International Journal of Computer Applications (0975 – 8887)
Volume 91 – No.10, April 2014

A Comparison of Evolutionary Algorithms: PSO, DE and GA for Fuzzy C-Partition

Assas Ouarda
Department of Computer Science, University of M’sila, Algeria.
Laboratory Analysis of Signals and Systems (LASS). University of M’sila, Algeria

M. Bouamar
Department of Electronics Science, University of M’sila, Algeria.
Laboratory Analysis of Signals and Systems (LASS). University of M’sila, Algeria

ABSTRACT
The fuzzy c-partition entropy technique for threshold selection is one of the best image thresholding techniques, but its complexity increases with the number of thresholds. In this paper, the selection of thresholds (fuzzy parameters) was seen as an optimization problem and solved using particle swarm optimization (PSO), differential evolution (DE), genetic (GA) algorithms. The proposed fast approaches have been tested on many images. For example, the processing time of four-level thresholding using PSO, DE and GA is reduced to less than 0.4s. PSO, DE and GA show equal performance when the number of thresholds is small. When the number of thresholds is greater, the PSO algorithm performs better than GA and DE in terms of precision and robustness. But the GA algorithm is the most efficient with respect to the execution time.

General Terms
Partitioning algorithms, Pattern recognition, Image segmentation.

Keywords
Entropy, Histograms, Optimization, Particle swarm optimization, Thresholding, Fuzzy c-partition, Differential Evolution Algorithm

1. INTRODUCTION
The image thresholding is the simplest method of image segmentation. Segmentation is a widely employed technique in many fields like: Optical Character Recognition, Signature Identification, Biomedical Imaging, and Target Identification.

However, the automatic selection of an optimum threshold has remained a challenge in image segmentation. Many approaches have been studied for thresholding [1][2][3][4][5][6][7][8][9][10][11]. Sezgin and Sankur [2] have developed classification of thresholding algorithms based on the type of information used, and they measure their performance comparatively using a set of objective segmentation quality metrics. They distinguish six categories, namely, thresholding algorithms based on the exploitation of: 1. histogram shape information, 2. Measurement space clustering, 3. histogram entropy information, 4. image attribute information, 5. spatial information, and 6. local characteristics.

The fuzzy set theory has been successfully applied in several areas such as control, image processing, pattern recognition, computer vision, medicine, social science, etc. With regard to automatic threshold selection and segmentation, the concept of fuzzy partition leads to a powerful and efficacious system [4] [9] [10].

Although, the thresholding results of the fuzzy c-partition entropy technique are much better than many existing approaches. The size of search space augments when the number of parameters of the membership function increases. Therefore, the computation time and storage space augment. For an image having L grey levels, and a membership function determined by c parameters, the size of search space is L!/((L-c)!.c!). For example, if L equals 256 and a membership function determined by two parameters, the search space will be 32 640. When the number of parameters is superior than or equal to 3, the exhausted search is too expensive or impracticable [4] [9]. To get optimal thresholds, it must find the optimal combination of the fuzzy parameters. Thus, the thresholding problem can be formulated as an optimization problem. The fuzzy entropy of the image has been chosen as the fitness function. Therefore, a strategy for effective research must be developed, where it can find the optimal combination of all the fuzzy parameters quickly.

In recent years there has been a growing interest in evolutionary algorithms for diverse fields of science and engineering. The differential evolution algorithm (DE), is relatively novel optimization technique to solve numerical optimization problems. The algorithm has successfully been applied to several sorts of problems as it has claimed a wider acceptance and popularity following its simplicity, robustness, and good convergence properties [11]. Particle Swarm Optimization (PSO) has the distinction of being one of the simplest heuristic algorithms in terms of complexity of equations. Genetic algorithms (GA) are optimization algorithms based on techniques derived from genetics and natural evolution: crossovers, mutations, selection, , etc. And it is global searching technique capable, most often, to prevent from trapping into locally optimal solutions.

In this work, we propose using PSO, DE and GA in finding the optimal combination of all fuzzy parameters efficiently, to render the multilevel thresholding technique more applicable and effective. The experimental study shows that the proposed approaches can obtain results with reduced computational time.

The organization of the paper is as follows. In section 2 the fuzzy c-partition entropy technique of thresholding is reviewed. The section 3, deals with a review of the optimization techniques used: PSO, DE and GA. In Section 4, a complete description of proposed thresholding algorithms is presented, where each step of the algorithm is developed in detail. Section 5 illustrates the obtained experimental results and discussions and section 6 concludes this paper.

2. FUZZY C-PARTITION ENTROPY TECHNIQUE

In the fuzzy c-partition entropy approach proposed in ref [4] and [9], an image is modelled by c fuzzy sets which have membership functions and there is no sharp boundary between these sets.

2.1 The Bi-level Thresholding

An image is modelled by two fuzzy sets dark and bright, whose membership functions are defined as follows:

$$\mu d = \begin{cases} 1 & x \le a, \\ \frac{x-c}{a-c} & a<x<c, \\ 0 & x \ge c. \end{cases} \quad (1)$$

$$\mu b = \begin{cases} 0 & x \le a, \\ \frac{x-a}{c-a} & a<x<c, \\ 1 & x \ge c. \end{cases} \quad (2)$$

Where x is the independent variable and a and c are parameters determining the shape of the above two membership functions.

The images have 256 gray levels ranging from 0 to 255. Then, an exhaustive search is used to find the pair aopt and copt which forms a fuzzy 2-partition that has the maximum entropy as follows:

For a = 0 to 254

For c = (a+1) to 255

1. For given a and c. new membership functions µd(i) and µb(i) are computed, for i =0, ..., 255.
2. Probabilities of the two fuzzy events dark and bright are defined as:

$$P(dark) = \sum_{i=0}^{255} \mu d(i) P(i) \quad (3)$$

$$P(bright) = \sum_{i=0}^{255} \mu b(i) P(i) \quad (4)$$

where P(i)is the probability of the occurrence of the gray level i =0 ,...,255.

3. The entropy of this fuzzy 2-partition is given by:

$$H = - P(dark) . \log(P(dark)) - P(bright) \cdot \log(P(bright)) \quad (5)$$

4. The selected threshold value Topt which is the mid-point of aopt and copt has to satisfy the following criterion function:

$$H(T_{opt}) = \max \left(\underset{t=0,\dots,255}{H(t)} \right) \quad (6)$$

Where $T_{opt} = \frac{a_{opt} + c_{opt}}{2}$ (7)

End For c

End For a

2.2 The Multi-level Thresholding

The 3-level thresholding is used to demonstrate how multiple thresholding can be conducted. Consider the following 3-fuzzy sets A1, A2, and A3, whose membership functions are defined as follows:

$$\mu_{A1} = \begin{cases} 1 & x \le a1, \\ \frac{x-c1}{a1-c1} & a1<x<c1, \\ 0 & x \ge c1. \end{cases} \quad (8)$$

$$\mu_{A2} = \begin{cases} 0 & x \le a1, \\ \frac{x-a1}{c1-a1} & a1 < x < c1, \\ 1 & c1 \le x \le a2, \\ \frac{x-c2}{a2-c2} & a2 < x < c2, \\ 0 & x \ge c2. \end{cases} \quad (9)$$

$$\mu_{A3} = \begin{cases} 0 & x \le a2, \\ \frac{x-c2}{a2-c2} & a2<x<c2, \\ 1 & x \ge c2. \end{cases} \quad (10)$$

Where x is the independent variable, a1, c1, a2 and c2 are parameters, and $0 \le a1 < c1 < a2 < c2 \le 255$.

Similar to bi-level thresholding, 3-level thresholding is also used to find a fuzzy partition in the fuzzy 3-partition space, such that the entropy is maximized. But at this time, we need to find two pairs of a’s and c’s. Because the search space is too large, the PSO algorithm is used to solve it.

A fuzzy c-partition can be determined by 2(c-1) parameters using the proposed approach. The problem becomes to find the best combinations of these parameters. It can be considered as a combinatorial optimization problem. The size of search space increases very rapidly when the number of parameters increases as given away in table 1 below.

Table 1. The size of search space for finding fuzzy parameters

Number of class	Number of fuzzy parameters	Size of search space
2	2	32640
3	4	2.7x106
4	6	1.7x108

3. REVIEW OF PSO, DE AND GA

3.1 Particle swarm optimisation algorithm

Particle Swarm Optimization (PSO) is population-based and evolutionary optimization technique developed by Kennedy and Eberhart in 1995 [12].

The particle swarm algorithm is introduced here in terms of social and cognitive behaviour, though it is widely used as a problem-solving method in engineering and computer science. PSO simulates a commonly observed social behaviour, where members of a group tend to follow the lead of the best of the group. The PSO algorithm consists of a swarm of particles, which are initialized with a population of random candidate solutions. They move to search the new solutions. Each particle has a position-vector xi, a velocity-vector vi and the fitness of solution f. Each particle remembers its own best position in a vector xli and the best position-vector among the swarm in a vector xg. During the iteration t, the position of each particle is updated by:

$$xi(t + 1) = xi(t) + vi(t + 1) \qquad (11)$$

and its vélocity by :

$$vi(t+1)= \omega vi(t) + C_1\rho1(xli(t)-xi(t)) + C_2\rho2(xg(t)-xi(t)) \qquad (12)$$

Where ω is called as the inertia factor, ρ1 and ρ2 are the random numbers, which are used to maintain the diversity of the population, and are uniformly distributed in the interval [0, 1]. C1 is a positive constant, called as coefficient of the self-recognition component; C2 is a positive constant, called: coefficient of the social component. From Eq.(11), a particle decides where to move next, considering its own experience, which is the memory of its best past position, and the experience of its most successful particle in the swarm. The particle swarm optimization algorithm is illustrated as follows:

Algorithm 1 *Particle Swarm Optimization Algorithm*

```
Initialize the size of the particle swarm n, and other parameters.
Initialize the positions and the velocities for all the particles randomly.
While (the end criterion is not met) do
t = t + 1;
    Calculate the fitness value of each particle;
    Determine the xg(global best);
    For i= 1 to n
        Determine the xli (t);
        For j = 1 to Dimension
        Update xi and vi according to Eqs.(11), (12)
        Next j
    Next i
End While.
```

3.2 Differential evolution algorithm

The Differential Evolution method DE was proposed by Storn and Price in 1997 [13]. DE is a powerful population-based stochastic search method for solving optimization problems. The DE technique is simple and easy to use.

A number of recent studies comparing DE with other heuristics, such as Genetic Algorithm (GA) and PSO indicate superiority of DE [14][15]. The literature includes many versions of differential evolution algorithm. The version of DE algorithm used in this work is known as the DE/Rand/1/exp. Classic DE algorithm works as follows: a population of N individuals is randomly initialized where each individual Xi represents a potential solution. A fitness function f is used to evaluate the quality of each solution. At each generation G, three operators named: mutation, crossover and selection are successively applied in order to form a new population.

The mutation generates a mutant vector Yi_G corresponding to each individual target vector Xi_G, such as:

$$Yi\_G = X\ m\_G + F(X\ l\_G - X\ j\_G) \qquad (13)$$

Where, the three vectors solutions j, l and m are generated randomly from the population (i ≠ j ≠ l ≠ m). After that, the 'binominal' crossover operation is applied to each pair of the generated mutant vector Y i_G and its corresponding target vector X i_G to generate a trial vector U i_G:

$$U_{i\_G} = \begin{cases} Y_{i\_G} & \text{if rand}(0,1) \le \text{Cr} \\ \text{X}_{\text{i\_G}} & \text{otherwise} \end{cases} \qquad (14)$$

Where, Cr is a predefined crossover rate which is a constant within the range [0,1]. Cr determines the portion of parameter values copied from the mutant vector. Finally, a selection operation is used to improve the solutions. If the fitness function of the vector Ui_G is less or equal to the target vector Xi_G then U i_G replaces the X i_G+1 on the next generation:

$$X_{i\_G+1} = \begin{cases} U_{i\_G} & \text{if } f(U_{i\_G}) \le f(X_{i\_G}) \\ X_{i\_G} & \text{otherwse} \end{cases} \qquad (15)$$

Where, f represents the fitness function. These steps were repeated until a predetermined generation number is reached.

Three factors control the evolution of DE algorithm: the population size N, the weight applied to the random differential F and the crossover rate Cr. Larger values for F leads to higher diversity in the generated population and lower values in faster convergence. The scaling factor F is a constant in the interval] 0, 1.2].

The pseudo-code for differential evolution algorithm is illustrated in Algorithm 2.

Algorithm 2 Differential Evolution Algorithm (Rand/1/exp)

```
Initialize the size of the population, and other parameters
Initialize population randomly
Evaluate population: evaluate all candidate solutions
Save the best solution X* with its fitness
While (the end criterion is not met) do
        For all individuals i in population pop
        Mutate individuals
        Apply crossover
        Evaluate the new candidate solution
        Apply selection
        Compare the best individual X with X*
            If    X has a fitness value better than X*
                then    Replace X* with X
            End if
        Endfor
End while
Output best recorded solution X*.
```

3.3 Genetic algorithm

The Genetic Algorithm (GA) has been designed the mid 1970s by John Holland at the University of Michigan. Genetic algorithms (GA) are optimization algorithms based on techniques derived from genetics and natural evolution (crossovers, mutations and selection) observed in biological populations.

The literature includes many versions of genetic algorithm [16] [17]. In this work, a basic binary encoded GA with tournament selection, uniform crossover and little probability mutation rate is used to find the fuzzy parameters. In the GA approach, the variables or solutions are first decoded into binary numbers (chromosomes) and so create a population. Each of these chromosomes is then converted into real value using specified lower and upper limits. Then a fitness function is calculated for each chromosome. The GA begins its search from a randomly generated population of designs space that evolve ever successive generations (iterations) converging to optimal solution. The GA uses three main operators to pass its population from one generation to another: selection, crossover and mutation. The first operator selects good chromosomes in a generation and forms the crossover population. No new chromosomes are produced in this operation. The crossover operator transmitted the best features of the current population to the next population, which will have a better fitness value on average. The crossover rate is usually quite large and is between 70% and 95% of the total population, while the rest of them are kept unchanged in the next population. The last operator is mutation allows diversity in population features and prevents the algorithm from getting trapped in local minimum by using mutation probability Pm. These steps were repeated until some termination criterion is reached. The pseudo-code for genetic algorithm is illustrated in Algorithm 3.

Algorithm 3 Genetic Algorithm

```
Initialize the size of the population, and other parameters.
Initialise population randomly
Evaluate population: evaluate all candidate solutions
Save the best solution X* with its fitness
While (the end criterion is not met) do
        SELECT parents;
        RECOMBINE pairs of parents;
        MUTATE the resulting children;
        EVALUATE children;
        SELECT individuals for the new generation
        Compare the best individual X with X*
        If      X has a fitness value better than X*
        then    Replace X* with X
        End if
End while
Output best recorded solution X*.
```

4. PROPOSED APPROACHES

PSO, DE and GA are used to find the a, c pairs which forms a fuzzy c-partition that has the maximum entropy. The proposed methods were presented through an example of multi level thresholding which is the 3-level thresholding. We have to find the two threshold values T1 and T2 of this fuzzy 3-partition such as: T1= (a1opt+c1opt)/2, and T2=(a2opt+c2opt)/2.

The images have 256 grey levels ranging from 0 to 255. The fuzzy parameters: a1, c1, a2 and c2 must satisfy the condition $0 \leq a1 < c1 < a2 < c2 \leq 255$. The fitness function represent the entropy of the fuzzy 3-partition H. The fuzzy 3-partition based on PSO, DE and GA approaches are shown as follows:

The Fuzzy 3-partition based PSO algorithm: It consists of the flowing steps;

1. Enter the image
2. Calculate the histogram
3. Compute the probability of the occurrence of each grey-level
4. Generating randomly particles positions and velocities. The positions represent the fuzzy parameters.
5. Velocity update according to equation.(11) : update the velocities of all particles at time t +1 using the particles fitness values at time t. The fitness function value of a particle determines which particle has the best global value in the current swarm, xg*k*, and also determines the best position of each particle over time, xlk.
6. Position update. The Position of each particle is updated using its velocity vector as shown in equation 12.
7. The steps of velocity update, position update, and fitness calculations are repeated until a fixed number of iterations without any improvement is reach.
8. The two threshold values of this fuzzy 3-partition are :

$$T1=(a_{1opt}+c_{1opt})/2, \text{ and } T2=(a_{2opt}+c_{2opt})/2.$$

In the proposed approach, the swarm contain 20 particles and the value assigned to C_1 is 1.5, C_2 is 1.5 too and w= 0.5.

The Fuzzy 3-partition based DE algorithm: It consists of nine steps;

1. Enter the image
2. Calculate the histogram
3. Compute the probability of the occurrence of each grey-level
4. Generating randomly population with the individuals represent the fuzzy parameters.
5. Evaluate population, evaluate all candidate solutions. Store the best solution $S^*=(a_{1opt}, c_{1opt}, a_{2opt}, c_{2opt})$ with its fitness.
6. Generate next population by performing mutation, crossover and selection operations.
7. Update the best solution. Compare the best individual S of the current population with S*. If S has a fitness value better than S* then replace S* with S.
8. The steps of the evaluation population, generating next population and updating of the best solution are repeated

until a fixed number of iterations without any improvement is reach.

9. The two threshold values of this fuzzy 3-partition are :

$T1=(a_{1opt}+c_{1opt})/2$, and $T2=(a_{2opt}+c_{2opt})/2$.

In the proposed approach, the population include 6 individuals in the population and the value assigned to *F* is 0.3 and Cr is 0.9.

The Fuzzy 3-partition based GA algorithm: It consists of nine steps;

1. Enter the image
2. Calculate the histogram
3. Compute the probability of the occurrence of each grey-level
4. Generating randomly population with the individuals represent the fuzzy parameters.
5. Evaluate population, evaluate all candidate solutions. Store the best solution S*=(a1opt, c1opt, a2opt ,c2opt) with its fitness.
6. Generate next population by performing selection (tournament selection), crossover (uniform crossover) and mutation (Gaussian mutation with fixed mutation rate) operations.
7. Update the best solution. Compare the best individual S of the current population with S*. If S has a fitness value better than S* then replace S* with S.
8. The steps of the evaluation population, generating next population and updating of the best solution are repeated until a fixed number of iterations without any improvement is reach.
9. The two threshold values of this fuzzy 3-partition are $T1=(a_{1opt}+c_{1opt})/2$, and $T2=(a_{2opt}+c_{2opt})/2$.

In the proposed approach, the population consist of 4 individuals in the population and the value assigned to Pc is 0.5 and Pm is 0.15.

5. EXPERIMENTAL RESULTS

The proposed PSO, DE and GA algorithms for a fuzzy c-partition thresholding have been tested on many images to certify its efficiency. Each image is presented by eight bits, that is, grey levels are ranging from 0 (the darkest) to 255 (the brightest). We present and discuss the experimental results of the proposed method through four examples of image segmentation. The images test used are the well known Lena Fig.1 (A), Cameraman Fig.1 (B), Peppers Fig 1(C) and Lake Fig 1(D). The histograms of these images are shown in Fig.1 (A1, B1, C1 and D1).

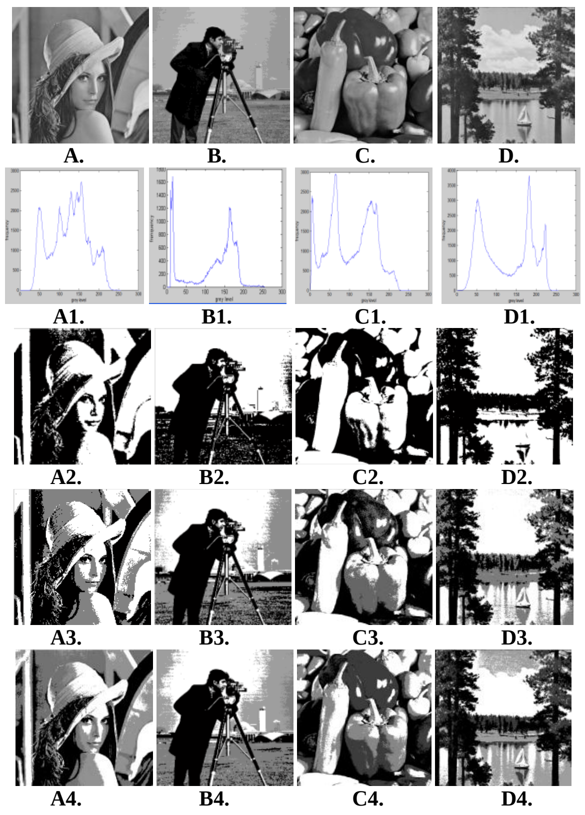

Fig 1: The tested images: (A) Lena, (B) Cameraman, (C) Peppers and (D) Lake Their Gray level histograms : (A1) Lena, (B1) Cameraman, (C1) Peppers and (D1) Lake and the thresholded images for c=2,3,4 : (A2,A3, A4) Lena , (b2,B3, B4) Cameraman, (C2,C3, C4) Peppers and (D2,D3, D4) Lake.

The proposed approaches can be simply extended to c level (c>3) thresholding. For c-level thresholding, there will be c membership functions μ1, μ2, . . . , μc, which includes 2(c-1) parameters a1; c1; a2; c2; . . . ; ac-1 ; cc-1 and these parameters satisfy the following condition: 0<a1<c1<a2<c2<. . .<ac-1<cc-1 < 255. Then we can also compute the optimal thresholds which correspond to the maximum of fuzzy entropy. The optimal combination of all 2(c-1) fuzzy parameters is obtained by using PSO, DE and GA as presented above, therefore, the optimal thresholds.

The value of the best fitness f(T*) is used to compare the three techniques adopted for multilevel thresholding, in order to evaluate the quality of the solutions obtained. Additional results are presented in order to examine the time execution of each approach. Peak signal to noise ratio (PSNR) is used to determine the quality of the thresholded image. The PSNR give the similarity of an image against a reference image based on the mean square error (MSE) of each pixel:

$$PSNR = 20\log_{10}\left(\frac{255}{RMSE}\right) \qquad (16)$$

Where, RMSE is the root mean-squared error, defines as:

$$RMSE = \sqrt{\frac{1}{MN}\sum^{M}\sum^{N}\left[I(i,j)-\hat{I}(i,j)\right]^2} \qquad (17)$$

Here I and Î are the original and thresholded images of size MxN, respectively.

The PSO, DE and GA algorithms are very good and permit a good thresholding (Fig. 1 (A2, B2, C2, D2, A3, B3, C3, D3, A4, B4, C4 and D4,). The main features of the images are well preserved. For example, the main features of the image of Cameraman such as man, sky, land and camera, are preserved.

The three techniques have several parameters. The values of these parameters considerably affect the performance of the algorithms. So, for each method, we varied the parameter values to be set while fixing the values of the other parameters. The optimal thresholds found using the three algorithms are: T, T1, T2 and T3. The numerical results obtained using algorithms PSO, DE and GA with c=2, 3 and 4 are presented in Table 2, 3 and 4. The performances of three approaches were compared using the fidelity criterion: the peak-to-signal-noise (PSNR) ratio. The methods adopted provide good results in terms of accuracy and robustness. However, the GA is the most efficient in terms of time execution (see figure 1). The results in table 2, 3 and 4 show that the best time processing is providing by PSO algorithm (see figure 2) and PSNR values of the best solutions (fitness) are obtained from PSO algorithm too.

Table 2. Experimental results using PSO algorithm for the test images

Image	Number of Classes	Value of Fitness	Optimal Thresholds	Time(s)	PSNR(dB)
Lena	2	0.6931	T=124	0.0013	27.4120
	3	1.0986	T1=104, T2=149	0.0331	27.4862
	4	1.3863	T1=85,T2=130,T3=160	0.1450	26.5774
Cameraman	2	0.6931	T=143	0.0011	27.7169
	3	1.0986	T1=113,T2=159	0.0311	28.2631
	4	1.3863	T1=70,T2=145,T3=167	0.1506	27.3029
Peppers	2	0.6931	T=108	0.0012	27.6000
	3	1.0986	T1=69,T2=139	0.0210	27.7169
	4	1.3863	T1=54,T2=135,T3=154	0.1522	27.9189
Lake	2	0.6931	T=124	0.0011	27.0587
	3	1.0986	T1=73,T2=162	0.0273	27.7966
	4	1.3863	T1=66,T2=149,T3=183	0.1539	26.1996
				m,	27.4207
				σ	0.5766

Table 3. Experimental results using DE algorithm for the test images

Image	Number of Classes	Value of Fitness	Optimal Thresholds	Time(s)	PSNR(dB)
Lena	2	0.6931	T=124	0.0053	27.4120
	3	1.0986	T1=103, T2=148	0.0094	27.6386
	4	1.3863	T1=85,T2=137,T3=157	0.0830	26.4282
Cameraman	2	0.6931	T=120	0.0055	29.3227
	3	1.0986	T1=113,T2=159	0.0166	28.2631
	4	1.3863	T1=68,T2=145,T3=166	0.0809	27.3753
Peppers	2	0.6931	T=104	0.0057	27.7169
	3	1.0986	T1=66,T2=138	0.0039	28.0448
	4	1.3863	T1=53,T2=117,T3=153	0.0846	28.2185
Lake	2	0.6931	T=125	0.0052	27.0249
	3	1.0986	T1=76,T2=175	0.0132	27.1966
	4	1.3863	T1=64,T2=131,T3=184	0.0823	26.0625
				m	27.5586
				σ	0.8685

As is apparent, from Figure 3, PSO thresholding has lowest average PSNR of 27.4207 with the lowest standard deviation of 0.5766. The fuzzy c-partition entropy using PSO, DE and GA algorithms perform equally well in terms of the processing time and the quality of image segmentation.

Table 4. Experimental results using GA algorithm for the test images

Image	Number of Classes	Value of Fitness	Optimal Thresholds	Time(s)	PSNR(dB)
Lena	2	0.6931	T=124	0.0317	27.4120
	3	1.0986	T1=103, T2=149	0.0738	27.5618
	4	1.3850	T1=82,T2=129,T3=157	0.1637	26.7007
Cameraman	2	0.6931	T=142	0.0326	27.7565
	3	1.0986	T1=106,T2=160	0.0457	28.2631
	4	1.3849	T1=74,T2=148,T3=167	0.1480	27.1966
Pe ppers	2	0.6931	T=108	0.0325	27.6000
	3	1.0986	T1=68,T2=142	0.1031	27.7169
	4	1.3853	T1=53,T2=123,T3=152	0.2407	28.0876
Lake	2	0.6931	T=125	0.0312	27.0249
	3	1.0986	T1=72,T2=162	0.0584	27.7966
	4	1.3858	T1=64,T2=148,T3=183	0.1125	26.2839
				m	27.4500
				σ	0.5685

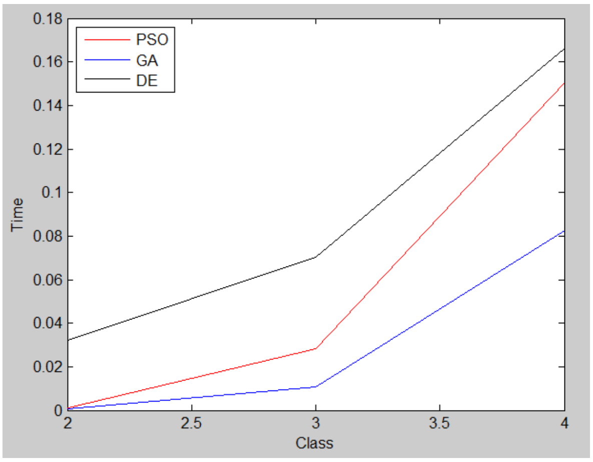

Fig 2: Average execution time variation according to the number of classes for the three approaches (PSO, GA and DE).

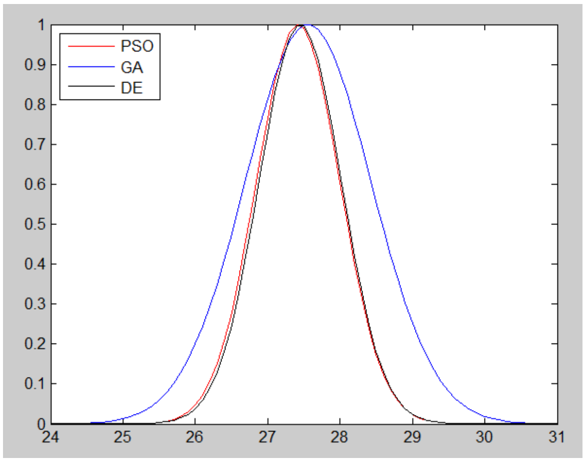

Fig 3: PSNR Distribution for the three approaches (PSO, GA and DE)

6. CONCLUSION

In this work, three approaches (PSO, DE and GA) were proposed to find the suitable thresholds of an image, based on the concept of fuzzy c-partition and maximum entropy principle. The image can hold as much information as possible when it is transformed to the fuzzy domain. The algorithms are extended to multiple thresholding. The use of PSO, GA and DE algorithms reduce greatly the time complexity. The three approaches were then compared by testing them on various images. The algorithms are comparable in terms of solution quality for c=2. While the value of c increases, PSO, DE and GA provide the same results in terms of accuracy and robustness of the results, but in terms of execution time the GA is most efficient. The experimental results have shown the effectiveness and usefulness of the proposed algorithms for image segmentation. The PSO, DE and GA approaches can deliver satisfactory performance.

7. REFERENCES

[1] J.-P. Cocquerez, et al., " Analyse d'images: filtrage et segmentation, Masson, Paris, pp. 239–280, 1995.

[2] M. Sezgin, B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation", Journal of Electronic Imaging 13(1), pp 146–165, 2004.

[3] O.D. Trier, A. Jain, "Goal-directed evaluation of binarization methods", IEEE Trans. PAMI, pp 1191–1201, 1995.

[4] H.D. Cheng, J.-K. Chen, J. Li, "Threshold selection based on fuzzy c-partition entropy approach", Pattern Recognition 1, pp 857–870, (1998).

[5] Synder W., Bilbro G., Logenthiran A., Rajala S.,"Optimal thresholding A new approach", Pattern Recognition Letters, 11, pp 803–810, 1990.

[6] Tsai W. H., "Moment-preserving thresholding: a new approach", Computer Vision,Graphics and Image Processing, Vol-29, pp 377-393, 1985.

[7] Weszka J., Rosenfeld A., "Histogram modifications for threshold selection", IEEE Transaction on Systems Man Cybernet, 9, 38–52, 1997

[8] N. Otsu, "A threshold selection using gray level histograms", IEEE Trans. Systems Man Cybernet, 9, pp 62–69, 1979.

[9] S. Benabdelkader, M. Boulemden, "Recursive algorithm based on fuzzy 2-partition entropy for 2-level image thresholding", Pattern Recognition 38, pp 1289–1294, 2005.

[10] S. Benabdelkader, M. Boulemden, S. Louifi, "Threshold selection by maximising the between class variance of a fuzzy 2-partition", Proceedings of the Ninth International Workshop on Systems, Signals and Image Processing, Manchester, UK, pp. 282–288, 2002.

[11] E. Cuevas, D. Zaldivar, "A novel multi-threshold segmentation approach based on differential evolution optimization", Expert Systems with Applications 37, pp 5265-5271, 2010.

[12] J. Kennedy, R.C. Eberhart.. "Particle Swarm Optimization", Proc. IEEE Int. Conf. On Neural Networks, pp.1942-1948, WA, Australia, 1995.

[13] R. Storn, K.Price, "Differential evolution-a simple and efficient heuristic foroptimization over spaces", Journal of global Optimization 11(4). pp 341-359, 1997.

[14] J. Vesterstrom, R. Thomsen."A comparative study of differential evolution, particle swarm optimization and evolutionary algorithms on numerical benchmark problems" proceedings of Sixth Congress on Evolutionary Computation. IEE Press Piscataway. NJ. USA. pp. 1980-1987., 2004

[15] S. Paterlini, T. Krink " Differential evolution and particle swarm optimisation in partitional clustering".Computational Statistics &Data Analysis 50. pp 1220-1247, 2006

[16] R. L. Haupt, S.E.. Haupt, Practical Genetic Algorithms,John wiley & sons, INC., Publication. ISBN 0-471-45565-2. 2004

[17] L. Davis, Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York. 1991