Genetic Algorithm Strategies for Voronoi Classifier Navigation

Matthew Skalny
U.S. Army TARDEC
6501 E. 11 Mile Road
Warren, MI 48397
matthew.skalny@us.army.mil

Jim Overholt
U.S. Army TARDEC
6501 E. 11 Mile Road
Warren, MI 48397
jim.overholt@us.army.mil

Greg Hudas, Graham Fiorani
U.S. Army TARDEC
6501 E. 11 Mile Road
Warren, MI 48397
greg.hudas, graham.fiorani
@us.army.mil

ABSTRACT
There are many approaches to guiding robots in partially known environments, including waypoints, D*, and various other methods. In this paper, we describe a new method for robot navigation that uses navigation "beacons" called *Voronoi classifiers* to guide a robot to a goal area, and the application of a genetic algorithm for optimizing the placement of these classifiers. Our results show that a genetic algorithm (GA) can be a good way of placing the classifiers.

Categories and Subject Descriptors
I.2.9 [**Artificial Intelligence**]: Robotics – *autonomous vehicles, sensors*

General Terms
Algorithms, theory

Keywords
Voronoi classifier, robot navigation, dynamic waypoints, genetic algorithm

1. INTRODUCTION
Successful navigation of robots or groups of robots from some start point to some goal is critical for military robots. Some robotic applications that require accurate navigation schemes include robotic mules for delivering supplies, and using robots for perimeter search/patrol.

There are a number of challenges involved in robot navigation. Some of these challenges are path planning, robot localization, and sensor inaccuracy [1,2,3,4]. To address these issues, a number of methods have been developed and implemented.

Currently, waypoint navigation is the *de facto* standard for laying out a path for a robot to follow. The U.S. Army has demonstrated the capabilities of waypoint navigation (sometimes known as laying out 'electronic breadcrumbs') in several well publicized experiments (such as Demo III leader-follower activities [5]). However, waypoint algorithms can exhibit problems when unknown obstacles or dynamically changing terrain alters the path between consecutive points.

To solve the issues with waypoint navigation, a number of methods have been developed.

Arkin's potential field methods [6] use attractive and repulsive forces to guide a robot – obstacles "emit" repulsive forces, and attractive forces lead the robot towards the goal. A resultant force from the combination of the various forces provides the robot with its navigation instructions. This method breaks down when there are several obstacles in the field of interest, generating points of *singularity* or *null points* in the field [7].

Choset developed a path planning method based on computing all possible geometric center paths in cluttered environments. A tree search and pruning technique is then used to trim the all the paths down to the feasible ones. Any location can be optimally moved to as long as these paths are followed until the robot get as close as possible to the final desired point. This method combines SLAM concepts with computational geometry methods using *Generalized Voronoi Graphs* [8].

Stentz has developed a popular method called D* [9] that has been used in a number of applications. D* combines A* search with a local re-planner that updates an optimal path based on sensor data. While effective, this means that the robot could be spending a good deal of time recalculating optimal paths. In an obstacle rich environment, a more robust, non-optimal solution may be more appropriate.

Analysis of all of the above methods lead use to develop a new robot path planning and navigation method that uses *Voronoi classifiers* to provide navigation instructions to a robot or robots.

2. ROBOT PATH PLANNING USING VORONOI CLASSIFIERS
2.1 Introduction
We have previously presented our Voronoi classifier method for robot path planning and navigation, and have shown it to have the potential to be a more robust and easier to visualize than other current methods [10]. The basis for our method is the following problem statement:

Genetic and Evolutionary Computation Conference (GECCO) '05, June 25-29, 2005, Washington, DC, USA.

*Given a playing field **'F'** with a set of obstacles and a goal region **'G'** place a minimal set of **'navigation classifiers'** that will allow any feasible starting location in **'F'** to terminate at **'G'** within a finite number of steps **'K'** (see figure 1).*

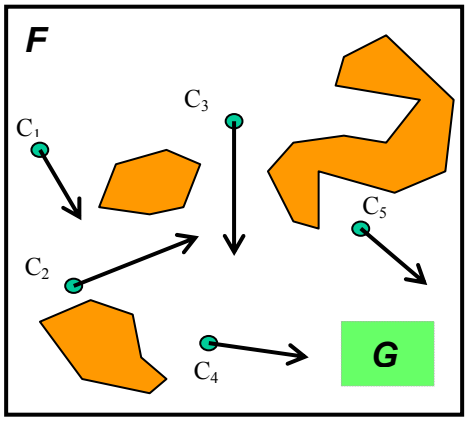

Figure 1: Playing field 'F' with obstacles and goal 'G'. Several *Voronoi classifiers* are shown.

The name *Voronoi classifier* comes from the fact that each navigation classifier C_i in figure 1 can be associated with a Voronoi region V_i. A Voronoi diagram [11] may be generated by any finite set of points (sites) in a plane. The partitioning of a plane with n sites into n convex polygons such that each polygon contains exactly one site and every point in a given polygon is closer to its site than to any other site yields a Voronoi diagram (see figure 2).

In the case of our Voronoi classifier navigation method, each site in figure 2 would be a classifier providing direction and distance to travel instructions to any robot that is requesting instruction while in the classifier's associated Voronoi region.

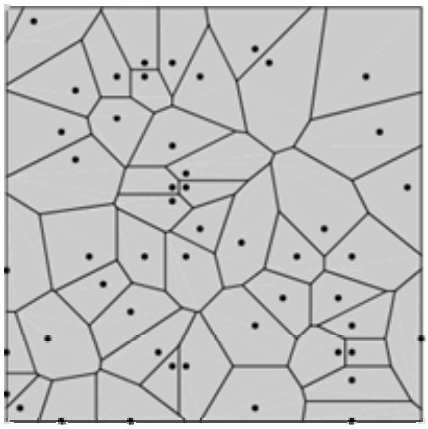

Figure 2: Example of a Voronoi diagram

2.2 Navigation Example using Voronoi Classifiers

Each Voronoi classifier can be viewed as a dynamic waypoint generator – it provides a direction and distance command to any robot within its region that is in need of instruction. This process continues until the robot either:

1) Reaches the goal,
2) Leaves the playing field, or
3) Exceeds some maximum limit on distance traveled or number of steps taken.

For example (see figure 3), a robot starting at b_0 will request and receive its first instruction from C_1, which tells it to go to b_1. In the case of path b, this process is repeated until the robot reaches the goal.

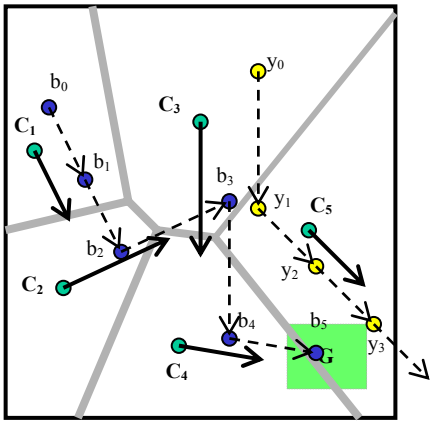

Figure 3: Trajectories on a playing field generated by *Voronoi Classifiers*. The blue trajectory terminates at the goal. The yellow trajectory leaves the playing field.

2.3 Optimizing Classifier Placement

Until now, we have used human placement for positioning the classifiers. This has been done through a software tool we developed using a concept called the *color map*. Figure 4 shows an example of a color map – green indicates areas from which a robot will be successfully directed to the goal, red indicates areas from which the robot will be directed to leave the playing field, and yellow indicates regions that do not lead the robot off the field and do not reach the goal within the number of steps specified.

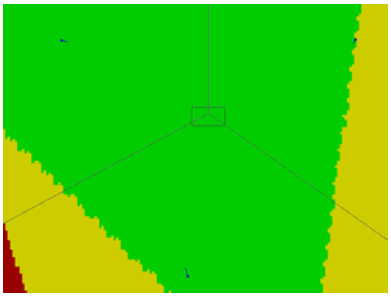

Figure 4: Example of Color Map, Goal in Center

Each colored region on the color map is based on the color for a sample point or points contained within that region. The goal of placing classifiers is to make the entire playing field green.

For an empty field, or even a field with few obstacles, it is not too difficult for a human to lay out a good set of classifiers. However, once the complexity of the playing field starts increasing, hundreds of classifiers may be needed to achieve the accuracy desired. This has led us to investigate automated methods for placing classifiers and determining classifier magnitude and direction parameters. The rest of this paper focuses on our research into the use of a genetic algorithm for optimizing classifier placement and magnitude and direction parameters.

3. CLASSIFIER PLACEMENT USING GENETIC ALGORITHM

3.1 Reason for using a Genetic Algorithm

We identified a genetic algorithm (GA) as a good optimization option for several reasons. These are:

1) A set of classifiers can be treated as a string of real values, where every four values on the string represent the x and y position of that classifier, and the magnitude and direction of the navigation instruction it provides.

2) We have not identified a solid mathematical expression for calculating how many steps a robot will take to get to a goal given some set of classifiers and a starting position. Each path is a linear combination of some non-negative integer number of each classifier's vector (direction + magnitude instruction), but the problem is highly constrained by obstacle and Voronoi region restrictions.

3) The fitness landscape is very likely characterized by a number of sharp fitness changes over relatively small changes in a single parameter. This is due to the discreet nature of taking classifier instructions, as well as to the properties of a Voronoi diagram. Figures 5a and 5b demonstrate how a slight change in the y value of one classifier can dramatically affect the overall fitness of a set of classifiers.

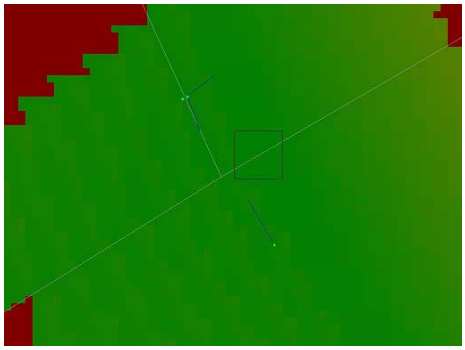

Figure 5a: Initially, two classifiers to upper left of goal are placed very close to each other, and third classifier is placed below goal – the playing field is mostly green, indicating a good rate of success.

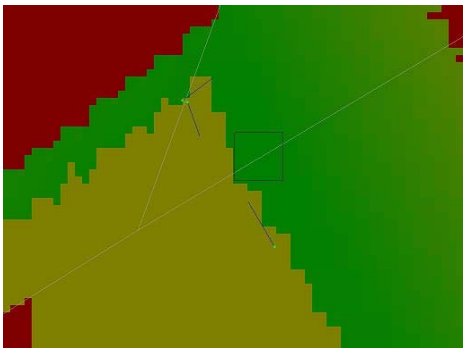

Figure 5b: The right-most classifier of the two that are to the upper left of the goal has its y position changed very slightly, relative to the playing field size – there is a dramatic increase in areas that are not directed to the goal (red + yellow)

3.2 Test Environment Setup

For this experiment, we used the test environment shown in figure 6. This environment provides a reasonably dense obstacle field, and provides a good test for the effectiveness of different GA approaches. This is also an environment for which a human can set up classifiers relatively easily, making comparisons possible.

The playing field is of size 1000 x 1000, normalized to be size 1.0 x 1.0. The goal is the small square just to the right and above of center, and has a normalized size of 0.1 x 0.1, or 1% the area of the playing field.

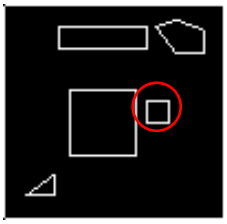

Figure 6: Test Environment – Goal is circled in red.

3.3 Strategy

We have used a standard GA representation as our main approach. Each individual in the population contains a fixed number of classifiers, where each classifier is represented by four consecutive values (x position on field, y position on field, magnitude instruction, direction instruction).

Because it can be difficult to guess the number of classifiers needed for any given field, we have also begun investigation into using methods that add classifiers to an initially small set of classifiers. The goal of this approach is to let the classifier set get built up to the proper size needed without having to make a decision on what the correct amount of classifiers is. The methods we have begun to investigate include a complexifying method that resembles the NEAT (Neuro-Evolution of Augmenting Topologies) methodology for neural networks [12], and another method that uniformly adds one classifier to every individual in the population after the population has not shown significant improvement for some number of generations.

3.4 Fixed Length GA

For the fixed length GA, each individual in the population was given a fixed number of classifiers, with each classifier on the individual's chromosome being a group of four consecutive scaled integer fields (x position, y position, magnitude, direction). N classifiers were used in each run, where N was varied to test the effectiveness of different sized classifier sets. The advantage of using the fixed GA representation is that there are a number of software tools and methods available to efficiently perform the search, and the population can focus on searching a larger group of potential solutions that have the same number of classifiers. The downside is that the designer of the GA must choose the number of classifiers.

3.5 Complexifying and Uniform Classifier Addition Genetic Algorithms

We have also begun to investigate using methods for starting from a minimal set of classifiers and adding classifiers until a solution is reached in order to take the guess work out of the problem. There are two methods we are investigating – a complexifying method inspired by the NEAT method for evolving neural networks [12], and a method that involves uniform addition of a new classifier to every population member when the fitness of the population stagnates.

NEAT evolves neural networks by starting with an initial population filled with very simple networks, and gradually adding new connections and nodes. It uses speciation to protect newly added structure, and to allow new structures to develop without competition from less complex, but more optimized networks [12]. Similarly, our complexifying GA starts with a minimal set of classifiers, and gradually adds new classifiers with some probability over the duration of a run. Kavka[13] also uses a similar scheme for evolving a Voronoi-based fuzzy controller. Like NEAT, we have also identified the use of speciation, based on the number of classifiers, as a way to separate different sized classifier sets and to protect innovation.

The uniform classifier addition (UCA) GA involves starting from a minimal set of classifiers and uniformly adding a new random classifier to every member of the population when needed. We base the need to add a new classifier on how much improvement the population has made over a specified number of generations – if the improvement is below some threshold, a new classifier is added to all members of the population. When the settings are right, this method becomes very similar to running a fixed length GA with different fixed values over and over again.

4. RESULTS AND OBSERVATIONS

4.1 Fitness Function Used

The fitness of each individual in each run is equal to the proportion of the color map that is green given a number of sample starting points. This indicates over how much of the playing field a given classifier setup is able to successfully guide the robot to the goal. For our experiments, about 1000 sample points arranged in a grid were used as starting points for the robot.

4.2 Results and Observations – Fixed Length GA

We used the following settings for each run of the fixed length GA:

- Two-point crossover, P(crossover) = 0.25
- Field mutation, P(mutation) = 1 / (4 * num classifiers)
- Tournament Selection, tourn. Size = 3
- Population Size = 150
- Number of Classifiers per Individual = 15, 25, 30
- Number of Fields per Individual = 60, 100, 120
- Number of generations = 400

Five runs were done for each different number of classifiers – the table below shows the average fitness of the best individual after 200 and 400 generations over all five runs, for each different fixed number of classifiers.

Table 1. Average Fitness of Best Individual after 200 and 400 Generations, for N = 15, 25, 30 Classifiers, Averaged over Five Runs

Number of Classifiers (N)	Fitness of Best Individual after 200 Generations	Fitness of Best Individual after 400 Generations
15	0.9172	0.9687
25	0.9420	0.9797
30	0.9518	0.9828

The table shows that both 25 and 30 classifier solutions both can reach a very good value – about 0.98 each. Generally, we consider 0.95 and above a successful result. These results also indicate the difficulty in choosing a value for N – 15 classifiers performs almost as well as 25 and 30, and there is an indication that adding more than 30 classifiers could give even better performance.

4.3 Initial Results and Observations – Complexifying GA and Uniform Classifier Addition GA

Initial results for the complexifying GA were not as good as achieved when choosing a fixed number of classifiers for the entire population. Our initial settings were as follows:

- P(crossover) = 0.25
- P(Adding Classifier) = 0.10 – 0.30
- P(Deleting Classifier) = 0.10 – 0.30
- P(Parameter Mutation) = 0.01 – 0.10
- Population Size = 150
- Number of Classifiers per Individual = Variable
- Number of Fields per Individual = Variable
- Number of generations = 400
- Speciation = Based on number of classifiers
- Threshold for Species Boundary = +/- (1–5)

As the setting values show, we have tried a number of different settings for the complexifying GA. Results for a typical run saw the maximum fitness topping out at about 0.75, which is significantly lower than the performance achieved by the fixed length GA. There were two cases that occurred. When the probability of adding new classifiers was kept low, and species were set up to include individuals with a greater variation in number of classifiers, premature convergence was prevalent. When the species threshold was reduced and the probability of adding a classifier was increased, solutions with newly added classifiers were able to survive, but classifier bloat began to occur. Possible reasons for each case are discussed in section 5.

For the uniform classifier addition GA, there are two critical settings that must be determined:

- How much the fitness of the best individual must improve in order to avoid having a new classifier added to all members of the population AND
- How many generations the best individual is given to make this improvement

Our initial observations show that when the population is given too long to make small improvements, the run will take very long even though the maximum fitness has flattened out for some time. When too much pressure is put on a population to increase its best fitness, new classifiers are added before the population has had a chance to optimize its current number of classifiers. Another important observation made during analysis of this method is that the initially evolved set of classifiers (from when the population's individuals contained a minimal set of classifiers) at times is not even present in later solutions – reasons for this are discussed in section 5.

5. DISCUSSION OF RESULTS

There are a number of points of interest regarding the results and observations from section 4. The first is the performance of the fixed length GA for this problem.

The fixed length GA comes up with very good solutions – a robot will be successfully guided to the goal from 95% or more of the starting positions that were sampled (within the desired number of steps). While this is a good value, the most important aspect of the GA performance is how it does in comparison to a human. The GA runs on a high end laptop in about 15 minutes for 400 generations – from observation, and experience humans can create classifier setups that achieve similar 0.95+ fitness values within five minutes. At some point, the complexity of the playing field will very likely make the GA faster at placing a classifier set than a human – additional studies are needed to analyze the performance of humans and a genetic algorithm on increasingly complex playing fields.

The reasons for the less than desired performance of the complexifying and UCA genetic algorithms also present several important discussion points. Potential reasons for the difficulties the complexifying and UCA methods have are directly related to the properties of the Voronoi diagram associated with the classifier set.

First, there is the concept of equivalent solutions that have significantly different classifier locations. Figure 7a and 7b both have the exact same Voronoi diagram and the exact same color map, even though the classifiers in 7a are very close to the goal and the classifiers in 7b are near the edges of the playing field.

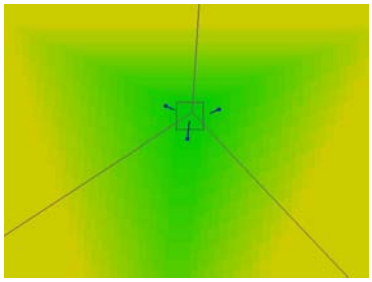

Figure 7a: Three Classifiers and Associated Color Map

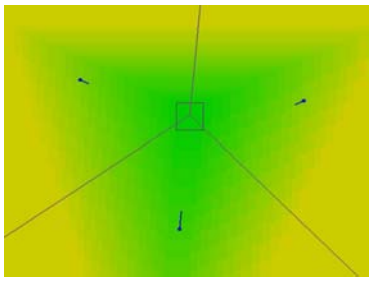

Figure 7b: Three Classifiers at Different Positions, but with Same Color Map

There is also the concept of global effect in Voronoi diagrams when there are very few sites. Figure 8a shows a set of classifiers used to provide a solution for a simple playing field with a goal in the middle of a few small obstacles. When a new classifier is added in figure 8b, it has a significant impact on both the Voronoi diagram and color map – this is an effect of there being so few classifiers.

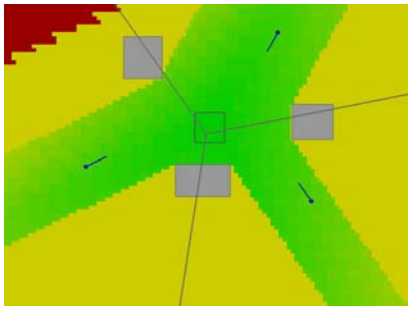

Figure 8a: Three Classifiers and Associated Color Map

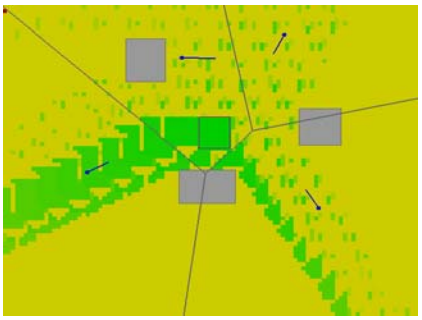

Figure 8b: Global Effects of Adding Classifier in Upper Right

Figures 7a, 7b, 8a, and 8b illustrate a critical reason why starting with a minimal set of classifiers and gradually adding new classifiers tends to perform poorly – the best performer in a population with few classifiers per individual may not include building blocks that will scale up to individuals with larger numbers of classifiers. This means that all the effort put into evolving a population with fewer classifiers may not translate into a benefit when more classifiers are added. This occurs in both the complexifying genetic algorithm and the UCA genetic algorithm.

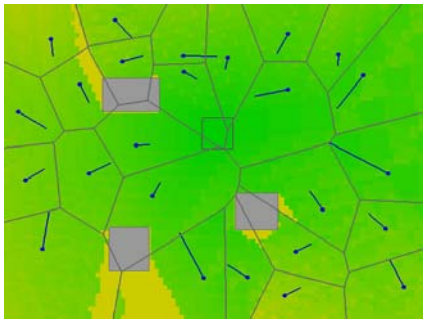

Figure 9a: Many Classifiers and Associated Color Map

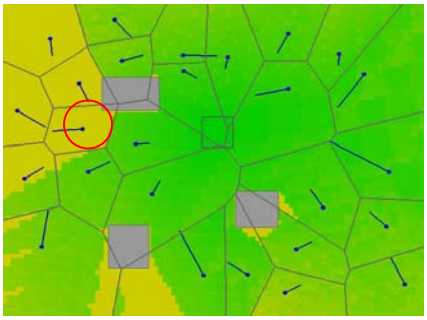

Figure 9b: Many Classifiers and Associated Color Map with Classifier Added at Left

In contrast to the global effects shown in figures 8a and 8b, there is also the concept of effects being limited to a local region of the Voronoi diagram when a new Voronoi site is added to a field that already contains many sites. Figure 9a shows a playing field with many classifiers, and figure 9b shows the effect of adding a new classifier to that field. Figure 9b shows relatively little change from figure 9a, especially when compared to the change that occurs between figure 8a and 8b. This has an important impact on the complexifying method when using speciation – at some point, adding new classifiers will have very little impact on the overall fitness of an individual solution. This leads to solutions becoming overly complex, and makes it much more time consuming and difficult for the GA to come up with a solution.

All of these issues with the observations and results of the various genetic algorithm setups leads to some valuable conclusions and action items for future work.

6. CONCLUSIONS AND FUTURE WORK

We have introduced our Voronoi classifier based method for robot navigation, and have demonstrated that a traditional GA setup is a feasible method for optimizing the placement of classifiers. We have also proposed alternate GA approaches, and demonstrated properties of our problem that have made those approaches challenging to implement.

There are a number of action items that we have identified for future work. Avoiding having to guess the best amount of classifiers is not the only reason for allowing the number of classifiers per individual to grow – we will in the near future be implementing the classifiers as hardware sensors/transmitters. This means that a cost will be associated with adding a classifier, making this a multi-objective optimization problem where we want to maximize the effectiveness of the classifier setup, while minimizing the number of classifiers used.

We are also investigating hybrid GA and non-GA approaches for optimizing the number and placement of classifiers. One thing that we have noticed while placing classifiers ourselves is that it is a very methodical process. One typical strategy we use when manually placing classifiers is to start by placing classifiers very near the goal. When we add new classifiers, we work backwards from the goal, each time making a new portion of the playing field green by pointing a newly added classifier towards a region that is already green. For these reasons, we are examining methods like dynamic programming and locally greedy heuristic measures to supplement or replace the use of a genetic algorithm.

7. REFERENCES

[1] Durrant-Whyte, H.F.. Where am I? A Tutorial on Mobile Vehicle Localization. *Industrial Robot*, 21 (2):11-16, 1994.

[2] Bornstein,J., Everett, B., and Feng, L. *Navigating Mobile Robots: Systems and Techniques,* A.K. Peters, Ltd., Wellesley, MA, ISBN 1-56881-058-X, 1996.

[3] Bonnifait, Ph., and Garcia G. A Multisensor Localization Algorithm for Mobile Robots and its Real-Time Experimental Validation. *Proc. IEEE Int'l Conf. on Robotics and Automation*, Minneapolis, MN, April, 1996.

[4] Hudas G., Cheok, Ka C., and Overholt, J. Two Dimensional Localization using Nonlinear Kalman Approaches. In *Proceedings SPIE Optics East*, Philadelphia, PA, October 2004.

[5] Shoemaker, C. M., and Bornstein, J. A. Overview of the Demo III UGV Program. In *Proceedings of the SPIE Robotic and Semi-Robotic Ground Vehicle Technology*, Vol. 3366, 1998.

[6] Arkin, R. C. Motor Schema-Based Mobile Robot Navigation. *The International Journal of Robotics Research*, August 1989, pp. 92-112.

[7] Koren, Y., and Borenstein, J. Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation. In *Proceedings of the IEEE Conference on Robotics and Automation*, 1991.

[8] Choset, H., and Burdick, J. Sensor-Based Exploration: The Hierarchical Generalized Voronoi Graph. *The International Journal of Robotics Research*, Vol. 19 No. 2, 2000.

[9] Stentz, A. Optimal and Efficient Path Planning for Partially-Known Environments. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1994.

[10] Overholt, J., Skalny, M., Fiorani, G., and Hudas, G. Robot Path Planning Using Voronoi Classifiers. In *Proceedings of SPIE*, Vol. 5804, March 2005.

[11] Fortune, S. Voronoi diagrams and Delaunay triangulations in Computing in Euclidian Geometry. *World Scientific*, 1992.

[12] Stanley, K., and Miikkulainen, R. Evolving Neural Networks Through Augmenting Topologies, *Evolutionary Computation* 10(2):99-127, 2002.

[13] Kavka, C., and Schoenauer, M. *Evolution of Voronoi-based Fuzzy Controllers*. In *PPSN 2004*, September 2004, Birmingham, England.