International Journal of Computer Information Systems and Industrial Management Applications (IJCISIM)
ISSN: 2150-7988 Vol.2 (2010), pp.121-136
http://www.mirlabs.org/ijcisim

Fast and Accurate Watermark Retrieval Using Evolutionary Algorithms

E.V. Gopal, Munaga.V.N.K. Prasad*, V. Ravi
Institute for Development and Research in Banking Technology, Castle Hills Road #1, Masab Tank, Hyderabad 500 057, AP, India
gopaledupuganti@gmail.com, mvnkprasad@idrbt.ac.in, rav_padma@yahoo.com

Abstract

A Watermark hidden in an image is retrieved differently from the original watermark due to the frequently used rounding approach. The simple rounding will cause numerous errors in the embedded watermark especially when it is large. Evolutionary algorithms (EA) are used to correct the rounding errors. The main issue before applying EAs to reduce the rounding errors is that the embedded data should be retrieved correctly. Evolutionary algorithms are commonly used as adaptive approaches that provide a randomized , parallel and global search method based on the mechanics of natural selection and natural genetics in order to find solutions of problems. We present the application of Genetic algorithms (GA), Differential Evolution (DE) and Simplified Threshold Accepting algorithm (STA) to enhance the watermark retrieval. Experimental results show that STA converges faster to the optimal solution than DE, which in turn converges faster than GA.

1. Introduction

With the large usage of Internet and the development in computer industry, the digital media, including images, audio and video, are easily obtained in the real world. Digital multimedia contents suffer from attacks like duplication of copyrights, easy modification and transmission over the Internet. Digital watermarking of electronic images inherently places a digital id entity or watermark into the original image itself, to protect the ownership of the original sources.

Digital watermarking can be classified as visible and invisible (Shih and Wu, 2005).

* Corresponding author. Phone: +91 40 23534981 Ext. 2132; FAX: +91 40 23535157.

1. The visible watermarks such as bills, company logos can be viewed by eyes. Although the watermarks are viewed without any calculation, the embedded watermarks can be destroyed easily.

2. In invisible watermarking scheme, the embedding locations of the watermark are secret, only the authorized persons can extract the watermark. Although some mathematical calculations required retrieving the watermark, the invisible watermarks are more secure and robust than visible watermark s.

On the other hand, digital watermarking can also be categorized as robust and fragile watermarking. Robust watermarks (Cox et al., 1997; Lin and Chen , 2000; Nikolaidis and pitas, 1998) are designed to have the ability to detect the watermark after some image processing operations such as image scaling, bending, cropping, and so on. Robust watermarks are mainly used for copyright protection. In contrast, fragile watermarks (Celik et al., 2002; Wong ,1998) are designed to b eco me invalid after even the slightest modification of the watermark ed image. Fragile watermarks are mainly used for authentication purposes.

Image watermarking can be performed in two ways, one in spatial domain and the other in frequency domain. In the spatial domain (Bruyndonck x et al., 1995; Nikolaidis and Pitas, 1998), insert the watermark into a host image by changing the gray levels of some pixels in the host image. The disadvantage of this approach is that the embedded information may be easily detected using computer analysis. On the other hand, in frequency domain (Bas et al., 2002; Huang et al., 2000; Lin and Chen, 2000) transform the image into frequency domain using Discrete Cosine Transform (DCT), Discrete Fourier Transform (DFT) or Discrete Wavelet Transform (DWT). Then embed the watermark into the coefficients of the transformed image. Though it is difficult to detect a watermark in this approach, there are two major defects. First, we cannot embed too much data in the frequency domain because the quality of the host image will be distorted significantly. To increase the watermark capacity and imperceptibility, combinational image watermarking in the spatial and frequency domains (Shih and Wu, 2003) is proposed. Second, the data embedded in coefficients of the transformed image will be somewhat disturbed due to deviations in converting real numbers into integers (rounding approach) in spatial domain (Shih and Wu, 2005). To reduce rounding errors evolutionary algorithms are used and is depicted in Figure 1. In this paper we employed DE to enhance the watermark retrieval. We also modified the TA algorithm named it STA (Simplified TA) and applied it to retrieve the watermark correctly. We compared the performance of GA, DE and STA.

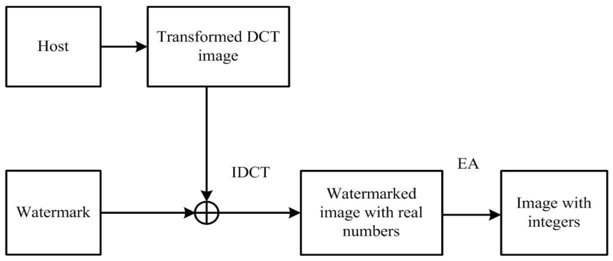

Figure 1. The flowchart showing the usage of evolutionary algorithms in reducing rounding

The paper is organized as follows. Section 2 presents the embedding algorithm. Section 3 introduces the errors caused by deviations in translating real numbers into integers and the overview of evolutionary algorithms (EA) used in reducing errors. Section 4 describes proposed methodology of applying EA's to solve the problem. Experimental results are presented in section 5. Finally conclusions are made in section 6.

2. Embedding Algorithm

Several approaches can be used in embedding a watermark in the frequency domain. In this paper we embed the watermark in the coefficients of the transformed image (Shih and Wu, 2005). The transformation functions generally used are DCT, DFT and DWT. Fig.1 shows the flowchart of our algorithm.

2.1 Algorithm

1. Divide the host image into sets of 8 x 8 blocks. Let H be the original host image with size $N \times N$. $H=\{h(i,j), 0 \le i,j < N\}$, $H^m=\{h^m(i,j), 0 \le i,j < 8\}$, where $h^m(i,j) \in \{0,1,2,\ldots,2^L-1\}$, L is the number of bits used to represent gray level of pixels and m is the total number of the 8 x 8 blocks.
2. Divide the watermark image into sets of 2 * 2 blocks. Let W be the binary watermark image with size $M \times M$. $W=\{w(i,j), ,0 \le i,j < M\}$, where $w(i,j) \in \{0,1\}$
$W^n=\{w^n(i,j), ,0 \le i,j < 2\}$, where n is the total number of 2×2 blocks.
3. Transform H^m to H^{m_DCT} by DCT. $h^{m\_DCT}(i,j) \in \mathbf{R}$
4. Insert W^n into the coefficients of H^{m_DCT}. $H^{m\_F}=\{h^{m\_F}(i,j)=h^{m\_DCT} \oplus w^n(i,j),\ 0 \le i,j < 8\}$, $h^{m\_F}(i,j) \in \mathbf{R}$
5. Transform the embedded host image H^{m_F}, by Inverse DCT to obtain H^{m_IDCT}, $h^{m\_IDCT} \in \mathbf{R}$.
6. Find the suitable solution to translate all real numbers in H^{m_IDCT} into integers, and obtain H^{EA}. $H^{EA}=\{h^{EA}(i,j), ,0 \le i,j < 8\}$, where $h^{EA}(i,j) \in \{0,1,2,\ldots,2^L-1\}$

3. Overview of Evolutionary Algorithms (EA) applied in reducing errors

In the previous section we observed that the data embedded in the coefficients of the transformed image will be somewhat disturbed due to deviations in converting real numbers to integers (rounding approach) in spatial domain. For instance, Figure 2 shows the extracted watermark after IDCT coefficients are rounded. Figure 2A is the original host image, an 8 X 8 gray-level image, in the spatial domain and Figure 2B is the transformed image of Figure 2A by DCT. Figure 2C is a binary watermark in which "0" and "1" represents the embedded data in its location; the minus sign "-" indicates no change in its position. Fig 2D obtained by embedding Figure 2C into Figure 2B based on LSB modification. We can find three differences when comparing Figures 2B and 2D; for instance (8.646 and 9.646), (6.199 and 7.199) and (-0.765

and 0.235). After transforming Figure 2D into its spatial domain by IDCT, we obtain Figure. 2E where all pixels are real numbers. After translating real numbers in to integers by round in g the real numbers, we obtain Figure 2F. Figure 2G is the transformed image from Figure. 2F by DCT. We obtain the watermark by extracting the bits from the same position in which we embedded the watermark, as shown in Figure. 2H. We can observe from the Figure.2 that the embedded and extracted watermarks are not equal.

165	163	161	161	162	161	159	158
164	162	160	160	161	160	159	157
162	160	158	158	159	159	157	156
160	158	157	157	158	158	156	155
158	157	155	156	157	157	156	155
158	156	155	156	157	158	156	155
158	156	155	156	158	158	157	156
158	156	155	156	158	159	158	156

Figure 2A. Original Host Image

1264.125	6.729	-0.765	9.158	0.145	0.803	-1.070	1.152
18.538	6.199	0.223	-0.366	0.197	-0.142	-0.184	0.171
8.646	-0.175	-0.020	0.568	0.063	0.156	-0.219	0.438
1.486	-0.006	-0.233	-0.351	0.328	-0.149	-0.081	-0.248
-1.595	-0.171	-0.163	-0.156	-0.125	-0.100	-0.067	-0.037
1.659	0.248	-0.189	-0.071	0.370	-0.249	0.580	0.106
-2.106	0.055	0.036	0.288	-0.160	-0.035	0.520	-0.459
1.195	-0.264	0.405	-0.173	0.163	-0.369	-0.439	-0.080

Figure 2B. Transformed image of 2A

-	-	1	1	-	-	-	-
-	1	-	-	-	-	-	-
1	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-

Figure 2C. Binary Watermark

1264.1 25	6.729	0.235	9.158	0.145	0.803	-1.070	1.152
18.538	7.199	0.223	-0.366	0.197	-0.412	-0.184	0.171
9.646	-0.175	-0.020	0.568	0.063	0.156	-0.219	0.438
1.486	-0.006	-0.233	-0.351	0.329	-0.149	-0.081	-0.247
-1.595	-0.171	-0.163	-0.156	-0.125	-0.100	-0.067	-0.037
1.659	0.248	-0.189	-0.072	0.370	-0.249	0.580	0.106
-2.106	0.055	0.036	0.288	-0.160	-0.035	0.520	-0.459
1.195	-0.264	0.405	-0.173	0.163	-0.369	-0.439	-0.080

Figure 2D. Transformed Image after embedding watermark

167.137	164.952	162.853	162.500	163.657	162.286	160.959	158.760
165.569	163.392	161.302	160.964	162.132	160.779	160.456	157.277
162.788	160.619	158.548	158.235	159.432	159.105	157.800	155.630
159.845	157.693	156.647	156.367	157.595	157.301	156.018	153.874
157.691	156.550	154.529	155.287	156.556	156.300	156.043	153.907
157.163	155.041	154.047	154.835	156.136	156.907	155.679	153.564
158.246	156.124	155.148	155.963	158.299	158.102	157.889	155.766
156.629	154.533	153.570	154.394	156.728	157.534	157.338	154.252

Figure 2E. Watermarked Image after IDCT Transform

167	165	163	162	164	162	161	159
166	163	161	161	162	161	160	157
163	161	159	158	159	159	158	156
160	158	157	156	158	157	156	154
158	157	155	155	157	156	156	154
157	155	154	155	156	157	156	154
158	156	155	156	158	158	158	156
157	155	154	154	157	158	157	154

Figure 2F. Rounded Image

1264.5	7.904	-0.196	9.957	-0.979	1.384	-1.978	2.858
27.158	7.250	0.363	-0.345	0.414	-0.364	-0.171	0.412
11.881	-0.260	-0.236	0.969	0.035	0.145	0.046	0.187
3.228	-0.608	-0.363	-0.540	0.611	-0.054	-0.414	-0.182
-3.181	0.353	-0.325	-0.035	-0.245	-0.868	-0.137	0.785
3.664	-0.241	4.861	-0.579	0.255	-0.263	0.241	0.289
-3.999	-0.101	0.302	-0.040	-0.552	-0.137	0.483	-0.395
3.598	-0.130	0.668	-0.492	-0.053	-0.425	-0.534	0.085

Figure 2G. DCT Image of the rounded image2F

-	-	0	1	-	-	-	-
- 1	1 -	- -	- -	- -	- -	- -	- -
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-

Figure 2H. extracted watermark

Since we cannot predict what will happen in the frequency domain o f a host image if we change some values of pixels in the spatial domain, correcting the rounding errors becomes a difficult task . To reduce the errors caused during conversion from real numbers obtained after IDCT to integers in spatial domain, we used evolutionary algorithms (EAs), such as Genetic algorithms (GA), Differential Evolution (DE) and Simplified Threshold Accepting (STA).

3.1 Genetic Algorithms (GA)

Genetic algorithms introduced by Holland (Holland, 1975) provide a randomized, parallel, and global search method based on the mechanics of natural selection and natural genetics in order to find solutions of problems. GAs are different from normal optimization and search procedures in four ways (Herrera et al., 1994)

(1) GAs work with a coded parameter set, not the parameters themselves, (2) GAs search from random selected points, not from a single point (3) GAs use objective function information, and 4) GAs use probabilistic transition rules, not deterministic ones.

Generally, GAs start with so me randomly selected genes in the first generation , called population. Each individual in the population corresponds to solution in the problem do main and is called chromosome. An objective (fitness function) is used to evaluate the quality of each chromo some. The chromosomes with high quality will survive and form the population of the n ext generation. By u sing the reproduction, crossover and mutation operations a new generation is recombined in order to find the best solution. This process will rep eat until a pre-specified condition called convergence criterion is satisfied , or a constant number of iteration s is reached. Genetic algorithms consume more iterations and time to minimize the rounding errors. So in order to reduce the time and iterations we used Differential evolution (Bhat et al., 2006; Karboga and Okdem, 2004) and threshold accepting algorithms (Ravi et al., 2000a).

3.2 Differential Evolution

The optimization method used for the solution of the inverse problem in our work, namely, the Differential Evolution, is a new generation Evolutionary Algorithm (EA) and we employed an improved version of this method in our work. EA derive their name from natural biological evolutionary processes, some of which are mimicked for obtaining solution to an optimization problem.

One can use EA for problems that are difficult to solve with traditional optimization techniques, including problems that are not well defined or are difficult to model mathematically. A popular optimization method, which belongs to the EA class of methods, is the Genetic Algorithm (GA). In recent times, Genetic Algorithm has replaced the traditional methods as the preferred optimization tool, as several studies have conclusively proved. Differential Evolution is relatively a new algorithm. Price and Storn (1997) have first proposed it in 1997. Its popularity has been catching up, of late. It is fast in numerical optimization and is more likely to find the true optimum (Karboga and Okdem, 2004; Price and Storn, 1997).

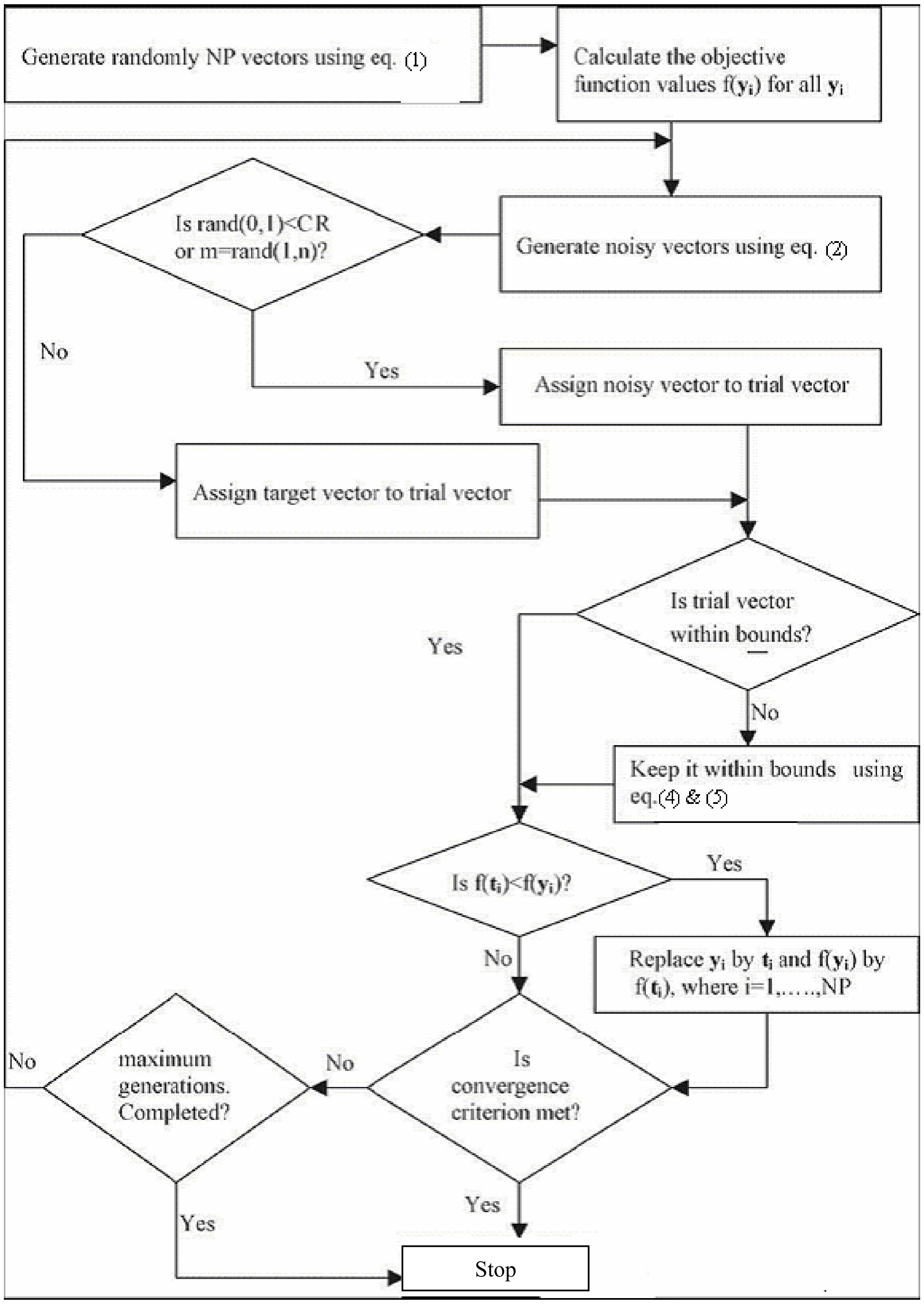
Generate randomly NP vectors using eq. (1)
Calculate the objective function values f(yᵢ) for all yᵢ
Generate noisy vectors using eq. (2)
Is rand(0,1)<CR or m=rand(1,n)?
No
Yes
Assign noisy vector to trial vector
Assign target vector to trial vector
Is trial vector within bounds?
Yes
No
Keep it within bounds using eq.(4) & (5)
Is f(tᵢ)<f(yᵢ)?
Yes
No
Replace yᵢ by tᵢ and f(yᵢ) by f(tᵢ), where i=1,…..,NP
No
maximum generations. Completed?
No
Is convergence criterion met?
Yes
Yes
Stop

Figure 3. Flowchart of DE

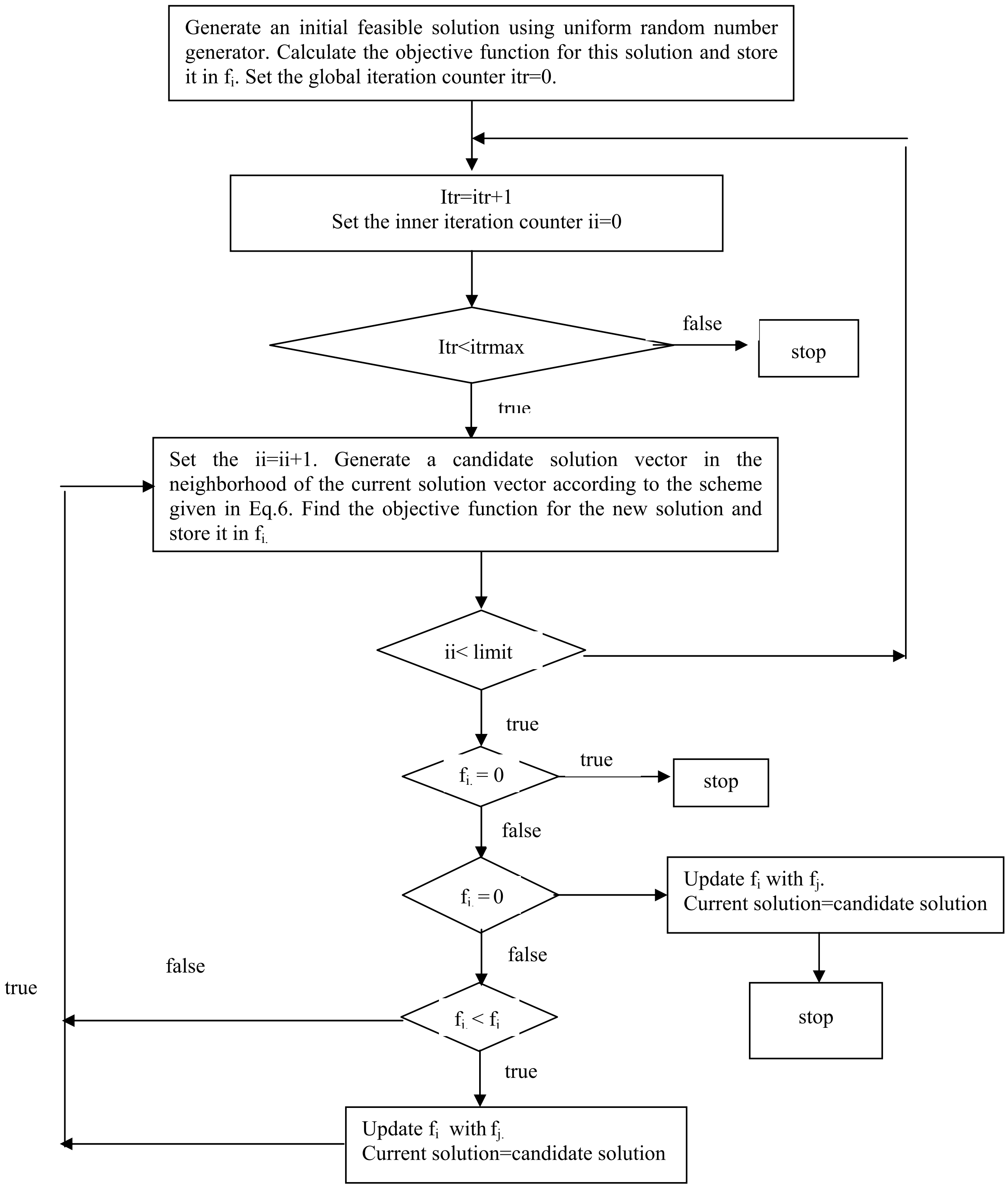
Generate an initial feasible solution using uniform random number generator. Calculate the objective function for this solution and store it in f_i. Set the global iteration counter itr=0.
Itr=itr+1
Set the inner iteration counter ii=0
Itr<itrmax
false
stop
true
Set the ii=ii+1. Generate a candidate solution vector in the neighborhood of the current solution vector according to the scheme given in Eq.6. Find the objective function for the new solution and store it in $f_{i.}$
ii< limit
true
$f_{i.} = 0$
true
stop
false
$f_{i.} = 0$
Update f_i with f_j.
Current solution=candidate solution
stop
false
false
true
$f_{i.} < f_i$
true
Update f_i with $f_{j.}$
Current solution=candidate solution

Figure 4. Flowchart of Simplified TA

3.2.1 Differential Evolution for unconstrained optimization.

The method of differential evolution (Price and Storn, 1997; Bhat et al., 2006,) consists mainly of four steps: Initialization, Mutation, Recombination and Selection. In the first step, a random population of potential solutions is created within the multi-dimensional search space. To start with, we define our objective function f(**y**) to be optimized , where **y**=(y^1, …, y^n) is a vector of n decision variables. The aim is to find a vector **y** in the given search space, for which the value of the objective function is an optimum. The search space is first defined by providing the lower and upper bounds for each of the n decision variables of **y**, i.e., $\mathbf{ymin} \le \mathbf{y} \le \mathbf{ymax}$. In the initialization step, NP vectors, each of n dimensions, are randomly initialized. The parameters are encoded as floating point numbers.

Mutation is basically a search mechanism, which, together with recombination and selection, directs the search towards potential areas of optimal solution. In this step, three distinct target vectors $\mathbf{y}_a$, $\mathbf{y}_b$ and $\mathbf{y}_c$ are randomly chosen from the NP parent population on the basis of 3 random numbers a, b and c. One of the vectors $\mathbf{y}_c$ is the base of the mutated vector. To this is added the weighted difference of the remaining two vectors, i.e. $(\mathbf{y}_a - \mathbf{y}_b)$ to generate a noisy random vector, $\mathbf{n}_i$. The weighting is done using a scaling factor F, which is u ser-supplied and is usually in the range 0 to 1.2. This mutation process is repeated to create a mate for each member of the parent population.

In the recombination (crossover) operation, each target vector of the parent population is allowed to undergo recombination by mating with a mutated vector. Thus, vector $\mathbf{y}_i$ is recombined with the noisy random vector, $\mathbf{n}_i$ to generate a trial vector, $\mathbf{t}_i$. Each element of the trial vector ($t_{i\ m}$, where i = 1, …., NP and m=1,….,n), is determined by a binomial experiment whose success or failure is determined by the user-supplied crossover factor, CR. The parameter CR is used to control the rate at which the crossover takes place. Trial vector, $\mathbf{t}_i$, is, thus, the child of two parent vectors: noisy random vector, $\mathbf{n}_i$ and the target vector, $\mathbf{y}_i$. DE performs a non-uniform crossover, that determines which trial vector parameters are inherited from which parent.

It is sometimes possible that some particular combinations of three target vectors from the parent population and the scaling factor F would result in noisy vector values, which are outside the bounds set for the decision variables. It is necessary, therefore, to bring such values within the bounds. For this reason, the value of each element of the trial vector is checked at the end of the recombination step. If it violates the bounds, it is heuristically brought back to lie within the bounded region. It is in the last stage of the 'selection ', the trial vector is pitted against the target vector and the fitness is tested and fitter of the two vectors survives and proceeds to the next generation.

After NP competitions of this kind in each generation , one will have a new population, which is fitter than the population in the previous generation. This evolution procedure consisting of the above four steps is repeated over several generations until the termination condition is reached, i.e. when the objective function attains a prescribed optimum or a specified number of generations are completed, whichever happens earlier.

3.3 Simplified Threshold Accepting

In the recent literature, global optimization techniques such as, Simulated Annealing, Tabu Search, Genetic Algorithms, Threshold Accepting are all grouped in one category and called metaheuristics (Osman and Kelly, 1996). Threshold Accepting, proposed by Dueck and Scheuer (Dueck and Scheuer, 1990), is a variant of the original simulated annealing algorithm in that the acceptance of a new move or solution is determined by a deterministic criterion rather than a probabilistic criterion. Dueck and Scheuer (Dueck and Scheuer, 1990) showed through numerical experimentation that threshold accepting is superior to simulated annealing for solving combinatorial global optimization problems. In this paper, the threshold accepting algorithm has been modified and adapted to the problem already described in the beginning of this section.

Threshold Accepting has been used in optimizing the fuzzy rule base while maximizing the classification rate in fuzzy rule based classifiers (Ravi et al., 2000, 2001; Ravi and Zimmermann, 2000), optimization of complex system reliability (Ravi et al., 2000a), in neural-fuzzy rule based classifiers (Ravi and Zimmermann, 2001), training of neural networks for classification and forecasting problems (Ravi and Zimmermann , 2003; Ravi et al., 2005) and in designing new fuzzy clustering algorithms (Ravi et al., 2006). Based on this experience on TA, we decided to investigate the effectiveness of TA to expedite the watermark retrieval.

4. Methodology of applying EAs to solve the problem

4.1 Genetic Algorithms

To apply GAs in solving our problem, a chromosome 'C' consisting of 64 genes C=g0 g1 g2 . . . g63, is considered, where g0, g1, g2, … g63 correspond to the pixels in 8 * 8 block of host image (Shih and Wu, 2005). Here each g is either 0 or 1. The evaluation (objective) function is difference between embedded and extracted watermarks.

$$\text{Evaluation}\, fn = \sum_{i=0}^{all\ pixels} \left| watermark^{em} - watermark^{ex} \right|$$

Where $watermark^{em}$ is the embedded watermark and $watermark^{ex}$ is the extracted watermark. Our goal is to minimize the objective function value. For each sample solution obtained by genetic algorithm we translate real numbers(r) into integers(r*) using the following rules: (1) If the signal is "1" ,r * = Trunc(r)+1 (2) If the signal is "0", r* = Trunc(r) where Trunc(r) is the integer part of r.

Algorithm

1. Define the fitness function, number of genes, sizes of population, crossover rate, critical value and mutation rate.
2. Initial population is randomly assigned with 0's and 1's.
3. Evaluate the fitness value for each corresponding chromosome.
4. Reproduction, Crossover and Mutation operators (Shih and Wu, 2005) are applied to generate next generation of chromosomes
5. Repeat steps 3-5 until a predefined condition is satisfied, or a constant number of iterations is reach ed.

4.2 Differential Evolution

To apply DE in solving our problem, a vector 'V' consisting of 64 parameters V=g0 g1 g2 . . . g63, is considered, where g0, g1, g2, … g63 correspond to the pixels in 8 * 8 block of host image (Shih and Wu , 2005). Here each g is a real value between 0 and 1.

Objective (Fitness) Function:

The evaluation (objective) function is difference between embedded and extracted watermarks.

$$Evaluation\, fn = \sum_{i=0}^{all\ pixels} \left| watermark^{em} - watermark^{ex} \right|$$

Because DE works with real values, we use the following two rules to convert real numbers(r) into integers(r*)
(1) If the signal is "≥ 0.5" ,r * = Trunc(r)+1
(2) If the signal is "< 0.5", r* = Trunc(r)
where Trunc (r) is the integer part of r.

Algorithm

1. The first step is the random initialization of the parent population. Generate randomly NP vectors, each of n dimensions:

$$y_i^m = y\min{}^m + rand(0,1) * (y\max{}^m - y\min{}^m)$$

where i = 1,2….., NP and m = 1,2…., n (1)

2. Calculate the objective function values f(**y**$_i$) for all **y**$_i$.
3. Select three random numbers (y_a, y_b, and y_c) within the range 1 to NP. The weighted difference (**y**$_a$ – **y**$_b$) is used to perturb (**y**$_c$) to generate a noisy vector **n**$_i$:

$$n_i = y_c + F * (y_a - y_b)$$

where i = 1, 2….., NP (2)

4. Recombine each target vector **y**$_i$ with the noisy random vector **n**$_i$ to generate a trial vector **t**$_i$:

$$t_i^m = n_i^m \; if\; rand(0,1) < CR\; or\; m = rand(1,n);\; t_i^m = y_i^m\; otherwise,$$

where i = 1,2….., NP and m = 1,2…., n (3)

5. Check whether each decision variable of the trail vector is within the bounds. Otherwise force it to lie within the bounds using:

$$t_i^m = y\min{}^m + 2.0*(p/q)*(y\max{}^m - y\min{}^m), if\; t_i^m > y\max{}^m$$
$$with\; p = (t_i^m - y\max{}^m)\; and\; q = (t_i^m - y\min{}^m) \qquad (4)$$
$$t_i^m = y\min{}^m + 2.0*(p/q)*(y\max{}^m - y\min{}^m), if\; t_i^m < y\min{}^m$$
$$with\; p = (y\min{}^m - t_i^m)\; and\; q = (y\max{}^m - t_i^m) \qquad (5)$$

6. Calculate the value o f the objective function for the two vectors **t**$_i$ and **y**$_i$. Fitter of the two (one with the lower objective function value) survives and proceeds to the next generation.
7. Check if convergence criterion met. If yes stop; otherwise go to step 8.
8. Check if maximum number of generations have been completed. If yes, stop; otherwise go to step 3.

The flowchart illustrating the sequence of operations is showed in Figure. 3.

4.3. Simplified Threshold Accepting

While applying the TA algorithm, we realized that the entire TA algorithm need not be employed and only some elements of it are necessary to solve the problems at hand. Hence, we devised, what we call 'simplified threshold accepting' (STA) in this paper. The idea of STA is very simple. Since, in this case, the objective function takes only 5 distinct integer values (0,1,2,3 and 4), the acceptance scheme of the neighborhood solutions is modified here and accordingly, by totally removing the concept of threshold parameter used in traditional TA. Further, the convergence criterion is tweaked so that the algorithm stops when the objective function value becomes equal to zero. These changes in TA were necessitated because of the peculiarities of the problem explained above.

To apply STA in solving our problem, a vector 'V' consisting of 64 parameters V=g0 g1 g2 . . . g63, is considered, where g0, g1, g2, … g63 correspond to the pixels in 8 * 8 block of host image (Shih and Wu, 2005). Here each g is a real value between 0 and 1.

Objective (Fitness) Function

The evaluation (objective) function is difference between the embedded and extracted watermarks.

Evaluation fn

$$= \sum_{i=0}^{all\ pixels} \left| watermark^{em} - watermark^{ex} \right|$$

Because STA works with real values, we use the following two rules to convert real numbers(r) into integers(r*)

(1) If the signal is "≥ 0.5" ,r * = Trunc(r)+1

(2) If the signal is "< 0.5", r* = Trunc(r)

where Trunc(r) is the integer part of r.

Algorithm

1. Specify the maximum number of global iterations, maximum number of inner iterations.
2. Start the global iteration.
3. Randomly generate the initial vector (old vector) following uniform random number generator between (0,1).
4. Compute the objective function and store it in f_i.
5. Start the inner iteration.
6. Generate the candidate solution vector (new vector) in the neighborhood of the old solution vector according to the scheme given as follows:

$$y^c = y^o + (2*u-1)^p; i = 1,2,...,n$$

where u is a random number drawn from Uniform distribution in the range (0,1), p is a pre - specified odd integer and the superscript c and o respectively indicate the candidate solution and the old solution

7. Compute the objective function value of the candidate vector and store it in f_j.
8. If f_i=0 return the old vector and stop.
9. Else, If f_j =0 update the old vector with new vector, f_i value is updated with f_j value, return the old vector and stop.
10. Else, If $f_j \le f_i$ update the old vector with new vector, f_i value is updated with f_j value , inner iteration value is incremented and go to step 6 if inner iterations value is less than the maximum number of inner iterations, else increment the global iterations go to step 3 until global iterations value reaches maximum number of global iterations.
11. Else, If $f_j > f_i$ inner iteration value is incremented and go to step 6 if inner iterations value is less than the maximum number of inner iterations otherwise increment the global iterations go to step 3 until global iterations value reaches maximum number of global iterations.

The Flowchart illustrating the sequence of operations is showed in Figure. (4).

5. Results & Discussions

We tested the effectiveness of GA, DE and STA on a variety of benchmark images Viz., lena256, lena512, baboon512, Barbara512, pepper512. The watermark images used are rose128 and the Chinese character used in (Shih and Wu, 2005). GA was implemented using JDEAL (http://laseeb.isr.ist.utl.pt/sw/JDEAL), DE and STA were coded in java. The computational experiments were conducted on a Pentium IV 1500 MHz system with 256MB RAM. For instance, Figure 5 shows the extracted watermarks after applying evolutionary algorithms to correct rounding errors. Figure 5A is the original host image, an 8 X 8 gray-level image, in the spatial domain and Figure 5B is the transformed image of Figure 5A by DCT. Figure 5C is a binary watermark in which "0" and "1" represents the embedded data in its location ; the minus sign "-" indicates no change in its position. We obtain Figure 5D by embedding Figure 5C into Figure 5B based on LSB modification. We can find three differences when comparing Figures 5B and 5D; for instance (8.646 and 9.646), (6.199 and 7.199) and (-0.765 and 0.235). After transforming Figure 5D into its spatial domain b y IDCT, we obtain Figure. 5E where all pixels are real numbers. Figure 5F is the vector obtained from EAs. After translating real numbers into integers by EAs, we obtain Figure 5G. Figure 5H is the transformed image from Figure. 5G by DCT. Finally we obtain the exact embedded watermark by extracting the bits from the same position in which we embedded the watermark, as shown in Figure. 5I.

165	163	161	161	162	161	159	158
164	162	160	160	161	160	159	157
162	160	158	158	159	159	157	156
160	158	157	157	158	158	156	155
158	157	155	156	157	157	156	155
158	156	155	156	157	158	156	155
158	156	155	156	158	158	157	156
158	156	155	156	158	159	158	156

Figure 5A. Original Host Image

1264.125	6.729	-0.765	9.158	0.145	0.803	-1.070	1.152
18.538	6.199	0.223	-0.366	0.197	-0.142	-0.184	0.171
8.646	-0.175	-0.020	0.568	0.063	0.156	-0.219	0.438
1.486	-0.006	-0.233	-0.351	0.328	-0.149	-0.081	-0.248
-1.595	-0.171	-0.163	-0.156	-0.125	-0.100	-0.067	-0.037
1.659	0.248	-0.189	-0.071	0.370	-0.249	0.580	0.106
-2.106	0.055	0.036	0.288	-0.160	-0.035	0.520	-0.459
1.195	-0.264	0.405	-0.173	0.163	-0.369	-0.439	-0.080

Figure 5B. Transformed image of 3A

-	-	1	1	-	-	-	-
-	1	-	-	-	-	-	-
1	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-

Figure 5C. Binary Watermark

1264.125	6.729	0.235	9.158	0.145	0.803	-1.070	1.152
18.538	7.199	0.223	-0.366	0.197	-0.412	-0.184	0.171
9.646	-0.175	-0.020	0.568	0.063	0.156	-0.219	0.438
1.486	-0.006	-0.233	-0.351	0.329	-0.149	-0.081	-0.247
-1.595	-0.171	-0.163	-0.156	-0.125	-0.100	-0.067	-0.037
1.659	0.248	-0.189	-0.072	0.370	-0.249	0.580	0.106
-2.106	0.055	0.036	0.288	-0.160	-0.035	0.520	-0.459
1.195	-0.264	0.405	-0.173	0.163	-0.369	-0.439	-0.080

Figure 5D. Transformed Image after embedding watermark

167.137	164.952	162.853	162.500	163.657	162.286	160.959	158.760
165.569	163.392	161.302	160.964	162.132	160.779	160.456	157.277
162.788	160.619	158.548	158.235	159.432	159.105	157.800	155.630
159.845	157.693	156.647	156.367	157.595	157.301	156.018	153.874
157.691	156.550	154.529	155.287	156.556	156.300	156.043	153.907
157.163	155.041	154.047	154.835	156.136	156.907	155.679	153.564
158.246	156.124	155.148	155.963	158.299	158.102	157.889	155.766
156.629	154.533	153.570	154.394	156.728	157.534	157.338	154.252

Figure 5E. Watermarked Image after IDCT Transform

0	0	1	1	0	1	0	1
1	0	1	0	1	1	0	0
0	1	1	1	0	0	1	1
0	1	0	1	1	1	1	0
1	0	0	1	0	1	0	1
1	0	1	1	1	1	0	1
0	1	1	1	1	0	0	1
0	0	0	0	1	0	1	0

Figure 5F EA. vector to reduce rounding errors

167	164	163	163	163	163	160	159
166	163	162	160	163	161	160	157
162	161	159	159	159	159	158	156
159	158	156	157	158	158	157	153
158	156	154	156	156	157	156	154
158	155	155	155	157	157	155	154
158	157	156	156	159	158	157	156
156	154	153	154	157	157	158	154

Figure 5G. Translation of real numbers into integers by EAs

1264.5	7.456	-1.677	9.992	0.728	1.682	-1.632	2.282
27.212	7.051	0.063	-0.267	-0.319	0.028	0.630	-0.377
11.283	-0.072	0.193	0.273	0.320	-0.351	0.466	1.255
4.045	0.753	-0.068	-0.211	1.290	-0.316	1.662	-0.531
-4.946	-1.357	-0.285	0.664	-0.246	1.146	-0.677	-1.130
4.289	0.367	-0.833	-0.731	-0.080	-0.637	0.089	-0.666
-4.507	-0.231	0.227	-0.461	0.137	0.540	0.561	-0.823
3.164	-0.326	0.760	-1.077	0.274	-0.089	-0.155	-1.167

Figure 5H. Transformed Image of 3G by DCT

-	-	1	1	-	-	-	-
-	1	-	-	-	-	-	-
1	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-

Figure 5I. extracted watermark after applying EA

For instance, Figure 6 is the example that shows the embedding of watermark into a real image. Figure 6A is the original host image, the Lena image with size 256 x 256. Figure. 6B is original binary watermark. We embed Figure 6B into the coefficients of the transformed image of Figure 6A by DCT. Figure 6C is the rounded image after app lying IDCT. Figure 6D is the extracted watermark from Figure 6C. Figure. 6E, 6F, 6G are the images after applying GA, DE, STA algorithms respectively. Figure. 6H, 6I, 6J are the extracted watermarks from 6E, 6F, 6G respectively.

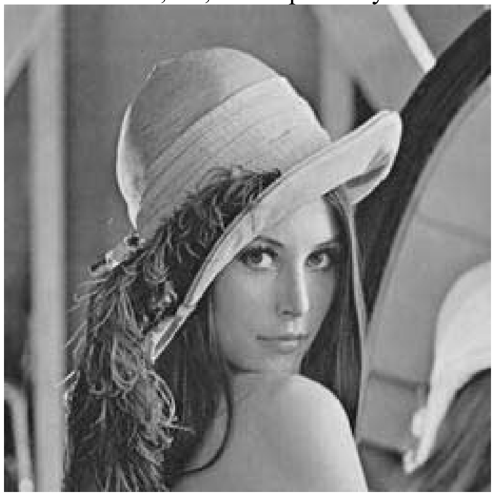
Figure 6A. Original lena 256 x 256 image

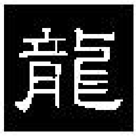
Figure 6B. original binary watermark

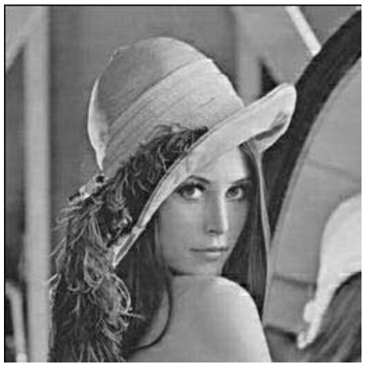
Figure 6C. Rounded IDCT image

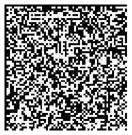
Figure 6D. extracted watermark from rounded IDCT image

Figure 6E. GA image

Figure 6F. DE image

Figure 6G. TA image

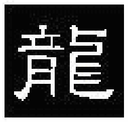
Figure 6H. GA watermark

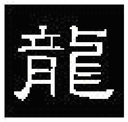
Figure 6I. DE watermark

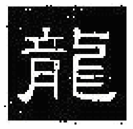
Figure 6J. TA watermark

Table.1 represents the comparisons when translating real numbers in integers by round and EAs. The iteration values are the average of 10 sample values. Each algorithm is applied 10 times iteratively and the average of these 10 values are mentioned in the Table 1. Each time STA takes lesser iterations than DE, which in turn takes lesser iterations than GA.

The definition of PSNR is

$$PSNR = 20.\log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right)$$

Where MAX_I is the maximum pixel value of the image. For gray-scale images this value is 255.

MSE is the Mean Square Error, it is defined as

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1} \| I(i,j) - K(i,j) \|^2$$

Where m and n are the width and height of the image respectively. "I" is the original host image and "K" is the watermarked image.

Table 1. The comparisons when translating real numbers in integers by round and EAs

	By Round	By GA	By DE	By TA
PSNR	36.760	36.678	36.680	36.683
NC	0.466	0.733	0.733	0.723
iterations	-----------	1739	1456	855

The error measure NC (Normalized Correlation) is defined as follows:

$$NC = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} w(i,j) * w^f(i,j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} [w(i,j)]^2}$$

Where *w(i,j)* is the origin al watermark with size N * N and $w^f(i,j)$ is the extracted watermark.

Table. 2 represents the results obtained b y applying GA, DE, STA on different images. From this table it is clearly observed that TA takes lesser iterations than DE, which in turn takes lesser iterations than GA. Although there is slight reduction in STA based extracted watermark NC value it converges quickly to the optimal solution. The iterations (Iter) value is the average of the 10 experiments conducted for each algorithm. In each experiment iterations value of STA is lesser than DE, which in turn lesser than GA.

Table 2. Comparison of GA, DE, TA on different images

Image	GA			DE			TA		
	PSNR	NC	Iter	PSNR	NC	Iter	PSNR	NC	Iter
Lena 256x256	36.678	0.733	1739	36.680	0.733	1456	36.683	0.723	855
Baboon 512x512	31.944	0.728	7602	31.946	0.728	6566	31.944	0.721	3866
Barbara 512x512	37.078	0.745	9751	37.079	0.745	8628	37.077	0.728	5695
Lena 512 x 512	38.123	0.722	10726	38.123	0.722	9614	38.188	0.703	6113
Pepper 512 x 512	34.048	0.742	11739	34.048	0.742	10063	34.042	0.726	5358

For instance, Figure. 7 shows the experiments conducted on a baboon 512 x 512 image. Figure 7A is the original host image, baboon 512 x 512. F ig 7B is a binary watermark. Figure 7C, 7D are the rounded IDCT image and corresponding extracted watermark respectively. Figure 7E, 7F, 7G are the GA, DE and STA based watermarked images respectively. Figure 7H, 7I, 7J are the GA, DE and STA based extracted watermarks respectively.

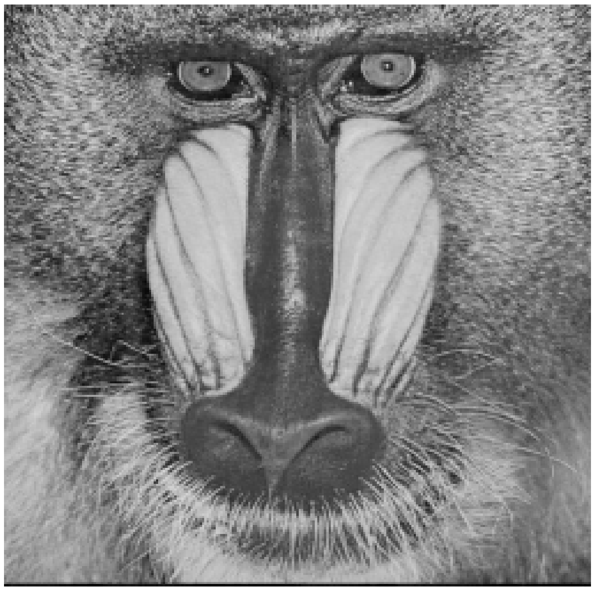
Figure 7A. Original baboon 512x512 image

128x128 binary

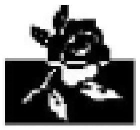

Figure 7B. Original Watermark

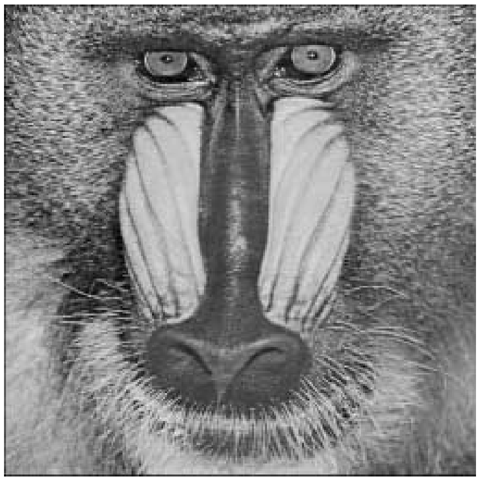
Figure 7C. Rounded baboon

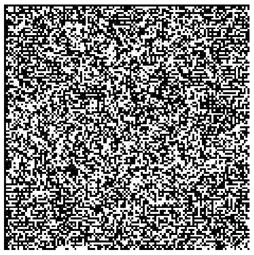

Figure 7D. extracted watermark from 9C

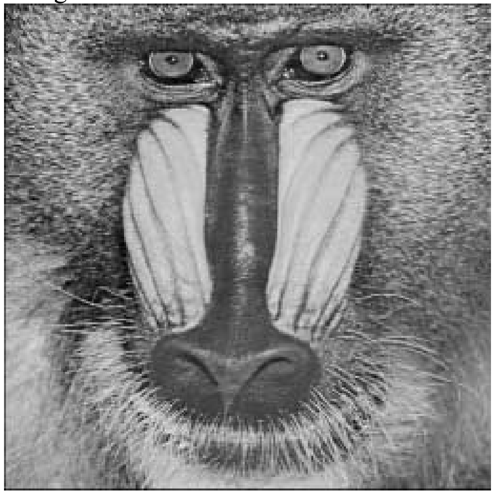

Figure 7E. Watermarked image after applying GA

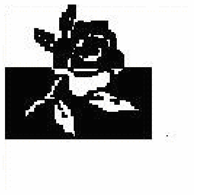

Figure 7H. extracted watermark From 9E

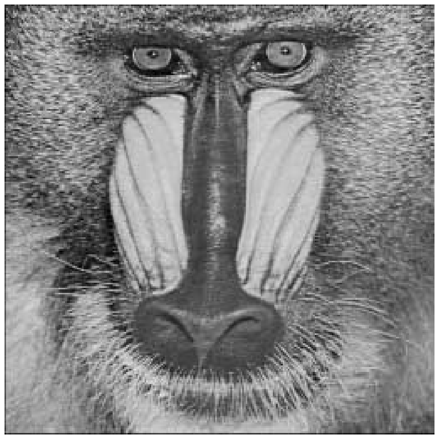

Figure 7F. Watermarked image after applying DE

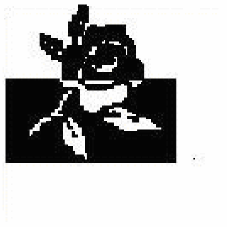

Figure 7I. extracted watermark from 9F

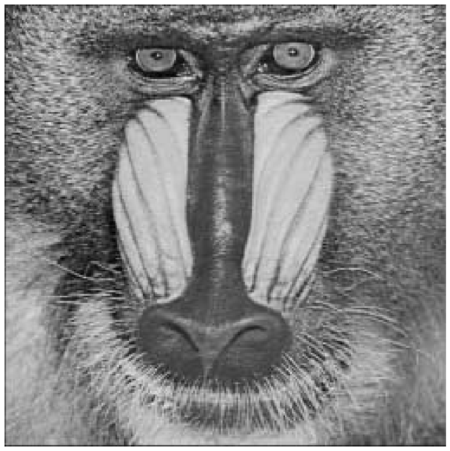

Figure 7G. Watermark ed image after applying TA

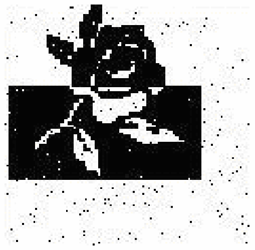

Figure 7H. extracted watermark from 9G

6. Conclusions

In this paper, we present the progression of correcting rounding errors based on EAs. The main issue here is the embedded data should be extracted correctly. To enhance the watermark retrieval we applied GA, DE and STA algorithms on the watermarked image. Our STA algorithm converges quicker to the optimal solution than DE, which inturn converges quicker than GA. We implemented the GA algorithm (Shih and Wu, 2005) and compared the results with DE and STA. We can observe from the previous section that STA takes half of the iterations consumed by GA algorithm. We can also observe that in all benchmark images DE consumes lesser iterations than GA. The extracted watermarks in the previous section shows that they are of high quality than those in (Shih and Wu, 2005).

7. References

[1] Bas, P., Chassery, J. -M, Macq, B, 2002. Image watermarking: an evolution to content based approaches. Pattern Recognit. 35, 545-561.

[2] Bhat, T. R., Venk ataramani, D., Ravi, V., Murty, C. V. S., 2006. An improved differential evolution method for efficient parameter estimation in biofilter modeling. Biochemical Engineering Journal 28, 167-176.

[3] Bruyndonckx, O., Quisquater, J. -J, Macq, B, 1995. Spatial method for copyright labeling o f digital images. In: P roc. IEEE Workshop on Nonlinear Signal and Image Processing, Neos Marmaras, Geece, pp. 456-459.

[4] Celik, M. U., Sharma, G., Saber, E., Tekalp, A. M., 2002. Hierarchial watermark ing for secure image authentication with localization. IEEE Trans. Image Process. 11(6), 585-595.

[5] Cox, I. J., Kilian, J., Leigh to n, F. T., Shamoon, T, 1997. Secure sp read spectrum watermarking for multimedia. IEEE Trans. Image Process. 6(12), 1673-1687.

[6] Dueck, G., Scheuer, T., 1990. Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. Journal of Computational Physics 90, 161-175.

[7] Herrera, F., Lozano, M., Verd egay, J. L, 1994. Applying genetic algorithms in fuzzy optimization problems. Fuzzy Syst. Artif. Intell. 3(1), 39-52.

[8] Holland, J. H., 1975. Adaptation in natural and artificial systems. The University of Michigan Press. Huang, J., Shi, Y. Q., Shi, Y, 2000. Embedding image watermarks in DC components. IEEE Trans. Circuits S yst. Video Technol. 10(6), 974-979.

[9] Karboga, D., Okdem, S., 2004. A Simple and Glob al Optimization Algorithm for engineering Problems : Differential Evolution Algorithm. Turk J Elec Engin, VOL.12, NO. 1, 53-60.

[10] Lin, S. D., Chen, C. –f, 2000. A robust DCT-based watermarking for copyright protection . IEEE Trans. Consumer Electron . 46(3), 415-421.

[11] Nikolaidis, N. , Pitas, I, 1998. Robust image watermark ing in the spatial domain. Sign al Process. 66(3), 385-403.

[12] Osman, H., Kelly, J. P., 1996. Meta-heuristics: An overview, in: I.H. Osman , J.P. Kelly Eds.), Meta-Heuristics: Theory and Applications. Kluwer Academic Publishers, Boston, MA, pp. 1-21.

[13] Price, K., Storn , R., 1997. Differential evolution. Dr. Dobb’s J., 18-24.

[14] Ravi, V., Ma Bin, Ravi Kumar, P., 2 006. Threshold Accepting Based Fuzzy Clustering. Algorithms. International Journal of Uncertainty, Fuzziness and knowledge-Based Systems, Vol. 14, No.5, 617-632.

[15] Ravi, V., Murty, B. S. N, Dutt, N. V. K, 2005. Forecasting the properties of carboxylic acid s by a Threshold accepting-train ed neural network. Indian Chemical Engineer, 47(3), pp. 147-51.

[16] Ravi, V., Reddy, P. J., Zimmermann, H.. J., 2000. Pattern classification with principal component analysis and fuzzy rules bases, EUROPEAN JOURNAL OF OPERATIONAL RESEARCH, 126, 3, 526-533.

[17] Ravi, V., Redd y, P. J., Zimmermann , H.. J., 2000a.. Fuzzy Global Optimization of Complex system Reliability. I EEE TRANSACTIONS ON FUZZY SYSTEMS, 8, 3, 241-248.

[18] Ravi, V., Redd y, P. J., Zimmermann, H.. J., 2001. Fuzzy rule base generation for classification and its Optimization via Modified Threshold Accepting. FUZZY SETS AND SYSTEMS, 120, 2, 271-279.

[19] Ravi, V., Zimmermann, H. J., 2000. Fuzzy rule based classification with Feature Selector and Modified Threshold Accepting. EUROPEAN JOURNAL OF OPERATIONAL RESEARCH, 123, 1, 16-28.

[20] Ravi, V., Zimmermann, H. J., 2001. A neural network and fuzzy rule base hybrid for Pattern classification. Soft Computing 5(2): 152-159.

[21] Ravi, V., Zimmermann, H. J., 2003. Optimization of Neural Networks via Threshold Accepting: A Comparison with Back propagation Algorithm. Conference on Neuro-Computing and Evolving Intelligence, Auckland, New Zealand. Shih, F. Y., Wu, Yi, -Ta, 2003. Combinational image watermarking in th e spatial and frequency domains. Pattern Recognition 36, 969-975.

[22] Shih, F. Y., Wu, Yi, -Ta, 2005. Enhancement of image watermark retrieval based on genetic algorithms. Journal of Visual Communications & Image Representation. 16, 115-133.

[23] Wong, P. W., 1998. A public key watermark for image verification and authentication. In :P roc. IEEE Int. Con f. Image Processing, Chicago, IL, pp. 425-429.

Venkata Gopal Edupuganti is a PhD student in the Department of Computer Science at New Jersey Institute of Technology, USA. He received Master of Technology in Information Technology from University of Hyderabad, India in 2007. His current research interests include object recognition in images and videos, pattern recognition, machine learning, and digital watermarking.

M. V. N. K. Prasad is an assistant Professor, Institute for Development and Research in Banking Technology, Hyderabad. He received B.E., M.S., Ph.D degrees, in Computer Engineering. His main research interests are Digital watermarking, Biometrics and Payment System Technologies.

Vadlamani Ravi is an Associate Professor in IDRBT, Hyderabad since April 2005. He holds a Ph.D. in Soft Computing from Osmania University, Hyderabad & RWTH Aachen Germany. Earlier, he was a Faculty at NUS, Singapore for three years. He published 84 papers in refereed Journals / Conferences and invited book chapters. He edited *"Advances in Banking Technology and Management: Impacts of ICT and CRM"* published by IGI Global, USA. He is a referee for several international journals and on the Editorial board of IJIDS, IJDATS, IJISSS, IJSDS & IJITPM. He is listed in the Marquis Who's Who in the World in 2009.