Int. J. Applied Decision Sciences, Vol. X, No. Y, xxxx 1

Design of genetic algorithms for topology control of unmanned vehicles

Cem Şafak Şahin*, Elkin Urrea,
M. Ümit Uyar and Michael Conner

Department of Electrical Engineering,
138th Street and Convent Avenue,
City College of the City University of New York,
New York, NY 10031, USA
E-mail: csahin@gc.cuny.edu
E-mail: eurrea@gc.cuny.edu
E-mail: uyar@ccny.cuny.edu
E-mail: conner@ccny.cuny.edu
*Corresponding author

Giorgio Bertoli and Christian Pizzo

US Army Communications-Electronic RD&E Center,
Offensive Information Operations (OIO) Branch,
Ft. Monmouth, NJ 07703, USA
E-mail: giorgio.bertoli@us.army.mil
E-mail: Christian.pizzo@us.army.mil

Abstract: We present genetic algorithms (GAs) as a decentralised topology control mechanism distributed among active running software agents to achieve a uniform spread of terrestrial unmanned vehicles (UVs) over an unknown geographical area. This problem becomes more challenging under the harsh and bandwidth limited conditions of military applications. Using only local neighbour information, a GA guides each UV to select a ‘fitter’ speed and direction among exponentially large number of choices, converging towards a uniform node distribution. In an observed occurrence of a threat situation during a mission where UVs are to spread uniformly over an unknown terrain, if the number of UVs change with time (e.g., losing assets due to hostile forces), the remaining units should reposition themselves to compensate the loss in area coverage. Our simulation software results show that GAs can be an effective tool for providing a robust solution for topology control of UVs in military applications.

Keywords: genetic algorithms; GAs; artificial intelligence; mobile ad hoc network; MANET; bio-inspired algorithms; unmanned vehicles; UVs; topology control.

Reference to this paper should be made as follows: Şahin, C.S., Urrea, E., Uyar, M.Ü., Conner, M., Bertoli, G. and Pizzo, C. (xxxx) ‘Design of genetic algorithms for topology control of unmanned vehicles’, *Int. J. Applied Decision Sciences*, Vol. X, No. Y, pp.000–000.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **2010**	2. REPORT TYPE	3. DATES COVERED **00-00-2010 to 00-00-2010**

4. TITLE AND SUBTITLE **Design of genetic algorithms for topology control of unmanned vehicles**	5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Department of Electrical Engineering,City College of the City University of New York,,138th Street and Convent Avenue,New York,NY,10031**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR’S ACRONYM(S) 11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
We present genetic algorithms (GAs) as a decentralised topology control mechanism distributed among active running software agents to achieve a uniform spread of terrestrial unmanned vehicles (UVs) over an unknown geographical area. This problem becomes more challenging under the harsh and bandwidth limited conditions of military applications. Using only local neighbour information, a GA guides each UV to select a ?fitter? speed and direction among exponentially large number of choices, converging towards a uniform node distribution. In an observed occurrence of a threat situation during a mission where UVs are to spread uniformly over an unknown terrain if the number of UVs change with time (e.g., losing assets due to hostile forces), the remaining units should reposition themselves to compensate the loss in area coverage. Our simulation software results show that GAs can be an effective tool for providing a robust solution for topology control of UVs in military applications.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**18**	

Biographical notes: Cem Şafak Şahin received his BS degree from Gazi University, Turkey in 1996, MS degree from Middle East Technical University, Turkey in 2000, and MPhil degree from the City University of New York, USA in 2009, all in Electrical Engineering. He is currently a PhD candidate at the City College and the Graduate Center of the City University of New York.

Elkin Urrea received his BS in Electronics Engineering from Santo Tomas University, Bogota, Colombia, in 1998 and MSc in Electrical Engineering from the City College of the City University of New York (CUNY) in 2002. He is currently a PhD candidate at the Graduate Center, CUNY. His interests include wireless mobile ad hoc networks, distributed mobile robotics and communication networks.

M. Ümit Uyar is currently with the City College and the Graduate Center of the City University of New York. He leads a group of researchers in bio-inspired computing. He was a Distinguished Member of Technical Staff at AT&T Bell Labs until 1993. He has a BS degree from Istanbul Technical University, and MS and PhD degrees from Cornell University, all in Electrical Engineering.

Michael Conner received his BSEE degree from Johns Hopkins University in 1971, and MSEE and PhD degrees from University of Maryland in 1974 and 1976. He has held a variety of positions at a number of institutions in government, industry and academia including Harry Diamond Laboratories (now ARL), AAI Corp., Bell Telephone Laboratories (now Lucent), and for the last 25 years, City College of New York (CUNY). He holds two US patents.

Giorgio Bertoli (CISSP) is an Electrical Engineer and Branch Chief for the Intelligence and Information Warfare Directorate, Communications-Electronics Research Development and Engineering Center, US Army Research Development and Engineering Command. He has a Bachelor and Masters Degrees in Electrical Engineering from New Jersey Institute of Technology and a second Masters Degree in Computer Science from the University of Massachusetts.

Christian Pizzo is an Electronics Engineer and Researcher at the Intelligence and Information Warfare Directorate (I2WD) at Fort Monmouth, NJ. He received his Bachelors in Computer Science and Engineering from Bucknell University, Pennsylvania and is a Certified Information Systems Security Professional (CISSP).

1 Introduction

Uniformly spreading autonomous mobile agents over an unknown terrain operating under the harsh conditions of military missions is a challenging task. To ensure military mission safety and success, it is desirable that mobile nodes such as terrestrial unmanned vehicles (UVs) are to employ local information in their limited wireless communication ranges without relying on global communication among mobile nodes, or the existence of a global controller. In our research, we use genetic algorithms (GAs) (Goldberg, 1989; Holland, 1995) as the tool to dynamically control a UVs speed and direction according to its local environment information (e.g., the number of neighbours, neighbours' locations, positions of the obstacles, etc.) so that a uniform node distribution over an unknown geographical terrain is obtained.

Many military and commercial applications, such as search and rescue missions, minefield clearing, and self-spreading of assets under bandwidth-limited conditions, operating in difficult-to-access areas (e.g., building rubble due to an earthquake or obscured vision due to smoke), require uniform distribution of autonomous mobile nodes controlled by active running software agents over an unknown area. In these applications, a number of UVs can gather information from multiple viewpoints simultaneously, allowing the system of mobile agents to share information and understand the environment quickly and comprehensively. For example, a group of mobile agents equipped with video cameras could be sent into a disaster site, where the agents, running GA-based topology control algorithms, uniformly deploy themselves to cover the unknown terrain. While UVs provide video transmission, some of them may be physically blocked by debris or may fail to operate, in which case the other agents should re-position themselves to compensate for the lost area coverage. Note that the terms node, mobile node, mobile agent, and UV are used interchangeably throughout this paper to refer to the physical mobile entities in a mobile ad hoc network (MANET), each of which is running a bio-inspired topology control algorithm.

The topology control of UVs using a decentralised solution over an unknown geographical terrain is a challenging problem since:

1. the geographical area may change dramatically in a short time-span during an operation
2. the number of mobile agents may change (increase or decrease) dynamically
3. mobile agents do not have access to navigation maps nor to GPS devices but can only have limited information from local neighbours
4. mobile agents are typically deployed into the terrain from a single entry point (more difficult to analyse than random or other types of initial distributions often seen in existing research).

Using the results of our earlier research introducing a GA-based approach for topology control problems in MANETs (Sahin et al., 2008; Urrea et al., 2009), we present here a force-based GA (FGA) as a decentralised topology control mechanism distributed among active running software agents to achieve a uniform spread of UVs. Using only local neighbour information, a GA guides each UV to select a ‘fitter’ speed and direction among exponentially large number of choices, converging towards a uniform node distribution over an unknown geographical area. Consider a mission where UVs are to spread uniformly over an unknown terrain; in an observed occurrence of a threat situation, if the number of UVs changes with time (e.g., losing assets due to hostile forces or equipment malfunction), the remaining units should reposition themselves to compensate the loss in area coverage. Our simulation software results show that GAs can be an effective tool for providing a robust solution for topology control of UVs.

The rest of this paper is organised as follows. In Section 2, we review prior research on the use of GAs on mobile agent deployment, target localisation in MANETs, swarm robotics, and statistical methods to extract probability distribution from unknown data set. In Section 3, we discuss the general properties of GAs and our distributed FGA. Section 4 and Section 5 present our simulation software and simulation experiment results for different configurations, respectively.

2 Related work

Our FGA is inspired by the force-based node distribution in physics where each molecule attempts to remain in a balanced position and to spend minimum energy to protect its own position (Heo and Varshney, 2003). We used the discrete-time random walk model from Hokelek et al. (2008) to calculate the mean node degree for the hexagonal area in Urrea et al. (2007, 2008). The results from Sahin et al. (2008) are used for the FGA to calculate the fitness function for different numbers of mobile nodes in a hexagonal lattice.

GAs have proven to be an efficient tool in different distributed robotic applications. In Chen and Zalzala (1995), a genetic approach is presented with distance-safety criteria for a mobile robot motion. An adaptive GA is proposed by Shinchi et al. (2000) to identify targets while avoiding obstacles. In Moreno et al. (2002), mobile robots are used with ultrasonic sensors to collect range-limited data from the environment. Unlike our FGA, this approach finds an optimal position using global map knowledge.

Swarm robotics is another approach of using multi-robot systems collaboratively instead of single complex robots. Among several promising results, e.g., Tuci et al. (2006) illustrates a complex transporting problem requiring collaboration for small robots. Li et al. (2009) uses quantum probability in the chromosome coding strategy to adapt coalition formation into multi-robot systems. In Hsiang et al. (2003), an algorithm for distributing a swarm of primitive robots in an unknown geographical area is proposed. A hierarchical behaviour-based model in which several parameters are adjusted with a GA for tuning the parameters of a swarm to surround a target is proposed in Soto and Lin (2005). Naghsh et al. (2008) outlines the interactive use of autonomous robots and human beings in fire emergency settings. This study shows that a swarm of robots which are capable of working in fire fighting operations. Dependability, robustness, and reliability of the swarm-based systems for distributed safety critical systems are discussed in Winfield et al. (2006).

Our FGA for self-spreading mobile nodes in a MANET differs from the cited approaches above in several aspects. We perform uniform distribution of nodes and their reconfiguration due to asset losses by using a very limited knowledge obtained from the neighbouring UVs. Another significant difference is that we assume no prior knowledge of the geographical area or the positions of the obstacles. Finally, our FGA is very resilient to agent losses since it is fully distributed without a central controller or any other type of privileged entities, and each agent uses only the local information available.

3 Our distributed forced-based GA

GAs are a class of stochastic search algorithms forming a subset of bio-inspired computation algorithms. GAs mimic the way biological trait information is transferred and improved under selection pressure. The desired phenotype traits (that is, those of individuals) are selected by the evaluation of a specified fitness (objective) function. Individuals with a higher objective function score are more likely to be selected for breeding process by the GA. The principal of survival of the fittest is the starting point of GAs, which is typically applied to problems where deterministic or heuristic methods are not present or cannot provide satisfactory results. GAs essentially is composed of a set of individual chromosomes (called the population) and biologically inspired operators that create a new (and potentially better) population from an old one. According to the theory of evolution, only those individuals in a population who are better suited to the environment are likely to survive and generate offspring, thereby transmitting their superior genetic information to new generations (Holland, 1995; Mitchell, 1998).

A GA is an iterative optimisation procedure. Instead of working with a single candidate solution in each iteration, it works with a number of candidate solutions (i.e., a population). Another important property of GAs is that it can work effectively with a randomly populated initial population in cases where there is a lack of information about the problem space.

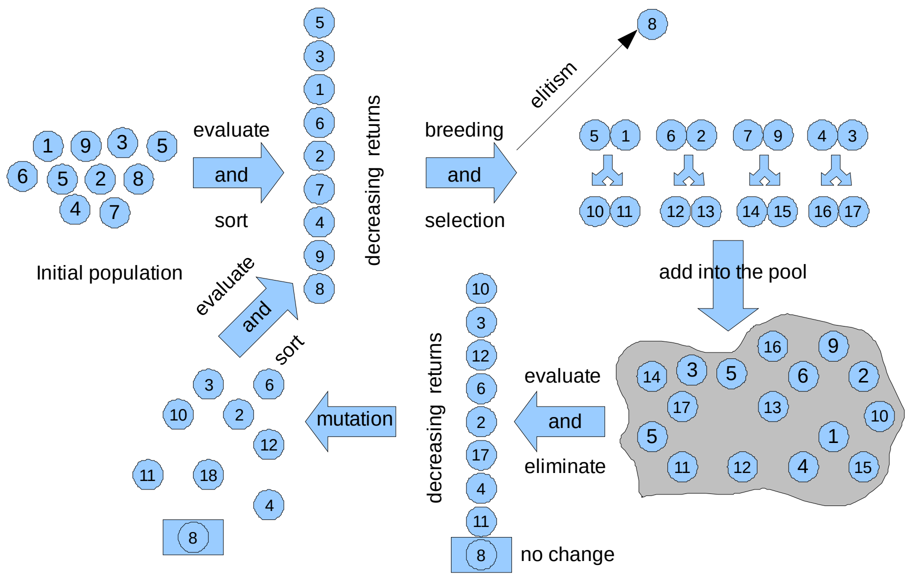
GAs are widely used for many applications from industry to research and development, including optimisation problems (e.g., data fitting, path finding, network traffic matrix calculation, etc.), financial problems (e.g., forecasting for stock market or gold prices, portfolio management, etc.), business management applications (e.g., scheduling, project management, task assignment), engineering problems (e.g., communication network design and optimisation, circuit boards design and optimisation, solving complex electromagnetic problems), and research and development applications (e.g., molecular and DNA modelling, curve fitting, etc.). However, one must be aware of the advantages and risks of applying GAs to a problem. The main advantage of GAs is their ability to quickly scan large problem spaces. The nature of GA provides a quick elimination of unsuitable solutions in the interested search space. On the other hand, one drawback of GAs is the need of computational power. The chromosome set and the objective function must be selected carefully because of their significant effects on the computational time and convergence to a final result.

Inheritance, selection, mutation, and crossover are the main operators for GA implementations (Goldberg, 1989; Sivanandam and Deepa, 2008). A tournament is the most well-known selection methodology in GAs. It is performed between two individuals chosen from a population. Between the two selected ones, the winner of the tournament is the one with the better fitness, which is permitted to reproduce. Crossover is a genetic operator which carries the fitter chromosomes from one generation to the next. Two randomly chosen parents generate two new offspring by the crossover operation, therefore creating a new generation as explained in Sivanandam and Deepa (2008) and Urrea et al. (2007). A mutation operator is used to keep genetic diversity alive through generations, and prevents the GAs from getting stuck at local extreme points.

Because GAs are stochastic in nature, they are not guaranteed to find the exact optimal solution. In practice, though, GAs often provides powerful solutions to certain classes of problems which prove intractable or too computationally intensive to be solved using classical deterministic or heuristic techniques.

Figure 1 describes, in general terms, a simple GA process. First a population of N individuals is randomly generated and evaluated using a fitness function score. The population is then sorted by their fitness scores. Individuals are selected for breeding (i.e., tournament and crossover) with probabilities proportional to their fitness scores. The offspring are added to a pool composed of candidate solutions for a new population. The offspring in the pool are then evaluated and sorted again. Only the better performing individuals are accepted into the newly created population. Mutation occurs on randomly selected individuals except for the best individual in the population. The elite individual has the best fitness value in the previous population and is typically chosen for the newly created population without any genetic operator selection. Therefore, the fitness score in the new population is better than, or at least the same as, the previous one. The population undergoes this process (without the initialisation step where chromosomes are chosen randomly) for many generations until some termination criterion is satisfied (e.g., convergence tolerance of the best individuals reaching a certain preset limit, satisfying a predefined fitness value, or reaching a limit on the number of generations).

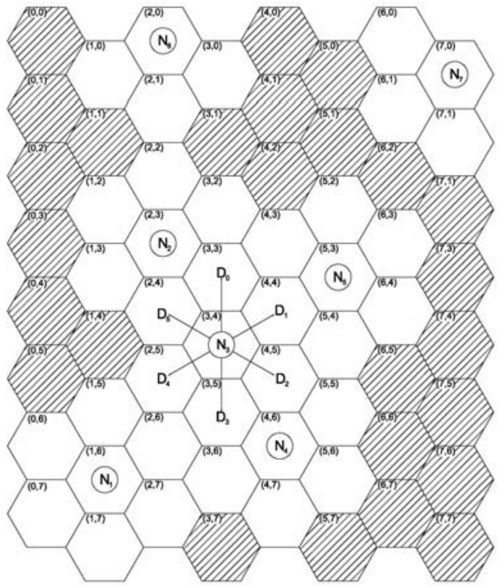
Figure 1 An example of operation flow in a GA (see online version for colours)



In our approach, a two-dimensional geographical area of ($d_{max} \times d_{max}$) is composed of logical hexagonal cells, where a unique Cartesian coordinate pair (x, y) is assigned to each one of the cells. For example, in Figure 2, there are 64 hexagonal cells and seven mobile nodes, each of which can move into six different directions (i.e., D_0 through D_5). A wireless communication link between two mobile stations is represented by a vector whose dimensions are in terms of layers. One layer is equal to the centre-to-centre distance between two neighbouring cells. In general, for a mobile node in location (0, 0) and another mobile node in location (x, y), the link state between these nodes is <x, y> (i.e., <x–0, y–0>). For example, in Figure 2, for a mobile node N_3 in location (3, 4) and another mobile node N_1 in location (1, 6), the vector representing wireless link between these nodes is <2, –2>.

A wireless link with the state of <x, y> ($0 < x, y < d_{max}$) between two mobile stations is called available if two nodes are communicating with each other; otherwise the link is said to be unavailable. If R_{com} is a positive integer representing the communication range of a node, and R is the centre-to-centre distance between two neighbouring cells, a wireless link can be available only if $R \leq R_{com}$. The number of available links of a node is called its degree. In Figure 2, N_3 communicates with N_1, N_2, N_4, and N_5 if $R_{com} = 3$; hence, the degree of N_3 is 4 for $R_{com} = 3$ (the cells that are not within the transmission range of any MANET node are coloured in grey).

Figure 2 An 8 × 8 hexagonal area partitioned into logical cells (R_{com} = 3)



In our model, each mobile node can move into one of the six neighbouring cells directed by its running software agent within the area boundaries. A mobile node uses the total force applied to it by the neighbouring nodes located in its communication range to decide the next direction and speed. In a similar approach given in Heo and Varshney (2003), where each molecule attempts to remain in a balanced position and to spend minimum energy to protect its own position, the optimal location is deterministically calculated as:

$$xf_n^{i,j} = \frac{D}{\mu^2}(R_{com} - |\, p_n^i - p_n^j \,|)\frac{p_n^i - p_n^j}{|\, p_n^i - p_n^j \,|} \tag{1}$$

where μ is the expected density (the average number of nodes required to cover an entire area when the mobile nodes are deployed uniformly), D is the local density (the number of nodes within the communication range), and p_n^i is the location of ith node at time step n.

The force between two nodes depends on the distance between them and the number of other nodes within their communication range (i.e., the force from a closer neighbour is greater than the farther one). The total force applied to a node by its neighbours can be used as the fitness of the corresponding mobile node. A smaller fitness shows a better position for a mobile node since it indicates that the total force applied to it by its neighbours cancels each other. The mean node degree ($\overline{N}$) is shown to be an effective measure indicating the number of neighbours to construct a fitness function for a given total number of nodes, communication range, and a geographical area as shown by Urrea et al. (2009) and Hokelek et al. (2008). $\overline{N}$ is the expected number of neighbours to maximise the coverage. Therefore, using $\overline{N}$, we can calculate the total force applied to a mobile node n as follows:

$$F(n) = \sum_{i=0}^{k} \sum_{j=0}^{k} \overline{N}.(R_{com} - |((x - x_i) + (y - y_j))| \quad (2)$$

where k is the total number of neighbours, (x, y) is the current coordinate value for the node, and (x_i, y_j) is the location of a neighbour node.

In our FGA, a mobile node gathers information about its neighbouring nodes' speed, direction, and location, and then, using the fitness function defined in equation (2), proceeds to run the FGA to generate new chromosomes representing candidate solutions for the next generation. These candidates are ordered according to their fitness values from the lowest to the highest. The lowest fitness corresponds to the solution representing the least amount of force applied to a node, and hence, the best one among the candidate solutions for that generation.

The software agent in each mobile node runs FGA for g generations, and then selects the chromosome with the best fitness value. The goal of each mobile node is to find a location where it is at equilibrium with respect to the total force applied to it by its neighbours. Suppose each node checks r possible positions and speeds for each generation. A mobile node ends up checking a total of $(r.\ g)$ possible outcomes for g generations. Let us assume that the nodes can move at speeds up to v (i.e., it can move over v cells at a time). In this case, there are $(6.\ v)$ possible locations at the end of the first movement, $(36.\ v^2)$ for the second movement, and $(6.\ v)^g$ for the gth movement. In total, there are up to $\sum_{i=0}^{g} (6.v)^i$ possible locations after g movements. Instead of calculating each possible solution, each node runs the FGA for r chromosomes until reaching g iterations. Consequently, the node adapts the fittest solution at the final iteration as its next direction and speed as imposed by FGA.

4 GA simulation software

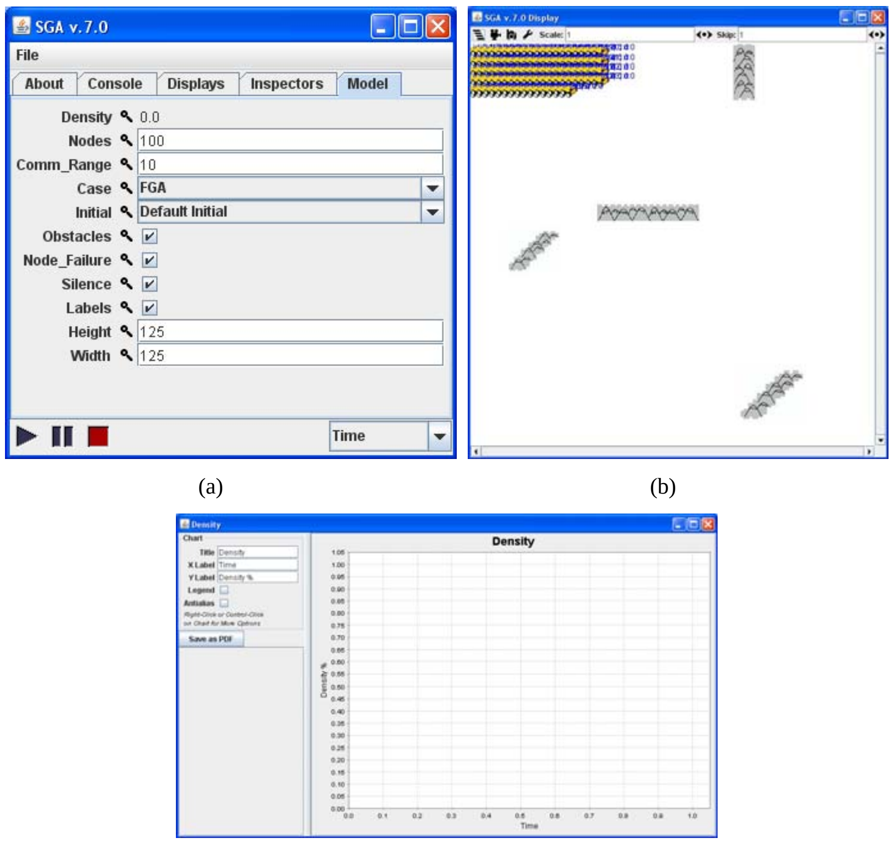
We implemented a simulation software system in Java to study the effectiveness of our distributed GA-based algorithms for a uniform distribution of knowledge sharing agents. Eclipse SDK© version 3.2.0 was used as the development environment, and Mason, a fast discrete-event multi-agent simulation library core developed by George Mason University ECJLab, was used for the GUI interface.

The simulation software implementation has more than 4,000 lines of algorithmic Java code. To avoid possible inefficiencies, we developed our algorithms without using any existing GA libraries. Our design philosophy was to build a GA-based application to which a programmer can easily add new features (e.g., different types of crossover, or different rules for tournament, etc.) and new evolutionary computation approaches. Our simulation software was implemented such that it runs as a multi-agent application which imitates a real-time topology control scenario. As a result, the observations from our simulation software match closely to those from the real testbed experiments reported by Dogan et al. (2009).

In our simulation software, a user can provide inputs for the following parameters:

1. N: total number of mobile nodes
2. R_{com}: communication range
3. T_{max}: maximum number of iterations
4. type of initial deployment
5. d_{max}: size of the geographical terrain
6. type of GA-based application.

Figure 3 Graphical user interface for our GA simulation software system (see online version for colours)



(a) (b)

(c)

Sample screen shots of our user interface are shown in Figure 3. In Figure 3(b) an initial deployment of 100 UVs starting from the north-west sector is displayed. Each UV can only communicate with the other mobile agents located within its range of communication, R_{com}. Currently, there are three different initial deployment strategies for the mobile nodes:

1 start from the north-west corner

2 place the nodes randomly over the terrain

3 start from a given coordinate (e.g., the centre).

The north-west initial deployment represents a more realistic approach of the topology control problem for the UVs compared to the other deployment possibilities over an unknown terrain. For example, in an earthquake rescue, mine clearing, a military mission in a hostile area, or a surveillance operation, all mobile nodes may be forced to enter the operation area from the same vicinity.

The software also has the ability to run experiments using an initial mobile node distribution with a given set of initial conditions that have already been created for previous runs (i.e., the initial data for each mobile agent includes a starting coordinate, speed, and direction). This ability is important since each experiment is repeated many times to eliminate the noise in the collected data. The mobile agents can move with one of three different speeds (i.e., fast, slow, or immobile) to any of the six possible directions in the hexagonal lattice. Our simulation software also provides the capability for a user to collect data without using GUI. Figure 3(c) shows the resulting network area coverage (NAC) values in real time to provide information about the performance of the GA-based algorithm.

5 Simulation experiment results

To evaluate the performance of our FGA, we consider a scenario in which a team of UVs enter an unknown geographical area without any prior information and a global control unit. Each UV has a limited communication range (R_{com}), and, hence, can only be aware of its neighbours and runs its own GA-based software application. Our main target is to keep the network fully connected among the mobile agents while covering the geographical terrain uniformly under realistic conditions such as arbitrary obstacles in the terrain, stoppages due to malfunctions and hostile attacks toward one or more mobile nodes (i.e., either isolated or concentrated losses). Our distributed FGA aims to provide each node with a near-optimal number of neighbours.

The behaviour of our distributed FGA may be modelled statistically so that we can extract the patterns in the data collected from simulation experiments. This model then can be a useful guide to predict behaviour of our FGA for similar experiments.

Statistical model is defined as a set of mathematical equations describing the behaviour of an object study in terms of random variables and their associated probability distributions. It is mathematically thought of as a pair of (X, P) where $X = (x_1, x_2,..., x_n)$ n is the number of observations is the set of possible observations and P is the set of possible probability distribution on X (McCullagh, 2002).

The data set includes NAC values for different number of UVs (N = 100, 200, and 300) with a fixed maximum communication distance (R_{com} = 10) for d_{max} = 125. The basic statistical model of NAC is obtained using statistical inference which deals with the problem of inferring properties of an unknown distribution from the data generated by that distribution. All probability distribution functions for continuous random variables have a form of:

$$\frac{1}{c.A_r(s)}.\Theta(\frac{x-l}{c}) \quad (3)$$

where $a \leq x \leq b$, l is range, a and b are integers, $\Theta(s)$ is the actual shape of the probability density function (pdf), $A_r(s)$ is the area under the function (i.e., l represents the location parameter which has the effect of translating the pdf or x-axis, c is the scale parameter that expands the scale of x-axis, and s is the shape parameter governing the actual shape of the characteristic function, Θ).

Scale parameter is also named as *measures of dispersion*. This metric presents information about how 'spread out' the values are around the central tendency of the random variables. Let us assume that we have a functional $\tau(F)$ (also denoted by $\tau(X)$ when X is a random variable with distribution F) defined over a sufficiently large class of symmetric distributions which is closed under changes of location and scale. We shall require τ to be non-negative and to satisfy:

$$\tau(aX) = |a|.\tau(x) \textbf{ for } a > \mathbf{0} \textbf{ and } \tau(X+b) = \tau(x) \textbf{ for all } b \quad (4)$$

A non-negative functional τ satisfying equation (4) will be called as a measure of dispersion if and only if it satisfies in addition $\tau(F) \leq \tau(G)$ whenever G is more dispersed than F as given by Bickel and Lehmann (1976).

Shape parameter is also called as *measures of shape*. It is calculated as

$$\gamma_1 = \frac{\mu_3}{\mu_2^{3/2}},$$

where μ_i is the ith central moment.

Each of our simulation experiments was run for T_{max} = 700 time units, and was repeated for 50 times so as to avoid transient results from the natural non-deterministic behaviour of our distributed FGA. At the beginning of each experiment all UVs were located at the north-west corner of the given geographical terrain as seen in Figure 3(b). In order to get a fair comparison for experiments with different number of nodes, the communication range and the hexagonal area were fixed at R_{com} = 10 and 15, 625 cells (*125 × 125*), respectively.

To evaluate the performance and effectiveness of our distributed FGA, we execute the combination of two types of military applications. First UVs are deployed in a hostile region where some mobile nodes can be lost during and after deployment. In this application, they are two events that affect the deployment of the mobile agents. Some nodes can lose their communication functionality due to malfunction, whereas others are destroyed due to enemy attacks. These affected nodes are considered out of the experiments after that point. As a consequence of such as events, the remaining nodes must reconfigure their positions to compensate the missing area coverage due to lost team members.

In a second application, mobile agents intentionally stop communicating with the neighbouring nodes during short periods of time to go unnoticed by adversary forces and avoid being the target of enemy attacks. Following such silent modes, all mobile agents resume transitions again. This military scenario was applied into our simulation software as follows. The UVs did not use any communication with their neighbours at the beginning of their mission to ensure that there are no hostile forces in the area they entered. After T = 100, the mobile agents start communicating with their mobile neighbours located in their communication range, so that the FGA was initiated by each UV.

Figure 4 shows the frequency of the normalised NAC for N = 100 UVs from T = 0 to T = 700. The improvement in NAC through the time for the experiment as the nodes perform our distributed FGA can be easily seen in Figure 4. Initially, NAC is very low since all UVs are at their initial positions of north-west corner of the area [Figure 4(a)]. The frequency is 50 for T = 0, meaning that all 50 experiments start with a low NAC value. As time progresses and coverage improves, frequency of higher NAC values is observed. For example, in Figure 4(b), higher NAC values are achieved by more experiments at time T = 400. Since there are not enough UVs to cover the geographical area for N = 100, NAC value and its frequency do not improve from T = 400 to T = 600. Figure 4(d) represents the frequency of NAC after two hostile attacks.

For the experiments shown in Figure 4(a)–Figure 4(d), the parameters of scale, location, and skew are shown in Figure 5(a)–Figure 5(c), respectively. The standard deviation (σ) reaches its highest value at T ≈ 230 in Figure 5(a). However, σ is always high for the case of 100 nodes. Figure 5(b) illustrates the average network coverage in percentage for our FGA. The boost in NAC can be seen after T = 100 when our FGA starts running. Furthermore, after each hostile attack at T = 400 and T = 600, the operational UVs successfully readjusted their position and speed to reoccupy the area under attack. There is no obvious effect of malfunctioned nodes in Figure 5(a)–Figure 5(c). The skew is high in Figure 5(c). In fact, since we cannot see any known distribution in Figure 4(a)–Figure 4(d), Figure 5(a) and Figure 5(c) have large oscillations. We can confirm that 100 nodes were not enough to uniformly cover the terrain. However, our FGA performed well to obtain the maximum coverage after each hostile attack.

The frequency of NAC for N = 200 nodes from T= 0 to 700 are displayed in Figure 6(a)–Figure 6(d). The UVs reach the maximum area coverage at time T = 400 as the frequency of high NAC values are observed in Figure 6(b). A Gaussian distribution shape is observed after 50 runs. Figure 7(a)–Figure 7(c) support the normal distribution in Figure 6(a)–Figure 6(d). However, we still observe more deviation and skew than Gaussian distribution in Figure 7(a) and Figure 7(c). The UVs using our distributed FGA move successfully around the obstacles and reoccupy the empty areas after hostile attacks as seen from Figure 7(a) through Figure 7(c). Compared to the case of N = 100, we see much improved results for N = 200 since there are more UVs to increase the area coverage.

Figure 8 (a)–Figure 8(d) shows the results for N = 300 from T = 0 to T = 700. The UVs spread out the entire terrain less than T = 400 units. The figures show a normal distribution supported by Figure 9(a)–Figure 9(c). In Figure 9(a), σ converges to a stable value (≈0.5). After each hostile attack (T = 400 and T = 600), σ value increases indicating a non-Gaussian distribution, however, NAC goes back to the normal distribution after a certain time. Figure 9(c) represents the skew and indicates the quick recovery after each hostile activity. Since the largest number of UVs are used (N = 300), the overall best results are obtained for this case.

Figure 4 Frequency of network area coverage (NAC) for N = 100 (a) T = 0 (b) T = 400 (c) T = 600 (d) T = 700 (see online version for colours)

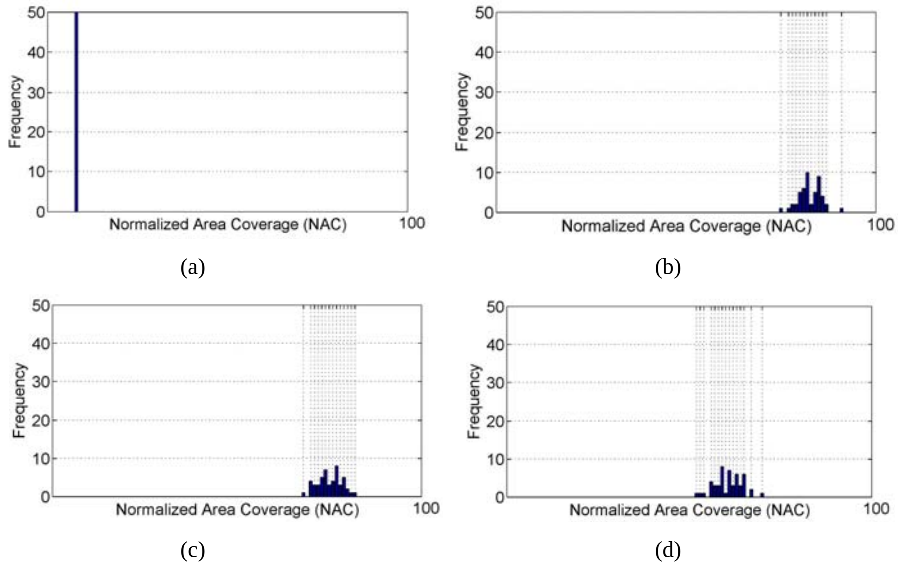


Figure 5 Standard deviation, mean, and skew for NAC experiments in Figure 4 (see online version for colours)

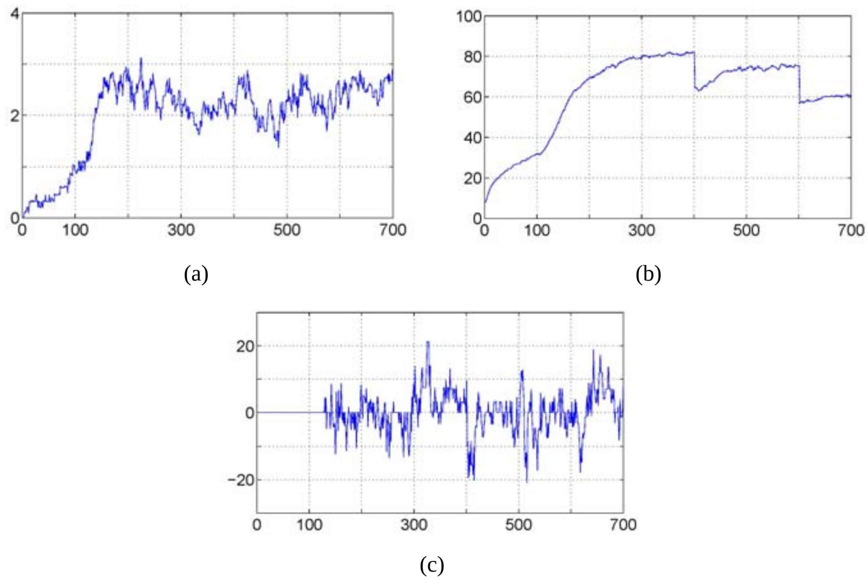


Figure 6 Frequency of network area coverage (NAC) for N = 200 (a) T = 0 (b) T = 400 (c) T = 600 (d) T =700 (see online version for colours)

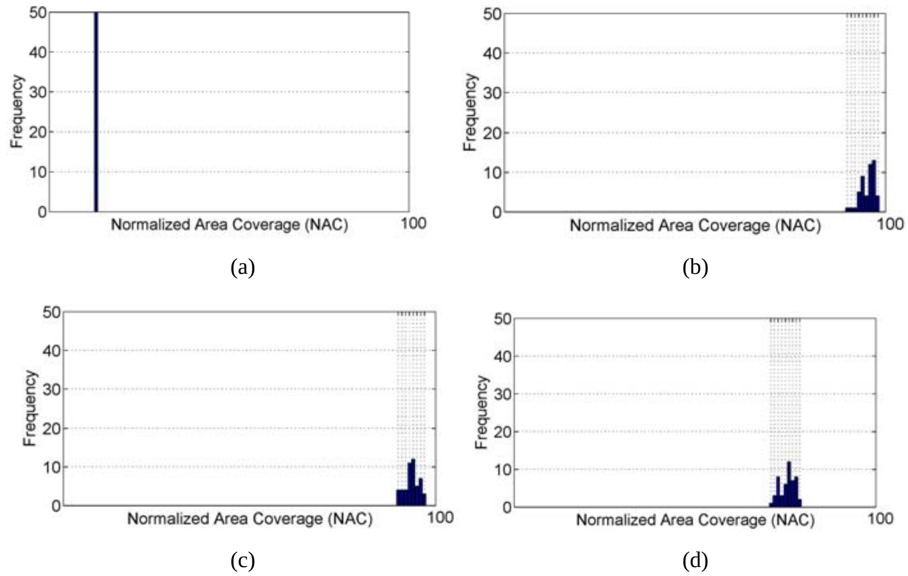


Figure 7 Standard deviation, mean, and skew for NAC experiments in Figure 6 (see online version for colours)

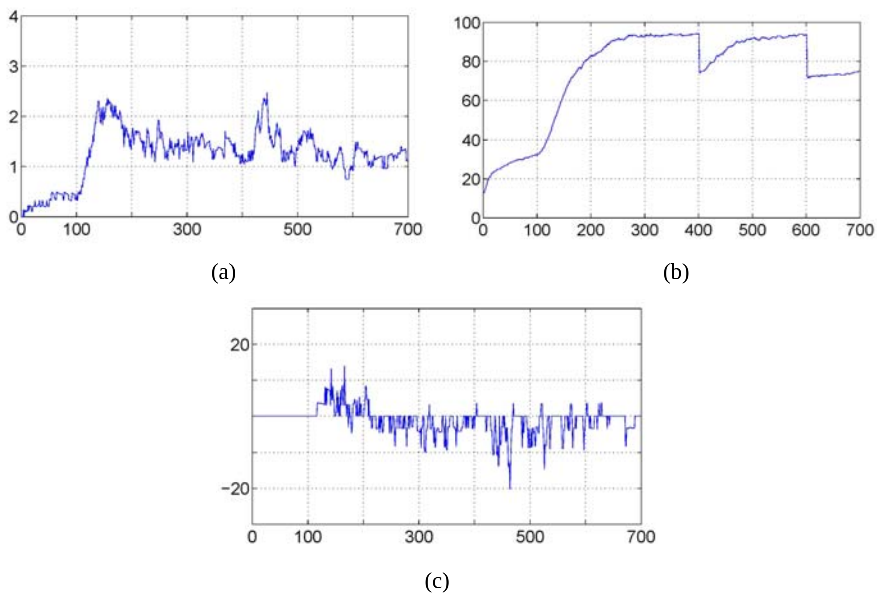


Figure 8 Frequency of network area coverage (NAC) for N = 300 (a) T = 0 (b) T = 400 (c) T = 600 (d) T =700 (see online version for colours)

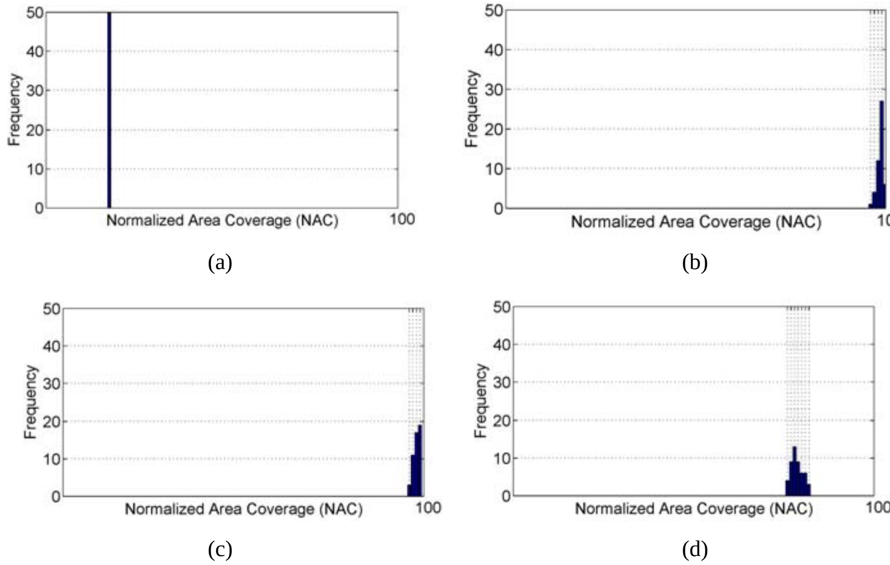


Figure 9 Standard deviation, mean, and skew for NAC experiments in Figure 8 (see online version for colours)

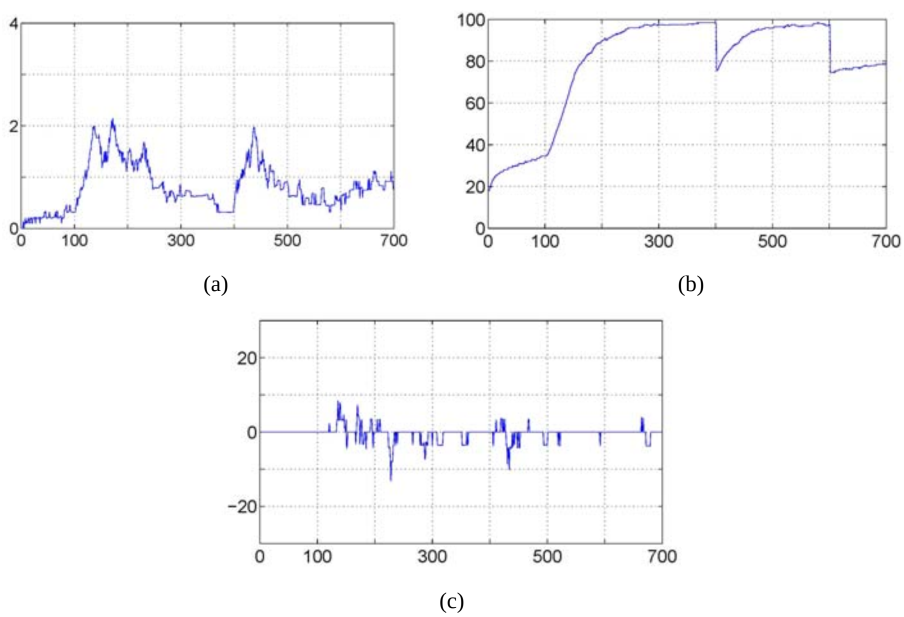
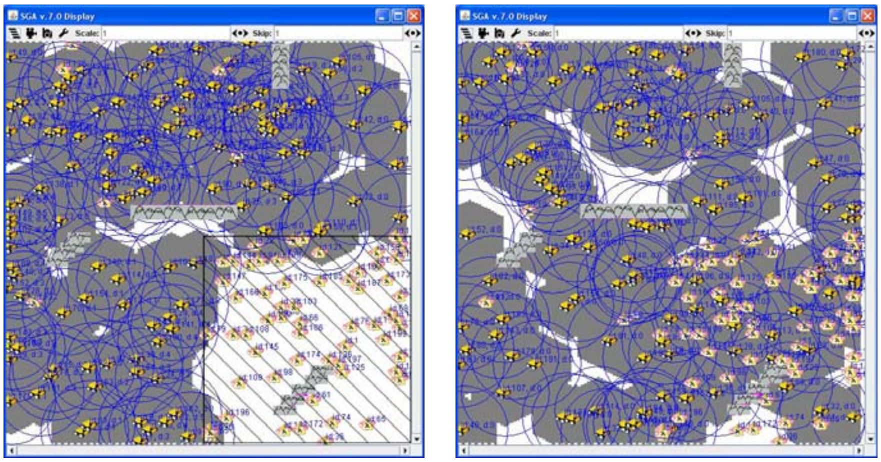


Figure 10 (a)–Figure 10(b) shows the screen captures after the second hostile attack at T = 600, and the recovery of NAC at T = 700 for N = 200 nodes. Large circles and UVs with a cross on them in Figure 10(a)–Figure 10(b) indicate the UVs communication range and failed/destroyed by hostile attacks, respectively. Dark grey regions represent the areas covered by at least one node. The crosshatch square represents the region where enemy attacks take place. At T = 401 and T = 601, the first and second hostile attacks destroy 25% of the UVs in the given geographical area, as shown in Figure 10(a). The remaining UVs keep performing FGA, and readjust their positions in less than 100 time units for a near uniform coverage as shown in Figure 10(b).

Figure 10 Screen shots after hostile attack and after recovery (a) T = 600 (b) T = 700 (see online version for colours)



(a) (b)

6 Concluding remarks

In this paper, we presented the application of GAs to topology control of UVs. Our FGA, inspired by the equilibrium of the molecules in physics, adjusts the speed and direction of each UV using only local information.

Two sets of military applications were considered as experimentation scenarios, namely hostile attacks, where considerable percentage of UVs become unavailable, and operation under silence mode, which requires intermittent stoppage of communication during a mission. Our simulation software results show that GAs can be an effective tool for providing a robust solution for topology control of UVs in both types of military applications.

Future work will include a formal analysis of FGA performance in various different operational conditions.

Acknowledgements

This work has been supported by US Army Communications-Electronics RD&E Center. The contents of this document represent the views of the authors and are not necessarily the official views of, or are endorsed by the US Government, Department of Defense, Department of the Army, or the US Army Communications-Electronics RD&E Center.

This work has been partially supported by the National Science Foundation grants ECS-0421159 and CNS-0619577.

References

Bickel, P.J. and Lehmann, E.L. (1976) 'Descriptive statistics for nonparametric models: III. Dispersion', *The Annals of Statistics*, Vol. 4, pp.1139–1158.

Chen, M. and Zalzala, A. (1995) 'Safety considerations in the optimization of the paths for mobile robots using genetic algorithms', *1st Intl. Conf. on Genetic Algorithms in Engineering Systems: Innovations and Applications*, pp.299–306, Sheffield, UK.

Dogan, C., Sahin, C.S., Uyar, M.U. and Urrea, E. (2009) 'Testbed for node communication in MANETs to uniformly cover unknown geographical terrain using genetic algorithms', *AHS'09: Proceedings of The NASA/ESA Conference on Adaptive Hardware and Systems*, pp.273–280, San Francisco, CA.

Goldberg, D.E. (1989) *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, New York, NY.

Heo, N. and Varshney, P.K. (2003) 'A distributed self spreading algorithm for mobile wireless sensor networks', *Wireless Communications and Networking*, Vol. 3, pp.1597–1602.

Hokelek, I., Uyar, M.U. and Fecko, M.A. (2008) 'A novel analytic model for virtual backbone stability in mobile ad hoc networks', *Wireless Networks*, Vol. 14, pp.87–102.

Holland, J.H. (1995) *Hidden Order: How Adaptation Builds Complexity*, Addison-Wesley, New York, NY.

Hsiang, T.R., Arkin, E.M., Bender, M.A., Fekete, S. and Mitchell, J.S.B. (2003) 'Online dispersion algorithms for swarms of robots', *Proceedings of the 19th Annual Symposium on Computational Geometry*, pp.382–383, San Diego, CA.

Li, Z., Xu, B., Yang, L., Chen, J. and Li, K. (2009) 'Quantum evolutionary algorithm for multi-robot coalition formation', *Proceedings of the 1st ACM/SIGEVO Summit on Genetic and Evolutionary Computation*, pp.295–302, Shanghai, China.

McCullagh, P. (2002) 'What is a statistical model?', *The Annals of Statistics*, Vol. 30, pp.1225–1310.

Mitchell, M. (1998) *An Introduction to Genetic Algorithms*, MIT Press, Boston, MA.

Moreno, L., Armingol, J.M., Garrido, S., Escalera, A. and Salichs, M. (2002) 'A genetic algorithm for mobile robot localization using ultrasonic sensors', *J. of Intelligence and Robotic Systems*, Vol. 34, pp.135–154.

Naghsh, A.M., Gancet, J., Tanoto, A. and Roast, C. (2008) 'Analysis and design of human-robot swarm interaction in firefighting', *Proceedings of the 17th IEEE International Symposium on Robot and Human Interactive Communication*, pp.255–260, Munich, Germany.

Sahin, C.S., Urrea, E., Uyar, M.U., Conner, M., Hokelek, I. Bertoli, G. and Pizzo, C. (2008) 'Genetic algorithms for self-spreading nodes in MANETs', *GECCO '08: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, pp.1141–1142, Atlanta, GA.

Shinchi, T., Tabuse, M., Kitazoe, T. and Todaka, A. (2003) 'Khepera robots applied to highway autonomous mobiles', *Artificial Life and Robotics*, Vol. 7, pp.118–123.

Sivanandam, S.N. and Deepa, S.N. (2008) *Introduction to Genetic Algorithms*, Springer-Verlag Berlin Heidelberg, New York, NY.

Soto, J. and Lin, K-C. (2005) 'Using genetic algorithms to evolve the control rules of a swarm of UAVs', *Proc. Intl. Symposium on Collaborative Technologies and Systems*, pp.359–365, St. Louis, MO.

Tuci, E., Gross, R., Trianni, V., Mondada, F., Bonani, M. and Dorigo, M. (2006) 'Cooperation through self-assembly in multi-robot systems', *ACM, Transactions on Autonomous and Adaptive Systems*, Vol. 1, pp.115–150.

Urrea, E., Sahin, C.S., Uyar, M.U., Conner, M., Hokelek, I. Bertoli, G. and Pizzo, C. (2007) 'Uniform MANET node distribution for mobile agents using genetic algorithms', *Proc. of Int. Conf. on Artificial Intelligence and Pattern Recognition*, pp.369–376, Orlando, FL.

Urrea, E., Sahin, C.S., Uyar, M.U., Conner, M., Hokelek, I. Bertoli, G. and Pizzo, C. (2008) 'Simulation experiments for knowledge sharing agents using genetic algorithms in MANETs', *Proc. of the 2008 Int. Conf. on Genetic and Evolutionary Methods*, pp.85–90, Las Vegas, NV.

Urrea, E., Sahin, C.S., Uyar, M.U., Conner, M., Hokelek, I. Bertoli, G. and Pizzo, C. (2009) 'Bio-inspired topology control for knowledge sharing mobile agents', *Mobile Ad Hoc Networks*, Elsevier, Special Issue on Bio-Inspired Computing, Vol. 7, No. 4, pp.677–689.

Winfield, A.F.T., Harper, C.J. and Nembrini, J. (2006) 'Towards the application of swarm intelligence in safety critical systems', *Proceedings of the 1st Institution of Engineering and Technology International Conference on the System Safety*, pp.89–95, London, UK.