Robotic Swarm Dispersion Using Wireless Intensity Signals

Luke Ludwig[1,2] and Maria Gini[1]

[1] Dept of Computer Science and Engineering, University of Minnesota
(ludwig,gini)@cs.umn.edu
[2] BAESystems Fridley, MN

Summary. Dispersing swarms of robots to cover an unknown, potentially hostile area is useful to setup a sensor network for surveillance. Previous research assumes relative locations (distance and bearing) of neighboring robots are available to each robot through sensors. Many robots are too small to carry sensors capable of providing this information. We use wireless signal intensity as a rough approximation of distance to assist a large swarm of small robots in dispersion. Simulation experiments indicate that a swarm can effectively disperse through the use of wireless signal intensities without knowing the relative locations of neighboring robots.

1 Introduction

Deploying large numbers of small simple robots in a decentralized swarm-like fashion is gaining recognition and popularity in many problem domains. One of the primary issues is how the swarm of robots will move. A common task is to spread out and cover an unknown area as thoroughly and quickly as possible in order to setup a sensor network for surveillance. This may be useful in areas hostile to humans such as disaster or military zones, or even planetary exploration.

In a swarm approach each robot is small, simple, and executes the same software program. Swarm methods bring many advantages. Since they are decentralized there is no single point of failure that can bring the entire system down. In fact many of the robots can fail and the swarm system will still function. Due to the simplicity of the robots they will be cheap to manufacture, another reason they are expendable. The designer of a swarm hopes that through individual actions based on local decisions the swarm as a whole will produce the intended emergent behavior. As in the natural world, unexpected emergent behavior may even occur.

A common theme in robot dispersion research is a connection to behavior in the natural world, whether it be of a biological, chemical, social, or physics based behavior. In particular there are many approaches that draw on the

general concept of repulsive and attractive forces between robots [13, 2, 9]. There are subtle differences between these approaches, but the general concept is the same; move away from neighboring robots, but not too far away. All of these approaches assume that the relative locations (distance and bearing) of neighboring robots is available to the robot through its sensors. This can be obtained using a 360^o laser range finder or an omnidirectional camera. Size is the limiting factor. The robots must be large enough to carry the laser range finder or have enough processing power to analyze the images from the camera which is inherently processor intensive. A common laser range finder, the SICK LMS 200, has dimensions of 15 cm x 18 cm x 15 cm. This is a rather large payload for a robot such as the University of Minnesota's Scout, which is only 11 cm long by 4 cm wide [12]. Small robots have the advantage of being cheaper, simpler, and less noticeable. Bob Grabowski from Carnegie Mellon has classified existing small robots by size in his Small Robot Survey, and at least half of the robots in this survey would be incapable of sporting either a 360^o laser range finder or an omni camera [6].

The question we attempt to answer is whether or not a swarm of small robots can be dispersed effectively without knowing the relative locations of neighboring robots. In particular, we use wireless signal intensity as an approximation of distance to assist in the dispersion. This requires a wireless 802.11 card on the robots, which is considerably smaller than the 360^o laser range finder.

Theoretically, signal intensity varies according to the law of inverse signal propagation, which simply means the signal intensity is proportional to the inverse square of the distance it travels. In a practical setting, the environment plays a huge role by providing obstacles that cause noise in this signal. However, it is unnecessary for the signal intensity to be very accurate in order to provide some indication to a robot for which way it should travel. The primary property needed is for the signal intensity to decrease over time as the distance between robots increases, and vice-versa.

2 Related Work

In 1992 Gage was the first to consider the problem of area coverage by a team of robots [4]. He categorized the problem into three types: blanket coverage, barrier coverage, and sweep coverage. Blanket coverage, the most similar to our research, has the objective of maximizing the total area covered by a static arrangement.

A promising experiment by Howard, Mataric, and Sukhatme considers how to deploy a mobile sensor network in an unknown hostile environment [9]. They use robots equipped with a 360^o laser range finder (4 meter range). No wireless communication is done. The dispersion is based on potential fields. Basically robots are repulsed by other robots and walls. They showed how a

mobile sensor network can be deployed without centralized control, localization, or communication, using only local rules based on potential fields. Their results are impressive, but this approach is not possible for very small robots due to the large sensors required.

Hsiang et. al [10] use a leader-follower approach based on local rules where the robots form chains emanating from a single source of robots. The robots attempt to follow walls by keeping the walls on their left. Their simulation experiment was ran in a discrete grid world and assumes "local sensors." It would be interesting to see if this algorithm could operate well in a more realistic simulation environment, such as provided by Player/Stage [5], while using only small proximity sensors (ex infrared) for following robots.

Batalin and Sukhatme [1] rely on the deployment of beacons into the environment to help coordinate a decentralized algorithm that uses only local interactions between the robots and beacons to cover an unknown area. To use this approach robots must be large enough and capable of carrying the static beacons. It is more flexible to mobilize and miniaturize everything. Any small robot can accomplish the same task as a static beacon by simply remaining stationary. One of the key aspects of the algorithm presented in this paper is deciding when a robot in motion should stop and become a stationary beacon.

Research has been done on utilizing wireless signal intensity to approximate distance for localization of sensor networks. Haeberlen et. al [7] show how accurate localization at the room-level can be obtained in an office setting in which signal intensity data has been predetermined. Tian He et. al. [8] examines the limitations of range based localization and proposes a novel range-free localization scheme.

3 Clique-Intensity Algorithm

We assume the robots have a few small and simple proximity sensors that extend at least a meter (ex. infrared) that allow the robot to avoid most collisions with walls and other robots. We also assume that the robots have a wireless 802.11 card and are capable of obtaining signal intensity measurements with incoming packets. This is a standard requirement of the 802.11 interface. No other sensors are needed during dispersion, although most likely robots will be carrying some form of a camera or other sensor which is meant to be utilized once the sensor network is in place. The processing power on small robots is limited, which makes analyzing images from a camera during real-time motion difficult. For many applications, it is likely the camera exists solely as a means of communicating images back to some central node for further processing or even human-in-the-loop analysis.

There are many ways to use wireless signal intensity to aid a swarm in dispersing throughout an unknown environment. In comparison to all of the repulsive/attractive dispersion research in which relative distance and bearing of neighboring robots is known [13, 2, 9], signal intensity gives only a rough

approximation of distance and no bearing information. This signal intensity must be tracked over time to determine which direction the robot should move. In a swarm of robots, each one may be in contact with many neighbors at a time. If it is known that one of the neighbors was stationary, then a robot could specifically reference the stationary robot's intensity and attempt to move in a direction of decreasing signal intensity until some threshold is reached. This is a key concept in the algorithm we developed.

The Clique Intensity Algorithm[3] is designed for a distributed homogeneous swarm, therefore the algorithm operates and runs from the perspective of a single robot in the swarm. The knowledge of each robot is a graph with robots as nodes and signal intensities between robots as weights. This graph is referred to as the connectivity graph. Robots share portions of their connectivity graphs with their neighbors such that each robot has the knowledge it needs to execute the algorithm. A clique is a graph or subgraph in which every node is connected to every other node. A maximal clique is not a subgraph of another clique. For each maximal clique in the connectivity graph a single robot is chosen to be the sentry for the clique, meaning it remains stationary. The other robots in the clique attempt to move away from the sentry, which is done by monitoring the change in the signal intensity over time. Each robot behaves in such a way that causes the entire swarm to disperse in an attempt to create cliques in the connectivity graph of size three or two. This is an attempt to triangulate the map which is known to be the most effective static configuration for the area coverage problem [11]. The algorithm is roughly composed of five basic steps (Figure 1).

Each robot in motion needs a sentry from which it monitors the signal intensity over time to determine which way to move. The primary decision to make is whether or not the robot is a sentry, and if not then the robot must decide which neighbor will be its sentry. This decision is made individually by each robot examining its connectivity graph and following a set of rules. The rules are structured such that each robot will arrive at the same decision as to which are sentries and which are in motion. Communication between robots of a bartering nature could be used to resolve the decision of which ones are sentries. This was not done since an attempt was made to avoid communication overhead and to keep the algorithm as simple as possible. The only communication between robots is the sharing of knowledge described above. See Figure 2 for a detailed description of the algorithm.

Considering the five steps in Figure 1, steps 1 and 3 dominate the time complexity. Step 3 is the Maximal Clique Enumeration Problem which is NP-Hard. One of the most common algorithms for this problem is the Improved BK by Bron and Kerbosch [3], whose worst-case time complexity has recently been proved to be $O(3^n/3)$ [14]. We use a variant of this algorithm with similar exponential complexity. To allow the algorithm to scale to a large number of robots, a quick calculation is done whenever the number of neighboring

[3] The idea behind the Clique Intensity Algorithm was proposed by Steven Damer.

Loop:
1. Update connectivity graph from neighbors' shared knowledge.
2. Share edges incident on me with neighbors.
3. Find all maximal cliques that I am in.
4. Determine the sentry for each maximal clique.
5. Choose and apply behavior based on sentries, cliques, and connectivity graph.

Behaviors:
1. Avoid Collisions Behavior:
   Utilize proximity sensors to avoid collisions
2. Seek Connection Behavior:
   Go in reverse for a bit and if this doesn't work pivot and move forward
3. Disperse Behavior:
   if my *sentry_intensity* is decreasing over time then go straight
   otherwise pivot for a bit and then move forward, and
   check for decreasing *sentry_intensity* again
4. Guard Behavior: Don't move.

Fig. 1. The high-level steps that roughly describes the Clique Intensity Algorithm. One of the four behaviors is applied each time through the loop.

robots exceeds a threshold. This quick calculation skips steps 1, 3, and 4, and instead each robot executes the disperse behavior and uses the neighbor with the lowest id as its sentry (Step 2 of Figure 2). See the discussion section for further explanation of this approach.

Step 1 of Figure 1, updating the connectivity graph, is the other dominant factor in the time complexity of this algorithm. Sharing the entire connectivity graph would provide the nice property that each robot has full knowledge of the connectivity graph across the swarm. Consider a scenario in which n robots begin close together such that the connectivity graph is fully connected, in which case the number of edges in the graph is $n*(n-1)/2$. Each robot would receive $n-1$ graphs from their neighbors, and must process each one to update its own knowledge. For this worst case scenario, the complexity in terms of the number of robots in the swarm becomes $O((n-1)*(n-1)*n/2) = O(n^3)$. In practice, when sharing the entire connectivity graph it becomes computationally difficult to have more than 20 robots in the swarm.

However, a robot does not need full knowledge of the connectivity graph for the Clique Intensity Algorithm to operate. There are two types of edges that are used by the algorithm. The first type are those that are incident on the robot, which are automatically given from the robot's wireless card. The second type are those that connect two robots which are both adjacent to the robot. These edges must be shared between robots. This is accomplished by each robot sharing only those edges incident on them to their adjacent neighbors. With a fully connected graph with n nodes, each robot receives

Loop:

1. If I am within proximity of a wall then apply Avoid Collisions Behavior
2. If number of neighbors is greater than *neighbor_threshold* then apply Disperse Behavior while using the neighbor with lowest id as my sentry.
3. Update connectivity graph from neighbors' shared knowledge
4. Share edges incident on me with neighbors.
5. If completely disconnected from all robots then apply Seek Neighbor Behavior
6. Find all maximal cliques that I am in.
   Two robots are connected if $signal\_intensity > \frac{1}{clique\_distance^2}$
7. For each maximal clique, choose a sentry
   A robot is a sentry at time i for clique j if:
   It is a sentry for another clique OR
   of all robots in clique j, it is in the most cliques OR
   if it is tied for being in the most cliques and has the lowest id in clique j
8. If I am a sentry then apply Guard Behavior
9. If I am an explorer, apply Disperse Behavior
   I am an explorer if:
   I am not a sentry AND
   my id is higher than at least 3 other robots in each of my cliques AND
   I am connected to more than 1 other robot.
10. Choose the sentry with the greatest intensity to be my sentry. Set *sentry_intensity* to be equal to the signal intensity between me and my sentry.
11. If $sentry\_intensity > \frac{1}{(clique\_distance-1)^2}$ then apply Disperse Behavior
12. Else apply Guard Behavior

Fig. 2. The detailed Clique Intensity Algorithm, from the perspective of a single robot in the swarm, hence the use of "I". Each time through the loop one of the four behaviors from Figure 1 is chosen and applied.

$n-1$ edges from $n-1$ adjacent neighbors, or a worst-case time complexity of $O(n^2)$. In practice this approach allows the algorithm to operate fluidly with 100 robots in the swarm.

A crucial aspect of the Clique Intensity Algorithm is how sentries are chosen. We would like as few robots as necessary to be sentries, so if a robot is a sentry for one clique, then it is considered a sentry for all of its cliques. If a robot on the perimeter becomes a sentry, then this will cause other robots to turn around and head back towards what is likely a more densely populated area. Therefore we want to encourage robots on the perimeter of the dispersion effort to continue dispersing. This is done by choosing the robot in a clique which is in the most cliques to be the sentry. This follows the idea that robots on the perimeter will be in fewer cliques. The tie-breaker for this situation is to simply choose the robot with the lowest id.

Creating cliques in the connectivity graph of size three or two is accomplished by designating certain robots as explorers. Given that they are still within range of at least one other robot, explorers will continue on out of range

of their sentries. Any robot that is not an explorer or sentry will continue moving in direction of decreasing sentry intensity until a certain threshold is reached and will then stop.

The desired clique distance is an important parameter that can be supplied as input to the algorithm. The algorithm tries to create cliques in which the robots are separated by a distance approximated by the clique distance. Depending on the problem at hand, it may be desirable to set the clique distance as high as the expected wireless connectivity range. In other situations, such as a map with small rooms, we may want the clique distance to be considerably smaller than the wireless connectivity range.

4 Simulation Experiments

Simulations were ran using the Player/Stage robot server and two- dimensional simulation environment [5]. Player/Stage is probably the highest fidelity, most realistic, robot simulation software in the world. Robot clients written for the Player server originally for simulation are often easily transferred to a real robot with little or no modification necessary. Player/Stage does not facilitate simulation of wireless message passing. The additional simulation infrastructure built for this experiment includes simulating wireless connectivity and message passing amongst robots via TCP/IP sockets, an automatic launching system to facilitate running the simulation in a distributed manner, and an embedded web server in each robot to track the state of the simulation. The wireless signal intensity was simulated by a direct calculation of the distance between each robot, using the inverse square law of signal propagation ($Intensity = \frac{1}{(distance\ traveled\ by\ signal)^2}$). The effect of the environment, walls in particular, was not taken into consideration.

The metrics tracked during each experiment include initial area coverage, final area coverage, and total running time. Each robot is considered to cover an area within a specified radius of the robot. The area coverage is computed in a post-processing step using a script that takes as input the number of meters per pixel, radius, and an image of the final dispersion map. The number of pixels within the radius of each robot, taking walls into consideration, is counted and then multiplied by the resolution of a single pixel to produce as output the area coverage in square meters and an image depicting the coverage. The maximum velocity allowed was 0.3 meters per second.

4.1 Experiment A

A simple experiment with 12 robots was performed on a cave-like map. The clique distance was set at 5, and it was assumed the robots covered an area within a radius of 2.5 meters. The robots disperse from the starting configuration shown in Figure 3 until the algorithm reaches an equilibrium in which the entire swarm is stationary. The initial area coverage is 25 square meters.

This experiment was ran 10 times and resulted in a mean area coverage of 128 square meters with a standard deviation of 6.27 square meters. It took between 60 and 100 seconds for the algorithm to reach an equilibrium. Notice there are no disconnections in the area coverage.

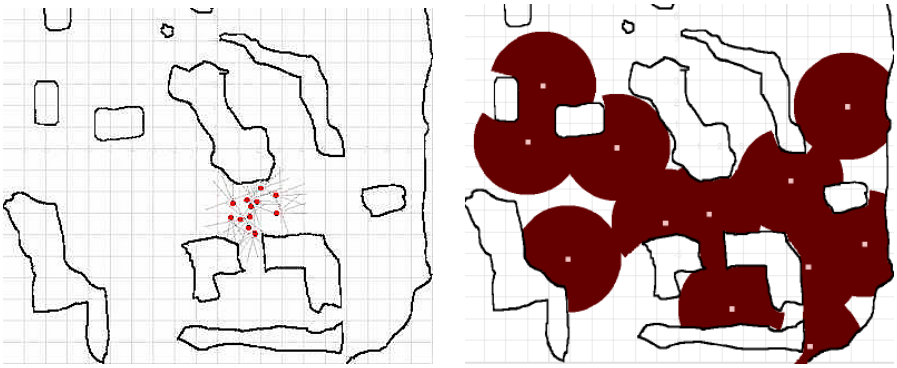

Fig. 3. Experiment A. Left: Starting configuration of experiment with 12 robots in a cave-like setting. A total of 25 square meters is covered at the start. Lines emanating from the robots indicate the range of the proximity sensors. Right: The output of the post-processing area coverage step for one run. Each robot covers an area within a 2.5 meter radius. This run resulted in an area coverage of 141 square meters in 80 seconds.

4.2 Experiment B

In this experiment, 100 robots started densely packed in the middle of a complicated floor of a hospital. The starting location is indicated by the light gray circle in Figure 4. The robots dispersed for 5 minutes and the clique distance was set at 5. It was assumed the robots covered an area within a radius of 4.5 meters. The algorithm was terminated after 5 minutes, even though an equilibrium was not achieved. The initial area coverage was 118 square meters. This experiment was ran 10 times and the mean area coverage was 1023 square meters with a standard deviation of 48 square meters. The is an improvement on average by a factor of 8.7.

5 Discussion

When a large number of robots are densely packed in a small area it is computationally too demanding for each robot to calculate cliques and share knowledge with its neighbors. The quick calculation done in step 2 of Figure 2 avoids this problem when the number of neighbors is above a threshold. Although

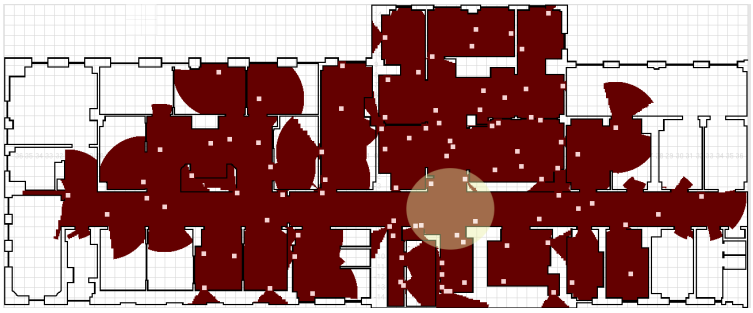

Fig. 4. Experiment B. 100 robots started in the light gray circle in the middle of the hospital map and dispersed outwards for 5 minutes. Each robot covers an area within a 4.5 meter radius. The total area covered for this particular run is 1063 square meters. The initial area coverage was 118 square meters.

this is necessary for a large number of robots, this causes inconsistencies that would not exist otherwise. As the swarm spreads out, some robots will switch modes to calculating cliques, while some will still be following the simpler approach. This means it is possible to have a situation in which one robot's sentry is moving (ex. 1 is connected to 2 and 2 is connected to 3). A way to avoid this problem would be to allow a robot to tell its sentry that it is using it as a sentry, in which case the sentry would know to stay put. Further experimentation is needed to verify this idea.

6 Conclusions and Future Work

The results obtained from both experiments indicate that a swarm of small robots can be dispersed effectively through the use of wireless signal intensities, without knowing the relative locations of neighboring robots. Using small robots instead of larger ones capable of carrying a laser range finder means that many more robots can be deployed for the same cost. It would be interesting to analyze how swarms with limited sensing abilities compare to swarms a fraction of their size with more powerful sensing abilities.

There is more work to be done on this topic. Variations in signal intensity should be modeled in a more realistic manner, by taking wall affects into consideration and applying gaussian noise. Eventually, experiments with real robots must be done to fully verify the potential of using wireless signal intensity in robot dispersion.

Acknowledgments

Partial funding provided by the National Science Foundation Industry/University Safety, Security, and Rescue Research Center, and by the Air Force under Contract No. FA8651-04-C-0191 to Architecture Technology Corporation.

References

1. M. Batalin and G. S. Sukhatme. Coverage, exploration and deployment by a mobile robot and communication network. *Telecommunication Systems Journal, Special Issue on Wireless Sensor Networks*, 26(2):181–196, 2004.
2. M. A. Batalin and G. S. Sukhatme. Spreading out: A local approach to multi-robot coverage. In *Proc. Int'l Symp. on Distributed Autonomous Robotic Systems"*, Fukuoka, Japan, 2002.
3. C. Bron and J. Kerbosch. Algorithm 457: Finding all cliques of an undirected graph. In *Proc. of the ACM*, 1973.
4. D. W. Gage. Command control for many-robot systems. In *19th Annual AUVS Technical Symposium*, pages 22–24, Huntsville, Alabama, June 1992.
5. B. P. Gerkey, R. T. Vaughan, K. Stöy, A. Howard, G. S. Sukhatme, and M. J. Matarić. Most valuable player: A robot device server for distributed control. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, pages 1226–1231, Oct. 2001.
6. B. Grabowski. Small robot survey. Carnegie Mellon University, http://www.andrew.cmu.edu/user/rjg/webrobots/small_robots_metric.html, 2003.
7. A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki. Practical robust localization over large-scale 802.11 wireless networks. In *Int'l Conf. on Mobile Computing and Networking (MOBICOM)*, pages 70–84, Philadelphia, PA, 2004.
8. T. He, C. Huang, B. M. Blumi, J. A. Stankovic, and T. Abdelzaher. Range-free localization schemes for large scale sensor networks. In *Int'l Conf. on Mobile Computing and Networking (MOBICOM)*. ACM, 2003.
9. A. Howard, M. J. Mataric, and G. S. Sukhatme. Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem. In *Proc. Int'l Symp. on Distributed Autonomous Robotic Systems"*, 2002.
10. T.-R. Hsiang, E. Arkin, M. A. Bender, S. Fekete, and J. Mitchell. Algorithms for rapidly dispersing robot swarms in unknown environments. In *Proc. 5th Workshop on Algorithmic Foundations of Robotics (WAFR)*, 2002.
11. B. Kadrovach and G. Lamont. Design and analysis of swarm-based sensor systems. In *Proc. of the Midwest Symposium on Circuits and Systems*, Aug. 2001.
12. P. E. Rybski, S. A. Stoeter, M. Gini, D. F. Hougen, and N. Papanikolopoulos. Performance of a distributed robotic system using shared communications channels. *IEEE Trans. on Robotics and Automation*, 22(5):713–727, Oct. 2002.
13. W. M. Spears, R. Heil, D. F. Spears, and D. Zarzhitsky. Physicomimetics for mobile robot formations. In *Autonomous Agents and Multi-Agent Systems*, 2004.
14. E. Tomita, A. Tanaka, and H. Takahashi. The worst-case time complexity for generating all maximal cliques. In *Proc. of Computing and Combinatorics Conference*, 2004.