International Journal of Grid and Distributed Computing
Vol. 9, No. 2 (2016), pp.157-168
http://dx.doi.org/10.14257/ijgdc.2016.9.2.14

Smart Deployment of Sensor Nodes Using SA & GFA

[1.] Shailendra Singh, [2.] Deepa verma

[1,2] Assistant Professor E&CE department
[1,2] ITM University Gwalior, India
[1]shailendrasingh.ece@itmuniversity.ac.in
[2]deepaverma.ec@itmuniversity.ac.in

Abstract

In wireless sensor networks, network connectivity problems and sensor coverage are mainly affected by the restricted communication and sensing range of the nodes. To solve all these problems, many studies have been undertaken till now but most of these works is limited to be applied to the real environments because these do not consider various environmental factors that affect wireless sensor network deployment. In this paper, we propose a node deployment strategy that considers environmental factors and the number of nodes for surveillance .Deployment of the sensor nodes, and relay nodes considering all features of the surveillance and reconnaissance sensor networks as well as environmental factors. Smarty deploying the sensor nodes is very important if we are considering the performance of wireless sensor nodes (wsn). This paper focuses on real time deployment of sensor nodes using unmanned aerial vehicle (UAV). Smart deployment has importance in battlefield surveillance, disaster monitoring, environmental and habitat monitoring etc. It plays a very important role mainly in disasters affecting areas like flood affected or fire affected forests for sensing different conditions and physical data out there. The objective of this paper is to deploy sensor nodes smartly only in terrain of interest not in fire or water so that it can conserve environment and can reduce the loss of sensor nodes from being deploy in fire or water. Image segmentation of the images captured by camera situated on a board of UAV.. Comparison of these optimization tools SA and GFA are presented here for different images. Simulation results show that GFA performs better than SA when compared with no. of individuals and SA is faster than GFA.

***Keywords**: Simulated Annealing Algorithm (SA), image thresholding, Gravitational Field Algorithm (GFA), Wireless Sensor Networks (WSNs).*

Introduction

Wireless sensor networks (WSNs) have been used in many applications, such as forest monitoring, flood monitoring space exploration, disaster management, factory automation, secure installation, border protection and battlefield surveillance. A sensor node sense and collects environmental information such as humidity, seismicity, temperature, and GFA Relay a node sends sensor data to the other nodes or base. Most sensor nodes and relay nodes have some certain constraints such as restricted communication and sensing range as well as some limited battery life. These limitations may cause some technical by means of coverage, network lifetime, network connectivity, scheduling. The problems associated with coverage and connectivity is caused by the limited detection range and communication range between nodes. The coverage problem is important that the sensor nodes can cover the whole area it should monitor and sense . One solution of this problem is to maximize coverage is to deploy sensor nodes not too near to other nodes so that we can have a maximum use of detection capability of sensor node and at the same time to deploy the sensor nodes not too far from other nodes so that it can avoid the coverage holes. Sensor nodes and relay nodes need to be placed close enough so that they are within each other's communication range to ensure good connectivity. Remote monitoring covers a wide range of all applications where wireless systems can complement the wired systems by reducing the wiring costs and allowing some new types of measurement applications. Applications of remote monitoring include:

1. Nature monitoring by means of water, soil and air
2. monitoring for building structures like bridges
3. machine monitoring

Terrain recognition by means of color image segmentation has importance to optimal and autonomous deployment of WSN nodes. This can be effectively used to prevent sensor nodes from being deployed into those areas which are not interested such as water or fire in the terrain. Such prevention is necessary from the point of view of conservation of the environment. Besides, it reduces the loss of sensor nodes and confirms that the required sensing coverage can be achieved with less number of nodes deployed only in the terrains of interest where we want to deploy. Thresholding method is commonly used for segmentation of a true color images into two or more classes [4]. Let f(x, y) denote a Grayscale image of size H × W pixels that has S intensity levels. Its Two-level thresholding deals with determining a value of threshold t to perform the operation expressed by (1) for x =1, 2... H and y =1, 2... W

$$\mathrm{f(x,y)} = \begin{cases} 0, & \text{if } \mathrm{f(x,y)} \le t \\ L, & \text{if } \mathrm{f(x,y)} > t \end{cases} \quad ...(1)$$

This 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:

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

This can be further extended to generic n-level thresholding in which $n-1$ threshold levels t are necessary. It is obvious from (1) and (2) that the effectiveness of multilevel colour segmentation largely depends on the values of threshold levels $t_1, t_2, t_3 ..... t_{n-1}$.The basic question is: how are the threshold levels determined? Many methods have been proposed in literature, a survey of which is presented in paper.

The primary contribution of this paper is as follows:

This paper is organized as follows. Brief reviews of SA and GFA are presented in Section II-A and II-B, respectively. SA and GFA based image thresholding methods for autonomous deployment of sensor nodes from a modern UAV are presented in Section III. Details of the MATLAB based numerical simulations are presented and the results are discussed in Section IV. And finally concluding remarks are given in Section V.

II. Bio Inspired Algorithm: Detailed

Real world problems in almost all engineering departments are considered as multi-dimensional parabolic problems. It requires an enormous computational effort by solving with analytical methods and this effort increases exponentially when the problem size increases. This provides a motivation for multidimensional optimization tool that require moderate memory and computational resources, and yet produce good results

.

A) Simulated Annealing Algorithm

Simulated annealing (SA) is a generic probabilistic metaheuristic for the global optimization problem of locating a good approximation to the global optimum of a given function in a large search space. It is often used when the search space is discrete (*e.g.*, all tours that visit a given set of cities). For certain problems, simulated annealing may be more efficient than exhaustive enumeration — provided that the goal is merely to find an acceptably good solution in a fixed amount of time, rather than the best possible solution. The name and inspiration come from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. Both are attributes of the material that depend on its thermodynamic free energy. Heating and cooling the material affects both the temperature and the thermodynamic free energy. While the same amount of cooling brings the same amount of decrease in temperature it will bring a bigger or smaller decrease in the thermodynamic free energy depending on the rate that it occurs, with a slower rate producing a bigger decrease. This notion of slow cooling is implemented in the Simulated Annealing algorithm as a slow decrease in the probability of accepting worse solutions as it explores the solution space. Accepting worse solutions is a fundamental property of metaheuristics because it allows for a more extensive search for the optimal solution.

Simulated annealing mimics the annealing process to solve an optimization problem. It uses a temperature parameter that controls the search. The temperature parameter typically starts off high and is slowly "cooled" or lowered in every iteration. At each iteration a new point is generated and its distance from the current point is proportional to the temperature. If the new point has a better function value it replaces the current point and iteration counter is incremented. It is possible to accept and move forward with a worse point. The probability of doing so is directly dependent on the temperature. This unintuitive step sometime helps identify a new search region in hope of finding a better minimum.

The global best version of SA for minimization of cost function f(.)

```
$x \leftarrow$ <initial random solution>
$i \leftarrow 0$
while <Global stop condition not satisfied>do
    $t \leftarrow i+1$
    while <Local stop condition not satisfied> do
        $x^* \leftarrow$ PickRandomNeighbor(x)
        $\Delta E = F(x^*) - F(x)$
        $f$ $\Delta E<0$ then
            $x \leftarrow x^*$
        else
            $r \leftarrow$ <random uniform number between 0 and 1>
            if $r < e^{-\Delta E/(k-t)}$ then
                $x \leftarrow x^*$
            end if
        end if
    end while
end while
<Display solution x>
```

B) Gravitational Field Algorithm

The most important advantage of GFA is the ability with the multimodal objective functions (*i.e.* the mass functions). GFA can converge for one independent variable mass functions with probability 1. We give a theorem and its strict mathematical proof as follows. We define the mass function as f(x), and the dusts in one group of variable space are $x_1,x_2,x_3...x_{max},...,x_n$.The mass functions are subject to Equation as follows: In $f(x_{max}) \geq f(x_m)$ the group, the dusts become xWhere $m = 1, 2, 3..., n$. After moving towards x_{max} $x_1^{'}, x_2^{'}, x_3^{'}...x_{max}^{'},..., x_n^{'}$

Now we have a theorem

GFA for one independent variable mass functions could converge in the global maximum with probability 1, when the three conditions as below are satisfied:

1) The scale of group is small enough; such that the number of peaks is at most one.
2) The motion of the surrounding dusts is smoothing.
3) The number of dusts in one group is big enough.

When the $x_{max} \neq x_{peak}$

The global best version of GFA for minimization of cost function f (.)

```
dusts[] $\leftarrow$ random
for all i
such that $0 \leq$ i<N do
    dusts[i] $\in$ group[random]
end for
or all i such that $0 \leq$ i<G do
    centre[i] $\leftarrow$ GetMax(group[i])
end for
end for i such that $0 \leq$ i<N do
    dusts[i] $\leftarrow$ dusts[i] + (GetCentre(dusts[i])
$-$dust[i] x 0.0618
    while dusts[i] $\notin$ solution space do
        dusts[i] $\leftarrow$ random
    end while
end for
for all i such that $0 \leq$ i<N do
    if Fun((GetCentre(dusts[i])) $-$
 Fun(dusts[i]))) $<$ threshold then
        delete dusts[i]
    end if
end for
for all i such that $0 \leq$ i<N do
    if dust[i] is not GetCentre(dusts[i])
 $and$ random<r **then**
22. $dusts[\mathrm{i}] \leftarrow dusts[\mathrm{i}] + \mathrm{random}$
23. **end if**
24. **end for**
25. **if** $GroupNotFinish$ **then**
26.  $goto[5]$
27. **else** 27
28. $goto\ [29]$
29. **end if**
30. **if** $Finish$ **then**
31. $return\ centre[]$
32. **else**
33. $update\ N, G$
34. $goto[2]$
35. **end if**
```

III. SA and GFA Based Image Thresholding for Autonomous Deployment

There are numerous methods for image thresholding and image segmentation classified on the basis of image information they comprises of:

1) Thresholding
2) Clustering methods
3) Compression-based methods
4) Histogram-based methods
5) Edge detection

Let p represent the number of pixels having intensity level *i.e.* can be observed that the total number of pixels N satisfies

$$N = HXW = n_1 + n_2 + ... + n_S$$

The 1-D vector n with $i = [1, 2, ..., \mathrm{S}]$ represents the image histogram. The histogram is normalized and regarded as a probability distribution, as in, as follows:

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

Suppose it is desired to dichotomize the pixels into classes C_1 and $C_{2.}$ The class C_1 contains all pixels having intensities less than or equal to t, and the class C_2contains all pixels having intensities greater than the probabilities of occurrence classes C_2 and C_2 given by respectively

$$\omega_1(\mathrm{t}) = \Pr(\mathrm{C}_1) = \sum_{i=1}^{t} p_i$$

$$\omega_2(\mathrm{t}) = \Pr(\mathrm{C}_2) = \sum_{i=t+1}^{S} p_i$$

The mean levels defined by classes C_1 and C_2 are given byare given by respectively

$$\mu_1(\mathrm{t}) = \sum_{i=1}^{t} \frac{ip_i}{\omega_1}$$

$$\mu_2(\mathrm{t}) = \sum_{i=t+1}^{S} \frac{ip_i}{\omega_2}$$

These mean levels and probabilities satisfy the conditions $\omega_1(\mathrm{t}) + \omega_2(\mathrm{t}) = 1$ And $\omega_1\mu_1 + \omega_2\mu_2 = \mu_T$ where,

$$\mu_T = \mu(\mathrm{S}) = \sum_{i=1}^{S} ip_i$$

is the total mean level of the image. The variance of distribution of pixels in classes C$_1$ and C$_2$ are given by

Respectively

$$\sigma_1^2(\mathrm{t}) = \sum_{i=1}^{t} \{i - \mu_1(\mathrm{t})\}^2 \frac{p_i}{\omega_1}$$

$$\sigma_2^2(\mathrm{t}) = \sum_{i=t+1}^{S} \{i - \mu_2(\mathrm{t})\}^2 \frac{p_i}{\omega_2}$$

In order to find the goodness of the threshold at level

t, Otsu introduced three objective functions (α), (β), and (χ), defined in , as follows:

$$\alpha = \frac{\sigma_B^2}{\sigma_W^2}$$

$$\beta = \frac{\sigma_T^2}{\sigma_W^2} \mathrm{W}$$

$$\chi = \frac{\sigma_B^2}{\sigma_T^2}$$

Where σ_W^2 is the within class variance, σ_B^2 is the between classvariance, and σ_T^2 is the total variance. These are defined respectively

$$\sigma_W^2 = \omega_1\sigma_1^2 + \omega_2\sigma_2^2$$

$$\sigma_B^2 = \omega_1\omega_2 + (\mu_2 - \mu_1)^2$$

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

. A simple approach to optimal thresholding is to perform an exhaustive sequential search for a threshold Level t^* which satisfies $\sigma_B^2(\mathrm{t}^*) = \max_{1 \le t < S} \sigma_B^2(\mathrm{t})$ this can be extended to n-level thresholding problem, which involves $\mathrm{n}-1$ Thresholds that satisfy

$$\sigma_B^2(\mathrm{t}_1^*, \mathrm{t}_2^*, \ldots., \mathrm{t}_{n-1}^*) = \max_{1 \le t_1 < t_2 \ldots < t_{n-1} < S} \sigma_B^2(\mathrm{t}_1, \mathrm{t}_2, \ldots., \mathrm{t}_{n-1})$$

The exhaustive search method based on the Otsu criterion is simple and straightforward, but it has a weakness that itis computationally expensive [11]. The ranges of n - 1 candidate thresholds for n-level thresholding are as follows:

$$1 \le t_1 < S - n + 1, t_1 + 1 \le t_2 < S - n + 2$$

And $t_{n-2} + 1 \le t_{n-1} < L - 1$ Exhaustive search for n - 1 optimal threshold involves evaluations of objective functions of $n(\mathrm{L} - \mathrm{n} + 1)^{n-1}$ combinations of thresholds. SA and GFA have been used to determine the thresholds that maximize the between class variance σ_B^2 of the intensity distributions. For SA, the position of particle i is defined as

$$X_i = \{\mathrm{t}_{1,} \mathrm{t}_{2,....,} \mathrm{t}_{n-1}\}$$

And for GFA, the position of a bacterium i is defined as $P_i = \{\mathrm{t}_{1,} \mathrm{t}_{2,....,} \mathrm{t}_{n-1}\}$ the particles in SA and the traits in GFA are evaluated for the fitness function, which is defined as the between-class variance $\sigma_B^2$of the image-intensity distributions. These are shown in , respectively

$$f(\mathrm{X}_i) = \sigma_B^2(\mathrm{X}_i)$$

$$J(\mathrm{P}_i) = \sigma_B^2(\mathrm{P}_i)$$

The goal of SA is to find the position in the search space that satisfies

$$\mathrm{Xgbest} = \max_{1 \le t_1 < t_2 ... < t_{n-1} < S} \sigma_B^2(\mathrm{t}_{1,} \mathrm{t}_2 ..., \mathrm{t}_{n-1})$$

Similarly, the goal of GFA is to determine the position in the
Search space that satisfies

$$P\Big|_{J\max} = \max_{1 \le t_1 < t_2 ... < t_{n-1} < S} \sigma_B^2(\mathrm{t}_{1,} \mathrm{t}_2 ..., \mathrm{t}_{n-1})$$

IV. Numerical Simulation and Results

A.Image Thresholding for Autonomous Deployment

SA and GFA are used to compute the optimal values of thethresholds by maximizing the between-class variance of the distribution of intensity levels in the given image. The parameters for SA and GFA algorithms are chosen as follows:

SA:
1) Population = 20;
2) Iterations = 10;
3) Acceleration constants $c_1 = c_2 = 2.0$
4) Inertia weight is decreased linearly fromthe 0.9 in the first
Iteration to 0.4 in the last iteration;
5) Limits on velocities: $v_{\max} = 10$ and $v_{\min} = -10$
6) Limits on particle positions: $X_{\min} = 0$ and $X_{\max} = 255$

GFA:
1) Number of traits in each individual=2
2) Upper limit of a trait HIGHTRAIT= [30 30]
3) Lower limit of a trait LOWTRAIT= [0 0]
4) Number of genes in each trait=6
5) Order of magnitude the trait=2
6) Probability of crossover CROSS_PROB=0.6

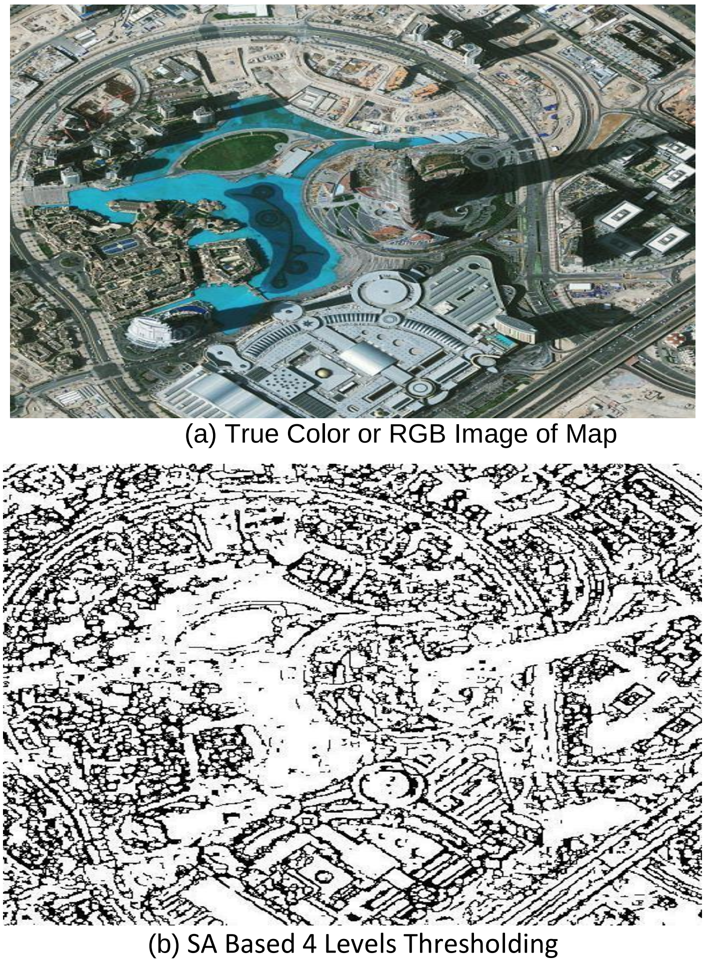
1) Case Study I :

SA and GFA are used to determine the optimal threshold level used to dichotomize the given image. The output of optimal thresholding of the *lake image* can be shown in Figure 1. This can be seen from the figure that the dark circles are clearly visible in SA while in GFA, it is little fade. In case study first, around 30 experiments are done by changing the no. of individuals in the case of GFA and no. of swarms in the case of SA and it is found that the fitness value from GFA is better than SA for different individuals/swarms.10 experiments are shown in table no.1 and Figure 2.

1) Case Study II :

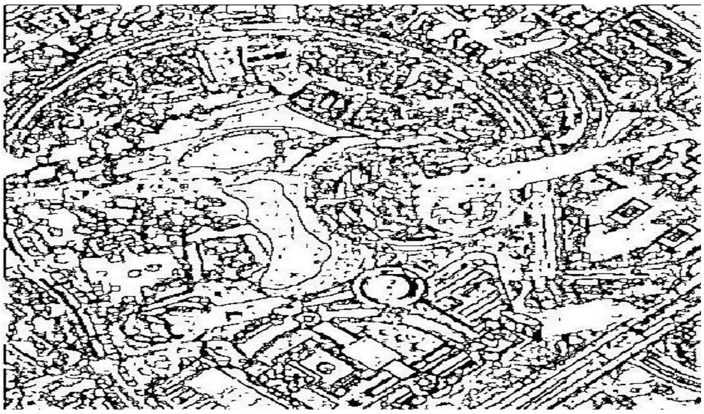
In the second case. About 30 experiments are performed and result is evaluated between the number of dusts or number of iterations versus fitness function. The graph shows the fitness value for both GFA and BFA. As shown in Figure 3. The BFA increase fast and becomes almost continuous after some iteration. This can be understand from the flock of birds, as one bird always be a leader in the flock and the leader is decided on the knowledge that which bird knows the place better and so on this pace changes so we get a good fitness value all the time While in the case of GFA it increases steeply with dusts

It is quite obvious because it follows Darwin principle "survival for the fittest". That means at every generation it eliminated the bad ones so after generation and generation we keep forward to the best value.



(a) True Color or RGB Image of Map

(b) SA Based 4 Levels Thresholding



(c) GFA Based 4 Level Thresholding

Figure 1. Results of SA and GFA based Colour Image Thresholding (Grey Levels are shown)

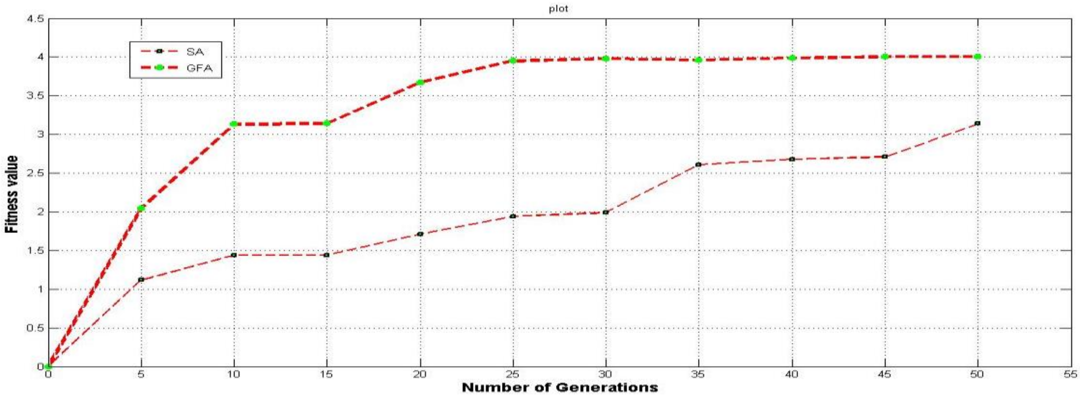


Figure 1. Plot of Number of Individuals or Number of Iterations versus the Fitness Value

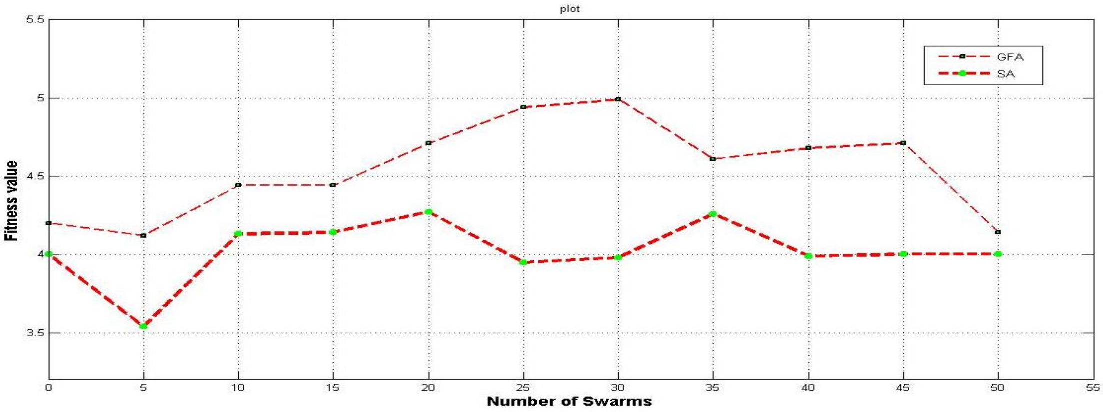


Figure 2. Plot of Number of Swarms or Number of Dusts versus the Fitness Value

Table 1. Summary of Results of SA and GFA Search for Optimal Multilevel Thresholding

Number of swarms/dusts	Fitness value	
	SA	GFA
20	4.442	4.584
25	5.003	3.983
30	5.068	3.983
35	5.256	3.889
40	5.000	4.117
45	4.889	4.017
50	5.002	4.264
55	4.783	4.389
60	4.941	4.026
65	4.888	4.292
70	4.885	4.337
75	4.759	4.338

Table II. Summary of Results of SA and GFA Search for the Image Optimal Multilevel Thresholding

Number of iterations	Fitness value	
	SA	GFA
0	0	0
5	1.24	1.99
10	1.54	2.13
15	1.56	3.24
20	2.03	3.75
25	2.74	3.20
30	3.03	4.06
35	3.36	4.24
40	3.54	4.39
45	3.71	4.34
50	4.14	4.36

V. Conclusion and Future Work

Bio inspired algorithms SA and GFA have been presented in this paper for segmentation of true color terrain images for autonomous deployment of WSN nodes from a modern UAV this tasks is treated as multidimensional optimization Problems and solved using the bio inspired algorithms. The algorithms have been briefly outlined and a statistical summary of their results is presented Segmentation based autonomous deployment presented in the paper reduces the number of sensor nodes deployed in the terrains of no interest. The algorithms proposed for multilevel thresholding are observed to be accurate than the exhaustive search for optimal thresholds [15]. The results show that the proposed algorithms have a tradeoff issue. While SA is more continuous but gives less fitness value, GFA gives better fitness value with no. of individuals so this algorithm should be used according to its application. If no. of individuals are more use GFA otherwise SA. A judicial choice between the algorithms depends on the desired localization accuracy and the desired quickness of localization be easily investigated. In future, SA and GFA can be applied for centralized localization in order to compare with the distributed localization method presented in this paper. Such a comparison with an emphasis on energy awareness is particularly very useful. Besides, a comparison of the stochastic localization methods with the available deterministic methods can give a useful insight. The analysis presented shows that localization errors propagate as iterations progress. The control of error propagation is a potential and important direction further study.

References

[1] R V. Kulkarni, Senior Member, IEEE, and GFAnesh Kumar VenayaGFAmoorthy, Senior Member, IEEE”,” Bio-inspired Algorithms for Autonomous Deployment and Localization of Sensor Nodes”, IEEE transactions on systems, man, and cybernetics—part c: applications and reviews (to appear in issue 5, volume 40, **2010**)

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “A surveyon sensor networks,” IEEE Commun. Mag., vol. 40, no. 8, Aug. **(2002)**,pp. 102–114.

[3] P. Corke, S. Hrabar, R. Peterson, D. Rus, S. Sampalli, and G. Sukhatme, “Autonomous deployment and repair of a sensor network using an unmanned aerial vehicle,” in Proc. IEEE Int. Conf. Robot. Autom., vol. 4, May **(2004)**, pp. 3602–3608.

[4] A. Ollero, M. Bernard, M. La Civita, L. van Hoesel, P. Marron, J. Lepley, and E. de Andres, “AWARE: Platform for autonomous self-deploying and operation of wireless sensor-actuator networks cooperating with unmanned aerial vehicles,” in Proc. IEEE Int. Workshop Saf., Secur. Rescue Robot. (SSRR), Sep. **(2007)**, pp. 1–6.

[5] A. Ollero and L. Merino, “Control and perception techniques for aerial robotics,” Annu. Rev. Control, vol. 28, May **(2004)**, pp. 167–178.

[6] M. Sezgin and B. Sankur, “Survey over image thresholding techniques and quantitative performance evaluation,” J. Electron.Imag., vol. 13, no. 1, Jan. **(2004)**, pp. 146–168.

[7] N. Otsu, “A threshold selection method from gray-level histograms,” IEEE Trans. Syst., Man Cybern., vol. SMC-9, no. 1, Jan. **(1979)**,pp. 62–66.

[8] G.S. Tomar and Shekhar Verma, “Dynamic multi-level hierarchal clustering approach for wireless sensor networks”, International Journal of Simulation- Systems, Science and Technology, vol.10, no. 4, May **(2009).** pp. 1-7.

[9] N. Patwari, 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, Jul. **(2005)**, pp. 54–69.

[10] J. Aspnes, 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, pp. 1663–1678, Dec. 2006.

[11] A. Boukerche, 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, Dec. **(2007)**, pp. 6–12.

[12] J. Hightower and G. Borriello, “Location systems for ubiquitous computing,” Computer, vol. 34, no. 8, Aug. **(2001)**, pp. 57–66.

[13] GS Tomar, Tripti Sharma, Brijesh Kumar, “Fuzzy based Ant colony Optimization Approach for Wireless Sensor Network”, Wireless Personal Communication, vol.84, no.1, May **(2015)**, pp.361-375.

[14] T. C. Liang, T. C. Wang, and Y. Ye, “A gradient search method to round the semidefinite programming relaxation solution for ad hoc wireless sensor network localization,” Stanford Univ., Stanford, CA, Tech. Rep., Dec. **(2004)**. Available: http://www.stanford.edu/~yyye/formal-report5.pdf

[15] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in Proc. IEEE Int. Conf. Neural Netw., Nov. 27–Dec. 1, **(1995)**, vol. 4, pp. 1942–1948.

[16] K. M. Passino, “Biomimicry of bacterial foraging for distributed optimization and control,” IEEE Control Syst. Mag., vol. 22, no. 3, Jun.**(2002)**,pp. 52–67.