such as darwinian evolution and swarm intelligence. One of the recent algorithms to this category is PSO, motivated by the observations in birds, fishes or other organisms that move in swarms. The PSO algorithm is easy to implement, has few parameters. Because of its simplicity, ease of implementation and high convergence rates, the PSO algorithm has been applied to a variety of problems like evolving the structure as well as weights for artificial neural nets, power system optimization, process control, dynamic optimization, adaptive control and electromagnetic optimization.

3.1. Mathematical Formulation

The dynamic behaviour of the particle swarm can be quantified as follows:

$$v(t+1) = v(t) + \varphi_1(x - x_p) + \varphi_2(x - x_n) \quad (2)$$

$$x(t+1) = x(t) + v(t+1) \quad (3)$$

Where, v particle velocity; x particle position representing test solution. x_p represents position of the particle with overall highest fitness thus far and the value x_p represents position of the particle with the highest fitness in the neighbourhood. φ_1 and φ_2 uniform random variables to prevent the convergence to a local solution. Historically, these random variables are a combination of acceleration constants c_1 and c_2, called the cognition and social constants respectively [5]. Low values allow the particles to roam far from the target regions before being pulled back, while high values result in abrupt movement towards or past target regions.

3.2. Basic Variants of PSO

Many variations have been developed to improve speed of convergence and quality of solution found by the PSO. The lacks of PSO have been reduced with a variation of parameters in PSO. The variation is influenced by a number of control parameters, namely the dimension of the problem, the number of particles (swarm size), acceleration coefficients, inertia weight, neighborhood size, number of iteration and the random values which scale the contribution of the cognitive and social component. Below are the basic variations of PSO:

1. *Velocity Clamping*: Velocity clamping will control the global exploration of the particle. If the velocity of a particle exceeds the maximum allowed speed limit, it will set a maximum value of velocity. High value of will cause global exploration, whereas lower values result in local exploration. Velocity clamping did not influence the position of the particle. This only reduces the size of the step velocity. Changes in the search direction not only can make a particle to perform a better exploration but also has negative effects and the optimum value cannot be found.
2. *Inertia Weight*: The inertia weight controls the momentum of the particle by weighing the contribution of the previous velocity-basically controlling how much memory of the previous flight direction will influence the new velocity.
3. *Constriction Coefficient*: The constriction approach was developed as a natural, dynamic way to ensure convergence to a stable point. Condition and of the swarm is guaranteed to convergence.

The performance of PSO depends on its parameters to a great extent. Among all other parameters of PSO, Inertia weight is crucial one that affects the performance of PSO significantly and therefore, needs a special attention to be chosen appropriately.

3.3. Different Inertia Weight Strategies for PSO

Inertia weight plays a key role in the process of providing balance between exploration and exploitation process. The inertia weight determines the contribution rate of a particle's previous velocity to its velocity at the current time step.

Initial studies depict the use of a constant inertia weight throughout the searching process, while the later studies emphasizes on choosing a dynamically varying inertia weight [28]. A linear inertia weight was considered in [7] and the nonlinear strategy was reported in [30, 21] which was shown to be more suitable for smoother spaces. The linearly decreasing strategy [31] enhances the efficiency and performance of PSO. In spite of its ability to converge optimum, it gets into the local optimum solving the question of more apices function. Al-Hassan *et al.* [1] introduced an optimized Particle Swarm technique (PSOSA) that uses simulated annealing for optimizing inertia weight and tested the approach on urban planning problem.

A fuzzy adaptive inertia weight was proposed by [15, 29] where the inertia weight is dynamically adjusted based on fuzzy sets and rules. An exponential based inertia is considered in some recent studies [9, 16] that enhance PSO performance significantly. An adaption of dynamically varying inertia is conceived and analyzed in [33].

Gao *et al.* [8, 32] proposed a new PSO algorithm which combined logarithm decreasing inertia weight with chaos mutation operator. The logarithm decreasing inertia weight can improve the convergence speed, while the chaos mutation can enhance the ability to jump out of the local optima. A research on dynamically changing inertia was carried out in [12] that compare different nonlinear dynamic strategies.

In this paper, we present inertia weight as a factor of gradient based rate of particle convergence to improve the diversity of the swarm and speed up convergence to global optima as an efficient extension of our previous work [18].

4. Improved Inertia Weight Factor for Effective Reindexing Scheme

The index image and palette table are first extracted from the palette image. The CE based solution [6] to TSP is used to re-index colors for which the edge weight between the colors is calculated as follows:

$$e_{ij}=e_{ji}=\begin{cases} if \quad i\neq j & S(c_i,c_j)+S(c_j,c_i) \\ else & 0 \end{cases} \tag{4}$$

Where, $S(c_i, c_j)$ denotes the number of times the pair of colors (c_i, c_j) appear together in the raster scan of the image for $i, j=1, ..., N$ the index image is now transformed as a complete non-directed weighted graph $G=(V, E, e)$ where, e is calculated as in Equation 4. The co-occurrence matrix thus, formed acts as the distance measure for the palette reordering with colors as vertices.

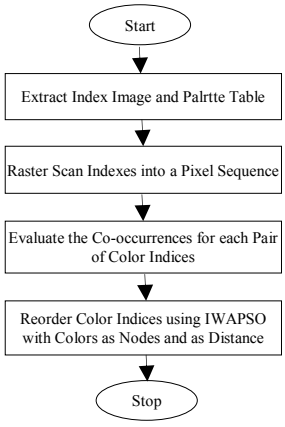


Figure 1. Flowchart for palette reindexing.

An overview of the proposed algorithm as shown in Figure 1, Inertia Weight Adaptive Particle Swarm Optimization Algorithm (IWAPSO) starts with a population of random solutions "particles" in a D-dimension space formed by the entries of the palette. The position of the i^{th} particle is represented by $X_i=\{x_1, x_2, \ldots, x_3\}$, while its velocity is given by $V_i=\{v_1, v_2,\ldots, v_{iD}\}$. The particle dynamics as explained in section 3 is decided by the velocity and position update equations. The position update equation is calculated from Equation 3.

The velocity update Equation in 2 is modified by introducing adaptiveness into the random variables as follows:

$$v(t+1)=w\,v(t)+c_1 rand_1()(x-x_p)+c_2 rand_2()(x-x_n) \tag{5}$$

Where, $rand_1()$ and $rand_2()$ are uniformly distributed random functions between [0, 1] which determine the tension in the system and c_1 and c_1 are called the cognition and social constants respectively [13] which determine exploration or exploitation of search space.

The term w, is the inertia weight which is employed to control the exploration abilities of the swarm as it scales the current velocity value affecting the updated velocity vector (5).

1. If $w>1$, then the velocity will decrease with time, the particle will accelerate to maximum velocity and the swarm will be divergent.
2. If $w<1$, then the velocity of particle will decrease until it reaches zero. Larger values will facilitate an exploration rather small values will promote the exploitation.

In order to, strike a balance between exploration and exploitation, inertia factor is made adaptive by the method of linearly decreasing inertia weight [7]. The linear expression is given as:

$$w=w_{initial}-\frac{iter}{iter_{\max}}\times(w_{initial}-w_{final}) \tag{6}$$

Where, w_{intial} expresses the initial weight, w_{final} expresses the final weight. The variables *iter* and $iter_{mix}$ represents the current number of iterations and maximum number of iterations respectively. In the above equation the change of inertia factor is associated only with the number of iterations and could not better adapt to changes in characteristics of those with complex nonlinear optimization problem.

To overcome the shortcomings in evaluating the inertia factor, this paper introduces a new algorithm i.e., improved IWAPSO. The introduction of the rate of particle convergence factor to the swarm optimization algorithm to modifies its inertia factor appreciably. It not only considers the role of the number of iterations, but also the impact of the current global optimal position and current time optimal position of all particles.

Global optimum of particle swarm is always better than or equal to the current optimum value of individual. When all the particles have reached the

global optimum value, the swarm converges to a point when k=0.

1. With value of k is higher, indicating the more dispersed distribution of particles in a particle swarm, the particles easily falls into local optimum.
2. With the decrease of k, the algorithm easily falls in local optimum.

Therefore, there is a need to modify the inertia weight, thus increasing the search space and improve the ability of global optimization of the particle swarm. In summary, inertia weight reduces along with decrease in k.

Stochastic search algorithms like the PSO algorithm perform a biased random walk to explore the search space. A random search allows stochastic optimization algorithms to escape from local minima and explore flat regions but is computationally expensive and leads to slow convergence rates. On the other hand, deterministic algorithms like gradient-based techniques converge faster by using derivative information to identify a good search direction but get stuck in local minima. Also, deterministic techniques perform poorly in minimizing functions for which the global minimum is surrounded by flat regions where the gradient is small.

By considering combination of various parameters, the rate of particle convergence factor is calculated from one of the three different Equations 7, 8 and 9.

$$k=\frac{\sum_{i=1}^{P_{size}} P_{ibest}(t)-P_{size}\times P_{gbest}(t)}{\sum_{i=1}^{P_{size}} P_{ibest}(t)} \tag{7}$$

$$k=\frac{\sum_{i=1}^{P_{size}} gradient\,(P_{ibest}(t)-P_{size}\times P_{gbest}(t))}{\sum_{i=1}^{P_{size}} P_{ibest}(t)} \tag{8}$$

$$k=\frac{\sum_{i=1}^{P_{size}} gradient\,(P_{ibest}(t))-P_{size}\times gradient\,(P_{gbest}(t))}{\sum_{i=1}^{P_{size}} P_{ibest}(t)} \tag{9}$$

Where, P_{gbest} expresses the current global optimum, P_{size} expresses the size of that particle swarm and P_{ibest} refers to the current optimum value of individual.

Equation 7 is formulated without using gradient function for the parameters used. In Equation 8 entire gradient is taken for the numerator of the parameter-rate of particle convergence. For Equation 9 gradient is taken separately to current global optimal position and current time optimal position. Experimentally evaluating the three different cases, Equation 9 appreciably contributed to weight factor in terms of distribution of particles in the swarm.

The introduction of gradient based rate of particle convergence makes significant improvement to Equation 6 and modifies the inertia factor.

$$w=w_{initial}-\frac{iter}{iter_{\max}}\times(w_{initial}-w_{final})\times k \tag{10}$$

The new inertia factor in Equation 10 is replaced for w in Equation 5. This equation influences the performance of the algorithm to achieve a better balance in global searching ability of particle swarm and to accelerate the convergence speed. The adaptiveness in the algorithm is introduced by updating the new inertia factor, acceleration coefficients and velocity of the particle.

The objective function for IWAPSO decides the criteria to select the optimal ordering of the color indices which is provided below:

$$f(p)=\sum_{i=1}^{N}\sum_{j=1}^{N} e(p(i),p(j))\,|i-j| \tag{11}$$

Where, $e(p(i), p(j))$ denotes the number of times that pixels with color $p(i)$ have neighboring pixels whose color is $p(i)$ calculated from Equation 4. Together with the modification in the inertia term w, stochastic factors and acceleration coefficients c_1 and c_2 are also included in Equation 5.

The introduction of stochastic factors may cause the system to enter a state of explosion because of increased global exploration [27] resulting in the particle velocities and positional coordinates tending to infinity. In order to, prevent such a scenario, a maximum value of velocity max is defined as follows:

$$v(t)\begin{cases} if\ v(t)>V_{\max} & v(t)=v_{\max} \\ if\ v(t)<-V_{\max} & v(t)=-v_{\max} \end{cases} \tag{12}$$

Where, v_{max} is the maximum velocity allowed for each particle. Acceleration coefficients c_1 and c_2 are adjusted with time to increase the social component and reduce the cognitive component over iterations. Particles are allowed to wander through the search space initially with a larger cognitive component and smaller social component; rather than moving toward the population best [25]. The convergence towards the global optima in the latter part of the optimization is achieved with a smaller cognitive component and larger social component. The acceleration coefficients change during runtime according to the following equation:

$$c_1=c_{min}+(c_{max}-c_{min})\times\exp[-(4\times iter/MAX)^2] \tag{13}$$

$$c_2=c_{min}-(c_{max}-c_{min})\times\exp[-(4\times iter/MAX)^2] \tag{14}$$

Where, c_{max} and c_{min} are the maximum and the minimum acceleration values selected arbitrarily, MAX, the maximum number of optimization steps and $iter$ represents the current iteration number. A check on exploration is placed by the adaptive velocity parameter v to limit the search space and not to wander into the illegal solutions. Convergence speed improves through the automatic control of improved inertia weight and acceleration coefficients c_1 and c_2 at run

time. Thus, the proposed method improves the diversity of the swarm in order to, avoid the stagnation phenomenon and a speeding convergence to global optima. The flow chart of the proposed algorithm is shown in Figure 2.

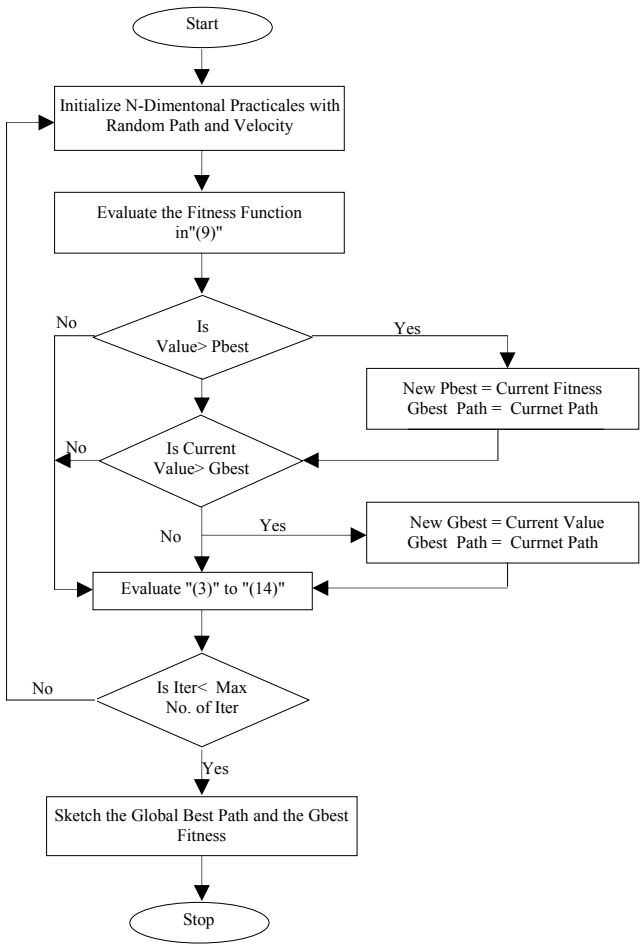
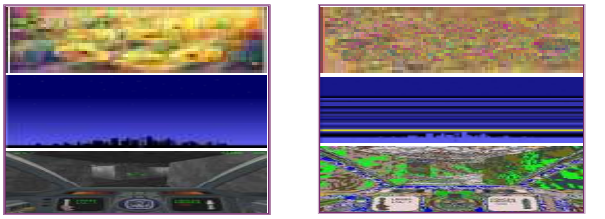


Figure 2. Flow chart for IWAPSO.

5. Experimental Results and Discussion

The images were initially obtained for the work from the FTP site ftp://ftp.ieeta.pt/~ap/images. The images were converted from PPM to BMP using gimp program. The IWAPSO is implemented in MATLAB. Maximized fitness function is found for the complete non-directed weighted graph by taking the following parameter values: c_{max}=3, c_{min}=1.2, w_{intial}=0.96 and w_{final}=0.6. The population size of 550 has been used for all the images. For images with less than 32 colors, the maximum iteration was fixed to 300 and the maximum velocity was set to 1.5 while for images with colors between 32 and 64, the maximum iterations and velocity were set to 700 and 2 respectively. For images with colors between 64 and 128, the maximum iterations was 1200 and maximum velocity was 3.4 and finally for images with colors more than 128, the maximum iterations and maximum velocity was set to 1800 and 4.2 respectively. The values were chosen through experimentation.

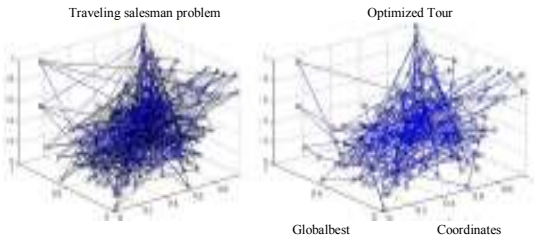
Figure 3-a, represents the different indexed images with distinct colors 256, 128 and 64 which are shown as an example from the list of test images. The reindexed images obtained through IWAPSO have been presented as an illustration in Figure 3-b.



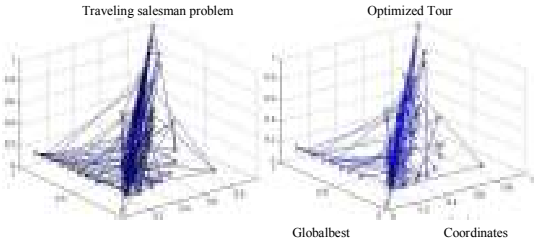
a) Original indexed images. b) Reindexed images through IWAPSO.

Figure 3. Indexed images with different color palettes.

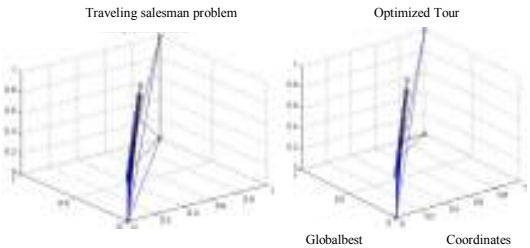
In Figure 4-a, b and c, the path obtained for the optimised tour using APSO and the tour obtained using PSO for travelling salesman problem are compared for only images with significantly distinct 256, 128 and 64 colors. Eventually it is evident that sub tours have been considerably reduced using the APSO method. Alternatively, in Figure 5-a, b and c the improved optimised tour using IWAPSO further reduces the sub tours and therefore, helps in reducing the run time of the algorithm. It is to be noted that this path is obtained by maximizing the co-occurrences between the pixel pairs as mentioned in the cost function. This according to information theory will eventually reduce the entropy of the image due to the reduced number of transitions between the colour indexes. Thus, a comparison is done for the Hamiltonian path obtained between the optimised tour using our previous method and improved optimised tour for the formulated IWAPSO.



a) Clegg image with a palette of 256 colours.

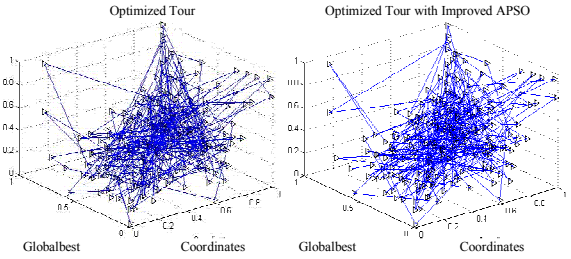


b) Descent image with a palette of 128 colours.

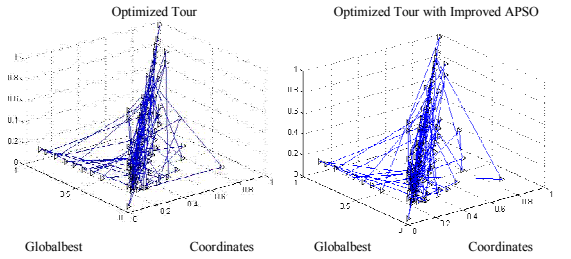


c) Seadusk image with a palette of 64 colours.

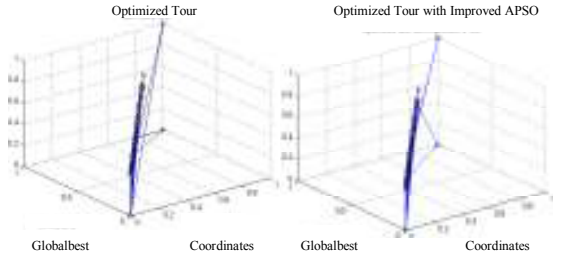
Figure 4. Comparison of optimised tour with APSO and Improved optimized tour with IWAPSO.



a) Clegg image with a palette of 256 colours.



b) Descent image with a palette of 128 colours.



c) Seadusk image with a palette of 64 colours.

Figure 5. Comparison of PSO with TSP and optimized tour with APSO.

Table 1. Comparison of re-indexing time.

Configuration			A	B	C
Sl. No.	**Images**	**No. of Colors**	**D**	**E**	**F**
1	Clegg	256	40	16.56	3.53
2	Descent	128	7	4.727	1.6388
3	Seadusk	64	2	0.77	0.35

A-intel dual core 2.8 GHz processor and 1GB of RAM; B and C-intel dual core 2.2 GHz processor and 1GB of RAM;

D-Re-indexing Time(Mins) From (Joshua van *et al.* 2009); E-Re-indexing Time(Mins) (Niraimathi *et al.* [18]); F-Re-indexing Time(Mins) proposed algorithm

Table 1 depicts the comparison of re-indexing time taken by the algorithm for images with 256, 128 and 64 colors. It is vivid that the re-indexing time drops down reasonably well than our previous work. This highlights the performance of the proposed method in comparison with earlier technique in [11] and also our previous work [18].

A graphical interpretation also is made between number of colors and re-indexing time, distinguishing the methods the optimized tour using APSO and the improved optimized tour using IWAPSO as shown in Figure 6. This proves that the proposed method is computationally less expensive than our previous work [18] and the previous algorithm [11].

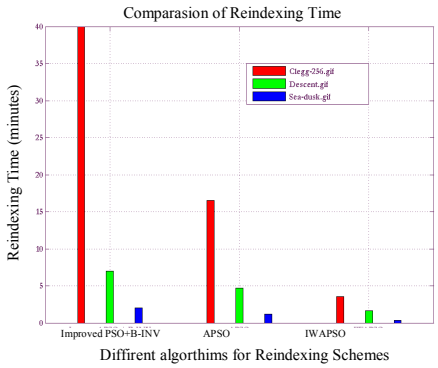


Figure 6. Graphical comparison of reindexing time.

The rate of convergence factor has been viewed graphically in Figure 7, obtained by plotting for various values of k and the number of colors from 8 to 256. Evaluation of k using Equations 7 and 8 yields constant value and there by does not notably contribute to the modification of inertia weight factor. The range of values obtained by considering k in Equation 9 with separate gradient for P_{gbest} and P_{ibest} parameters (-0.02 to 0.14) shows the introduction of diversity of the particles in the swarm.

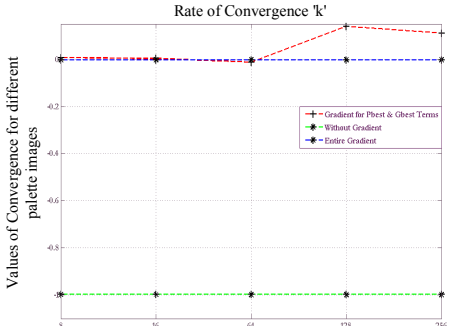


Figure 7. Graphical representation of rate of convergence parameter.

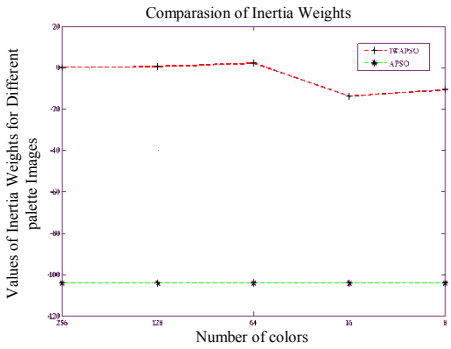


Figure 8. Graphical comparison of Inertia weight factor.

In Figure 8 the inertia factor contributed by IWAPSO is compared with APSO algorithm. The modified inertia factor shows the exploration abilities of the swarm to better adapt changes in characteristics of the new reindexing scheme.

Figure 9 compares the performance of the IWAPSO and APSO algorithms by evaluating the swarm fitness for different iterations. It can be seen that the APSO easily falls into local optimum and emerges premature phenomenon while the proposed algorithm has strong ability of global searching and higher precision of

optimum than the former. The graph shows steep gradients throughout the solution space for the IWAPSO algorithm. Therefore, the algorithm shows significant improvement in search ability and converges faster to a more accurate final solution than the APSO algorithm. This is because good solutions found by the APSO are refined using a deterministic gradient based inertia factor with high convergence rate avoiding costly random search.

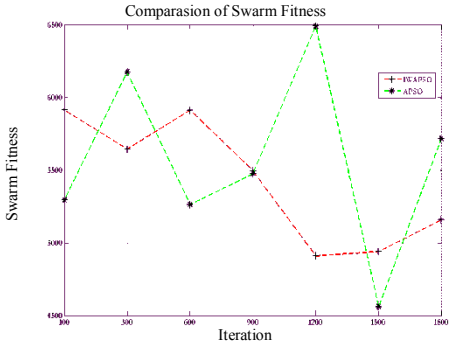


Figure 9. Graphical comparison of swarm fitness.

6. Conclusions

In this paper, an efficient algorithm is proposed which includes improved inertia weight formulated as a factor of the significant parameter rate of change of particle convergence. This parameter involves the role of global optimal position and current optimal position of all the particles to contribute better inertia weight factor to the algorithm. As the extension of our previous work, updating the inertia weight accelerates the convergence speed. Comparing our previous method, experimental results assert that the proposed modification considerably reduces the sub tours in the hamiltonian path obtained through IWAPSO algorithm. Finally, this paper also, demonstrates the competence of the proposed algorithm by graphically interpreting performance of its parameters namely, inertia weight, rate of change of particle convergence and swarm fitness.

References

[1] Al-Hassan W., Fayek M., and Shaheen S., "PSOSA: An Optimized Particle Swarm Technique for Solving the Urban Planning Problem," *in Proceedings of the International Conference on Computer Engineering and System*, Cairo, Egypt, pp. 401-405, 2006.

[2] Arnavut Z. and Sahin F., "Lossless Compression of Color Palette Images with One Dimensional Technique," *the Journal of Electronic Imaging*, vol. 15, no. 2, pp. 1-11, 2006.

[3] Battiato S., Gallo G., Impoco G., and Stanco F., "An Efficient Re-Indexing Algorithm for Color-Mapped Images" *the IEEE Transactions Image Processing*, vol. 13, no. 11, pp. 1419-1423, 2004.

[4] Battiato S., Rundo F., and Stanco F., "Self Organizing Motor Maps for Color-Mapped Image Re-Indexing," *the IEEE Transactions Image Processing*, vol. 16, no. 12, pp. 2905-2915, 2007.

[5] Clerc M., "Think Locally, Act Locally: The Way of Life of Cheap-PSO," *Technical report*, available at: http://clerc.maurice.free.fr/pso/, last visited 2001.

[6] De P., Kroese D., Mannor S., and Rubinstein R., "A Tutorial on the Cross-Entropy Method," *in Proceedings of Annuals of Operation Research*, Germany, pp. 19-67, 2005.

[7] Eberhart R. and Shi Y., "Particle Swarm Optimization: Developments, Applications and Resources," *in Proceedings of IEEE Congress on Evolution Computation*, Seoul, Korea, pp. 81-86, 2001.

[8] Gao Y., An X., and Liu J., "A Particle Swarm Optimization Algorithm with Logarithm Decreasing Inertia Weight and Chaos Mutation," *in Proceedings of International Conference on Computational Intelligence and Security*, Suzhou, China, pp. 61-65, 2008.

[9] Ghali N., El-Dessouki N., Mervat N., and Bakrawi L., "Exponential Particle Swarm Optimization Approach for Improving Data Clustring," *the International Journal of Electrical and Electronics Engineering*, vol. 3, no. 6, pp. 208-212, 2009.

[10] Hook J., Sahin F., and Arnavut Z., "Application of Particle Swarm Optimization for Traveling Salesman Problem to Lossless Compression of Color Palette Images," *in Proceedings of IEEE SMC System of System Engineering*, Singapore pp. 1-5, 2008.

[11] Hook J., Sahin F., and Arnavut Z., "Improved Lossless Compression of Color Palette Images by Re-indexing with Particle Swarm Optimization," *in Proceedings of Soft Computing, Computing with words and Perception in System Analysis, Decision and Control*, Famagusta, Cyprus, pp. 1-4, 2009.

[12] Hu J., Xu J., Wang J., and Xu T., "Research on Particle Swarm Optimization with Dynamic Inertia Weight," *in Proceedings of the International Conference on Management and Service Science*, Wuhan, China, pp. 1-4, 2009.

[13] Hu X., Eberhart R., and Shi Y., "Recent Advances in Particle Swarm," *in Proceedings of the Congress on Evolution Computation*, Oregon, USA, pp. 90-97, 2004.

[14] Kennedy J. and Eberhart R., "Particle Swarm Optimization," *in Proceedings of IEEE International Conference on Neural Networks*, Perth, Western Australia, pp. 1942-1948, 1995.

[15] Liu C., Ouyang C., Zhu P., and Tang W., "An Adaptive Fuzzy Weight PSO Algorithm," *in*

Proceedings of the 4th International Conference on Genetic and Evolution Computing, Shenzhen, China, pp. 8-10, 2010.

[16] Li H. and Gao Y., "Particle Swarm Optimization Algorithm with Exponent Decreasing Inertia Weight and Stochastic Mutation," *in Proceedings of the 2nd International Conference on Information and Computing Science*, Manchester, UK, vol. 1, pp. 66-69, 2009.

[17] Menon N. and Venkateswaran A., "On Ordering Color Maps for Lossless Predictive Coding," *the IEEE Transactions Image Processing,* vol. 5, no. 11, pp. 1522-1527, 1996.

[18] Niraimathi P., Sudhakar M., and Bhoopathy K., "Efficient Reordering Algorithm for Color Palette Image using Adaptive Particle Swarm Technique," *the Applied Soft Computing*, vol. 12, no. 8, pp. 2199-2207, 2012.

[19] Semary N., Hadhoud M., Abdul-Kader H., and Abbas A., "Novel Compression System for Hue-Saturation and Intensity Color Space," *the International Arab Journal of Information Technology*, vol. 10, no. 6, pp. 546-552, 2013.

[20] Pei S., Chuang Y., and Chuang W., " Effective Palette Indexing for Image Compression using Self-organization of Kohonen Feature Map," *the IEEE Transactions Image Processing*, vol. 15, no. 9, pp. 2493-2498, 2006.

[21] Peram T., Veeramachaneni K., and Mohan C., "Fitness-Distance-Ratio Based Particle Swarm Optimization," *in Proceedings of IEEE Swarm Intelligence Symposium*, Indiana, USA, pp. 174-181, 2003.

[22] Pinho A. and Neves A., "A Note on Zeng's Technique for Color Reindexing of Palette-Based Images," *the IEEE Signal Processing letters*, vol. 11, no. 2, pp. 232-234, 2004.

[23] Pinho A. and Neves A., "Survey on Palette Reordering Methods," *the IEEE Transactions Image Processing*, vol. 13, no. 11, pp. 1411-1418, 2004.

[24] Pinho A. and Neves A., "On the Relation between Memon's and the Modified Zeng's Palette Reordering Methods," *the Image Vision Computing*, vol. 24, no. 5, pp. 534-540, 2006.

[25] Sahin F. and Devasia A., "Distributed Particle Swarm Optimization for Structural Bayesian Network Learning," *Swarm Intelligence: Focus on Ant and Particle Swarm Optimization: I-Tech Education and Publishing*, Vienna, pp. 505-532, 2007.

[26] Sayood K., *Lossless Compression Handbook*, Academic New York, USA, 2002.

[27] Shi X., Liang Y., Lee H., Lu C., and Wang Q., "Particle Swarm Optimization-based Algorithms for TSP and Generalized TSP," *the Information Processing Letters*, vol. 103, no. 5, pp. 169-176, 2007.

[28] Shi Y. and Eberhart R., "A Modified Swarm Optimizer" *in Proceedings of IEEE International Conference of Evolution Computation*, Anchorage, Alaska, 1998.

[29] Shi Y. and Eberhart R., "Fuzzy Adaptive Particle Swarm Optimization" *in Proceedings of IEEE Congress on Evolutionary Computation*, Seoul, Korea, pp. 101-106, 2001.

[30] Suganthan P., "Particle Swarm Optimiser with Neighborhood Operator," *in Proceedings of IEEE Congress on Evolutionary Computation*, Washington, USA, pp. 1958-1962, 1999.

[31] Trelea I., "The Particle Swarm Optimization Algorithm: Convergence Analysis and Parameter Selection," *the Information Processing Letters*, vol. 85, no. 6, pp. 317-325, 2003.

[32] Yang C., Cheng Y., and Chuang L., "A Novel Chaotic Inertia Weight Particle Swarm Optimization for PCR Primer Design," *in Proceedings of the International Conference on Technology and Applications of Artificial Intelligence*, Hsinchu, Taiwan, pp. 373-378, 2010.

[33] Zhou Z. and Shi Y., "Inertia Weight Adaption in Particle Swarm Optimization Algorithm Advances in Swarm Intelligence," *in Proceedings of the 2nd International Conference*, *ICSI*, Chongqing, China, pp. 71-79, 2011.

Niraimathi Ponnusamy graduated from Bharath University, India and received her MTech in VLSI design. She is a faculty member in the Department of electronics and eommunication engineering, MIT Campus Anna University. Her research areas include image processing, soft computing, VLSI and signal processing algorithms.

Bhoopathy Krishnaswamy completed his doctoral degree from IIT Madras. He is presently working as Professor, ECE Department, in MIT campus, Anna University, India. His areas of interest include signal processing, image processing and network security.