Progress In Electromagnetics Research B, Vol. 29, 289–309, 2011

AN ADAPTIVE DIFFERENTIAL EVOLUTION ALGORITHM FOR AUTONOMOUS DEPLOYMENT AND LOCALIZATION OF SENSOR NODES

S. Roy, S. Minhazul Islam, S. Ghosh, and S. Das

Dept. of Electronics and Telecommunication Engineering
Jadavpur University, Kolkata 700032, India

A. Abraham

Machine Intelligence Research Labs (MIR Labs)
Auburn, Washington 98071-2259, USA

Abstract—In recent years, Wireless Sensor Networks (WSNs) have transitioned from being objects of academic research interest to a technology that is frequently being employed in real-life applications and rapidly being commercialized. The performance of a WSN is largely affected by high quality deployment and precise localization of sensor nodes. This article deliberates autonomous deployment of sensor nodes from an Unmanned Aerial Vehicle (UAV). This kind of deployment has importance in emergency applications, such as disaster monitoring and battlefield surveillance. The goal is to deploy the nodes only in the terrains of interest, which are distinguished by segmentation of the images captured by a camera on board the UAV. In this article we propose an improved variant of a very powerful real parameter optimizer, called Differential Evolution (DE) for image segmentation and for distributed localization of the deployed nodes. Image segmentation for autonomous deployment and distributed localization are designed as multidimensional optimization problems and are solved by the proposed algorithm. Performance of the proposed algorithm is compared with other prominent adaptive DE-variants like SaDE and JADE as well as a powerful variant of the Particle Swarm optimization (PSO) algorithm, called CLPSO. Simulation results indicate that the proposed algorithm performs image segmentation faster than both types of algorithm for optimal thresholds. Moreover in case of localization it gives more accurate

Received 23 February 2011, Accepted 11 April 2011, Scheduled 13 April 2011
Corresponding author: Swagatam Das (swagatamdas19@yahoo.co.in).

results than the compared algorithms. So by using the proposed variant of Differential Evolution improvement has been achieved both in the case of speed and accuracy.

1. INTRODUCTION

Wireless Sensor Networks (WSNs) are networks of distributed autonomous nodes that can sense or monitor physical or environmental conditions cooperatively [1]. An ad-hoc WSN consists of a number of sensors spread across a geographical area. Each sensor has wireless communication capability and some level of intelligence for signal processing and networking of the data. The development of WSNs was originally motivated by military applications such as battlefield surveillance. However, they are currently being employed in many industrial and civilian application areas including industrial process monitoring and control, machine health monitoring, environment and habitat monitoring, healthcare applications, home automation, and traffic control [2–8]. A few excellent surveys on the present state-of-the-art research on sensor networks can be traced in [9–13]. Usually in case of environment and habitat monitoring the sensor nodes are deployed by a helicopter. But this method of deployment cannot be used in case of monitoring a territory which is dangerous or hostile. Thus the use of autonomous unmanned aerial vehicle is evident [14]. Modern UAVs use control and perception to undergo coordinated deployment missions [15].

For UAVs Computer Vision can be used as a perception for estimation of motion and position, detection and tracking of objects, autonomous takeoff and landing, and in other practical applications, such as detection, monitoring, and terrain mapping. Terrain recognition by means of image segmentation has high relevance in case of autonomous node deployment because this methodology can be used for deploying the nodes only in the terrains of interest, i.e., avoiding water or fire. The loss of sensor nodes can be reduced by such a deployment scheme and so the same sensing coverage can be done by less number of nodes.

Image segmentation is used as a pre-processing technique by most machine vision methods. For segmenting an image into two or more classes usually thresholding technique is applied. Let $f(x, y)$ denote a grayscale image of size $H \times W$ pixels that has L intensity levels. For two level thresholding a threshold value $t$is to be found which performs the operation expressed by (1) for $x = 1, 2, \ldots, H$ and $y = 1, 2, \ldots, W$.

$$F(x, y) = \begin{cases} 0 & \text{if} \quad f(x, y) \leq t \\ L & \text{if} \quad f(x, y) > t \end{cases} \tag{1}$$

The above concept can be extended to three-level thresholding, in which there exist two threshold levels t_1 and t_2 such that $t_1 < t_2$, and the thresholding operation is performed, as expressed in the following equation:

$$F(x,y) = \begin{cases} 0 & \text{if} \quad f(x,y) \le t_1 \\ \frac{1}{2}(t_1 + t_2) & \text{if} \quad t_1 < f(x,y) \le t_2 \\ L & \text{if} \quad f(x,y) > t_2 \end{cases} \tag{2}$$

This can be further extended to generic n-level thresholding in which $n-1$ threshold levels $t_1, t_2, \ldots, t_{n-1}$ are necessary. It is obvious from Equations (1) and (2) that the effectiveness of multilevel segmentation largely depends on the values of threshold levels $t_1, t_2, \ldots, t_{n-1}$. Many methods for obtaining this threshold values are reported, a survey of which is presented in [16]. The finding of this threshold values can be formulated as an optimization problem.

Location is very important in WSNs for monitoring and tracking applications because Location information of the nodes can be utilized for detecting and recording events or for routing packets by means of geometric aware routing [17, 18] and sometimes the data is the location itself that is to be sensed [19]. Providing each node with a GPS is not an efficient solution because of cost, size, and energy constraints associated with it. Node localization, which is used to locate the positions of all deployed sensors, has emerged as an area of active research. Majority of the localization algorithms available so far possess a common characteristic that they evaluate the locations of the deployed nodes utilizing a priori knowledge of the coordinates of a particular type of nodes termed as beacons, landmarks, or anchors [20].

A WSN contains N nodes, each characterized by a communication range of r, dispersed in a 2-D mission field. The WSN can be formulated as the Euclidean graph $G = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ denotes the collection of sensor nodes. $\langle i, j \rangle \in E$ if the separation between v_i and v_j is $d_{ij} < r$. Unknown nodes (also called free or dumb nodes) are the set U of nonbeacon nodes. Nonbeacon nodes are unaware of their localization information. Settled nodes are the set S of nodes that have been able to locate their positions by means of the localization algorithm. Given a WSN $G = (V, E)$, and a set of beacon nodes B and their positions (x_b, y_b) for all $b \in B$, we need to find the position coordinates (x_u, y_u) of as many $u \in U$ as possible, converting the unknown nodes into settled nodes S.

WSN localization is basically a two-phase phenomenon. In the first phase nodes evaluate their distances from beacons (or settled nodes) by means of the information regarding signal propagation time or the received signal strength. The signal propagation time is calculated through the estimation of the arrival time, the round trip

time of flight, or the difference in arrival time of the signal. This phase is regarded as the ranging phase. Noise has an adverse effect on the precise measurement of these parameters. So, the localization algorithms that depend on these parameters are likely to provide inaccurate results. In the second phase, position estimation of the target nodes is performed using the information obtained in the first ranging phase. This is done in two methods, either by solving a set of simultaneous equations, or by applying an optimization algorithm that diminishes the localization error. In iterative localization algorithms, the settled nodes perform the function of beacons and the localization process is repeated until either all nodes become a member of the set of settled nodes, or no more nodes are available to be localized. The WSN localization problem has been tackled by several interesting approaches, a survey of which is presented in [21]. An overview of Sensor Localization Systems is presented in [22]. A survey on Localization for mobile wireless sensor networks is portrayed in [23].

The Differential Evolution (DE) [24, 25] algorithm emerged as a very competitive form of evolutionary computing more than a decade ago. Since then, the DE family of algorithms has been frequently adopted to tackle multi-objective, constrained, dynamic, large scale, and multimodal optimization problems and the resulting variants have been achieving top ranks in various competitions held under the IEEE CEC (Congress on Evolutionary Computation) conference series (e.g., see http://www3.ntu.edu.sg/home/epnsugan/index_files/cec-benchmarking.htm). In this paper we propose a new variant of DE, called ADE, where we introduce a new group-based mutation strategy, novel schemes for the adaptation of control parameters scale factor (F) and crossover rate (Cr), and also an exploitative crossover strategy (p-best crossover). We use this algorithm for thresholding of images captured from a downward-pointing camera on board the UAV used for autonomous deployment of WSN nodes. The same algorithm is proposed for post deployment distributed node localization in the WSN. The results of ADE have been compared with SaDE [26], JADE [27] and CLPSO [28]. The remainder of the paper is organized as follows: a short account of Differential Evolution is given in Section 2. In Section 3 the proposed algorithm 'Adaptive Differential Evolution' (ADE) is described. In Sections 4 and 5 ADE-based image segmentation and ADE based iterative node localization is described respectively. Section 6 provides details of the simulation experiments conducted, presents the results and discusses their implications. Finally, Section 7 concludes the paper.

2. CLASSICAL DIFFERENTIAL EVOLUTION

DE is a simple real-coded evolutionary algorithm. It works through a simple cycle of stages, which are detailed below.

2.1. Initialization of the Parameter Vectors

DE searches for a global optimum point in a D-dimensional continuous hyperspace. It begins with a randomly initiated population of NPD dimensional real-valued parameter vectors. Each vector, also known as *genome/chromosome*, forms a candidate solution to the multi-dimensional optimization problem. We shall denote subsequent generations in DE by $G = 0, 1, \ldots, G_{\max}$. Since the parameter vectors are likely to be changed over different generations, we may adopt the following notation for representing the i-th vector of the population at the current generation:

$$\vec{X}_{i,G} = [x_{1,i,G}, x_{2,i,G}, x_{3,i,G}, \ldots, x_{D,i,G}]. \tag{3}$$

The initial population (at $G = 0$) should cover the entire search space as much as possible by uniformly randomizing individuals within the search space constrained by the prescribed minimum and maximum bounds: $\vec{X}_{\min} = \{x_{1,\min}, x_{2,\min}, \ldots, x_{D,\min}\}$ and $\vec{X}_{\max} = \{x_{1,\max}, x_{2,\max}, \ldots, x_{D,\max}\}$.

Hence we may initialize the j-th component of the i-th vector as:

$$x_{j,i,0} = x_{j,\min} + rand_{i,j}[0,1) \cdot (x_{j,\max} - x_{j,\min}), \tag{4}$$

where *rand* is a uniformly distributed number lying between 0 and 1 and is instantiated independently for each component of the i-th vector.

2.2. Mutation with Difference Vectors

After initialization, DE creates a *donorvector* $\vec{V}_{i,n}$ corresponding to each population member or *target* vector $\vec{X}_{i,G}$ in the current generation through mutation. Five most frequently referred mutation strategies implemented in the public-domain DE codes available online at *http://www.icsi.berkeley.edu/∼storn/code.html* are listed below:

$$\text{“DE/rand/1”: } \vec{V}_{i,G} = \vec{X}_{r_1^i,G} + F \cdot (\vec{X}_{r_2^i,n} - \vec{X}_{r_3^i,G}). \tag{5}$$

$$\text{“DE/best/1”: } \vec{V}_{i,G} = \vec{X}_{best,G} + F \cdot (\vec{X}_{r_1^i,G} - \vec{X}_{r_2^i,G}). \tag{6}$$

"DE/target-to-best/1":

$$\vec{V}_{i,G} = \vec{X}_{i,G} + F \cdot (\vec{X}_{best,G} - \vec{X}_{i,G}) + F \cdot (\vec{X}_{r_1^i,G} - \vec{X}_{r_2^i,G}). \quad (7)$$

"DE/best/2":

$$\vec{V}_{i,G} = \vec{X}_{best,G} + F \cdot (\vec{X}_{r_1^i,G} - \vec{X}_{r_2^i,G}) + F \cdot (\vec{X}_{r_3^i,G} - \vec{X}_{r_4^i,G}). \quad (8)$$

"DE/rand/2":

$$\vec{V}_{i,G} = \vec{X}_{r_1^i,G} + F \cdot (\vec{X}_{r_2^i,G} - \vec{X}_{r_3^i,G}) + F \cdot (\vec{X}_{r_4^i,G} - \vec{X}_{r_5^i,G}). \quad (9)$$

The indices r_1^i, r_2^i, r_3^i, r_4^i, and r_5^i are mutually exclusive integers randomly chosen from the range $[1, NP]$, and all are different from the index i. These indices are randomly generated once for each donor vector. The scaling factor F is a positive control parameter for scaling the difference vectors. $\vec{X}_{best,G}$ is the best individual vector with the best fitness (i.e., lowest objective function value for minimization problem) in the population at generation G. The general convention used for naming the various mutation strategies is DE/x/y/z, where DE stands for Differential Evolution, x represents a string denoting the vector to be perturbed and y is the number of difference vectors considered for perturbation of x. z stands for the type of crossover being used (exp: exponential; bin: binomial). The following section discusses the crossover step in DE.

2.3. Crossover

To enhance the potential diversity of the population, a crossover operation comes into play after generating the donor vector through mutation. The donor vector exchanges its components with the target vector $\vec{X}_{i,G}$ under this operation to form the *trial* vector $\vec{U}_{i,G} = [u_{1,i,G}, u_{2,i,G}, u_{3,i,G}, \ldots, u_{D,i,G}]$. In this article we focus on the widely used binomial crossover that is performed on each of the D variables whenever a randomly generated number between 0 and 1 is less than or equal to the Cr value. In this case, the number of parameters inherited from the donor has a (nearly) binomial distribution. The scheme may be outlined as:

$$u_{j,i,G} = \begin{cases} v_{j,i,G}, & \text{if} \quad (rand_{i,j}[0,1) \leq Cr \quad \text{or} \quad j = j_{rand} \\ x_{j,i,G}, & \text{otherwise,} \end{cases} \quad (10)$$

where, as before, $rand_{i,j}[0,1)$ is a uniformly distributed random number, which is called anew for each j-th component of the i-th parameter vector. $j_{rand} \in [1, 2, \ldots, D]$ is a randomly chosen index, which ensures that $\vec{U}_{i,G}$ gets at least one component from $\vec{V}_{i,G}$.

2.4. Selection

The next step of the algorithm calls for *selection* to determine whether the target or the trial vector survives to the next generation, i.e., at $G = G + 1$. The selection operation is described as:

$$\begin{aligned} \vec{X}_{i,G+1} &= \vec{U}_{i,G}, \quad \text{if} \quad f(\vec{U}_{i,G}) \leq f(\vec{X}_{i,G}) \\ &= \vec{X}_{i,G}, \quad \text{if} \quad f(\vec{U}_{i,G}) > f(\vec{X}_{i,G}), \end{aligned} \tag{11}$$

where $f(\vec{X})$ is the objective function to be minimized.

3. THE ADE ALGORITHM

In this section, we outline ADE and discuss the steps of the algorithm in sufficient details. The algorithm employs a new mutation scheme called DE/target-to-gr_best/1, a p-best crossover scheme, and rules for adapting the control parameters F and Cr in each generation.

3.1. DE/Target-to-Gr_best/1

In DE, greedy strategies like DE/current-to-best/k and DE/best/k benefit from their fast convergence by guiding the evolutionary search with the best solution so far discovered, thereby converging faster to that point. But, as a result of such exploitative tendency, in many cases, the population may lose its diversity and global exploration abilities within a relatively small number of generations, thereafter getting trapped to some locally optimal point in the search space. Taking into consideration these facts and to overcome the limitations of fast but less reliable convergence performance of DE/current-to-best/1 scheme, in this article, we propose a less greedy and more explorative variant of the DE/current-to-best/1 mutation strategy by utilizing the best vector of a dynamic group of $q\%$ of the randomly selected population members for each target vector. The new scheme, which we call DE/current-to-gr_best/1, can be expressed as:

$$\vec{V}_{i,G} = \vec{X}_{i,G} + F \cdot (\vec{X}_{gr\_best,G} - \vec{X}_{i,G} + \vec{X}_{r_1^i,G} - \vec{X}_{r_2^i,G}), \tag{12}$$

where $\vec{X}_{gr\_best,G}$ is the best of $q\%$ vectors randomly chosen from the current population whereas $\vec{X}_{r_1^i,G}$ and $\vec{X}_{r_2^i,G}$ are two distinct vectors picked up randomly from the current population. Under this scheme, the target solutions are not always attracted towards the same best position found so far by the entire population and this feature is helpful in avoiding premature convergence at local optima. The parameter q is known as the group size which controls the greediness of the mutation scheme DE/target-to-gr_best/1.

3.2. The *p*-best Crossover

The crossover operation used in ADE is named as p-best crossover where for each donor vector, a vector is randomly selected from the p top-ranking vectors (according to their objective function values) in the current population and then normal binomial crossover is performed as in Equation (10) between the donor vector and the randomly selected p-best vector to generate the trial vector at the same index.

3.3. Parameter Adaptation Schemes in ADE

3.3.1. Scale Factor Adaptation

At every generation, the scale factor F_i of each individual target vector is independently generated as:

$$F_i = Cauchy(F_m, 0.1), \tag{13}$$

where $Cauchy(F_m, 0.1)$ is a random number sampled from a Cauchy distribution with location parameter F_m and scale parameter 0.1. The value of F_i is regenerated if $F_i \leq 0$ or $F_i > 1$. Denote $F_{success}$ as the set of the successful scale factors, so far, of the current generation generating better trial vectors that are likely to advance to the next generation. Also let $mean_A(F_{G-1})$ is the simple arithmetic mean of all scale factors associated with population members in generation $G-1$. Location parameter F_m of the Cauchy distribution is initialized to be 0.5 and then updated at the end of each generation in the following manner:

$$F_m = w_F \cdot F_m + (1 - w_F) \cdot mean_{Pow}(F_{success}) \tag{14}$$

The weight factor w_F is set in the following way:

$$\textbf{Case 1}: \text{If} \quad mean_A(F_{G-1}) < 0.85 \quad w_F = (0.9 + 0.01 \cdot (rand(0,1))) \tag{15}$$

$$\textbf{Case 2}: \text{If} \quad mean_A(F_{G-1}) \geq 0.85 \quad w_F = (0.8 + 0.01 \cdot (rand(0,1))) \tag{16}$$

where $rand(0,1)$ stands for a uniformly distributed random number in $(0,1)$ and $mean_{Pow}$ stands for power mean given by:

$$mean_{Pow}(F_{success}) = \sum_{x \in F_{success}} \left(x^n / \left|F_{success}\right|\right)^{1/n}, \tag{17}$$

with $|F_{success}|$ denoting the cardinality of the set $F_{success}$. We have taken the value of n as 1. Small random perturbations to the weight terms of F_m and $mean_{Pow}$ puts slightly varying emphasis on the two terms each time an F is generated, and improves the performance of ADE as revealed through our parameter tuning experiments.

3.3.2. Crossover Probability Adaptation

At every generation the crossover probability Cr_i of each individual vector is independently generated as:

$$Cr_i = Gaussian(Cr_m, 0.1), \tag{18}$$

where $Gaussian(Cr_m, 0.1)$ is a random number sampled from a Gaussian distribution according with mean Cr_m and standard deviation 0.1. Cr_i is regenerated if it falls outside the interval $[0, 1]$. Denote $Cr_{success}$ as the set of all successful crossover probabilities Cr_i's at the current generation. The mean of the normal distribution Cr_m is initialized to be 0.6 and then updated at the end of each generation as:

$$Cr_m = w_{Cr} \cdot Cr_m + (1 - w_{Cr}) \cdot mean_{Pow}(Cr_{success}), \tag{19}$$

with the weight being set as:

$$w_{Cr} = 0.9 + 0.001 * rand(0, 1), \tag{20}$$

The power mean is calculated as:

$$mean_{Pow}(Cr_{success}) = \sum_{x \in Cr_{success}} \left(x^n / \left|Cr_{success}\right|\right)^{1/n}, \tag{21}$$

where $|Cr_{success}|$ denotes the cardinality of the set $Cr_{success}$. Here also we took $n = 1.5$.

4. BASED IMAGE THRESHOLDING FOR AUTONOMOUS DEPLOYMENT

There exist numerous image segmentation methods that are classified into the following categories based on the image information they exploit:

- histogram-shape-based methods;
- clustering-based methods;
- entropy-based methods;
- object-attribute-based methods;
- spatial methods;
- local methods.

The ADE based image thresholding method used here exploit the image histogram shape. Otsu [29] proposed a nonparametric and unsupervised method of automatic threshold selection for image segmentation. This method establishes three appropriate criteria for evaluating the suitability of a given threshold level from the image

histogram. The following paragraphs discuss the basics of the Otsu evaluation criteria and point out strengths and weaknesses of the Otsu-based exhaustive search. Consider a digital image having a height of H pixels and a width of W pixels, in which the intensities are represented in L gray levels $[1, 2, \ldots, L]$. Let n_i represent the number of pixels having intensity level i. It can be observed that the total number of pixels N satisfies $N = H \times W = n_1 + n_2 + \ldots + n_L$. The $1-D$ vector n_i with $i = [1, 2, \ldots, L]$ represents the image histogram. The histogram is normalized and regarded as a probability distribution as follows:

$$p_i = \frac{n_i}{N}, \quad p_i > 0 \quad \text{and} \quad \sum_{i=1}^{L} p_i = 1 \tag{22}$$

Suppose it is desired to dichotomize the pixels into classes C_1, which represents background, and C_2, which represents an object, using a threshold level t. The class C_1 contains all pixels having intensities less than or equal to t, and the class C_2 contains all pixels having intensities greater than t. The probabilities of occurrence classes C_1 and C_2 are given as follows:

$$w_1(t) = \Pr(C_1) = \sum_{i=1}^{t} p_i \tag{23}$$

$$w_2(t) = \Pr(C_2) = \sum_{i=t+1}^{L} p_i \tag{24}$$

The mean levels of classes C_1 and C_2 are given by,

$$\mu_1(t) = \sum_{i=1}^{t} \frac{i p_i}{w_1} \tag{25}$$

$$\mu_2(t) = \sum_{i=t+1}^{L} \frac{i p_i}{w_2} \tag{26}$$

These probabilities and mean levels satisfy the conditions $w_1(t) + w_2(t) = 1$ and $w_1\mu_1 + w_2\mu_2 = \mu_T$, where

$$\mu_T = \mu(L) = \sum_{i=1}^{L} i p_i \tag{27}$$

is the total mean level of the image. The variance of distribution of pixels in classes C_1 and C_2 are given by,

$$\sigma_1^2(t) = \sum_{i=1}^{t} \{i - \mu_1(t)\}^2 \frac{p_i}{w_1} \tag{28}$$

$$\sigma_2^2(t) = \sum_{i=t+1}^{L} \{i - \mu_2(t)\}^2 \frac{p_i}{w_2} \tag{29}$$

In order to evaluate the goodness of the threshold at level t, Otsu introduced three objective functions (λ), (κ), and (η), defined as follows:

$$\lambda = \frac{\sigma_B^2}{\sigma_W^2}, \quad \kappa = \frac{\sigma_T^2}{\sigma_W^2} \quad \text{and} \quad \eta = \frac{\sigma_B^2}{\sigma_T^2} \tag{30}$$

where σ_W^2 is the within-class variance, σ_B^2 is the between-class variance, and σ_T^2 is the total variance. These are defined in as follows:

$$\sigma_W^2 = w_1\sigma_1^2 + w_2\sigma_2^2 \tag{31}$$

$$\sigma_B^2 = w_1 w_2 + (\mu_2 - \mu_1)^2 \tag{32}$$

$$\sigma_T^2 = \sum_{i=1}^{L} (i - \mu_T)^2 p_i \tag{33}$$

With this, the problem of two-level thresholding is reduced to an optimization problem to search for the threshold t^* that maximizes one of the objective functions defined in Equations (32)–(34). It also follows that the threshold t^* that maximizes σ_B^2 also minimizes σ_W^2. A simple approach to optimal thresholding is to perform an exhaustive sequential search for a threshold level t^*, which satisfies $\sigma_W^2(t^*) = \min_{1\le t<L} \sigma_B^2(t)$. This can be extended to n-level thresholding problem, which involves $n-1$ thresholds that satisfy $\sigma_W^2(t_1^*, t_2^*, \ldots, t_{n-1}^*) = \min_{1\le t_1<t_2<\ldots<t_{n-1}<L} \sigma_W^2(t_1, t_2, \ldots, t_{n-1})$.

The exhaustive search method based on the Otsu criterion is simple and straightforward, but it has a weakness that it is computationally expensive. The ranges of $n-1$ candidate thresholds for n-level thresholding are as follows: $1 \le t_1 < L-n+1, t_1+1 \le t_2 < L-n+2$ and $t_{n-2}+1 \le t_{n-1} < L-1$. Exhaustive search for $n-1$ optimal thresholds involves evaluations of objective functions of $n(L-n+1)^{n-1}$ combinations of thresholds. Therefore, it is not a suitable choice for the applications that require real-time multilevel image thresholding.

The task of determining $n-1$ optimal thresholds for n-level image thresholding can be formulated as a multidimensional optimization problem. In this study, ADE been used to determine the thresholds that minimizes the within-class variance σ_W^2 of the intensity distributions. For ADE, the position of a particle i is defined as $X_i = \{t_1, t_2, \ldots, t_{n-1}\}$. The vectors of the population are evaluated for the fitness function, which is defined as the within-class variance σ_W^2 of the image-intensity distributions.

This is shown in the following equation:

$$f(X_i) = \sigma_W^2(X_i) \tag{34}$$

The goal of ADE is to determine the position in the search space that satisfies Equation (36).

$$X_{best} = \min_{1 \le t_1 < t_2 \ldots < t_{n-1} < L} \sigma_W^2(t_1, t_2, \ldots, t_{n-1}) \tag{35}$$

5. BASED ITERATIVE NODE LOCALIZATION

The goal of WSN node localization is to obtain distributed evaluation of coordinates of the maximum of N target nodes using M stationary beacons that are aware of their positions. The node localization in a WSN is performed in this study as described below.

1) N dumb nodes and M beacons each having a transmission radius of r units are deployed from a UAV in a sensor field. Beacon nodes know their locations, and the information regarding their coordinates is utilized to locate the dumb nodes. The nodes that become settled at the termination of each iteration serve as references in the next generation, in which they function as the beacons do.
2) Each node that is situated within the transmission range of three or more beacons or settled nodes is termed as a localizable node.
3) Each localizable node in the deployment field evaluates its distance from each of its neighbouring beacons or settled nodes. The measurement noise considered here is taken as Additive White Gaussian noise (AWGN). A node calculates its distance from a beacon i as $\hat{d}_i = d_i + n_i$, where d_i represents the actual distance formulated as
$$d_i = \sqrt{(x - x_i)^2 + (y - y_i)^2}.$$
Here, (x, y) represents the position of the target node, and (x_i, y_i) is the location coordinates of the ith beacon present in the neighbourhood of the target node. The measurement noise n_i takes on a random numerical value uniformly distributed within the interval $d_i \pm d_i(P_n/100)$. It is evident that the localization result is dependent on the amount of P_n, the percentage noise that adversely affects the parameters involved in distance measurements.
4) Each localizable node independently applies ADE to locate the coordinates (x, y) that minimize the objective function. The

objective function is basically the error defined as follows:

$$f(x,y) = \frac{1}{M}\sum_{i=1}^{M}\left(\sqrt{(x-x_i)^2+(y-y_i)^2} - \hat{d}_i\right)^2 \quad (36)$$

where $M \geq 3$ is the number of beacons or settled nodes falling within the transmission radius of the target node.

5) ADE searches for the coordinates (x, y) that will minimize the error in Equation (36), therefore, the fitness space is basically a 2-dimensional landscape.
6) After all the localizable nodes are able to find out their coordinates, the total localization error is calculated as the mean of squares of distances between actual locations $(\hat{x}_i, \hat{y}_i)$, $i = 1, 2, \ldots, N_L$ determined by ADE. This is evaluated as follows:

$$E_l = \frac{1}{N}\sum_{i=1}^{L}\left((x_i-\hat{x}_i)^2+(y_i-\hat{y}_i)^2\right) \quad (37)$$

7) Steps 2 to 6 are repeated until either all the dumb nodes become localized or no more nodes can be localized. The performance metric of a localization algorithm is governed by the doublet (N_{N_L}, E_l), where $N_{N_L} = N - N_L$ is the number of nodes that are unable to be localized. The lower the values of N_{N_L} and E_l, the better the performance is.

The number of localized nodes increases with iterations. This increases the number of references already available for localized nodes. A node that localizes using just three references in an iteration k can possess more references in iteration $k+1$ due to increase in localizable nodes. This diminishes the probability of the flip ambiguity. On contrary, if a node has more references in iteration $k+1$ than in iteration k, the localization time increases. It is observed from exhaustive experimentation that the maximum number of references can be safely restricted to six.

6. EXPERIMENTS AND RESULTS

All the simulations are done on a Pentium core 2 duo machine with 2 GB RAM and 2.23 GHz speed.

6.1. Image Thresholding for Autonomous Deployment

All the algorithms are used to calculate the optimal thresholding values by minimizing the within-class variance of the distribution of intensity levels in the given image. The parameters for ADE are set as follows:

1) Parameter q which controls the greediness of the mutation scheme DE/current-to-gr_best/1 is set to 1/4th of the population size.
2) Parameter p in p-best crossover is also taken as 1/4th of the population size.

For the compared algorithms we employ the best suited parametric set-up chosen as guidelines from their respective literature.

We have conducted three set of experiments on the given image lake in Figure 1. This image consists of 256 intensity levels. The comparison is made on the basis of the number of iterations taken by the algorithms to reach the optimal threshold values as computed by the exhaustive search algorithm which is basically a deterministic process. All the computations are averaged over 50 independent runs.

6.1.1. 2-level Thresholding

The population size for all the contestant algorithms is kept as 20 for 2-level thresholding of the image. All the algorithms are successful to find the optimal threshold values but ADE can converge to the optimal values in minimum iterations as evident from Table 1. The output image is shown in Figure 2.

6.1.2. 3-level Thresholding

Here the goal is to find the two threshold values t_1^* and t_2^*. The population size is kept as same as in the 2-level thresholding case. This is a two-dimensional problem. The dimensions are initialized

Figure 1. Aerial image of the lake used for ADE based thresholding.

Figure 2. 2-level thresholding.

as random integers between 1 and 256 so that $1 < t_1 < t_2 < 256$. The optimal thresholds computed by the algorithms are identical with the exhaustive search results but it is clear from Table 1 that ADE converges faster to the threshold values outperforming the other algorithms. The 3-level thresholding image is shown Figure 3.

6.1.3. Multilevel Thresholding

Here the algorithm searches for the $n-1$ optimal threshold values. The population size is kept at the same value (20) for all the algorithms for fair comparison. Here also ADE can converge to the optima in lesser number of iterations as shown in Table 1. Figure 4 shows the 4-level thresholding of the given lake image.

Table 1. Summary of results for image thresholding.

Number	SaDE		JADE	
	Optimal threshold(s)	Iteration	Optimal threshold(s)	Iteration
2	112	10	112	8
3	103,146	20	103,146	18
4	94,125,155	30	94,125,155	27
5	90,119,143,179	46	90,119,143,179	42
Number	CLPSO		ADE	
	Optimal threshold(s)	Iteration	Optimal threshold(s)	Iteration
2	112	10	112	6
3	103,146	22	103,146	15
4	94,125,155	32	94,125,155	25
5	90,119,143,179	48	90,119,143,179	40

The objective is to deploy the sensors on dry land avoiding the water and vegetated areas in which any sensor deployment will be counted as a waste. 50 experiments of ADE-based autonomous deployment are conducted on the image lake terrain where 40 nodes and eight beacons are deployed. The outcome of one such experiment is shown in Figure 5 which shows all sensor nodes and beacons being deployed on dry land avoiding water and vegetative zones.

It is to be noted from Table 1 that all the contestant algorithms are successful in finding out the optimal thresholding values. But in emergency applications like disaster management and battlefield surveillance, preference should be given to that algorithm which can

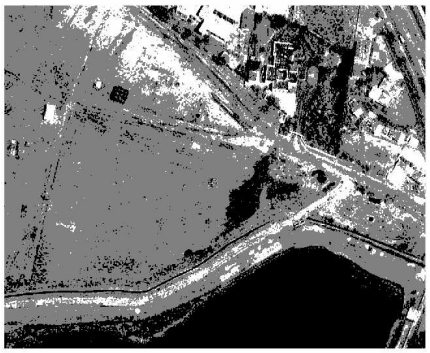

Figure 3. 3-level thresholding.

Figure 4. 4-level thresholding.

Figure 5. Node deployment.

provide us the optimal thresholding values in minimum number of iterations. Table 1 clearly shows the superiority of ADE as an efficient image thresholding algorithm. The superior performance of ADE can be attributed to the p-best crossover operation which is exploitative resulting in quick convergence. At the same time to avoid premature convergence a less greedy and more explorative mutation scheme DE/current-to-gr_best/1 has been used and parameters F and Cr are also adapted based on information of previous generations.

In autonomous deployment of wireless sensor nodes, the thresholding of the terrain image taken from a downward-pointed camera is done using ADE. The threshold information indicates whether a node can be dropped at that location or not. The output of thresholding of the image lake shows water and vegetated areas as black pixels and dry land as white pixels.

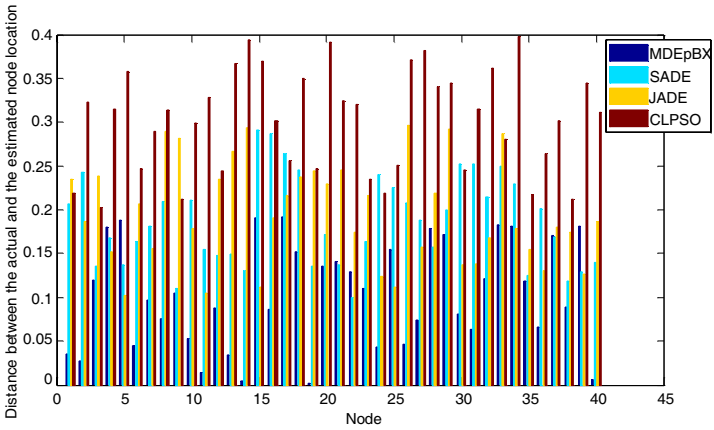

Figure 6. Distance between the actual node location and location estimated by the algorithms for each node.

6.2. Iterative Node Localization

Since all the wireless sensor nodes are deployed randomly in the dry land avoiding water bodies, we are unaware of the location of the nodes. In this section we are going to find out the location of the nodes with the help of the beacons as references using ADE. Performance of ADE is compared with the same algorithms, i.e., SaDE, JADE and CLPSO. For the experiment we have deployed forty nodes and eight beacons randomly in a sensor field of dimensions 100×100 square units. The transmission radius of each beacon is taken to be $r = 30$ units. In the experiment each target node that can be localized, i.e., falls within the transmission radius of three or more beacons uses ADE to localize itself. The population size for all the algorithms is kept as 30 and the number of iterations taken to localize each node is 100. The positions of each localizable node are randomly initialized between 0 and 100. The parameters p and q in ADE are kept at the same value as in the case of image thresholding. For the contestant algorithms we employ the same parametric set-up as in the previous case.

Fifty ADE-based localization experiments are conducted for $P_n = 2$ and $P_n = 5$. In each experiment each algorithm is allowed to run upto 4 iterations and at the end of each iteration the performance metric (N_{N_L}, E_l) is computed for each contestant algorithm. Table 2 shows the mean values of N_{N_l} and E_l for each algorithm at the end of four iterations both for $P_n = 2$ and $P_n = 5$. The lower the values of these quantities, the better will be the performance of the algorithm and ADE has clearly shown its superiority as evident from

Table 2. (N_{N_L}, E_l) for $P_n = 5$ and $P_n = 2$ after 4 iterations.

	$P_n = 5$		$P_n = 2$	
Algorithms	Mean (N_{N_L})	Mean (E_l)	Mean (N_{N_L})	Mean (E_l)
SaDE	0.86	0.0838	0.16	0.0123
JADE	0.72	0.0518	0.02	0.0085
CLPSO	1.26	0.2635	0.44	0.0205
ADE	0.48	0.0263	0.00	0.0015

Table 3. N_L and E_l after the end of each iteration.

Algorithms	Parameters	Iteration 1	Iteration 2	Iteration 3	Iteration 4
SaDE	Mean (N_L)	22.42	36.04	38.32	39.84
	Mean (E_l)	4.0532	0.1343	0.0827	0.0123
JADE	Mean (N_L)	23.28	36.42	38.92	39.98
	Mean (E_l)	2.4653	0.1034	0.0553	0.0085
CLPSO	Mean (N_L)	21.54	35.12	37.46	39.56
	Mean (E_l)	8.7835	0.5594	0.1648	0.0205
ADE	Mean (N_L)	24.66	37.82	39.74	40
	Mean (E_l)	1.0734	0.1432	0.0147	0.0015

the table x. Further, in Table 3 we have reported the mean values of N_l and E_l at the end of each iteration for each algorithm. ADE has outperformed the well known DE and PSO variants and this superior performance can be attributed to its algorithmic components-DE/current-to-gr_best/1, p-best crossover and parameter adaptation based on information of previous generations.

The maximum amount of Gaussian additive noise, P_n affects the distance accuracy measurements. It is to be noted form Table 2 that the localization error E_l increases as the amount of noise P_n increases. In Figure 6, we have plotted a bar graph showing the distance between the actual node and the node location estimated by the algorithms

7. CONCLUSION

In this paper we propose a modified Differential Evolution scheme namely ADE for segmentation of terrain images taken from an UAV for autonomous deployment of sensor nodes and for distributed localization of the deployed nodes in an iterative manner. Both the

assignments have been formulated as multidimensional optimization problems and solved by the proposed algorithm. The algorithm has been briefly outlined and a statistical summary of the simulation results are presented. It is clearly seen that ADE is able to perform image segmentation faster than the competitor algorithms as well as reducing the number of sensor nodes from being deployed in the terrains of no interest. The distributed localization method presented in this paper diminishes the number of transmissions to the base station assisting the nodes to conserve their energy, which is a grave concern encountered in majority of the applications of WSNs. Simulation results clearly indicate that the positions of the nodes obtained by ADE are more accurate. Our future work includes utilizing a vision system which considers the colour and texture property of terrains.

REFERENCES

1. Akyildiz, I. F., W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Commun. Mag.*, Vol. 40, No. 8, 102–114, Aug. 2002.
2. Callaway, Jr., E. H., *Wireless Sensor Networks: Architectures and Protocols*, CRC Press, Aug. 2003.
3. Zhao, F. and L. Guibas, *Wireless Sensor Networks: An Information Processing Approach*, Morgan Kaufmann, 2004.
4. Bulusu, N. and S. Jha, *Wireless Sensor Network: A Systems Perspective*, Artech House, Jul. 2005.
5. Chong, C. and S. Kumar, "Sensor networks: Evolution, opportunities, and challenges," *Proc. IEEE*, Vol. 91, No. 8, 1247–1256, Aug. 2003.
6. Halgamuge, M. N., M. Zukerman, K. Ramamohanarao, and H. L. Vu, "An estimation of sensor energy consumption," *Progress In Electromagnetics Research B*, Vol. 12, 259–295, 2009.
7. Liu, H. Q., H. C. So, K. W. K. Lui, and F. K. W. Chan, "Sensor selection for target tracking in sensor networks," *Progress In Electromagnetics Research*, Vol. 95, 267–282, 2009.
8. Gay-Fernandez, J. A., M. G. Sanchez, I. Cuinas, A. V. Alejos, J. G. Sanchez, and J. L. Miranda-Sierra, "Propagation analysis and deployment of a wireless sensor network in a forest," *Progress In Electromagnetics Research*, Vol. 106, 121–145, 2010.
9. Al-Karaki, J. N. and A. E. Kamal, "Routing techniques in wireless sensor networks: A survey," *IEEE Wireless Communications*, 6–28, Dec. 2004.
10. Akyildiz, I. F., W. Su, Y. Sankarasubramaniam, and E. Cayirci,

"Wireless sensor networks: A survey," *Computer Networks*, Vol. 38, No. 4, 393–422, 2002.

11. Pottie, G. and W. Kaiser, "Wireless sensor networks," *Communications of the ACM*, Vol. 43, No. 5, 51–58, May 2000.
12. Bojkovic, Z. and B. Bakmaz, "A survey on wireless sensor networks deployment," *WSEAS Trans. on Communications*, Vol. 7, No. 12, 1172–1181, Dec. 2008.
13. Yick, J., B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, Vol. 52, No. 12, 2292–2330, Aug. 2008.
14. Corke, P., S. Hrabar, R. Peterson, D. Rus, S. Sampalli, and G. Sukhatme, "Autonomous deployment and repair of a sensor network using an unmanned aerial vehicle," *Proc. IEEE Int. Conf. Robot. Autom.*, Vol. 4, 3602–3608, May 2004.
15. Ollero. A. and L. Merino, "Control and perception techniques for aerial robotics," *Annu. Rev. Control*, Vol. 28, 167–178, May 2004.
16. Sezgin, M. and B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation," *J. Electron. Imag.*, Vol. 13, No. 1, 146–168, Jan. 2004.
17. Patwari, N., J. N. Ash, S. Kyperountas, A. O. Hero, R. L. Moses, and N. S. Correal, "Locating the nodes: Cooperative localization in wireless sensor networks," *IEEE Signal Process. Mag.*, Vol. 22, No. 4, 54–69, Jul. 2005.
18. Aspnes, J., T. Eren, D. K. Goldenberg, A. S. Morse, W. Whiteley, Y. R. Yang, B. D. O. Anderson, and P. N. Belhumeur, "A theory of network localization," *IEEE Trans. Mobile Comput.*, Vol. 5, No. 12, 1663–1678, Dec. 2006.
19. Mitilineos, S. A., D. M. Kyriazanos, O. E. Segou, J. N. Goufas, and S. C. A. Thomopoulos, "Indoor localization with wireless sensor networks," *Progress In Electromagnetics Research*, Vol. 109, 441–474, 2010.
20. Boukerche, A., H. A. B. Oliveira, E. F. Nakamura, and A. A. F. Loureiro,"Localization systems for wireless sensor networks," *IEEE Wireless Commun. Mag.*, Vol. 14, No. 6, 6–12, Dec. 2007.
21. Hightower, J. and G. Borriello, "Location systems for ubiquitous computing," *Computer*, Vol. 34, No. 8, 57–66, Aug. 2001.
22. Mao, G., B. Fidan, and B. D. O. Anderson, "Wireless sensor network localization techniques," *Computer Networks*, Vol. 51, No. 10, 2529–2553, Jul. 2007.
23. Amundson, I. and I. Amundson, "A survey on localization for

mobile wireless sensor networks," *MELT'09 Proceedings of the 2nd International Conference on Mobile Entity Localization and Tracking in GPS-less Environments*, Vol. 5801, 235–254, Springer Berlin/Heidelberg, 2009.

24. Storn, R. and K. V. Price, "Differential evolution — A simple and efficient adaptive scheme for global optimization over continuous spaces," *Technical Report TR-95-012, ICSI*, http://http.icsi.berkeley.edu/∼storn/litera.html, 1995.
25. Storn, R. and K. Price, "Differential evolution — A simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, Vol. 11, No. 4, 341–359, 1997.
26. Qin, A. K., V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. on Evolutionary Computation*, Vol. 13, No. 2, 398–417, Apr. 2009.
27. Zhang, J. and A. C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," *IEEE Trans. on Evolutionary Computation*, Vol. 13, No. 5, 945–958, Oct. 2009.
28. Liang, J. J., A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Trans. on Evolutionary Computation*, Vol. 10, No. 3, 281–295, 2006.
29. Otsu, N., "A threshold selection method from gray-level histograms," *IEEE Trans. Syst., Man Cybern.*, Vol. 9, No. 1, 62–66, Jan. 1979.