BRADLEY University

ECE497 Project Proposal

Project Title: Area Coverage Optimization using Heterogeneous Mobile Robots

Eric Jones and Dakota Adra
Advisor: Dr. Suruz Miah

September 27, 2018

Contents

(a) (b) (c)

Figure 1: Mobile robots (agents) used in the current work: (a) Customized eduMIP, (b) Pioneer 3-DX, and (c) Khepera IV.

1 Introduction

In a framework of multi-agent systems, the coverage optimization of a spatial area using a team of mobile agents (robots) is of paramount importance due to its variety of uses in robotic applications. Most notability in roles where an agent (or a mobile robot) is preferable to a human or when a task is simply impossible for a human to perform. Typical applications for area coverage optimization in the field of mobile robotics include search and rescue, surveillance, environmental monitoring, cooperative estimation indoor navigation, among others. These problems are usually solved using an array of networked mobile agents operating collectively. Recently the implementation of area coverage algorithms has been restricted to a fleet of homogeneous robots at high monetary cost. The purpose of this research is to lower the costs of entry due to the selected robot platform by providing an easily accessible framework for heterogeneous robots that can be implemented both expediently and efficiently. Note that the terms robots and agents will be used interchangeably from now on.

This research is a continuation of a previous project conducted by authors in [3] wherein an area coverage algorithm with a static density is implemented in real-time. The current work seeks to extend this idea by implementing a density that is dynamic. The term density refers to the importance of a particular point in a given two-dimensional plane. In addition, we are looking to rigorously define a framework that multiple agents will use to implement the area coverage optimization algorithm. Therefore, we focus on developing a multi-agent framework using open-source software (MAFOSS) to standardize this algorithm. The MAFOSS is also aimed to be employed as an implementation platform for other multi-robot control algorithms, the leader-follower and the cyclic pursuit, for example. Furthermore, as a stretch goal, the MAFOSS is expected to be developed such that a user operating a smart phone is able to provide input to a networked mobile agents regarding the location of the high-density points. Additional stretch goals include the implementation of external sensors and collision detection or mapping.

To conduct this research, a team of mobile robots will be used for testing the area coverage algorithm. The mobile robots used in the current work are shown in Figure 1. Note that the mobile robot shown in Figure 1(a) is customized eduMIP robot that will be used for the current work. The details of these robots will be given in the implementation section of this document.

2 Background Study

Many papers have been published on algorithms requiring multiagent systems to optimize area coverage. However, few of these works describe a flexible framework that provides the means to achieve such a coverage task.

Authors in [1] Kilinic attempts to optimize area coverage of an interconnected network of sensors and actuators that act as a control system. Kilinic's goal is to maximize the area coverage of the wireless network control system (WNCS) whilst still maintaining convergence of the system in a large environment. Possible applications for the such a control system include smart grid, automatic management and navigation systems.

The system is arranged in several heterogeneous subnetworks that communicate ad-hoc. A single packet of information will hop multiple times through the network until it reaches the Kalman filter. Each of the subnetworks has a connection to the Kalman filter whilst remaining separate from each other. The Kalman filter is used to account for asymmetric packet arrival times as well as packet loss.

In [2] Lee attempts to optimize the area of a given region of interest for a time variant density. Dynamic density coverage has additional application compared to static density coverage. An example of this additional utility can be seen in a search and rescue scenario where the probability of a missing person being found in a particular area is time variant.

Lee's algorithm works by using voronoi regions and a robotic cost function to determine area coverage. Each robot is responsible for covering a particular voronoi region. Time information is used in the robot movement to account for the rapid changes in the density functions.

In [3] Miah adapts previous attempts of area coverage by using a fleet of heterogeneous modular cost-effective robots in real time. This allows for heterogeneous or non-uniform resource allocation and stabilizes the algorithm to allow for variation in the abilities of the constituent robots.

The algorithm in Miah's research is very similar to that of Lee's. Miah adds additional consideration for the robot actuation limitations for a heterogeneous case whilst Lee does not. Each individual robot is given a coverage metric to describe its area coverage performance.

In "Sensing and coverage for a network of heterogeneous robots" [4] Pimenta is attempting to adapt the area coverage model for intruder tracking. Multiple intruders in a specified region can be tracked using this method even if their location is unknown. The coverage is optimized and therefore the probability or detecting any existing intruders in the area is also maximized.

Pimenta's algorithm allows an individual robot to track an intruder in its Voronoi cell whilst the other robots respond to provide optimal area coverage. When an intruder is located within a given distance from a robot it will trigger that particular robot to follow the intruder.

In [5] Varposhti uses area coverage optimization to distribute mobile directional sensors over an area. The sensors can move around freely and can adapt in the case of an outage in a particular area. Such networks can be used in target tracking, search and rescue, and surveillance.

Varposhti uses a distributed learning algorithm to achieve area coverage in his research. Each sensor collaborates with its neighbors to determine the best position and orientation for each sensor. Each sensor moves in a random direction and turns in a random position until the coverage is maximized.

In [6] Yu attempts to find the optimal area coverage for deployable reconnaissance sensors. His approach involves considering the areas in which the sensors can be deployed, the connectivity and the coverage. Such an algorithm provides the ability to deploy sensors almost anywhere for the purpose of surveillance and intelligence.

Yu's algorithm uses genetic neural networks and particle swarm optimization to achieve optimal deployment. In the genetic algorithm the best performing configurations are passed on to the next generation whilst the poorly performing configurations are slowly phased out. Additionally a mutation rate is maintained to allow for other behavior that was not previously available in an older generation.

3 Mathematical Model

The following sub-sections will cover the work we have done so far. This includes the modelling of algorithm, its derivation, system simulation results in MATLAB, current and future design decisions, and some experiments we have run in the lab.

We began this work by looking at the project previously conducted and attempting to simulate their results in MATLAB from scratch. To start, we developed a way to map a set density to a square area. For this we represented any x-y coordinate within the square area as the set $\mathbf{q}$. Furthermore, we assigned $\bar{\mathbf{q}}$ to be the desired point for that agent to travel to in order to maximize coverage. As an example:

$$\bar{\mathbf{q}} = \begin{bmatrix} \bar{x} \\ \bar{y} \end{bmatrix} \tag{1}$$

The phi function is the measure of the density at any given point $\mathbf{q}$. Through simplification it can be seen that:

$$\phi(\mathbf{q}) = exp(\frac{-0.5 * norm(\mathbf{q} - \bar{\mathbf{q}})^2}{\sigma^2}) \tag{2}$$

$$\tag{3}$$

The variable σ seen in Equation (1) represents how the density is spread throughout the area. A larger σ denotes a larger spread of density. Further simplification yields:

$$\phi(\mathbf{q}) = exp(\frac{-0.5 * norm(\begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} \bar{x} \\ \bar{y} \end{bmatrix})^2}{\sigma^2}) \tag{4}$$

$$\phi(\mathbf{q}) = exp(\frac{-0.5 * norm((x - \bar{x})^2 + (y - \bar{y})^2)}{\sigma^2}) \tag{5}$$

See figure 2 for an example of the square area we are attempting to map.

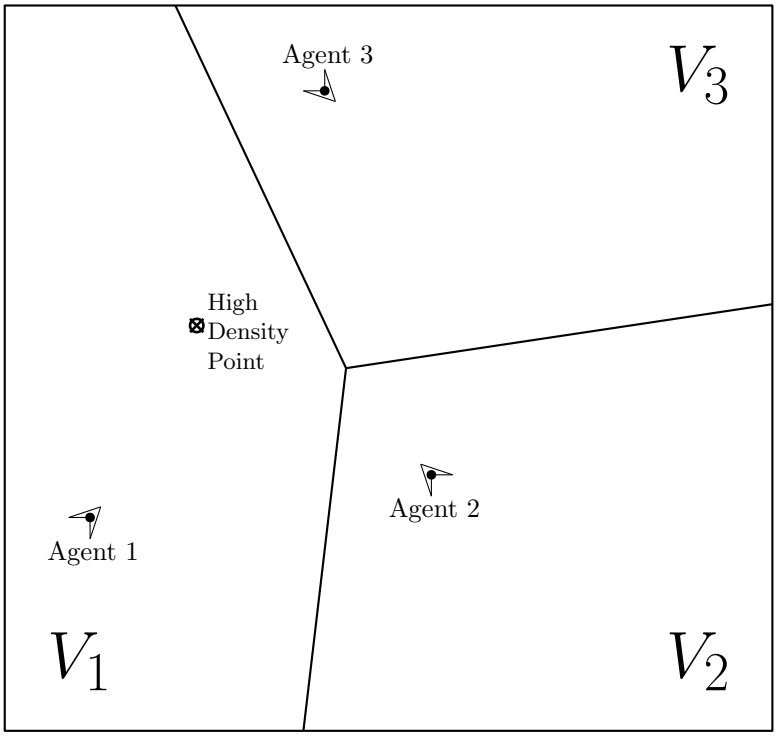


Figure 2: Plot of Agents in their Respective Voronoi Cells

This square area has been divided up into i Voronoi regions, for i number of agents operating in this area. These regions are represented in figure 2 as V1, V2, and V3. In our case we are using three agents, though note that this number is somewhat arbitrary and can be scaled to any reasonable number.

A Voronoi region is a partition of a plane such that all points within the partition's boundary must be closer to points within its boundary than to points outside. As such, the regions themselves must be convex two dimensional shapes. The Voronoi regions themselves represent a specific area covered by each agent within the region.

The density function itself is applied to each x-y coordinate in the square area. Figure 3 shows how a single density region is represented in three-dimensional space.

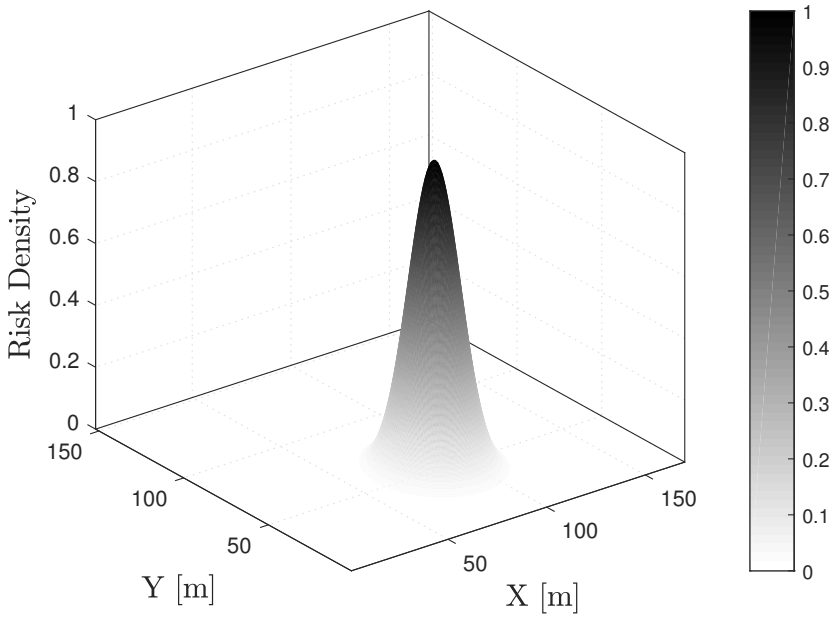


Figure 3: Density distribution centered at point (50,85)m.

Now that we had an idea of the how we would represent density we began looking at we would address the concept of coverage in our system.

Dr. Miah provided us with a derivation for coverage that I will detail here. The area coverage metric H in our system for i number of agents is defined as:

$$H = \sum_{i=1}^{n} \int_{V_i} \phi(\mathbf{q}) f(r_i^2) \mathrm{d}\mathcal{Q} \tag{6}$$

Where $f(r_i^2)$ is a function that models sensor characteristics.

$$f(r_i^2) = \alpha * e^{-\beta r_i^2} \tag{7}$$

Note that alpha and beta from the above represent ideal sensor parameters that we are not interested in modelling exactly. Our goal is to find a point in each Voronoi region $\mathbf{p}^{[i]}$ such that H is maximized. In order to do such we first find $\frac{\mathrm{d}H}{\mathrm{d}\mathbf{p}^{[i]}}$ and set it to zero.
For that let us define an individual case:

$$H_i = \int_{V_i} \phi(\mathbf{q}) f(r_i^2) \mathrm{d}\mathcal{Q}. \tag{8}$$

First let us take the derivative:

$$\frac{\mathrm{d}H}{\mathrm{d}\mathbf{p}^{[i]}} = \frac{\mathrm{d}}{\mathrm{d}\mathbf{p}^{[i]}} \sum_{i=1}^{n} H_i = \sum_{i=1}^{n} \int_{V_i} f(r_i^2) \phi(\mathbf{q}) \mathrm{d}\mathcal{Q} \tag{9}$$

As the derivative of the coverage metrics $\sum_{i=1}^{n} H_i$ is zero at all other indices other than the current agent index i, we can ignore those values and further simplify Equation (9).

$$\frac{\mathrm{d}H}{\mathrm{d}\mathbf{p}^{[i]}} = \frac{\mathrm{d}H_i}{\mathrm{d}\mathbf{p}^{[i]}} = \frac{\mathrm{d}}{\mathrm{d}\mathbf{p}^{[i]}} \int_{V_i} f(r_i^2)\phi(\mathbf{q})\mathrm{d}\mathcal{Q} = \int_{V_i} \frac{\mathrm{d}f(r_i{}^2)}{\mathrm{d}r_i{}^2} \frac{\mathrm{d}r_i{}^2}{\mathbf{p}^{[i]}} \phi(\mathbf{q})\mathrm{d}\mathcal{Q} \tag{10}$$

After the algebraic manipulation of derivative variables, note that we can find the derivative of the introduced range:

$$\frac{\mathrm{d}r_i}{\mathrm{d}\mathbf{p}^{[i]}} = \frac{\mathrm{d}}{\mathrm{d}\mathbf{p}^{[i]}} \sqrt{(x - x^{[i]]})^2 + (y - y^{[i]]})^2} \tag{11}$$

The function is then squared in order to simplify the derivative calculation:

$$\frac{\mathrm{d}r_i{}^2}{\mathrm{d}\mathbf{p}^{[i]}} = \frac{\mathrm{d}}{\mathrm{d}\mathbf{x}^{[i]}\mathrm{d}\mathbf{y}^{[i]}}[(x - x^{[i]})^2 + (y - y^{[i]})^2] = (-2x + x^{[i]}) + (-2y + y^{[i]}) = -2(\mathbf{q} - \mathbf{p}^{[i]}) \tag{12}$$

Using Equations (11) and (12), we are able to simplify Equation (10) even further:

$$\frac{\mathrm{d}H}{\mathrm{d}\mathbf{p}^{[i]}} = \int_{V_i} [-2\phi(\mathbf{q}) \frac{\mathrm{d}f}{\mathrm{d}r_i{}^2}](\mathbf{q} - \mathbf{p}^{[\mathbf{i}]})\mathrm{d}\mathcal{Q} \tag{13}$$

The quantity $[-2\phi(\mathbf{q})\frac{\mathrm{d}f}{\mathrm{d}r_i{}^2}]$ shown in Equation (13) is introduced as the modified $\phi(\mathbf{q})$ function. This function takes into account the sensor parameters. We call this new function $\tilde{\phi}(\mathbf{q}, \mathbf{p}^{[i]})$. From Equation (13):

$$\frac{\mathrm{d}H}{\mathrm{d}\mathbf{p}^{[i]}} = \int_{V_i} \tilde{\phi}(\mathbf{q}, \mathbf{p}^{[i]})(\mathbf{q} - \mathbf{p}^{[i]})\mathrm{d}\mathcal{Q} \tag{14}$$

Using properties of integrals, we split the integral into two:

$$\frac{\mathrm{d}H}{\mathrm{d}\mathbf{p}^{[i]}} = \int_{V_i} \mathbf{q}\tilde{\phi}(\mathbf{q}, \mathbf{p}^{[i]})\mathrm{d}\mathcal{Q} - \mathbf{p}^{[i]} \int_{V_i} \tilde{\phi}(\mathbf{q}, \mathbf{p}^{[i]})\mathrm{d}\mathcal{Q} \tag{15}$$

Finally, we make an algebraic substitution and compare this equation to the mass and centroid equations previously discussed:

$$\frac{\mathrm{d}H}{\mathrm{d}\mathbf{p}^{[i]}} = \int_{V_i} \tilde{\phi}(\mathbf{q}, \mathbf{p}^{[i]})\mathrm{d}\mathcal{Q} \frac{\int_{V_i} \mathbf{q}\tilde{\phi}(\mathbf{q}, \mathbf{p}^{[i]})\mathrm{d}\mathcal{Q}}{\int_{V_i} \tilde{\phi}(\mathbf{q}, \mathbf{p}^{[i]})\mathrm{d}\mathcal{Q}} - \mathbf{p}^{[i]} \int_{V_i} \tilde{\phi}(\mathbf{q}, \mathbf{p}^{[i]})\mathrm{d}\mathcal{Q} \tag{16}$$

We know that the mass is defined as the integral of our density function relative to a bounded area V_i. As such the modified mass is expressed as $\tilde{M}_{V_i} = \int_{V_i} \tilde{\phi}(\mathbf{q}, \mathbf{p}^{[i]})\mathrm{d}\mathcal{Q}$. The modified centroid is expressed as any point within the bounded area scaled by our modified $\tilde{\phi}(\mathbf{q}, \mathbf{p}^{[i]})$ function. We call it $\tilde{C_{V_i}} = \frac{\int_{V_i} \mathbf{q}\tilde{\phi}(\mathbf{q}, \mathbf{p}^{[i]})\mathrm{d}\mathcal{Q}}{\int_{V_i} \tilde{\phi}(\mathbf{q}, \mathbf{p}^{[i]})\mathrm{d}\mathcal{Q}}$.
Therefore we can express:

$$\frac{\mathrm{d}H}{\mathrm{d}\mathbf{p}^{[i]}} = \tilde{M}_{V_i}\tilde{C_{V_i}} - \mathbf{p}^{[i]}\tilde{M}_{V_i} = 0 \tag{17}$$

$$\tilde{M}_{V_i}(\tilde{C_{V_i}} - \mathbf{p}^{[i]}) = 0 \tag{18}$$

$$\tilde{C_{V_i}} = \mathbf{p}^{[i]} \tag{19}$$

In conclusion, as the mass can not be zero, the coverage is maximized when the position of the agent is equal to the modified centroid of a bounded area.

4 Milestones

We have decided to split up the implementation of the project into three main parts. A homogeneous case, a heterogeneous case, and various stretch goals that we will attempt to implement. In both the heterogeneous and homogeneous cases we will be testing the systems functionality with a static (time-invariant) and variable (time-variant) density.

- Homogeneous Case
  - Static Density
  - Variable Density
- Heterogeneous Case
  - Static Density
  - Variable Density
- Stretch Goals
  - Smart Phone Application
  - Interfacing External Sensors for Collision Detection or Mapping

5 Functional Requirements

5.1 High-Level System Block Diagram

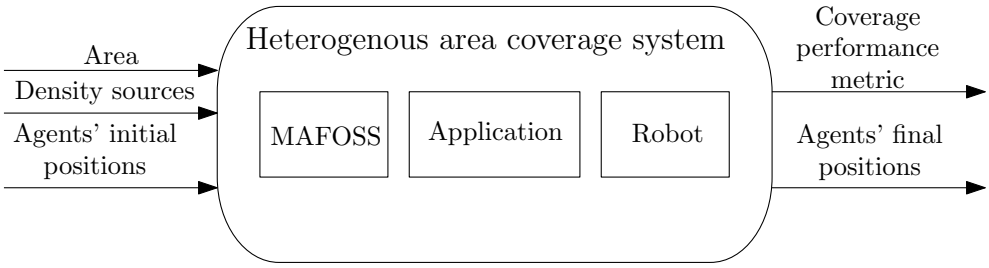


Figure 4: Functional Diagram

As can be seen in figure 4 the Heterogeneous area coverage system works by receiving a defined convex area, density sources, and agent positions from the user. The system then moves the robots around using the MAFOSS framework until they maximize their coverage metric or meet some coverage criteria. At this point, the agents will be in their final and optimal positions. Additional user input can be given via the smart phone application functionality such as, altering the density.

5.1.1 Inputs

The inputs for our system are the area that the robots will be operating in, as well as the locations of the density sources in this area. The robots initial positions are also given as this is crucial to the algorithm's operation.

- Area - The predefined two-dimensional operational area of the robots in our system. Note that this area must be convex for the algorithm to function.
- Density source - A simulated metric used to excite the system. In order to optimize their coverage the robots will try to cover regions that have higher densities. All other things being equal, the algorithm determines that a region of higher density requires coverage more than a region of lower density.
- Agents' initial positions - The agents' initial X and Y positions in a two dimensional X-Y plane.

5.1.2 Heterogeneous area coverage system

The heterogeneous area coverage system is the high level system that provides the networking infrastructure, agent movement, and application services.

- MAFOSS - A Multi-Agent Open-Source Framework that is used as the backbone for the multi-agent logistics.
- Application - The communication of a user's desired density to the system from a mobile device. Note that this sub-block of the Heterogeneous Area Coverage System is not required for it's operation.
- Agent - The mobile robot or vehicle that is being used to achieve area coverage.

5.1.3 Outputs

- Coverage performance metric - A description of how optimally the area is covered.
- Agents' final positions - The agents' final X and Y coordinates prior to the execution of the area coverage algorithm. The final positions of the robots should be at the centroids of their respective Voronoi regions.

5.2 MAFOSS System Architecture

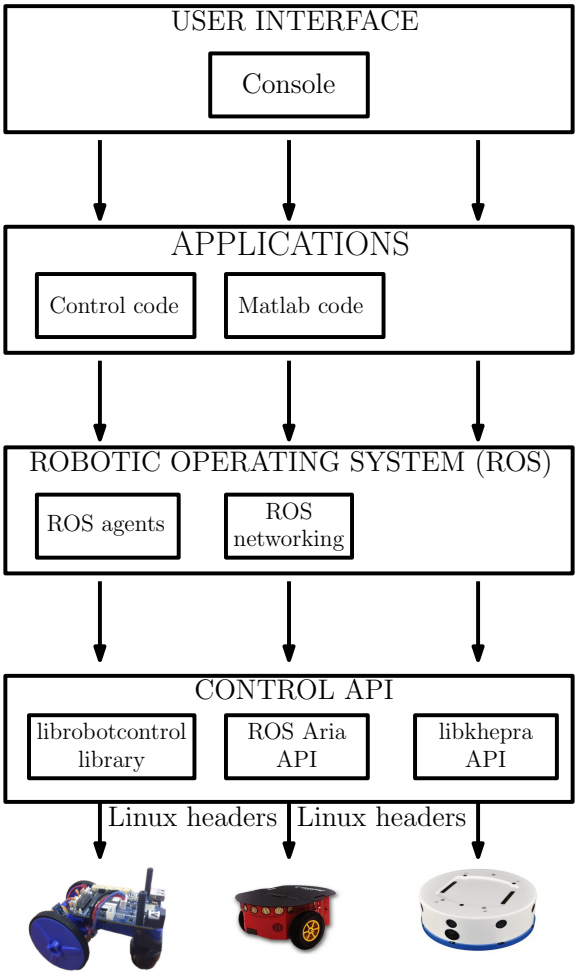


Figure 5: MAFOSS Software Architecture

- **User Interface** - The UI in the MAFOSS system framework takes user input regarding the position of high density points and feeds that into the Application. This data is sent through the console, when the algorithm is running, and recorded in a ROS Log file.
- **Applications** - The Applications section relates to the control code running on the agents in the system. This code will be ran in a loop, sending and receiving the currentPose and desiredPose of the robots. The data transfer will be accomplished through the Robotic Operating System (ROS) and Matlab's Robotic System Toolbox. Note that Matlab is not open source; however, in order to speed up implementation we decided to include it. The alternative is to facilitate all of the ROS master connections through one of the Beaglebone blues.
- **Robotic Operating System** - The Robotic Operating System (ROS) is a flexible framework

for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms. In our project we will be using ROS's networking abilities as well as it's ability to define individual agents as nodes to standardize the way we approach our algorithms implementation.

- **Control API** - The Control APIs in this architecture are each dependent on the target robot we are attempting to interface with. For the Beaglebone we use the well-documented *libbrobotcontrol* library, as it is specifically intended to be implemented on the Beaglebone with ROS. For the Khepra and the Pioneer, we will be using the proprietary interfacing firmware developed by their respective companies. The direct hardware interfacing will be done through the Linux Headers, as described in each API.

5.3 ROS Networking

In the MAFOSS System architecture ROS handles the networking for us. Figure 6 details on how ROS handles the connections between the agents.

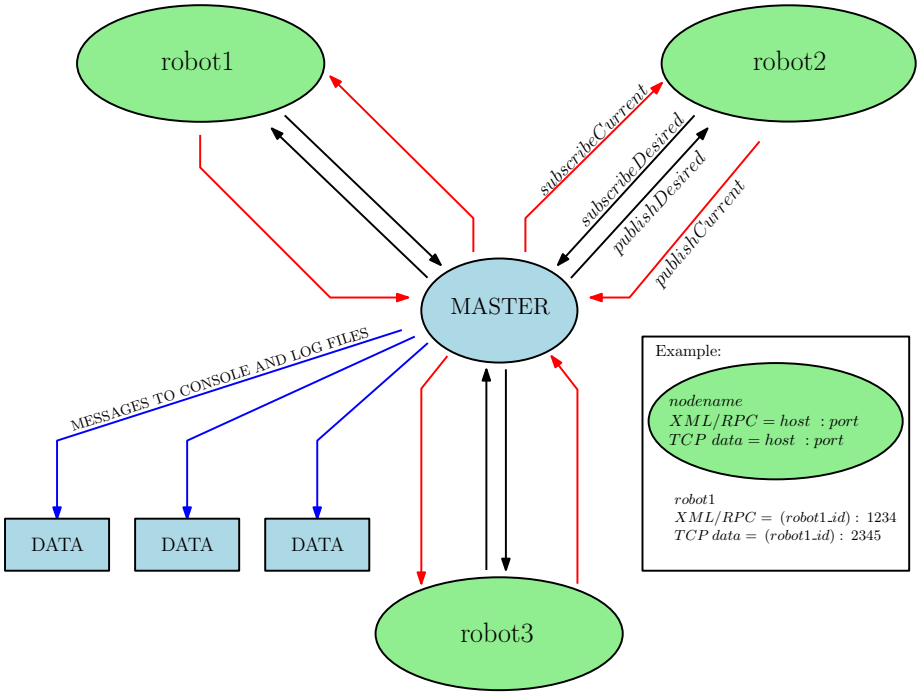


Figure 6: ROS Networking

- **Nodes** - Nodes are executables that uses ROS to communicate to other nodes through TCPROS across a corresponding Topic. In the above graph, the nodes in question are *robot1*, *robot2*, and *robot3*.

- **Topics** - Topics are named buses over which nodes exchange messages. In general, nodes are not aware of who they are communicating with. Instead, nodes that are interested in

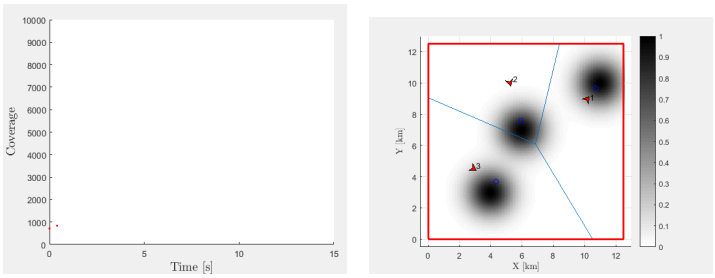
data *subscribe* to the relevant topic; nodes that generate data *publish* to the relevant topic. The graph lists the 4 topics associated with this system as *subscribeCurrent*, *subscribeDesired*, *publishCurrent*, and *publishDesired*. Each node communicating with the master in the system will get its own set of 4 topics. For robot one a full topic name could be *robot1PublishDesired*: this convention will be carried throughout the system independent of the number of agents.

- **Master** - The ROS Master provides naming and registration services to the rest of the nodes in the ROS system. It tracks publishers and subscribers to topics as well as services. The role of the Master is to enable individual ROS nodes to locate one another. Once these nodes have located each other they communicate with each other peer-to-peer.
- **Output Data** - Output Data in our system will be sent to the laptop computer running the master node as the raw x and y positions. This data will also be sent to the corresponding ROS system log file for each node. For simulation purposes, we will be using the log files created after running tests to create graphs relating our robot's desired position and it's actual position.
- **Example** - Note the example next to the graph. This gives a more in-depth view at what occurs inside the nodes on the graph. Given a publisher URI, a subscribing node negotiates a connection, using the appropriate transport, with that publisher, via XMLRPC (Extensible Markup Language Remote Procedure Call). The result of the negotiation is that the two nodes are connected, with messages streaming from publisher to subscriber. Each transport has its own protocol for how the message data is exchanged. For example, using TCP, the negotiation would involve the publisher giving the subscriber the IP address and port on which to call connect. The subscriber then creates a TCP/IP socket to the specified address and port. The nodes exchange a Connection Header that includes information like the message type and the name of the topic, and then the publisher begins sending serialized message data directly over the socket.

For our system, the above graphic gives a holistic view on its operation. Note that the ports specified to the TCP/IP sockets for the nodes are arbitrary. Further, the full URI for the master node will include the standard 11311 port.

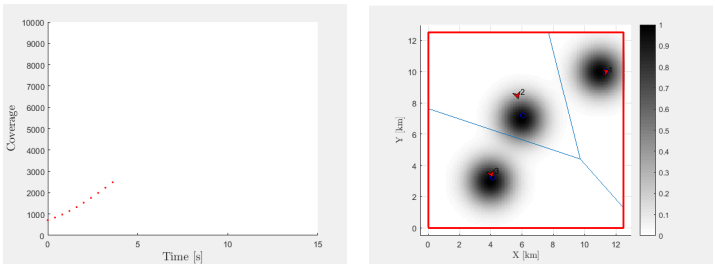
5.4 Simulation Results

Now that we had a algorithm, we could begin simulation in MATLAB. We began testing a series of algorithms in Matlab: line-following, leader-follower, and the final area coverage algorithm. After we validated these algorithms in this environment we moved on to Coppelia Robotic's virtual experimentation platform (VREP). VREP was chosen because it is based on a distributed control architecture wherein each object/model can be individually controlled via an embedded script, a plugin, a ROS node, a remote API client, or a custom solution. This makes VREP very versatile and ideal for multi-robot applications. Furthermore, the simulator has already designed a physics model for the Pioneer and Khepra. Team members designed the Beaglebone model in the VREP environment to be used for a simulators. The left-most picture in the following figures is the coverage metric H. When this is value reaches its maximum value, the coverage in the system is maximized. We can see from the following graphs that this is the case when the agents are moving to their modified centroids.



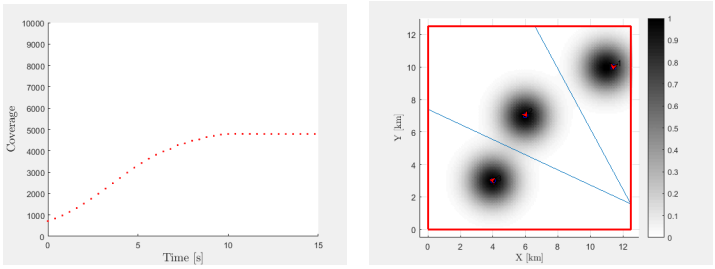
(a) (b)

Figure 7: Starting Matlab area coverage algorithm.



(a) (b)

Figure 8: In progress Matlab area coverage algorithm.



(a) (b)

Figure 9: End Matlab area coverage algoritm.

6 Parts List

Vendor	Part	Price	Quantity
Digikey	JST SH Jumper 6 Wire Assembly	$4.50	3
Digikey	JST SH Jumper 4 Wire Assembly	$9.00	6
Amazon	SanDisk Ultra 8GB Memory Card	$23.25	3
Amazon	USB 2.0 Male to Micro B cable (3 Pcs)	$7.99	1
Amazon	10 Sets of Mini Micro ZH 2Pin JST	$4.99	1
Amazon	*M*2.5*x*30*mm* Socket Head Cap Screws (20 Pcs)	$4.99	1
Renaissance Robotics	Beaglebone Blue	$219.00	3
Renaissance Robotics	Robotics eduMIP Kit	$150.00	3
Polulu	Pololu Ball Caster with 1″ Ball	$29.75	5

Table 1: List of ordered parts

7 Timeline for Milestones

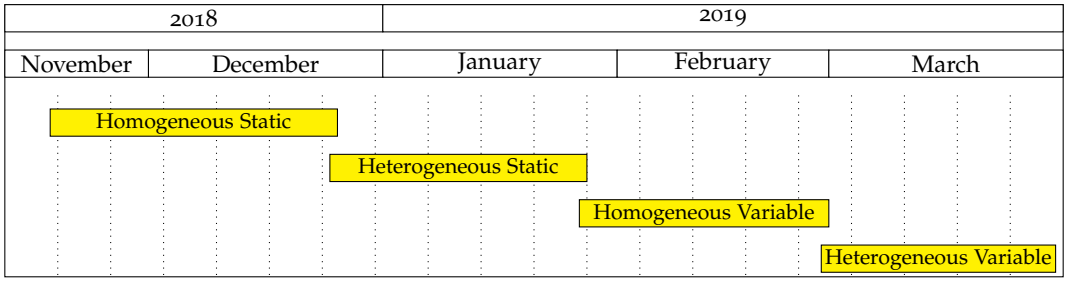


Figure 10: Timeline for Completion

- Homogeneous Case
  - Static Density
  - Variable Density
- Heterogeneous Case
  - Static Density
  - Variable Density

References

[1] D. Kilinc, M. Ozger, and O. B. Akan, "On the maximum coverage area of wireless networked control systems with maximum cost-efficiency under convergence constraint," *IEEE Transactions on Automatic Control*, vol. 60, no. 7, pp. 1910–1914, July 2015. 4

[2] Y. D.-M. S. G. Lee and M. Egerstedt, "Multirobot control using time-varying density functions," *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 489–493, April 2015. 4

[3] M. S. Miah and J. Knoll, "Area coverage optimization using heterogeneous robots: Algorithm and implementation," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 6, pp. 1380–1388, June 2018. 4

[4] Q. L. V. K. D. R. R. C. M. Luciano C. A. Pimenta, Mac Schwager and G. A. S. Pereira, "Simultaneous coverage and tracking (scat) of moving targets with robot networks," *IEEE Conference on Decision and Control*, vol. 47, no. 1, pp. pp. 3947–3952, September 2009. 4

[5] S. A. M. Varposhti, P. Saleh and M. Dehghan, "Distributed area coverage in mobile directional sensor networks," *2016 8th International Symposium on Telecommunications (IST)*, vol. 0, no. 0, pp. 18–23, September 2016. 4

[6] G. X. Z. Yu, G. Shan and X. Duan, "Method of multi-sensor optimal deployment for area coverage," *2018 International Conference on Electronics Technology (ICET)*, vol. 0, no. 0, pp. 116–119, May 2018. 4