

Automatic Circle Detection on Images with Annealed Differential Evolution

Swagatam Das, Sambarta Dasgupta, Arijit Biswas and Ajith Abraham*

*Department of Electronics and Telecommunication Engineering,
Jadavpur University, Kolkata, India*

**Center for Quantifiable Quality of Service, Norwegian University of Science and Technology, Norway
ajith.abraham@ieee.org*

Abstract

This article presents an algorithm for the automatic detection of circular shapes from complicated and noisy images. The algorithm is based on a hybrid technique composed of simulated annealing and differential evolution. A new fuzzy objective function has been derived for the edge map of a given image. Minimization of this function with a hybrid annealed differential evolution algorithm leads to the automatic detection of circles on the image. Simulation results over several synthetic as well as natural images with varying range of complexity validate the efficacy of the proposed technique in terms of its final accuracy, speed and robustness.

Keywords

Object recognition, simulated annealing, differential evolution computer vision, hand drawn shape location.

1. Introduction

Circle and ellipse detection from digital images have received considerable attention over the last few decades in computer vision [1]. Until date the most popular approaches for circle detection have been based on Hough transform (HT) [2, 3]. Let (x, y) be the location of an edge pixel on a circle with center coordinates (a, b) and radius r then the circle can be expressed as:

$$(x-a)^2 + (y-b)^2 = r^2$$

From this equation, every edge pixel of the image can be mapped into a conical surface in a three-dimensional (a, b, r) parameter space and this leads to the conventional HT [4]. In order to overcome limitations of HT researchers have proposed new approaches to HT e.g., the probabilistic HT [5], the randomized HT [6], and the fuzzy HT [7]. Lam and Yuen proposed an approach based on hypothesis filtering and HT to detect circles [8].

As an alternative to the HT based techniques, the shape recognition problem in computer vision has also been handled with stochastic search methods that include

random sample consensus [9], simulated annealing [10] and Genetic Algorithm (GA) [11]. In particular, GA has recently been used for important shape detection task e.g. Roth and Levine proposed use of GA for primitive extraction of images [11]. Lutton et al carried out a further improvement of the aforementioned method recently [12]. Yao *et al* came up with a multi-population GA to detect ellipses [13]. Ayala-Ramirez *et al* presented a GA based circle detector [14]. Their approach is capable of detecting multiple circles on real images but fails frequently to detect small and imperfect circles.

Differential Evolution (DE), proposed by Storn and Price [15], is a simple yet powerful population-based stochastic search technique over the continuous search space. DE has been successfully applied in diverse domains of science and engineering problems, such as mechanical engineering design [16],[17], data communication [18], chemical engineering [19], machine intelligence and pattern recognition [20]. In many cases, it has reportedly outperformed the GA or the particle swarm optimization (PSO) [21].

In this paper, the selection mechanism of the classical DE has been modified by employing the concepts of the *Simulated Annealing* (SA) algorithm. SA belongs to a class of heuristic search algorithms called *probabilistic hill-climbing* [22], which dynamically alter the probability of accepting the inferior solutions. The hybrid algorithm developed in the scope of this work is referred to as the Annealed Differential Evolution (AnDE). Besides using an SA based selection scheme, the AnDE introduces a center of mass based mutation strategy, in which the trial vectors are stochastically attracted towards the mean vector of the current population, instead of the best one as done in the case of DE/current-to-best/bin and DE/best/2 schemes. The efficacy of AnDE over a few other popular DE variants have been established through benchmark simulations in [23].

The remainder of this paper is organized as follows. Section 2 provides a brief outline of the classical DE family of algorithms and also reviews the previous works for the improvement of the performance of DE. Section 3 briefly discusses the spirit of the SA algorithm and then

introduces the proposed hybrid algorithm.. Section 4 describes candidate solution representation and also lays out the mathematical basis of the objective function. Results of computer simulation over several images have been presented in Section 5 and finally the paper is concluded in Section 6 with a discussion of future research directions.

2. The DE and its variants – an overview

DE starts with a population of N, D-dimensional search variable vectors or *chromosomes* in the terminology of evolutionary computing. The i-th vector of the population at iteration (time) t is:

$$\vec{X}_i(t)=[x_{i,1}(t),x_{i,2}(t),\dots,x_{i,D}(t)] \quad (1)$$

In each iteration of the algorithm, for each population member $\vec{X}_i(t)$, a *donor* or *trial* vector $\vec{V}_i(t+1)$ is created.

It is the method of creating this donor vector that distinguishes various DE schemes from one another. In “scheme DE1” (DE/rand/1), to create $\vec{V}_i(t+1)$, three vectors (say r_1 , r_2 , and r_3) are randomly chosen from the current population. Next the difference of any two of these three vectors is scaled by a scalar F and the scaled difference is added to the third vector whence we obtain the donor vector:

$$v_{i,j}(t+1) = x_{r_1,j}(t) + F.(x_{r_2,j}(t) - x_{r_3,j}(t)) \quad (2)$$

The DE family of algorithms use two kinds of crossover, namely ‘exponential’ and ‘binary’.

‘Binary’ crossover is implemented as follows:

$$u_{i,j}(t+1) = \begin{cases} v_{i,j}(t+1) & \text{if rand}(0, 1) < Cr \\ x_{i,j}(t) & \text{otherwise} \end{cases} \quad (3)$$

DE actually involves the Darwinian principle of “Survival of the fittest” in its selection process which may be outlined as,

$$\vec{X}_i(t+1) = \begin{cases} \vec{U}_i(t+1) & \text{if } f(\vec{U}_i(t+1)) < f(\vec{X}_i(t)) \\ \vec{X}_i(t) & \text{if } f(\vec{X}_i(t)) < f(\vec{U}_i(t+1)) \end{cases} \quad (4)$$

where f () is the function to be minimized. So if the new offspring yields a better value of the fitness function, it replaces its parent in the next generation; otherwise the parent is retained in the population. DE/current to best/1 follows the same procedure except that the donor vector is created using two randomly selected members of the population as well as the best vector of the current time step:

$$\vec{V}_i(t+1) = \vec{X}_i(t) + \lambda.(\vec{X}_{best}(t) - \vec{X}_i(t)) + \beta.(\vec{X}_{r_2}(t) - \vec{X}_{r_3}(t)) \quad (5)$$

where λ is another control parameter of DE in [0, 2]. To reduce the number of control parameters a usual choice is to put $\lambda = F$. Storn and Price [15] suggested a total of ten different working strategies for DE.

3. The Annealed DE algorithm

As its name implies, the Simulated Annealing (SA) exploits an analogy between the way in which a metal cools and freezes into a minimum energy crystalline structure (the annealing process) and the search for a minimum in a more general system. The algorithm is based upon that of Metropolis *et al.* [24], which was originally proposed as a means of finding the equilibrium configuration of a collection of atoms at a given temperature. It was Kirkpatrick *et al.* [25] who proposed it as the basis of an optimization technique for combinatorial (and other) problems.

The algorithm employs a random search which not only accepts better solutions that decrease the objective function f (in case of a minimization problem), but also some inferior solutions that increase it. The latter are accepted with a probability:

$$P = \exp\left(-\frac{\Delta f}{T}\right) \quad (6)$$

Where Δf is the change of the objective function over two successive iterations and T is a control parameter, known as the system ‘temperature’.

Unlike the *greedy* selection strategy employed in the classical DE, the AnDE algorithm proposed here incorporates a typical SA type selection mechanism that conditionally accepts the inferior solutions to the next generation. Suppose at time step $t = t$, the i-th chromosome was $\vec{X}_i(t)$ and its offspring created through the DE type mutation and crossover operations at the next time step be $\vec{U}_i(t+1)$. Now if $f(\vec{U}_i(t+1)) < f(\vec{X}_i(t))$, i.e. the offspring is better than the parent w.r.t the objective function, then $\vec{X}_i(t)$ is surely replaced in the next generation by $\vec{U}_i(t+1)$. But even if $f(\vec{U}_i(t+1)) > f(\vec{X}_i(t))$, $\vec{U}_i(t+1)$ may replace the parent chromosome $\vec{X}_i(t)$ with a probability of:

$$P_i = \exp\left[-\frac{f(\vec{U}_i(t+1)) - f(\vec{X}_i(t))}{T}\right] \quad (7)$$

Which is same in spirit as (5).

The SA algorithms require an annealing schedule for decreasing the control temperature T from an initial value T_0 to a final value T_f . The AnDE algorithm, however, employs a very common temperature decrement rule known as the exponential cooling schedule (ECS), proposed by Kirkpatrick *et al.* [25]. According to this

rule, the temperatures $T(t)$ and $T(t+1)$ over two successive time steps are related as:

$$T(t+1) = \alpha T(t) \quad (8)$$

Where α is constant close to, but smaller than 1.

The AnDE also modifies the basic mutation scheme of the DE/best/1 given by (5) by replacing the best vector $\vec{X}_{best}(t)$ by the mean of all the vectors belonging to the current generation. This mean vector can be regarded as the center of mass of the DE population (especially if we assume that the chromosomes are agents of equal mass in the multi-dimensional search space) and is given by:

$$\vec{X}_{CM} = \frac{1}{NP} \sum_{i=1}^{NP} \vec{X}_i(t) \quad (9)$$

Where NP is the total number of search variable vectors in the population. With this modification, the mutation scheme for trial vector generation in AnDE becomes:

$$\begin{aligned} \vec{V}_i(t+1) &= \vec{X}_i(t) + F \cdot (\vec{X}_{CM}(t) - \vec{X}_i(t)) \\ &+ F \cdot (\vec{X}_{r_2}(t) - \vec{X}_{r_3}(t)) \end{aligned} \quad (10)$$

The time-variation of Cr may be expressed in the form of the following equation,

$$Cr = (Cr_{max} - Cr_{min}) \cdot \frac{(T_{max} - t)}{t} \quad (11)$$

where Cr_{max} and Cr_{min} are the maximum and minimum values of Crossover rate Cr, t is the current time step and T_{max} is the maximum time steps (i.e. the maximum number of iterations allowed). The algorithm encapsulates all these changes and pseudo-code of algorithm is available in [23].

4. The AnDE Based Fuzzy Circle Detection Algorithm

4.1 The Scheme for Population Initialization

An individual member of the population of sample circle is actually a trial solution. Each sample circle is represented as a trial solution position $\vec{X} = [x_0, y_0, r_0]^T$, where first two components of the vector are x and y coordinates of the center of that circle and third term stands for the radius.

Let (x_p, y_p, r_p) be the p -th test circle in the population, where $p = 1(1)NP$, NP is the population size i.e. it denotes total number of test circles. In initialization suitable value is assigned to each of the three entries of the vectors. Two statistical properties of sample points on a circle are used

in this regard, where one property is used to find a suitable value for co-ordinates of probable center of the test circle and, another is used to model its probable radius. These two properties are described below. Let us consider $2N$ number of equally spaced sample points on a circle of radius r_0 and center at (x_0, y_0) . Now, i -th sample point may be designated by $(x(i), y(i))$.

Where, $x(i) = x_0 + r_0 \cos \frac{\pi}{N} i$

and $y(i) = y_0 + r_0 \sin \frac{\pi}{N} i, i = 1(1)2N$

Let mean-point (\bar{X}, \bar{Y}) be defined as,

$$\bar{X} = \frac{1}{2N} \sum_{i=1}^{2N} x(i), \bar{Y} = \frac{1}{2N} \sum_{i=1}^{2N} y(i)$$

Now, $\bar{X} = \frac{1}{2N} \sum_{i=1}^{2N} x(i) = \frac{1}{2N} \sum_{i=1}^{2N} (x_0 + r_0 \cos \frac{\pi}{N} i)$

$$\therefore \bar{X} = x_0 + r_0 \frac{\sin \pi}{\sin \frac{\pi}{2N}} \cos \frac{\pi(2n+1)}{4n} \Rightarrow \bar{X} = x_0$$

$$\text{Similarly, } \bar{Y} = y_0 \therefore \bar{X} = x_0, \bar{Y} = y_0 \quad (12)$$

Let us also define standard deviation,

$$\sigma_0 = \sqrt{\frac{1}{2N} \sum_{i=1}^{2N} (x(i) - \bar{X})^2 + (y(i) - \bar{Y})^2} \quad (13)$$

$$\Rightarrow \sigma_0^2 = r_0^2 \frac{1}{2N} \left(\sum_{i=1}^{2N} (\cos \frac{\pi}{N} i)^2 + (\sin \frac{\pi}{N} i)^2 \right)$$

$$\Rightarrow \sigma_0^2 = r_0^2 \quad (14)$$

To initialize k -th ($k = 1(1)NP$) test circle we form a set of $2m_k$ edge-points drawn at random from the edge-map, where $2m_k = rand(L, U)$ such that L, U specify upper and lower limit of $2m_k$. Let $(x(i), y(i))$ be the i -th edge-point, $i = 1(1)2m_k$ and (x_k, y_k, r_k) denotes the k -th test circle to be initialized. Now we assign values to three entries of the vector.

$$\text{We assign, } x_k = \frac{1}{2m_k} \sum_{i=1}^{2m_k} x(i), y_k = \frac{1}{2m_k} \sum_{i=1}^{2m_k} y(i)$$

$$\text{and } r_k = \sqrt{\frac{1}{2m_k} \left(\sum_{i=1}^{2m_k} (x(i) - x_k)^2 + (y(i) - y_k)^2 \right)}$$

This initialization scheme is adopted with the expectation that if the selected edge points may lie on the actual circle or very close to it so that we may reach a good approximation of the center and radius.

4.2 The Fuzzy Objective Function

We are assuming that in the edge-map edge pixels are simply represented by '1' and back-ground pixels by '0'.

Let us consider a test circle be denoted by (x_0, y_0, r_0) . And let A be the edge matrix. We denote objective function be f . Let for the above mentioned test circle its value be f_0 . $\therefore f_0 = f(A, x_0, y_0, r_0)$

We define another function $P(x, y)$, which returns pixel value of a point (x, y) .

$$P(x, y) = 1, \quad \text{if } (x, y) \text{ is an edge-point} \\ = 0, \quad \text{otherwise.}$$

For sampling we consider the family of circles with center at (x_0, y_0) with a radius varying in a range from $r_0 - \delta$ to $r_0 + \delta$. This forms the test-band, δ being a parameter.

After some trial and error we found that taking $\delta = \frac{r_0}{6}$ is

comfortable enough for this specific application. After this, N_s sample points are taken for each circle in test-band. Sample points are positioned on the circle maintaining equal distance between each other. Let i -th sample point on a circle of test-band, having radius $r_0 + j$ ($-\delta \leq j \leq \delta$) be denoted by (x_i^j, y_i^j) .

$$\therefore x_i^j = (r_0 + j) \cos \frac{2\pi}{N_s} i, y_i^j = (r_0 + j) \sin \frac{2\pi}{N_s} i \quad (15) \\ i = 1(1)N_s \text{ and } j = -\delta, -\delta+1, \dots, -1, 0, 1, \dots, \delta-1, \delta.$$

An organized method for dealing with imprecise data is fuzzy logic [26]. It is basically a multi-valued logic that allows more human-like interpretation and reasoning in machines by resolving intermediate categories between notations such as true/false, hot/cold used in Boolean logic. In this work, to deal with real world images containing incomplete and noisy circles, the set of the points belonging to the circumference of the actual circle has been considered as a fuzzy set and we assign a membership function to each point over the fuzzy set. This membership function depends upon distance of sample point from the central circle of the test band (i.e. circle with radius r_0). Let μ be the membership function and its value be μ_i^j for sample point (x_i^j, y_i^j) . We define membership function as,

$$\mu_i^j = P(x_i^j, y_i^j) \exp\left(-\frac{j^2}{2\sigma^2}\right). \quad (16)$$

If (x_i^j, y_i^j) is not an edge-point, we can infer $\mu_i^j = 0$ i.e. a zero membership [$\because P(x_i^j, y_i^j) = 0$]. Let us consider now the case that (x_i^j, y_i^j) is an edge-point. Then,

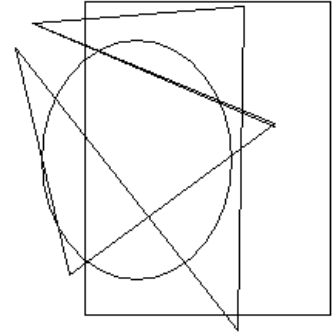
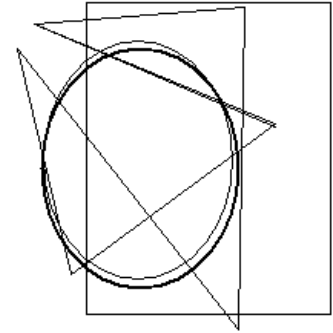
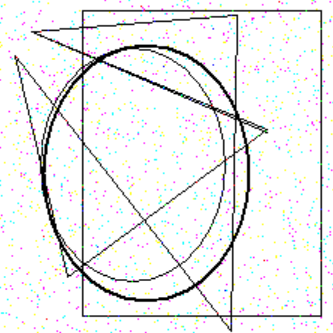
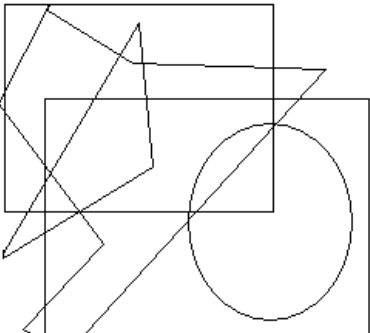
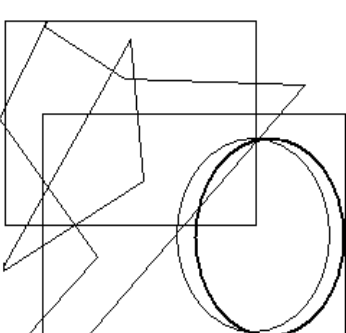
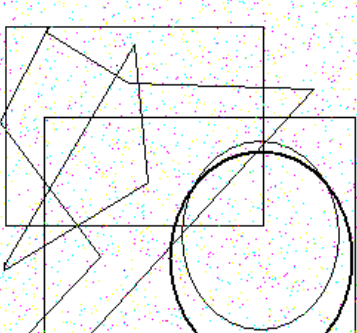
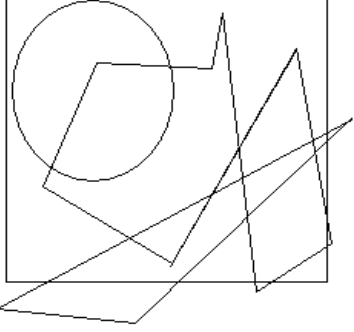
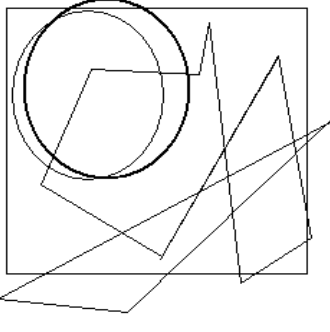
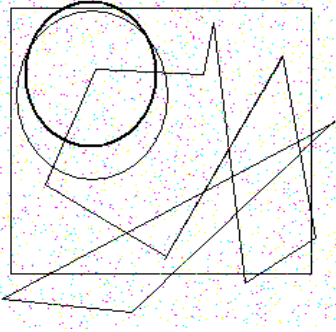
$\mu_i^j = \exp\left(-\frac{j^2}{2\sigma^2}\right)$. Clearly $\mu_i^j = 1$, if $j = 0$. Membership is unity or maximum when sampled edge-point lies on central circle. Membership value decreases when $|j|$ increases. If σ decreases the function becomes more sharp cut-off or narrow-band. Now, normalized objective function corresponding to (x_0, y_0, r_0) for A edge-map is defined as following,

$$f(A, x_0, y_0, r_0) = 1 - \frac{1}{(2\delta+1)N_s} \sum_{i=1}^{N_s} \sum_{j=-\delta}^{\delta} \mu_i^j \\ \Rightarrow f(A, x_0, y_0, r_0) = 1 - \frac{1}{(2\delta+1)N_s} \sum_{i=1}^{N_s} \sum_{j=-\delta}^{\delta} P(x_i^j, y_i^j) \exp\left(-\frac{j^2}{2\sigma^2}\right) \\ \Rightarrow f(A, x_0, y_0, r_0) = 1 - \frac{1}{(2\delta+1)N_s} \sum_{i=1}^{N_s} \sum_{j=-\delta}^{\delta} P(r_0 + j) \cos \frac{2\pi}{N_s} i, (r_0 + j) \sin \frac{2\pi}{N_s} i \exp\left(-\frac{j^2}{2\sigma^2}\right) \quad (17)$$

5. Experimental Results

The test bed includes ten synthetic (handcrafted) gray-scale images each of size 256×256 pixels. The natural images include a circular-shaped object among various other configurations. All the images were preprocessed using a standard edge-detector (Canny edge-detector in image-processing toolbox, MATLAB 7.0). We have reported detailed results for three images in order to save space. In order to test the robustness of our algorithm, we have added salt and pepper noise to the synthetic images before the algorithm was applied. It also illustrates the performance of the algorithm in presence of such noisy and corrupted pixels. Results are shown in Table 1. Detected circle is shown in thick line.

Table 1: Test images with detected circles

Image index	Original image	Circle detected image	Circle detected image with noise
1			
2			
3			

6. Conclusions

This paper has presented a novel application of the AnDE to the task of automatic circle detection from gray images. Also a new fuzzy fitness function was derived specifically for the circle detection task. Future research may focus on hybridizing DE with different Hough Transform based techniques for automatic shape extraction.

Also application of the DE based techniques for automatic shape recognition by mobile robots may also be studied. In future we shall also attempt to compare the performance of our DE based algorithm with other evolutionary computation techniques on the circle detection problem quantitatively.

References

- [1] Davies, E. R.: Machine Vision: Theory, Algorithms, Practicalities, *Academic Press*, London, 1990.
- [2] Illingworth, J. and Kittler, J.: Survey: A survey of the Hough transform, *Comput. Vision, Graphics, Image Process.* 44, 1988, 87–116
- [3] Leavers, V. F.: Survey: Which Hough Transforms, *CVGIP: Image Understanding* 58, 1993, 250–264.
- [4] Duda, R. O. and Hart, P. E.: Use of the Hough transformation to detect lines and curves in pictures, *Comm.Assoc. Comput. Mach.* 15, 1972, 11–15.
- [5] Shaked, D., Yaron, O., Kiryati, N.: Deriving stopping rules for the probabilistic Hough transform by sequential analysis. *Computer Vision Image Understanding* 63, 512–526, 1996.
- [6] Xu, L., Oja, E., and Kultanen, P.: A new curve detection method: Randomized Hough Transform (RHT). *Pattern Recognition Letters* 11(5), 331–338, 1990.
- [7] Han, J.H., Koczy, L.T., Poston, T.: Fuzzy Hough Transform. In *Proc. 2nd Int. Conf. on Fuzzy Systems*, vol. 2, pp. 803–808, 1993.
- [8] Lam, W., Yuen, S.: Efficient techniques for circle detection using hypothesis filtering and Hough transform. *IEEE Proc. Visual Image Signal Process.* 143 (5), 292–300, 1996.
- [9] Fischer, M., Bolles, R., 1981. *Random sample consensus: A paradigm to model fitting with applications to image analysis and automated cartography. CACM* 24 (6), 381–395.
- [10] Bongiovanni, G., and Crescenzi, P.: Parallel Simulated Annealing for Shape Detection, *Computer Vision and Image Understanding*, vol. 61, no. 1, pp. 60–69, 1995.
- [11] Roth, G. and Levine, M. D.: Geometric primitive extraction using a genetic algorithm. *IEEE Trans. Pattern Anal. Machine Intell.* 16 (9), 901–905, 1994.
- [12] Lutton, E., Martinez, P.: A genetic algorithm for the detection 2-D geometric primitives on images. In: *Proc. of the 12th Int. Conf. on Pattern Recognition*, vol. 1, pp. 526–528, 1994.
- [13] Yao, J., Kharma, N., and Grogono, P.: Fast robust GA-based ellipse detection. In: *Proc. 17th Int. Conf. on Pattern Recognition ICPR-04*, vol. 2, Cambridge, UK, pp. 859–862, 2004.
- [14] Ayala-Ramirez, V., Garcia-Capulin, C. H., Perez-Garcia, A., and Sanchez-Yanez, R. E.: Circle detection on images using genetic algorithms, *Pattern Recognition Letters*, 27, 652–657, 2006.
- [15] Storn, R., Price, K.: Differential evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, *Journal of Global Optimization*, 11(4) (1997) 341–359.
- [16] Rogalsky, T., Derksen, R.W., and Kocabiyik, S.: Differential Evolution in Aerodynamic Optimization, 46th Annual Conf of Canadian Aeronautics and Space Institute (1999) 29–36.
- [17] R.Joshi and A.C. Sanderson, Minimal representation multisensor fusion using differential evolution, *IEEE Trans. Systems, Man, and Cybernetics*, Part A, vol 29, no. 1, pp. 63–76, 1999.
- [18] Das, S., Konar, A.: Design of Two dimensional IIR Filters with Modern Search Heuristics: A Comparative Study, *International Journal of Computational Intelligence and Applications*, 2008 (in press).
- [19] Lampinen, J.: A Bibliography of Differential Evolution Algorithm. Technical Report. Lappeenranta University of Technology, Department of Information Technology, Laboratory of Information Processing, 1999. <http://www.lut.fi/~jlampine/debiblio.htm>
- [20] Omran, M., Engelbrecht, A. P., Salman, A.: Differential Evolution Methods for Unsupervised Image Classification, Congress on Evolutionary Computation (CEC-2005), IEEE Press.
- [21] Vesterström, J., Thomson, R.: A Comparative Study of Differential Evolution, Particle Swarm Optimization, and Evolutionary Algorithms on Numerical Benchmark Problems, Congress on Evolutionary Computation (CEC04), IEEE Press.
- [22] Romeo, F. and Sangiovanni-Vincentelli, A.: Probabilistic Hill-Climbing Algorithms: Properties and Applications, Computer Science Press, Chapel Hill. NC, 1985.
- [23] Das, S., Konar, A., and Chakraborty, U. K., Annealed Differential Evolution, Congress in Evolutionary Computation, CEC07, IEEE press.
- [24] Metropolis, N., Rosenbluth, A. W.: Rosenbluth, M., Teller, A. H. and Teller, E.: Equation of State Calculations by Fast Computing Machines. *J. Chem. Phys.* 21, 1087–1092, 1953.
- [25] Kirkpatrick, S., Gelatt, C. and Vecchi, M.: Optimization by Simulated Annealing. *Science*, 220: 671–680, 1983.
- [26] Zadeh, L.A.: *Fuzzy sets, Information and Control* 8 (3): 338–353, 1965.