Applying Swarm Rule Abstraction to a Wireless Sensor Network Domain

Peter A. Hamilton

***Abstract*—Rule abstraction is a powerful tool for modeling abstract behaviors in swarm systems. The research presented in this paper examines the application of rule abstraction to the wireless sensor network domain. I specifically analyze the potential of rule abstraction to accurately model and control the connectivity, coverage, and density of a simple sensor network. In this paper, I also present a simulation tool developed to facilitate the discovery and creation of new abstract rules and discuss preliminary experimental results that will hopefully lead to the development of new abstract rules.**

***Index Terms*—network layout, optimization, rule abstraction, swarm, simulator, wireless sensor network**

I. Introduction

Planning the optimal layout and organization of a wireless sensor network (WSN) can be a difficult task. Specific knowledge about the environment must be gathered and analyzed before planning can begin, and depending upon the application, the best locations for gathering data must be identified [7]. Various network characteristics must also be taken into consideration. For example, how much target area coverage is required to obtain an acceptable sampling of data? How does the physical layout affect possible weaknesses in the network, such as energy consumption or network congestion? These questions inspire improvements to the network layout, but modifications may change other network characteristics, requiring additional modifications. It is possible for an unending cycle of modifications to result from this process, complicating network design. Ideally, it would be helpful to understand how different network characteristics are related and how manipulating one characteristic will affect the others.

To address this problem, I propose applying a new form of swarm intelligence, known as *rule abstraction*, as an organizational method for WSN design [8]. To help in the application of rule abstraction, I have designed and implemented a WSN simulator that integrates network and swarm functionality, enabling user control of the swarm while facilitating analysis of network characteristics.

II. Background & Related Work

A. Swarm Intelligence

Swarm intelligence is a field of artificial intelligence that models the intelligent behavior observed in creature swarms by using multiagent systems [2]. Many organisms demonstrate swarm behavior, including various species of birds, fish, and insects. Swarm behavior can result in unexpected yet organized emergent behaviors that make the creature swarm more robust. For example, fish swarm in schools for protection from predators. Ants lay pheromone trails while searching for food, reinforcing the most traveled paths when a food source is located, thereby guiding more workers to the source.

The agents used to model a swarm are often autonomous and self-organizing, following a predefined set of low-level rules that govern individual behavior. Each rule has a specific weight factor that determines the magnitude of its influence on an agent's actions. As each agent follows the rule set, the overall behavior of the swarm may coordinate to produce more complicated emergent behaviors. A traditional example of emergent behavior is flocking. Demonstrated by Reynolds, flocking is produced by following three low-level rules: avoid colliding with neighboring agents (*AvoidCollision*), move towards the center of mass of neighboring agents (*MoveToMassCenter*), and move in the direction that neighboring agents are moving (*MoveWithNeighbors*) [12].

First introduced in relation to cellular robotics, swarm intelligence has since been associated with multiagent systems [1] [2]. Many applications and algorithms have been developed that utilize swarm intelligence. A common example, particle swarm optimization, is a technique where swarm agents represent problem solutions for a specific problem, searching the problem solution space for optimal solutions and, over time, moving towards the best solutions discovered [11].

B. Swarm Rule Abstraction

Swarm rule abstraction is a new swarm intelligence technique that seeks to represent the emergent behaviors of a swarm system as individual rules [8]. Since emergent behaviors are products of individual agent interactions as the agents follow the rule set, there is an intuitive connection between the low-level rules, their rule weights, and the emergent behavior. The discovery of relationships between low-level and abstract rules can lead to a swarm rule hierarchy that can be used in the development of additional abstract rules. An abstract rule, while representing a simple emergent behavior, can also accept a quantifier that specifically describes how or to what degree the swarm should demonstrate the desired behavior. For example, the rule *FormCircle()* might have a parameter value to define what the radius of the circle should be. *FormCircle(60)* would direct the swarm to form a circle of radius 60.

Swarm rule abstraction is a fairly new approach to the problem of efficient swarm control. The original work validating and demonstrating the concept focused primarily on simple geometric shape formation and ant colony models [8]. The *FormCircle()* rule was one of the first rules developed to demonstrate the ability of top-down mapping, where the abstract rule parameter is used to manipulate the low-level rule

weights to produce specific behavior. To create this mapping process, multiple experiments were conducted, measuring the resulting swarm behavior as the individual rule weights were adjusted. A more robust process has recently been developed for creating abstract rules, involving a systematic sampling of simulation data to create a top-down mapping [9]. Due to the novelty of swarm rule abstraction, this paper is the first to consider applying abstract rules to the WSN domain.

C. Wireless Sensor Networks

A wireless sensor network is composed of physically disconnected sensors that must communicate wirelessly to collect data and coordinate activity [13]. The nature and exact specifications of the network can vary depending upon the application. The traditional WSN model, originally focused on military applications, involved a random distribution of sensors throughout the target area, in addition to one or more sink nodes. Since the network is isolated, the sensors have limited sensing and communication capabilities as well as limited battery life, unless fitted with energy-generation technology (e.g., solar panels). Transmissions are multi-hop, passing along the most cost-effective path in the network graph from the originating sensor to the sink. Sink nodes are typically more powerful then the ordinary sensors and may have greater energy capacity and better sensing and communicating equipment; they may even be physically connected to other sink nodes in the network. Sensors may be stationary or mobile, allowing for dynamic configurations and layouts of the network depending on certain events or commands. Applications of WSNs include target detection, tracking, pursuit, and passive area monitoring for security purposes.

In recent WSN developments, networks can be heterogeneous, incorporating sensors with different capabilities and resources. The environment and surrounding topology can also affect how the network operates: sensors at higher elevations may be able to communicate further, while those in valleys or surrounded by obstacles may have limited sensing range. The motion of a sensor may be outside its control, as in the case of cell phones passing in and out of range of various cell towers or cars travelling along a highway. These adhoc networks must be robust to sudden changes and must be able to quickly coordinate sensor communications to preserve quality of service.

Environment monitoring has also become an important application. WSNs have been applied to monitoring wild life in isolated regions as well as passively performing temperature, atmospheric, and seismic data gathering. Networks have been established to monitor glacier activity, to measure salinity and sand bank formation in coastal regions hosting wind farms, and to locate victims of an avalanche [13]. Urban areas have also become testbeds for WSN applications; applications range from tracking building power consumption to monitoring patient activity and vital signs in hospitals [13]. As a domain composed of multiple cooperating entities, WSNs are ideally suited to the swarm intelligence paradigm.

D. Swarm Intelligence and Wireless Sensor Networks

Much work has been done in applying swarm intelligence to the optimization of sensor network performance. Particle swarm optimization has been applied to the problem of selecting network configurations that maximize network lifetime [15]. Groups of swarms have been designed to learn the parameters of a fuzzy-logic controller used to dictate source search strategies in an environment [4].

Ant colony models are also popular in addressing network optimization problems. Pheromone-based routing algorithms have been developed that allow networks to dynamically route data packets along paths that use sensors with the most energy resources, prolonging network lifetime and connectivity [10]. Pheromone tracing has also been used to organize mobile networks in dynamic event detection, task completion, and sleep scheduling [6]. For example, if a target is detected crossing an area of interest, the network controller determines the number of sensors needed to adequately track and monitor it. If there are too few sensors in range, those present drop pheromones to attract more sensors. The pheromones accumulate over time if the task is left uncompleted. If the task is completed, the sensors stop laying pheromones, which disappear over time.

Applying swarm intelligence to WSN design and optimization is not a novel idea. However, due to the increasing overlap between the fields, there is still much research to be done and the application of swarm rule abstraction is an appropriate extension of swarm intelligence.

III. WSN Simulator

To facilitate and demonstrate the application of swarm rule abstraction to the WSN domain, I implemented a WSN simulator that conducts network layout simulations. In the background, the simulator treats the network sensors as agents in a single swarm. Due to the novelty of swarm rule abstraction, no preexisting simulation tool exists that can support both rule abstraction and WSNs while also providing the low-level control needed to adjust the low-level rule weights. The simulator is implemented in Python 2.5.1.

The simulations presented have used the traditional definition of a wireless sensor network. Specifically, the network is homogeneous: all sensors have the same sensing and transmission range and all sensor agents have the same sets of rules applied to their behavior. To simplify analysis, the network environment is modeled as a two-dimensional featureless Euclidean grid, with no obstacles in the environment aside from the other sensor agents; also, all possible grid configurations are restricted to rectangular shapes, further limiting environment complexity.

The simulator is composed of two applications: the *network parameter controller* and the *network layout display*. The controller is a GUI that allows the user to specify various network parameters, including environment grid size, the number of sensors in the network, the sensing/transmission range of the sensors, and the number of time steps to run the simulation. The display shows the movement of the swarm over time, connecting and disconnecting sensor agents who move in and out of transmission range of each other.

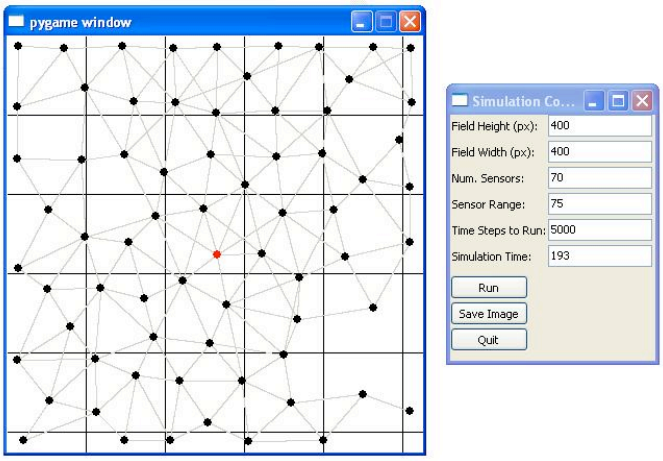

Figure 1. The WSN Simulation controller and display. Black dots represent sensors while the red dot acts as the sink. Gray lines indicate connections between sensors. Black lines indicate the boundaries of the geometric hashing buckets.

In an attempt to optimize performance, a version of geometric hashing is used. The environment grid is split into buckets of length and width equal to the sensing range. Sensor agents located in one bucket can only communicate with other agents in the same bucket or with agents in a neighboring bucket. This limits the time needed to update the list of neighboring sensor agents for any given agent. Worst-case performance is still observed when all agents move into one or two buckets.

Once input is provided and the simulation begun, the simulator applies each swarm rule to each agent's movement, displaying the network's layout as it changes over time. When the maximum time count is reached, the simulation ends, storing the network layout and simulation parameters for later use. The simulation can be run again or a new configuration can be generated. For running large numbers of experiments, a batch mode accepts a range of values for each simulation parameter, running through all possible configurations. The network layout is then analyzed and the results are stored in a data file.

IV. Abstract Characteristics

I analyze three potential abstract rules for the WSN domain: *connectivity*, *coverage*, and *density*. In practice, *coverage* and *density* both rely on *connectivity*, allowing for optimal computation time by overlapping and reusing calculations from one rule to another.

A. Connectivity

Connectivity represents the overall functionality of the network. A sensor v is connected to the network $G = (V, E)$, where V is the set of vertices in the network graph and E is the set of edges in the network graph, if $\exists P$ s.t. $\{P \mid \{e_1, e_2, \ldots, e_n\}\ where\ e_i \epsilon E\}$, a path connecting v to the sink. The more connected a network is, the more sensors it has available to monitor the area and supply data. Connectivity can also be related to the degree to which the network is partitioned. A single network partition implies total connectivity. A total of n partitions, where n is the number of sensors in the network, implies no connectivity.

Network connectivity can be computed by running a breadth-first search (BFS) starting from the sink node, marking each sensor found as connected. The network connectivity percentage is just the number of connected sensors divided by the total number of sensors in the simulation. The running time is dominated by BFS, which is $O(|V| + |E|)$.

B. Coverage

Coverage measures how efficiently the network monitors the target area. A region in range of a sensor is only covered if that sensor is itself connected to the network. Coverage is simply the total area within sensing range of all connected network sensors divided by the total target area. The coverage algorithm relies on the computation of connectivity to determine which sensors are connected, so both abstract parameters can be measured at the same time. The algorithm used to compute coverage considers every point in the environment grid, where X represents the width of the grid, Y the height of the grid, and XY the total number of grid points. Despite the use of geometric hashing, the worst case still involves scanning through all $|V|$ sensors, calculating the number of sensors in range of the current grid point. The running time of this algorithm is therefore $O(X \cdot Y \cdot |V|)$.

C. Density

Network density is the most nebulous of the three high-level characteristics under analysis. Density can be used to measure the uniformity of the network. It can also be thought of as the degree of sensor distribution throughout the target area. One method of measuring network density is to simply compute the average number of neighbors per sensor. Another method considers the total area covered by the network and computes the ratio of connected sensors to area covered. I chose to measure density in terms of coverage. Specifically, a network is said to be *M-dense* if at least 50% of all grid points are within sensing range of M connected sensors. M-denseness provides a general idea of how distributed the network is by revealing how evenly spread out the individual sensors are throughout the covered area.

Computing the density estimate is implicit in computing coverage, and therefore also takes $O(X \cdot Y \cdot |V|)$. Specifically, in these experiments I measured from 1 to 10-dense. Therefore, the time required to generate all three abstract parameters is $O(X \cdot Y \cdot |V|)$.

V. Experiments & Results

A. Low-Level Rules

The network swarm model uses the following four low-level rules:

- *AvoidCloseness* models repulsion between swarm agents according to an inverse-squared relationship. The closer two agents are, the stronger the repulsion. The further apart two agents are, the weaker the repulsion. This rule is primarily responsible for agent dispersion throughout

the environment grid. The parameter of this rule varies the strength of the repulsion [8].

- *MoveToCenter* is a conditional rule that a swarm agent follows when it has no neighbors. By convention, the agent will move toward the center of the environment grid in an attempt to find and connect with other agents. Since the network sink is always located near the center, the swarm agent will eventually connect with other agents, provided that the magnitude of the sensing/transmission range is reasonable. Note that this does not imply that the agent will be connected to the network. It is possible for a small group of agents to connect in an isolated region of the environment grid but not be connected to the sink. Since these agents have neighbors, they are never pulled to the center and may never connect. This rule is used primarily to connect as much of the network as possible during simulation. The parameter of this rule varies the speed at which an isolated agent moves to the center of the grid.
- *Slowdown* is a simple mechanical rule that bounds the maximum speed an agent can attain during simulation. If not applied, the swarm tends to cyclic instability and never reaches equilibrium. The parameter of this rule varies the slowdown rate. For these experiments, this parameter value is set to 0.22.
- *StayInBorders* is a simulation-specific rule that prevents the sensor agents from moving outside of the target area. When an agent moves within a specific range of the area border, a repulsive force is applied, pushing the agent away from the border. If the specified range is too large, agents are constantly pushed around and no equilibrium is reached. If the range is too small, an agent with high velocity can push through the repulsion and escape the environment, forever isolated from the swarm. The parameter of this rule varies the repulsion when in range of the border. For our experiments, the border range is set to 10 pixels and the parameter value to 1.0.

B. Experiment Setup

In the experimental network layouts, the *AvoidCloseness* values were chosen from the set {0, 1, 10, 20, 40, 60, 80, 100} and the *MoveToCenter* values from {0, 0.001, 0.002, 0.003, 0.004}. The grid sizes are {300, 400, 500} and the sensor count and range magnitude are both {50, 60, 70, 80, 90, 100}. All combinations of these parameter values were tested with 20 runs for each configuration, each with random initial network layouts to ensure statistically valid results. The average of the results for each set of 20 was then calculated along with a corresponding 95% confidence interval (CI). The results for specific data sets are provided in Appendix A. In each data table in Appendix A, the CIs are represented by the offset, x, from the mean of the characteristic being measured. To calculate the actual CI, simply add and subtract the offset from the mean, yielding $(mean - x, mean + x)$.

The primary goal of these experiments is to reveal the effects of both the environment and swarm parameters on network connectivity, coverage, and density. By demonstrating that the swarm parameters can yield results comparable to layouts with excessive numbers of sensors or transmission capability, these experiments reveal the power and potential of rule abstraction.

C. Graphs & Analysis

Due to the amount of data generated in these experiments, only a small portion can be presented here. To simplify analysis, five different data sets were selected and the corresponding results for connectivity, coverage, and density graphed. The first two data sets demonstrate the effects of the network and environment parameters (grid size, no. sensors, and sensing and transmission range) on the network while the other three data sets reveal the effects of the swarm parameters (*AvoidCloseness* factor and *MoveToCenter* factor) on the network. The parameters not listed in each data set serve as axes for the data graphs. The data sets selected are:

- Grid Size = 300, No. Sensors = 50, AvoidCloseness Factor = 0
- Grid Size = 500, Range = 100, AvoidCloseness Factor = 0
- Grid Size = 400, No. Sensors = 50, Range = 50
- Grid Size = 400, No. Sensors = 70, Range = 70
- Grid Size = 500, No. Sensors = 60, Range = 60

See Appendix A for the data tables corresponding to the graphs. For full results see the project website[1].

Each graph displays the magnitude of the CI offset for each data point, in addition to the average characteristic value at each data point. The offset magnitude is represented by a gray-scale shading of the xy-plane underneath each point; the darker the shade of a grid cell, the larger the magnitude of the CI offset for the corresponding point. This additional data depicts the consistency of the experiment results and further reveals the effects of the swarm parameters.

1) Connectivity: For network configurations with either larger numbers of sensors or large transmission ranges, I intuitively expect to see improved connectivity, regardless of the swarm rule factors. Figures 2 and 3 display examples of this relationship. In Figure 2, there is a direct correlation between increasing range and connectivity. A single sensor can cover more area with increasing range, also implying that it can communicate with more sensors. As the range increases, more sensors have a direct connection to the sink, increasing network connectivity. These sensors then act as intermediate nodes in the graph, allowing sensors further removed from the sink to connect, again increasing connectivity. In fact, once the range reaches 80 pixels, the network is almost entirely connected, and the corresponding CI offsets reveal that the results are very consistent.

For Figure 3, there is a direct relationship between the number of sensors and connectivity, similar to Figure 2. Due to the fixed size of the grid environment and the range, the results converge for more configurations as the number of sensors increases. The variation in average connectivity still exists with large numbers of sensors, though average connectivity never

[1] http://userpages.umbc.edu/~pete5/Projects/Honors_Thesis/

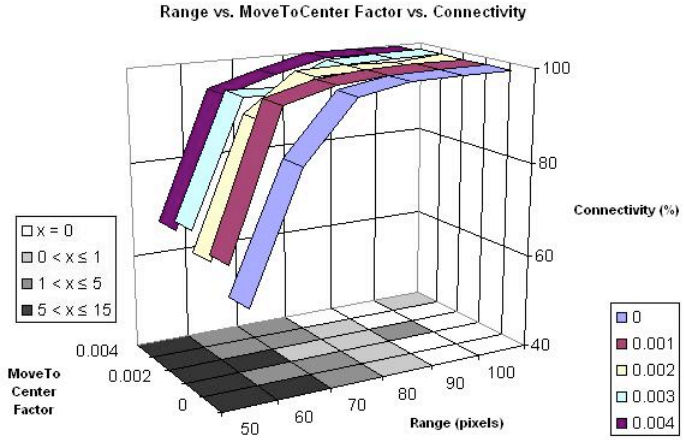

Figure 2. Grid size = 300 pixels, No. sensors = 50, *AvoidCloseness* factor = 0. The legend on the left defines the different ranges for the CI offset, x. The legend on the right associates each plot with its corresponding *MoveToCenter* factor. This scheme is used for all following data graphs.

measures below 90%. From the graph, *MoveToCenter* factors of 0 and 0.001 provide the most consistent results over various numbers of sensors.

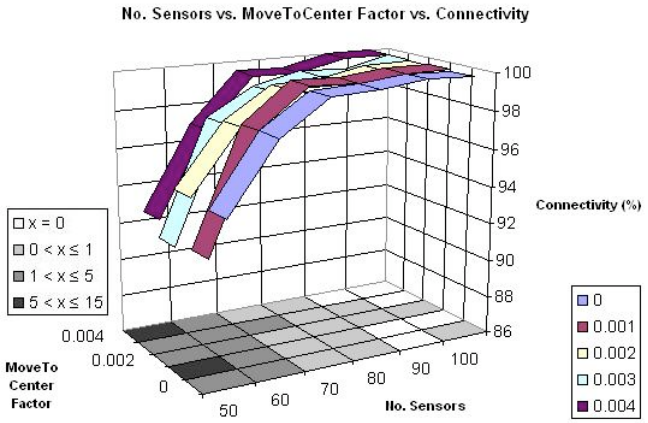

Figure 3. Grid size = 500 pixels, Range = 100 pixels, *AvoidCloseness* Factor = 0.

Figures 4, 5, and 6 demonstrate the effects of *AvoidCloseness* on connectivity performance. For Figure 4, the performance improves as the *AvoidCloseness* factor increases but general performance is poor for low *MoveToCenter* factor values. The magnitude of the CI offset does not decrease as the *AvoidCloseness* factor increases, unlike Figures 2 and 3. The most consistent performance that provides near optimal connectivity for the network configuration occurs when the *MoveToCenter* factor is 0.002.

It is possible for a combination of network and swarm parameters to create ideal performance. In Figure 5, maximum connectivity is achieved for very low *AvoidCloseness* factor values, and the corresponding CI intervals are as tight as possible. The *MoveToCenter* factor does not make a difference beyond an *AvoidCloseness* factor value of 10, conflicting with the intuition that the *MoveToCenter* rule is primarily responsible for improving network connectivity. However, the performance can also be interpreted as the result of excessive

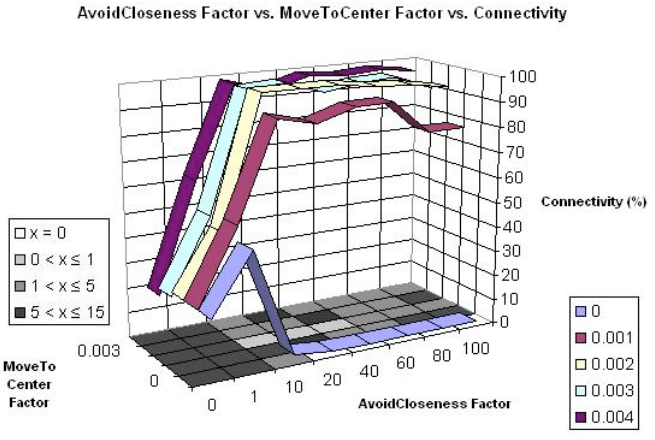

Figure 4. Grid size = 400 pixels, No. sensors = 50, Range = 50 pixels: When both rule factors are 0, the network is simply a random configuration with consistently poor performance.

numbers of sensors with adequate range flooding the target area. In this case, the *AvoidCloseness* factor would spread the sensors evenly around the grid, yielding consistent network density as depicted in the corresponding density graph in Figure 15.

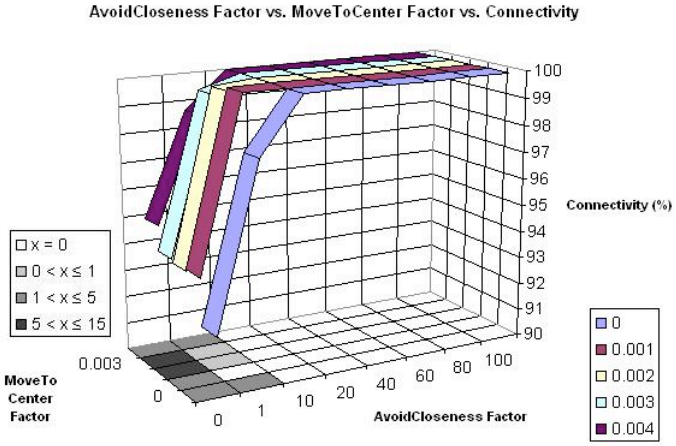

Figure 5. Grid size = 400 pixels, No. sensors = 70, Range = 70 pixels. The CI offsets are as tight as possible for most of the tests in this configuration.

The connectivity performance for a smaller network in the largest environment is displayed in Figure 6. The poor performance of the configuration with a *MoveToCenter* factor of 0 is due to the even dispersal of the sensors with no tendency to ensure connectivity with the sink. Overall, the *MoveToCenter* factor does not affect connectivity performance. The *AvoidCloseness* factor, beyond values of 10 or 20, leads to maximum connectivity which is consistent for all remaining factor values.

For connectivity performance, the *MoveToCenter* factor, while expected to greatly influence connectivity, in general only affects performance consistency as shown by the CI offsets. Instead, the *AvoidCloseness* factor produces even dispersal of the sensor agents, leading to better connectivity at low factor values, depending upon the network configuration.

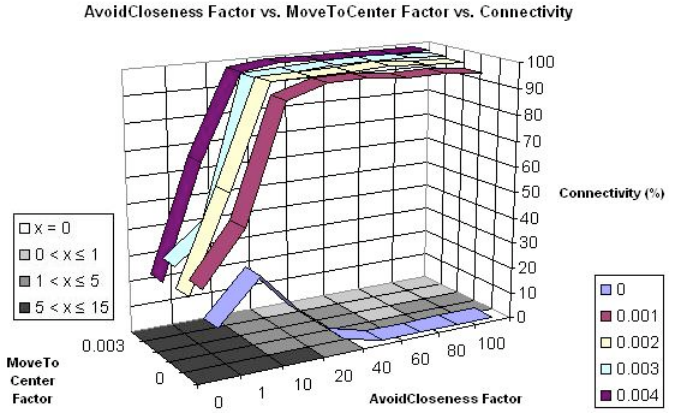

Figure 6. Grid size = 500 pixels, No. sensors = 60, Range = 60 pixels. The *MoveToCenter* factor value, aside from 0, does not change connectivity performance. However, it does improve performance consistency, as shown by the CI offsets.

In terms of swarm rule abstraction, the consistency over multiple *AvoidCloseness* factor values, in addition to small CIs, leads me to believe that the *AvoidCloseness* rule can be used to control connectivity with acceptable accuracy.

2) Coverage: Network coverage is related to network connectivity. It is usually presumed that greater connectivity leads to greater coverage. Therefore, there should be obvious similarities between the graphs for both coverage and connectivity.

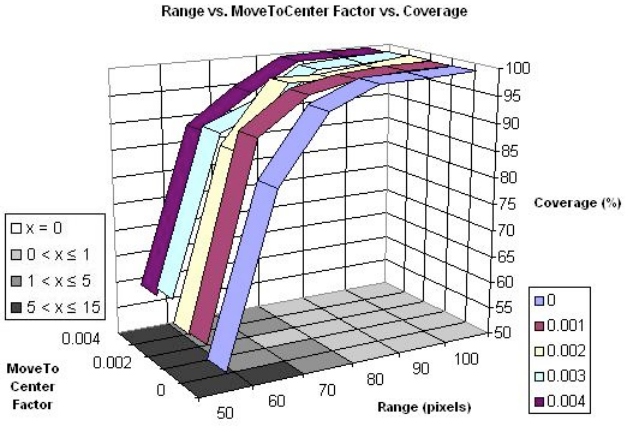

Figure 7. Grid size = 300 pixels, # sensors = 50, *AvoidCloseness* factor = 0: The relationship between coverage and range is almost identical to Figure 2.

The plots of Figures 7 and 8 correspond to Figures 2 and 3, representing the same network configurations. The coverage increases as both the range and the number of sensors increases. The *MoveToCenter* factor does not affect coverage performance nor coverage consistency, as both graphs show roughly the same decrease in CI offset magnitude as the range and the number of sensors increases.

Figure 9 is interesting due to the inconsistent CI offset magnitudes recorded for various data points. As in Figure 4, it appears that the optimal *MoveToCenter* factor value is 0.002, but it is also interesting to note the rate of decline of each plot from the initial peaks observed at an *AvoidCloseness*

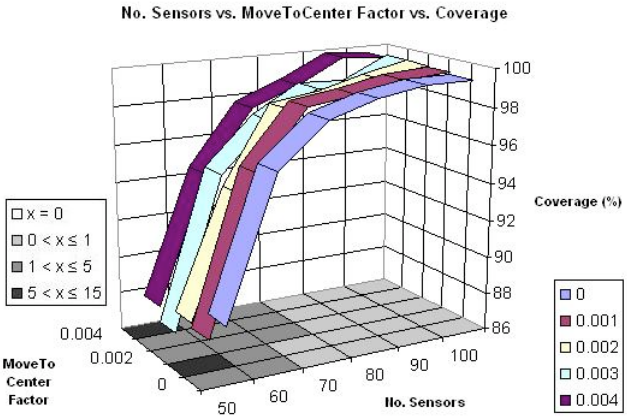

Figure 8. Grid size = 500 pixels, Range = 100 pixels, *AvoidCloseness* factor = 0.

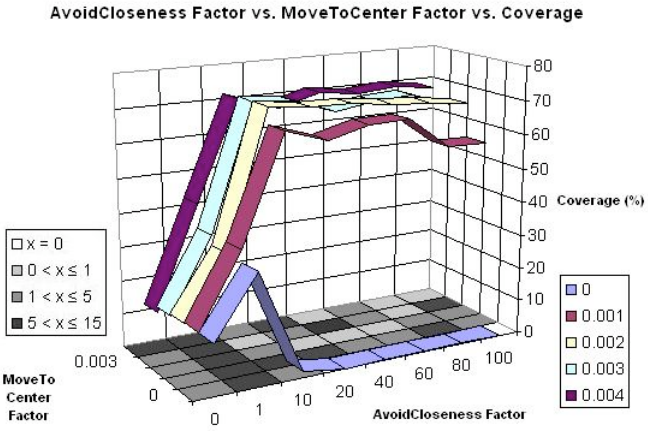

Figure 9. Grid size = 400 pixels, No. sensors = 50, Range = 50 pixels: This graph is virtually identical to Figure 3. The reason optimal coverage beyond 80% is not observed is because of the physical limitations of the network in a medium-sized environment.

factor value of 10. A possible explanation for this could be that the increase in repulsion among sensor agents overcomes the connectivity attraction, leading segments of the network to form small local networks that are isolated from the sink. This phenomenon also appears in Figure 4, though Figure 9 shows tighter average consistency for greater *MoveToCenter* and *AvoidCloseness* factor values.

In Figure 10, the maximum coverage is reached quickly, indicating that the physical network configuration may be more responsible than the swarm parameters.

Figure 11 is similar to previous coverage and connectivity graphs. Overall, coverage performance is consistent as the *AvoidCloseness* factor changes. Given the parameter modifications, I believe that *AvoidCloseness* can also be used to model coverage, possibly in determining the necessary size of the network needed to yield optimal coverage for specific target areas.

3) Density: Density is dependent on both connectivity and coverage, therefore the effects of both characteristics should be detected in the density performance.

The relationships in Figures 12 and 13 are linear. The only reason the density values stop increasing in Figure 12 is due

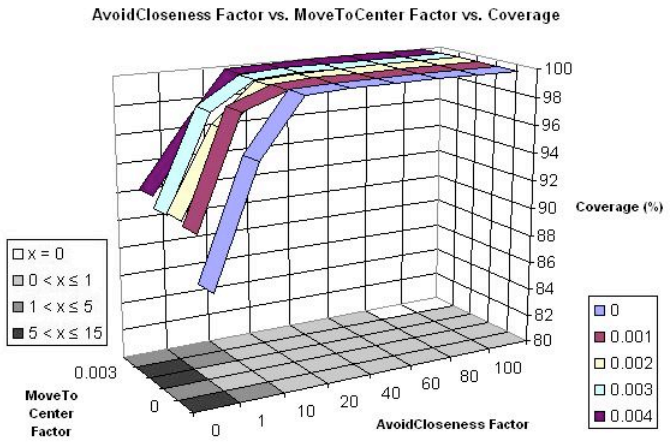

Figure 10. Grid size = 400 pixels, No. sensors = 70, Range = 70 pixels. Performance matches connectivity performance in Figure 5.

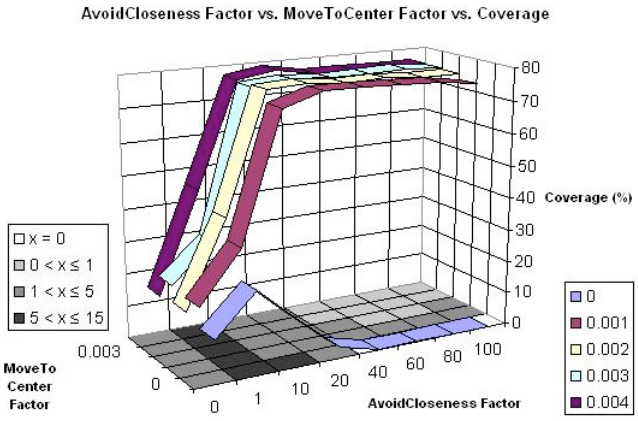

Figure 11. Grid size = 500 pixels, No. sensors = 60, Range = 60 pixels. Coverage performance is independent of the *MoveToCenter* factor and is consistent for higher *AvoidCloseness* factor values.

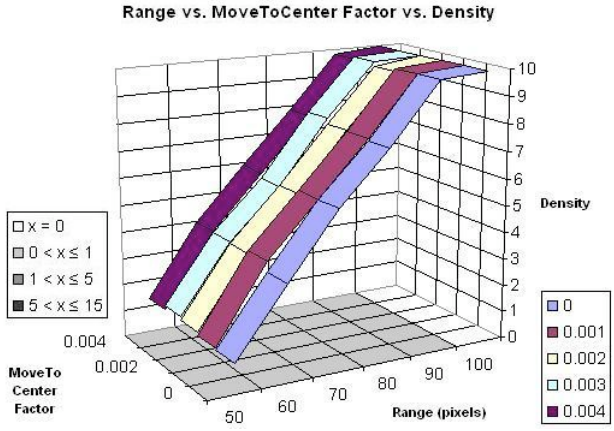

Figure 12. Grid size = 300 pixels, No. sensors = 50, *AvoidCloseness* factor = 0: Since the experiments only measure up to 10-dense, the linear relationship between Density and Range is truncated. Again, swarm rule factors have no impact on network density in this network configuration.

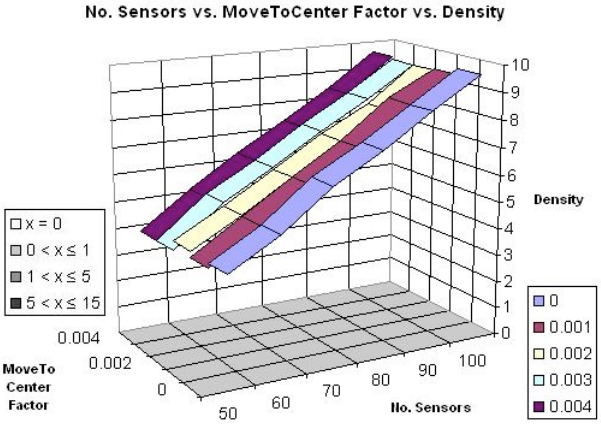

Figure 13. Grid size = 500 pixels, Range = 100 pixels, *AvoidCloseness* factor = 0.

to the fact that 10-dense was the maximum density level measured in the experiments. I would expect the density to continue increasing with the range until any individual sensor can cover the entire target area. At this point, the density would reach a maximum possible value, equal to the total number of sensors in the network. Similarly in Figure 13, increasing the number of sensors in the target area will always increase the density, since the maximum density is dependent on the total number of network sensors.

Figure 14 shows the first instance where the *MoveToCenter* factor influences the density measure. A decrease in density is observed as the *AvoidCloseness* factor increases, but it is most pronounced when the *MoveToCenter* factor value is 0.001. Since density is primarily dependent on coverage, and a decrease occurs in the corresponding coverage graph in Figure 9, a similar decrease in density is expected. *MoveToCenter* values beyond 0.001 produce satisfactory average density, though the optimal configuration in terms of both maximum possible density and CI offset magnitude occurs when the *MoveToCenter* factor value is 0.002, as in Figures 4 and 9.

The variability of average density displayed in Figure 15 is at first confusing. However, the density axis shows that the variability mainly occurs between 4.5-dense and 5-dense, which is actually pretty consistent. The CI offset magnitudes are also small, indicating that the measures are accurate over multiple simulations for each configuration.

For the configuration in Figure 16, the relationship is again obvious. Higher *AvoidCloseness* factor values lead to greater network dispersal throughout the environment, leading to redundant coverage of the same areas by multiple sensors, creating higher network density. An exact CI offset for higher *AvoidCloseness* factor values is expected, since the sensor spread throughout the target area is confined by the area bounds and the *StayInBorders* rule, preserving optimal density for the configuration even with differing *AvoidCloseness* factor values.

For each network configuration, the density values measured are very consistent, if not exact, making density the most precise of the three network characteristics measured. It also indicates that density is the most robust, making it resistant to changes in the swarm parameters. This could result from the artificial limits placed on measuring the density. However, I do not believe this to be the case because density is tied

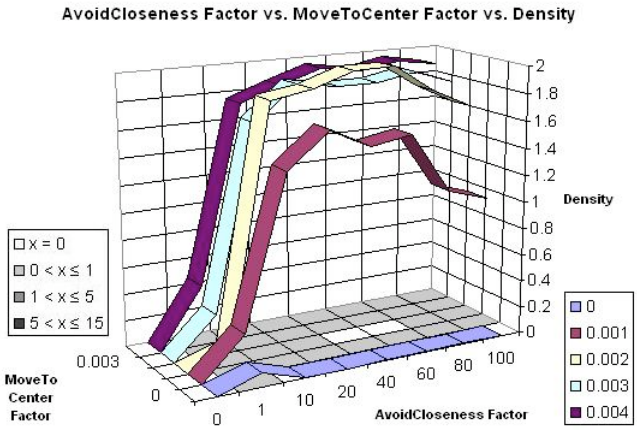

Figure 14. Grid size = 400 pixels, No. sensors = 50, Range = 50 pixels: No satisfactory density is recorded when both swarm rule factors are 0, as expected.

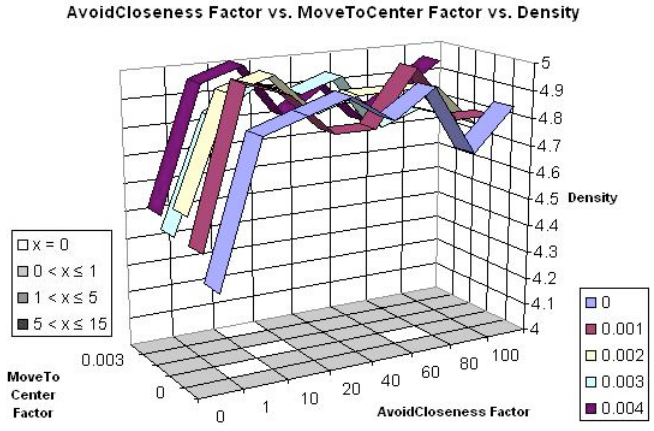

Figure 15. Grid size = 400 pixels, No. sensors = 70, Range = 70 pixels. While there appears to be a lot of variability in the average density measures, the density axis reveals that all values, aside from those with *AvoidCloseness* = 0, are between density measures of 4.5 and 5.0.

directly to connectivity and coverage in this model, and both generally show consistent behavior for each data set.

VI. Future Work

This work has only begun to reveal the potential for swarm rule abstraction to model high-level network characteristics. Ongoing work is currently focused on fully developing the connectivity, coverage, and density rules. However, due to the number of variable parameters available in the WSN domain, the analysis can also be expanded to include new network types and environments.

To simplify testing and analysis for these experiments, the network was limited to a homogeneous set of sensors and the network environment to a two-dimensional featureless Euclidean grid. However, this model does not accurately represent the real world: most WSN applications employing swarm rule abstraction would be deployed in more complicated environments with a more varied network composition. Simulation obstacles could be added to the environment to interact and interfere with sensor movement and communication. I am also

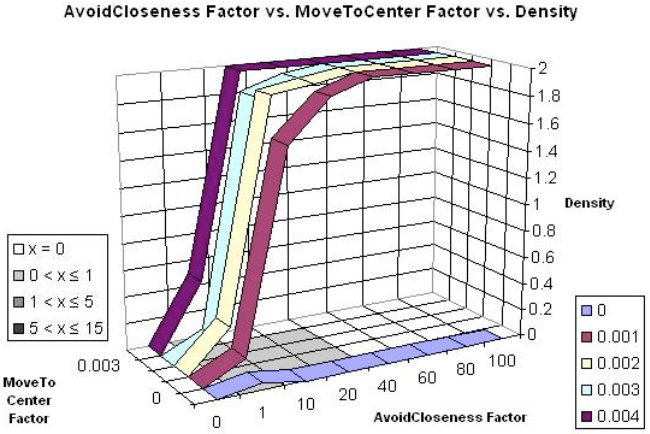

Figure 16. Grid size = 500 pixels, No. sensors = 60, Range = 60 pixels. For this configuration, the maximum density possible with available resources is 2-dense. The *MoveToCenter* factor affects the speed at which the maximum is reached but is does not prevent the maximum from being reached in any case.

interested in extending the environment to three dimensions, allowing for variable terrain and producing hills and valleys that may enhance or hinder network operations. Finally, I plan on investigating how the current abstract rule models generalize to heterogeneous networks.

In addition to modifying the problem domain, there are several modifications and enhancements that can be made to the WSN simulator. These updates include saving external copies of network layout and simulation parameters, adding data load functionality so the simulator can upload a previously created network layout, adding GUI functionality to allow the user to dynamically pick the swarm rules to use along with their respective weight values, and displaying an actual test of the network in a target-tracking scenario. The long-term goal of this research is to develop a robust simulation application that will be instrumental in facilitating the analysis of swarm rule abstraction with new network types and environments.

VII. Conclusions

I have presented swarm rule abstraction and defined how it can be applied to the wireless sensor domain, specifically in modeling network characteristics such as connectivity, coverage, and density. I have also presented a WSN simulator that was designed specifically to assist in creating and analyzing relationships between the low-level swarm rules and the observed abstract network characteristics. I have conducted experiments revealing the potential of rule abstraction and presented selected findings in both graphical and tabular form.

I believe these initial results warrant additional investigation into the applicability of swarm rule abstraction, both in further developing rules modeling connectivity, coverage, and density, and in modeling new rules, like network lifetime. I believe that the contributions of this paper and research will lead to advances in the design and control of wireless sensor networks and I look forward to participating in this future work.

REFERENCES

[1] Beni, G. and Wang, J. (1989). Swarm Intelligence in Cellular Robotic Systems. In the *Proceedings of the NATO Advanced Workshop on Robots and Biological Systems*.

[2] Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999). *Swarm Intelligence: From Natural to Artificial Systems*. Oxford-University Press.

[3] Collier, T.C. and Taylor, C. (2004). Self-Organization in Sensor Networks. In the *Journal of Parallel and Distributed Computing.* Vol. 64, Is. 7, pp. 866-873.

[4] Cui, X., Hardin, T., Ragade, R.K., and Elmaghraby, A.S. (2004). A Swarm-based Fuzzy Logic Control Mobile Sensor Network for Hazardous Contaminants Localization. In *IEEE International Conference on Mobile Ad-hoc and Sensor Systems.* pp. 194-203.

[5] Gruenwald, C., Hustvedt, A., Beach, A., and Han, R. (2007) SWARMS: A Sensornet Wide Area Remote Managements System. In *3rd International Conference on Testbeds and Research Infrastructure for the Development of Networks and Communities, 2007.* TridentCom 2007.

[6] Hong, J., Lu, S., Chen, D., and Cao, J. (2008). Towards Bio-Inspired Self-Organization in Sensor Networks: Applying the Ant Colony Algorithm. In *22nd International Conference on Advanced Information Networking and Applications.* pp. 1054-1061.

[7] Jourdan, D.B. (2006). Wireless Sensor Network Planning with Application to UWB Localization in GPS-Denied Environments. Doctoral Thesis at Massachusetts Institute of Technology.

[8] Miner, D., Hamilton, P., and desJardins, M. (2008). Learning Abstract Rules for Swarm Systems. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (2008).*

[9] Miner, D. and desJardins, M. (2008). Learning Abstract Properties of Swarm Systems. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems.*

[10] Muraleedharan, R. and Osadciw, L.A. (2003). Sensor Communication Network Using Swarm Intelligence. In *Proceedings of the 2nd IEEE Upstate New York Workshop on Sensor Networks.*

[11] Parsopoulos, K.E. and Vrahatis, M.N. (2002). Recent Approaches to Global Optimization Problems Through Particle Swarm Optimization. In *Natural Computing*. Vol. 1, No. 2-3, pp. 235-206.

[12] Reynolds, C. (1987). Flocks, Herds, and Schools: A Distributed Behavioral Model. In *SIGGRAPH Comput. Graph.* Vol. 21, Iss. 4, pp. 25-34.

[13] Römer, K. and Mattern, F. (2004). The Design Space of Wireless Sensor Networks. In *IEEE Wireless Communications.* pp. 54-61.

[14] Selvakennedy, S., Sinnappan, S., and Shang, Y. (2007). A Biologically-inspired Clustering Protocol for Wireless Sensor Networks. In *Computer Communications*. Vol. 30, Is. 14-15, pp. 2786-2801.

[15] Zhao, C. and Chen, P. (2007). Particle Swarm Optimization for Optimal Deployment of Relay Nodes in Hybrid Sensor Networks. In *IEEE Congress on Evolutionary Computation, 2007.* pp. 3316-3320.

Appendix A

Grid Size	No. Sensors	Range	AvoidCloseness Factor	MoveToCenter Factor	Coverage	Coverage CI	Density	Density CI	Connectivity	Connectivity CI
300	50	50	0	0	53.2765	8.367509	1.1	0.413462	59.6	10.179177
300	50	50	0	0.001	55.868111	7.550509	1.2	0.429414	65.9	8.828598
300	50	50	0	0.002	55.363167	8.608237	1.3	0.482096	64.3	11.576534
300	50	50	0	0.003	60.064333	8.310611	1.45	0.448557	68.7	9.765738
300	50	50	0	0.004	58.535555	6.958272	1.3	0.520416	66.8	9.199063
300	50	60	0	0	83.659889	6.046749	3.55	0.448557	85.7	7.127909
300	50	60	0	0.001	90.831833	3.464694	3.95	0.258356	96.1	3.478379
300	50	60	0	0.002	87.042833	4.603972	3.85	0.286556	91.5	5.287461
300	50	60	0	0.003	89.271278	4.54872	3.9	0.413462	95.1	4.482148
300	50	60	0	0.004	88.297389	3.447639	3.95	0.293182	94.1	4.226289
300	50	70	0	0	95.652389	2.371015	5.85	0.156493	97.9	1.719335
300	50	70	0	0.001	97.329222	1.677661	5.85	0.209041	98.9	1.399034
300	50	70	0	0.002	98.286611	0.920388	5.75	0.189776	99.4	0.836165
300	50	70	0	0.003	92.917111	4.388307	5.7	0.418082	94.7	4.976046
300	50	70	0	0.004	94.591056	3.007722	5.9	0.191037	97.3	2.976039
300	50	80	0	0	99.345333	0.294278	7.7	0.20084	99.9	0.191037
300	50	80	0	0.001	99.087833	0.64245	7.75	0.189776	99.8	0.382074
300	50	80	0	0.002	98.114611	0.979114	7.8	0.175308	99.6	0.525923
300	50	80	0	0.003	98.842722	0.75337	8	0.138593	99.6	0.525923
300	50	80	0	0.004	99.428944	0.459151	7.85	0.209041	100	0
300	50	90	0	0	99.524833	0.371153	9.85	0.156493	100	0
300	50	90	0	0.001	99.802111	0.1605	9.85	0.156493	100	0
300	50	90	0	0.002	99.0105	0.852986	9.75	0.189776	99.3	1.009923
300	50	90	0	0.003	99.291111	0.628538	9.9	0.131481	100	0
300	50	90	0	0.004	99.730167	0.242698	9.8	0.175308	100	0
300	50	100	0	0	99.971889	0.032821	10	0	100	0
300	50	100	0	0.001	99.926333	0.081123	10	0	100	0
300	50	100	0	0.002	99.953445	0.048931	10	0	100	0
300	50	100	0	0.003	99.826556	0.170268	10	0	100	0
300	50	100	0	0.004	99.590278	0.50812	10	0	99.6	0.764149

Table I
DATA FOR FIGURES 2, 7, AND 12.

Grid Size	No. Sensors	Range	AvoidCloseness Factor	MoveToCenter Factor	Coverage	Coverage CI	Density	Density CI	Connectivity	Connectivity CI
500	50	100	0	0	88.98318	4.04987	4.05	0.258356	94.4	3.609807
500	50	100	0	0.001	87.63866	6.548258	3.9	0.457567	91.9	7.383115
500	50	100	0	0.002	87.89214	3.591745	4.1	0.236015	94.5	3.242897
500	50	100	0	0.003	86.72304	4.709131	3.75	0.27282	91.5	4.620573
500	50	100	0	0.004	87.60086	5.223909	3.75	0.435522	92.5	5.430827
500	60	100	0	0	96.1383	1.660361	5.1	0.131481	97.916667	1.219543
500	60	100	0	0.001	95.61384	1.721023	5.05	0.095519	97.5	1.869435
500	60	100	0	0.002	94.13406	2.316872	5.1	0.131481	97.75	1.588201
500	60	100	0	0.003	94.75676	2.18843	5.25	0.189776	97.583333	1.825757
500	60	100	0	0.004	94.3521	2.311424	5.1	0.191037	96.916667	1.950107
500	70	100	0	0	98.36462	0.731693	6.5	0.219135	99.642857	0.335708
500	70	100	0	0.001	98.6262	0.81789	6.2	0.175308	99.857143	0.27291
500	70	100	0	0.002	98.19206	1.100674	6.2	0.175308	99.357143	0.576385
500	70	100	0	0.003	97.27606	1.635604	6.25	0.189776	99	1.049533
500	70	100	0	0.004	97.61086	1.233599	6.25	0.189776	99.5	0.955186
500	80	100	0	0	99.13226	0.36592	7.5	0.219135	99.75	0.371561
500	80	100	0	0.001	98.9619	0.828915	7.3	0.20084	99.75	0.477593
500	80	100	0	0.002	98.55624	0.814579	7.3	0.20084	99.375	0.703709
500	80	100	0	0.003	98.71746	0.838854	7.4	0.214707	99.5625	0.46727
500	80	100	0	0.004	98.59658	0.900111	7.3	0.244018	99.4375	0.659114
500	90	100	0	0	99.54588	0.292302	8.65	0.209041	100	0
500	90	100	0	0.001	99.48766	0.235023	8.65	0.209041	100	0
500	90	100	0	0.002	99.2235	0.480239	8.2	0.175308	99.888889	0.212264
500	90	100	0	0.003	98.6266	0.917158	8.4	0.214707	99.444445	0.606268
500	90	100	0	0.004	99.83848	0.099231	8.35	0.209041	100	0
500	100	100	0	0	99.53612	0.244192	9.75	0.189776	99.9	0.131481
500	100	100	0	0.001	99.64166	0.202201	9.55	0.218036	100	0
500	100	100	0	0.002	99.73924	0.210428	9.5	0.219135	99.9	0.191037
500	100	100	0	0.003	99.78334	0.127685	9.5	0.219135	100	0
500	100	100	0	0.004	99.67074	0.197194	9.6	0.214707	100	0

Table II
DATA FOR FIGURES 3, 8, AND 13.

Grid Size	No. Sensors	Range	AvoidCloseness Factor	MoveToCenter Factor	Coverage	Coverage CI	Density	Density CI	Connectivity	Connectivity CI
400	50	50	0	0	14.757438	3.144839	0	0	23.7	5.412404
400	50	50	0	0.001	15.016969	3.937554	0	0	23.9	6.44405
400	50	50	0	0.002	14.600438	4.917551	0	0	23.8	7.961075
400	50	50	0	0.003	14.027125	4.385217	0	0	21.8	7.039102
400	50	50	0	0.004	12.021563	5.017726	0	0	19	7.866904
400	50	50	1	0	32.375219	6.40112	0.1	0.131481	46.8	9.072916
400	50	50	1	0.001	35.869125	8.276937	0.3	0.20084	52.8	12.322774
400	50	50	1	0.002	32.631687	7.70682	0.1	0.131481	47.1	11.264374
400	50	50	1	0.003	35.21775	9.273718	0.3	0.20084	51.1	13.317836
400	50	50	1	0.004	42.212281	8.147184	0.45	0.218036	62.2	12.072391
400	50	50	10	0	1.825844	2.598431	0	0	4.1	3.055023
400	50	50	10	0.001	67.043688	6.984454	1.4	0.255553	90.7	9.177636
400	50	50	10	0.002	72.322281	1.024785	1.85	0.156493	98.6	0.738585
400	50	50	10	0.003	72.057156	0.934694	1.65	0.209041	97.8	1.267197
400	50	50	10	0.004	71.571406	1.111637	1.75	0.189776	98.3	1.009923
400	50	50	20	0	0	0	0	0	2	0
400	50	50	20	0.001	63.073563	9.330674	1.65	0.286556	87.1	12.588252
400	50	50	20	0.002	70.786094	0.813981	1.9	0.131481	98.2	0.826925
400	50	50	20	0.003	71.003563	1.093893	1.9	0.131481	98.8	1.220877
400	50	50	20	0.004	67.335563	6.809331	1.85	0.209041	94.2	9.300814
400	50	50	40	0	0	0	0	0	2	0
400	50	50	40	0.001	66.083	2.719969	1.5	0.259284	91.7	3.98102
400	50	50	40	0.002	70.588375	0.763901	2	0	98.4	0.858829
400	50	50	40	0.003	66.656094	6.73542	1.85	0.209041	93.5	9.238547
400	50	50	40	0.004	70.654438	0.857969	1.95	0.095519	98.7	1.047271
400	50	50	60	0	0	0	0	0	2	0
400	50	50	60	0.001	66.399563	1.535084	1.55	0.218036	92.3	2.323654
400	50	50	60	0.002	69.217938	0.827651	2	0	96.5	1.104404
400	50	50	60	0.003	68.497781	1.482568	1.85	0.156493	96.2	2.016039
400	50	50	60	0.004	69.131156	1.204077	1.9	0.131481	97.1	1.579595
400	50	50	80	0	0	0	0	0	2	0
400	50	50	80	0.001	56.875094	7.065447	1.1	0.306789	79	9.72721
400	50	50	80	0.002	68.261812	1.152025	1.8	0.175308	95.6	1.699671
400	50	50	80	0.003	69.020625	0.905386	1.9	0.131481	96.9	1.254247
400	50	50	80	0.004	70.160437	0.696697	1.95	0.095519	98.4	1.022211
400	50	50	100	0	0	0	0	0	2	0
400	50	50	100	0.001	56.640406	4.759279	1	0.24005	78.8	6.700793
400	50	50	100	0.002	66.919781	1.460609	1.65	0.209041	93.4	2.304147
400	50	50	100	0.003	65.066531	6.628129	1.8	0.223474	91.1	9.11126
400	50	50	100	0.004	68.767031	0.810466	1.9	0.131481	96.5	1.38246

Table III
DATA FOR FIGURES 4, 9, AND 14.

Grid Size	No. Sensors	Range	AvoidCloseness Factor	MoveToCenter Factor	Coverage	Coverage CI	Density	Density CI	Connectivity	Connectivity CI
400	70	70	0	0	87.704781	5.106454	4.35	0.444255	92.142857	4.577379
400	70	70	0	0.001	91.028531	4.364078	4.45	0.352664	93.857143	4.48481
400	70	70	0	0.002	91.356656	5.548648	4.55	0.46948	93.785714	6.25872
400	70	70	0	0.003	91.206188	5.7126	4.45	0.48951	93.928571	6.363073
400	70	70	0	0.004	92.007875	2.776451	4.5	0.259284	94.857143	3.555008
400	70	70	1	0	95.873219	2.284415	4.85	0.156493	98	1.923663
400	70	70	1	0.001	98.62875	0.43092	5	0	100	0
400	70	70	1	0.002	97.062281	0.758056	4.95	0.095519	99.857143	0.27291
400	70	70	1	0.003	97.889594	1.33272	4.85	0.156493	99.571429	0.59726
400	70	70	1	0.004	96.144781	2.32116	4.95	0.095519	98.642857	1.934585
400	70	70	10	0	99.804156	0.135469	4.9	0.131481	100	0
400	70	70	10	0.001	99.777875	0.175265	4.95	0.095519	100	0
400	70	70	10	0.002	99.703906	0.15524	5	0	100	0
400	70	70	10	0.003	99.740625	0.14454	4.95	0.095519	100	0
400	70	70	10	0.004	99.882656	0.065712	5	0	100	0
400	70	70	20	0	99.962219	0.023361	4.95	0.095519	100	0
400	70	70	20	0.001	99.961344	0.025976	4.8	0.175308	100	0
400	70	70	20	0.002	99.905563	0.107495	4.85	0.156493	100	0
400	70	70	20	0.003	99.979844	0.016022	4.9	0.131481	100	0
400	70	70	20	0.004	99.935906	0.056111	4.8	0.175308	100	0
400	70	70	40	0	99.996344	0.006799	4.85	0.156493	100	0
400	70	70	40	0.001	99.994344	0.005702	4.8	0.175308	100	0
400	70	70	40	0.002	99.9975	0.002797	4.9	0.131481	100	0
400	70	70	40	0.003	99.996125	0.004463	4.95	0.095519	100	0
400	70	70	40	0.004	99.992906	0.006237	4.9	0.131481	100	0
400	70	70	60	0	99.998969	0.001235	4.95	0.095519	100	0
400	70	70	60	0.001	99.998312	0.002282	5	0	100	0
400	70	70	60	0.002	99.999906	0.000179	4.9	0.131481	100	0
400	70	70	60	0.003	99.997844	0.004057	4.75	0.189776	100	0
400	70	70	60	0.004	99.998219	0.003103	4.75	0.189776	100	0
400	70	70	80	0	99.998406	0.001787	4.7	0.20084	100	0
400	70	70	80	0.001	99.998875	0.001808	4.8	0.175308	100	0
400	70	70	80	0.002	99.999094	0.001134	4.9	0.131481	100	0
400	70	70	80	0.003	99.997187	0.003798	4.8	0.175308	100	0
400	70	70	80	0.004	100	0	4.85	0.156493	100	0
400	70	70	100	0	99.999281	0.000954	4.85	0.156493	100	0
400	70	70	100	0.001	99.99925	0.001433	4.8	0.175308	100	0
400	70	70	100	0.002	99.999781	0.000418	4.75	0.189776	100	0
400	70	70	100	0.003	99.998125	0.00271	4.75	0.189776	100	0
400	70	70	100	0.004	100	0	4.95	0.095519	100	0

Table IV
DATA FOR FIGURES 5, 10, AND 15.

Grid Size	No. Sensors	Range	AvoidCloseness Factor	MoveToCenter Factor	Coverage	Coverage CI	Density	Density CI	Connectivity	Connectivity CI
500	60	60	0	0	13.3439	4.273899	0	0	18.75	6.184599
500	60	60	0	0.001	19.62988	4.777076	0	0	29.25	7.113818
500	60	60	0	0.002	15.50612	4.907697	0	0	22.916667	7.26007
500	60	60	0	0.003	20.81954	4.457498	0	0	30.916667	6.724396
500	60	60	0	0.004	14.77412	4.66502	0	0	21.416667	7.528497
500	60	60	1	0	25.24722	6.527271	0.05	0.095519	34.666667	8.98066
500	60	60	1	0.001	34.81156	6.892943	0.15	0.156493	48.916667	10.02413
500	60	60	1	0.002	41.24912	7.100445	0.3	0.244018	58.833333	10.140093
500	60	60	1	0.003	31.88528	7.387464	0.15	0.156493	43.333333	10.542278
500	60	60	1	0.004	45.30376	7.587624	0.5	0.259284	65.083333	10.901751
500	60	60	10	0	14.03208	6.911349	0	0	18	8.122775
500	60	60	10	0.001	73.26554	3.385701	1.6	0.255553	92.333333	4.255452
500	60	60	10	0.002	77.2172	1.830751	1.9	0.131481	97.083333	2.012052
500	60	60	10	0.003	76.47712	1.800346	1.85	0.156493	96.666667	2.355627
500	60	60	10	0.004	77.14604	1.44947	2	0	97.25	1.352317
500	60	60	20	0	3.41092	3.134585	0	0	5.5	3.638339
500	60	60	20	0.001	76.9569	1.320358	1.9	0.131481	96.916667	1.447598
500	60	60	20	0.002	77.66396	1.025438	1.95	0.095519	97.416667	1.356257
500	60	60	20	0.003	77.55374	1.268512	1.95	0.095519	97.416667	1.540451
500	60	60	20	0.004	78.92426	1.216253	2	0	99.166667	0.632587
500	60	60	40	0	0	0	0	0	1.666667	0
500	60	60	40	0.001	76.78072	1.050322	2	0	97.25	1.42905
500	60	60	40	0.002	76.71402	1.132274	2	0	97.666667	1.334944
500	60	60	40	0.003	76.81528	0.907542	2	0	98.583333	1.014115
500	60	60	40	0.004	77.5179	0.752025	2	0	98.75	0.648209
500	60	60	60	0	0	0	0	0	1.666667	0
500	60	60	60	0.001	76.33636	1.091995	2	0	95.75	1.67327
500	60	60	60	0.002	77.08746	0.633266	2	0	98.5	0.649237
500	60	60	60	0.003	77.05612	0.806623	2	0	98.833333	0.734092
500	60	60	60	0.004	77.0006	0.858568	2	0	98.666667	0.91233
500	60	60	80	0	0	0	0	0	1.666667	0
500	60	60	80	0.001	76.00798	1.186268	2	0	96.083333	1.42905
500	60	60	80	0.002	76.1673	0.768257	2	0	97.583333	1.045206
500	60	60	80	0.003	77.37224	0.776328	2	0	98.75	0.860412
500	60	60	80	0.004	76.66856	0.704565	2	0	98.416667	0.672449
500	60	60	100	0	0	0	0	0	1.666667	0
500	60	60	100	0.001	74.3367	1.332957	2	0	94.833333	1.831956
500	60	60	100	0.002	76.15028	1.066849	2	0	97.083333	1.344403
500	60	60	100	0.003	76.54798	0.58296	2	0	97.916667	0.761736
500	60	60	100	0.004	76.71286	0.675331	2	0	98.75	0.795988

Table V
DATA FOR FIGURES 6, 11, AND 16.