Connectivity Maintenance and Coverage Control in Dynamic Environments

Jacob Kruse*
*School of Computing, University of Georgia
Email: jrk25310@uga.edu

***Index Terms*—Voronoi, Coverage Control, Connectivity Maintenance, and Dynamic Environments.**

I. INTRODUCTION

Multi-robot systems are being increasingly deployed in the real world to solve a wide range of tasks. Using communication and cooperation, these systems have enhanced capabilities, allowing them to perform tasks such as environmental mapping, search and rescue, or warehouse automation. Utilizing multiple robots opens up a world of possibilities and applications that are not possible for a single robot to complete efficiently or even at all.

One of the oldest and most researched topics in multi-robot systems is coverage control. In coverage control, the objective is to effectively spread the robots throughout the environment in order to achieve the best sensing capability. This is better known as the Locational Optimization problem, and much work has been done to solve this problem. The most widely recognized solutions make use of Lloyd's controller or the Move-to-Centroid controller, where the robots cover the environment by traveling to the centroid of their respective Voronoi regions. Although this solution guarantees optimal coverage of the environment, it works under the assumption that the robots never lose connection or the ability to communicate with one another. In environments where connection ranges may be limited, the classic solution will fail once the robots move out of communication range, as the necessary information will not be able to propagate between the robots.

To solve this problem, research has been done for the coverage control problem with the implementation of connectivity constraints. For large environments or those with tough communication conditions, it is necessary to ensure that each robot maintains the proper connections to effectively collaborate. One of the most notable works is [3], which utilizes control barrier functions to enforce connectivity constraints. Although the paper provides an effective approach that optimally covers the environment while maintaining connectivity, it is flawed in the sense that it assumes a static and constant communication range. This algorithm is designed to be applicable to the real world, but the assumption of a static communication range is unrealistic. The purpose of this paper is to provide a solution to this problem.

II. RELATED WORK

A. Centroidal Voronoi Tessellations

The classic solution to the Locational Optimization problem divides the environment into Voronoi regions, calculates the centroid, and assigns control inputs to each robot so that they move toward the centroid of their cell [1], [2]. It has long been known that this approach is optimal and much work has been done to improve on the basic algorithm. Using weights to represent different qualities of each robot, this idea can be extended to produce larger or smaller regions for each robot. This approach produces what is known as the Power Diagram, which has large applications in heterogeneous systems.

B. Connectivity Maintenance

A number of methods have tried to address the issue of maintaining connectivity in coverage control. Although many of the works guarantee the preservation of communication, they do not guarantee any optimality in terms of covering the environment. However, a recent approach [3] provides an optimal controller for coverage while also accounting for connectivity maintenance. In order to produce an optimal solution, the robots have to be as unrestricted as possible. The approach determines the connectivity graph of each robot by calculating the Minimum Spanning Tree where the weights relate to each robot's distance and speed. This tree represents the least-constrained connections and is used for determining the constraints. These constraints are calculated between the robots that share a connection based on the Minimum Spanning Tree, and ensure that while the robots are attempting to reach their centroids, they do not lose communication. This method has guaranteed optimality but assumes a static communication range to build their constraints. In my work, I have allowed these communication ranges to adapt to the environment.

III. BACKGROUND

The Voronoi Diagram is a classical way to divide an environment into regions, and it is commonly used in coverage control. For all points q in a domain Q, each robot is assigned all of the points that are closest to it. The Voronoi region for robot i is defined by the following,

$$V_i = \{q \in Q \mid \|q - x_i\| \leq \|q - x_j\|,\ \forall j \neq i\},$$

where x_i and x_j are the positions of robot i and all of its neighbors j, respectively. For each of these Voronoi partitions, the centroid of each cell is calculated with the following,

$$C_{V_i} = \frac{\int_{V_i} q\,\phi(q)\,dq}{\int_{V_i} \phi(q)\,dq},$$

where $\phi(q)$ is the density function for the environment. With the centroids of each cell, each robots' control input can be calculated to drive them towards these points. This is known as the Move-to-Centroid controller, and it is defined by the following,

$$\dot{x}_i = \hat{u}_i = k_p(C_{V_i} - x_i).$$

Although this algorithm will provide an optimal solution to cover the environment, communication constraints are not considered. Building on the idea of the Move-to-Centroid controller, the method from [3] implements control barrier functions in order to build these constraints. In order to ensure that each robot stays within the defined communication range R_c, they define the following condition,

$$h^c_{i,j}(x) = R_c^2 - \|x_i - x_j\|^2$$
$$\mathcal{H}^c_{i,j} = \left\{\mathbf{x} \in \mathbb{R}^{2n} : h^c_{i,j}(\mathbf{x}) \geq 0\right\},$$

where $\mathcal{H}^c_{i,j}$ is the set of all possible positions where robots i and j will be connected. In order to make the solution optimal, the condition above has to be as least constraining as possible to the desired control input, $\hat{u}$, from the Move-to-Centroid controller. Because of this, the conditions are only applied to a connectivity graph G_c. In order to quantify the connectivity of each robot, they define weights with the following,

$$w_{i,j} = -[\dot{h}_{i,j}(\mathbf{x}, \hat{u}_i, \hat{u}_j) + \gamma h_{i,j}(\mathbf{x})], \forall (v_i, v_j) \in \mathcal{E},$$

where $(v_i, v_j) \in \mathcal{E}$ represents each possible connection between the system of robots. These are used to build a weighted graph G of the entire system, where the optimal connectivity graph G_c^* corresponds to the minimum spanning tree of the weights. With this connectivity graph, they define the complete constraints with the barrier certificate defined by,

$$\mathcal{B}(\mathbf{x}, G_c^*) = \{\mathbf{u} \in \mathbb{R}^{2n} : \dot{h}_{i,j}(\mathbf{x}, \hat{u}_i, \hat{u}_j) + \gamma h_{i,j}(\mathbf{x}) \geq 0, \forall (v_i, v_j) \in \mathcal{E}\},$$

where γ is a user-defined variable for the constraints. It has been proven that if the original state of robots is connected and if the control input $\mathbf{u}$ stays within the barrier certificate, the system will stay connected. With these definitions in place, they define their objective function with the following,

$$\mathbf{u}^* = \arg\min_{G_c, \mathbf{u}} \sum_{i=1}^{n} \|u_i - \hat{u}_i\|^2,$$

such that the following constraints are held,

$$G_c = (V, \mathcal{E}) \subseteq G$$
$$\mathbf{u} \in \mathcal{B}(\mathbf{x}, G_c)$$
$$\dot{\mathcal{H}}(\mathbf{x}) \leq 0.$$

Using a Quadratic Programming solver, the control inputs $\mathbf{u}^*$ can be calculated and applied to the robots to achieve a connection aware, optimal coverage solution.

As effective as this method is, it falls short by assuming a constant communication range R_c. Real-world environments will cause problems for the algorithm, as they will likely have areas with higher and lower communication capabilities. For areas with lower bandwidth, the described method will not adapt to the new changes, causing it to disconnect as the communication range shrinks. In areas with higher communication capabilities compared to what was defined, the solution will fail to find an optimal solution. This is addressed with the method in this paper and is described in the next section.

IV. Methodology

Although it has been stated that the method from [3] is flawed, the main contributions of this paper are implemented with some changes to allow for adaptation in dynamic environments in the proposed algorithm. The first change occurs with the connectivity constraint, which is defined by the following,

$$h^c_{i,j}(x) = R_c^2 - \|x_i - x_j\|^2.$$

Because they are assuming a constant communication range R_c for all robots, this needs to be reformulated to a different value. The location of each robot is the main factor that determines their communication range. For example, dense areas may cause a robot's range to shrink, while open regions may cause a robot's communication to increase. Because of this, the constant communication rangeR_c is replaced with a dynamic variable. The new condition is now defined as

$$h^c_{i,j}(x) = R_{i,j}^2 - \|x_i - x_j\|^2,$$

where $R_{i,j}$ is the minimum communication range of robot i and j. The minimum value is used to ensure that both robots are within the smallest of the two ranges, which ensures that the robots can communicate in both directions. Due to the modification of condition $h^c_{i,j}(x)$, the time derivative $\dot{h}^c_{i,j}(x)$ will be affected. Although it is never explicitly stated in the paper, the following is true for their definition,

$$\dot{h}^c_{i,j}(x) = -2(x_i - x_j)(\dot{x}_i - \dot{x}_i)^T.$$

As you can see, the communication range is not included in the above equation because it is constant and does not change over time. However, due to the inclusion of a dynamic communication range $R_{i,j}$, the new time derivative will be the following,

$$\dot{h}^c_{i,j}(x) = 2R_{i,j}(\dot{R}_{i,j}) - 2(x_i - x_j)(\dot{x}_i - \dot{x}_i)^T.$$

Although $h^c_{i,j}(x)$ and $\dot{h}^c_{i,j}(x)$ have been reformulated, the remaining equations from the original paper will remain the same. Implicitly, they will have changed due to the new values for $h^c_{i,j}(x)$ and $\dot{h}^c_{i,j}(x)$, but they will still use these variables in the same way.

V. EXPERIMENTS

In order to evaluate the effectiveness of the proposed algorithm, experiments within the Robotarium environment were conducted. These experiments were performed with five robots and within a 3.0 by 2.0 meter environment. The robots began in different positions for each scenario, but they were initially all connected in each case. The code for the experiments can be found on GitHub.

The proposed algorithm was compared with two baselines: the standard Voronoi / Move-to-Centroid controller [1], [2] and the Connectivity Maintenance controller from [3]. The Connectivity Maintenance baseline assumes a communication range of 0.901 meters, which corresponds to one-fourth of the range necessary to cover the entire environment. An underlying density function is generated for each scenario which determines the dynamic communication range of each robot depending on where it is located. This communication range can vary by 30% of the static communication range, which corresponds to a range between 0.631 and 1.172 meters. Note that this density function has nothing to do with the function $\phi(q)$, and it is assumed that the entire environment has a constant importance factor.

Experiments were carried out on four different scenarios. The thing that changed between these four scenarios was the underlying density function, which represents the dynamic communication range for each point in the environment. Figures 1 and 2 show the results for two different environments generated from Gaussian functions. Figures 3 and 4 depict the results for two randomly generated density functions. The top 3 images in each figure represent the final configurations for both baselines and the proposed algorithm. The colored dots represent the robots, and the colored circles represent the corresponding communication ranges determined by the environment. The black lines portray the Voronoi regions for each configuration and the black x's represent the centroids of those regions. The underlying distribution function is represented by the shading, where yellow corresponds to the minimum communication range and purple corresponds to the maximum communication range. The Connectivity Maintenance baseline and the Dynamic Communication extension contain solid white lines to represent the connectivity graphs for the system. The Connectivity Maintenance baseline has dashed white lines to represent the assumed static communication range.

The bottom three images show the quantitative results for each scenario. The first graph is a comparison of Algebraic Connectivity (λ_2) throughout each algorithm's execution, which represents the connection strength for a given graph. If this value is positive, then the graph is connected, where larger values represent a stronger connection. If this value is zero, the graph is disconnected. The second graph displays a comparison of the sensing cost throughout the execution of each method. This is calculated with the following equation,

$$\mathcal{H} = \sum_{i=1}^{n} \int_{q \in V_i} \|q - x_i\|^2 \phi(q) dq$$

where the lowest possible cost is desired. The final graph compares the iteration of convergence for each method. Convergence refers to the time in which the algorithm has reached its optimal coverage. For the experiments, the algorithm is determined to have converged once all robots have moved less than 0.002 meters from the previous iteration.

Looking first at the connectivity graph for each scenario, it is clear that the Standard Voronoi and Connectivity baselines lose connection in all four of the scenarios. However, the positive connectivity value throughout the entire execution of the Dynamic Communication extension shows that the system maintains a connection in all four scenarios. Looking at the cost comparison, it might seem that the proposed method is inferior, especially looking at Figure 3. Although this is correct, the difference in cost is usually negligible and is one of the trade-offs for maintaining connectivity. When it comes to the convergence comparison, there is not enough consistency to make any reasonable assumptions. Because the environment is dynamic, the communication ranges can differ, which can lead to less or more restrictive constraints than those in the Connectivity Maintenance baseline. Despite this, both connection-aware algorithms generally converge quicker than the Standard Voronoi method because their constraints prevent them from traveling the full distance to the centroids.

VI. CONCLUSION

This paper provides an extension to the Connectivity Maintenance algorithm from [3]. Incorporating a dynamic communication range allows the environment to adapt to difficult environments and maintain communication with its neighbors. With this being said, it is still able to find a solution to cover the domain effectively. This is reflected in the quantitative results in the Experiments section.

With this being said, there is still room for improvement. In my algorithm, the underlying density function that determines the communication range is known from the beginning. This makes it much easier to estimate the time derivative of the communication range. However, in the real world this may not be possible, so it would be necessary to find a way to accurately calculate the instantaneous change of the communication range. Despite this, this paper has presented an effective solution to the coverage control problem with communication constraints.

REFERENCES

[1] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," IEEE Transactions on robotics and Automation, vol. 20, no. 2, pp. 243–255, 2004.

[2] A. Okabe and A. Suzuki, "Locational optimization problems solved through voronoi diagrams," European Journal of Operational Research, vol. 98, no. 3, pp. 445–456, 1997.

[3] W. Luo and K. Sycara, "Voronoi-based coverage control with connectivity maintenance for robotic sensor networks," in Proc. 2019 Int. Symp. on Multi-Robot and Multi-Agent Systems (MRS). IEEE, 2019.

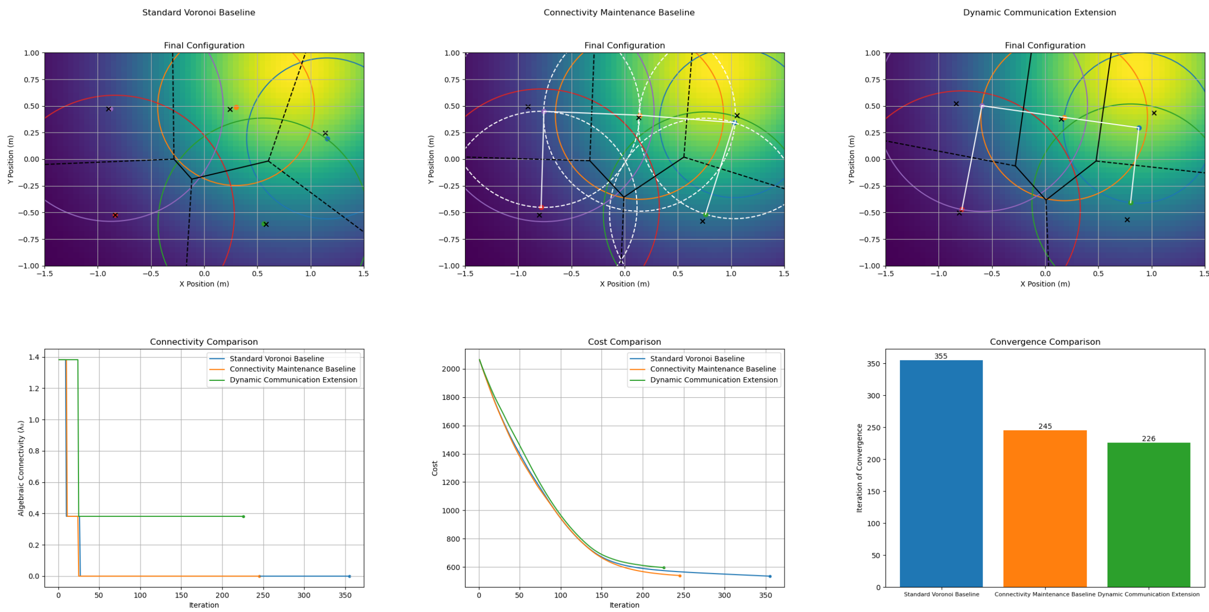

Fig. 1. Experimental Results for the 1st Gaussian distribution function.

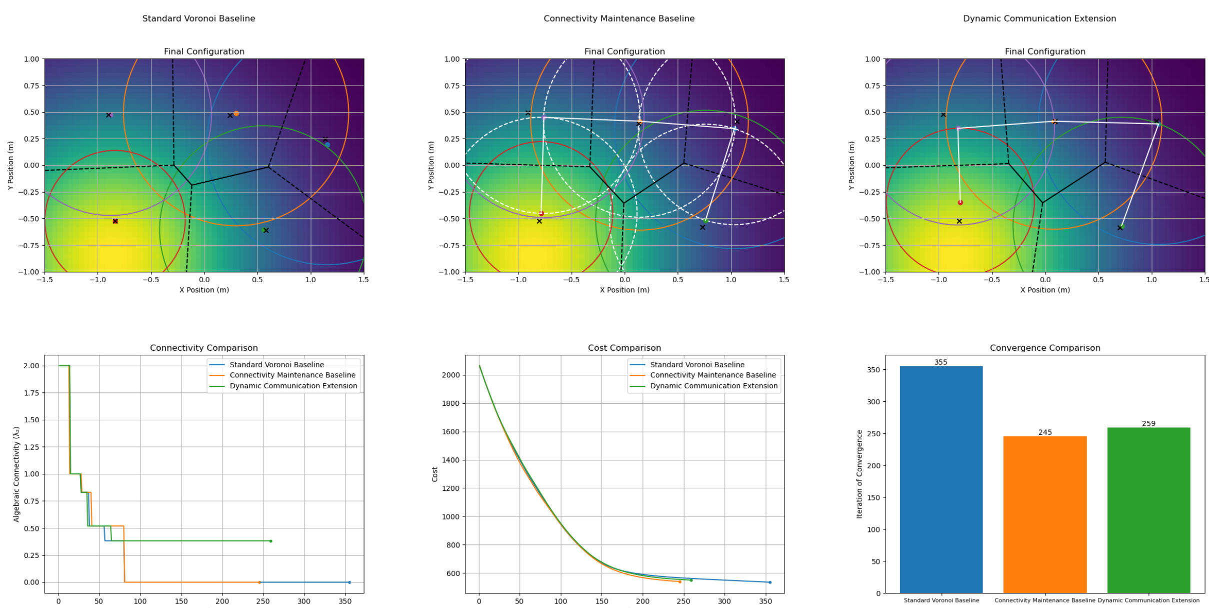

Fig. 2. Experimental Results for the 2nd Gaussian distribution function.

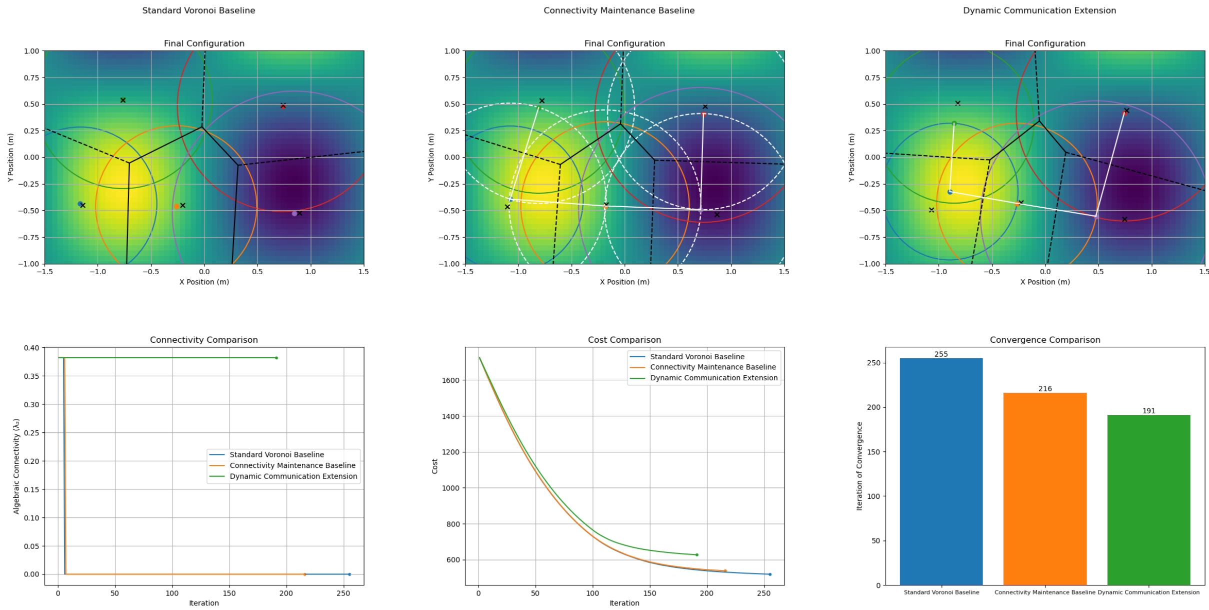

Fig. 3. Experimental Results for the 1st Random distribution function.

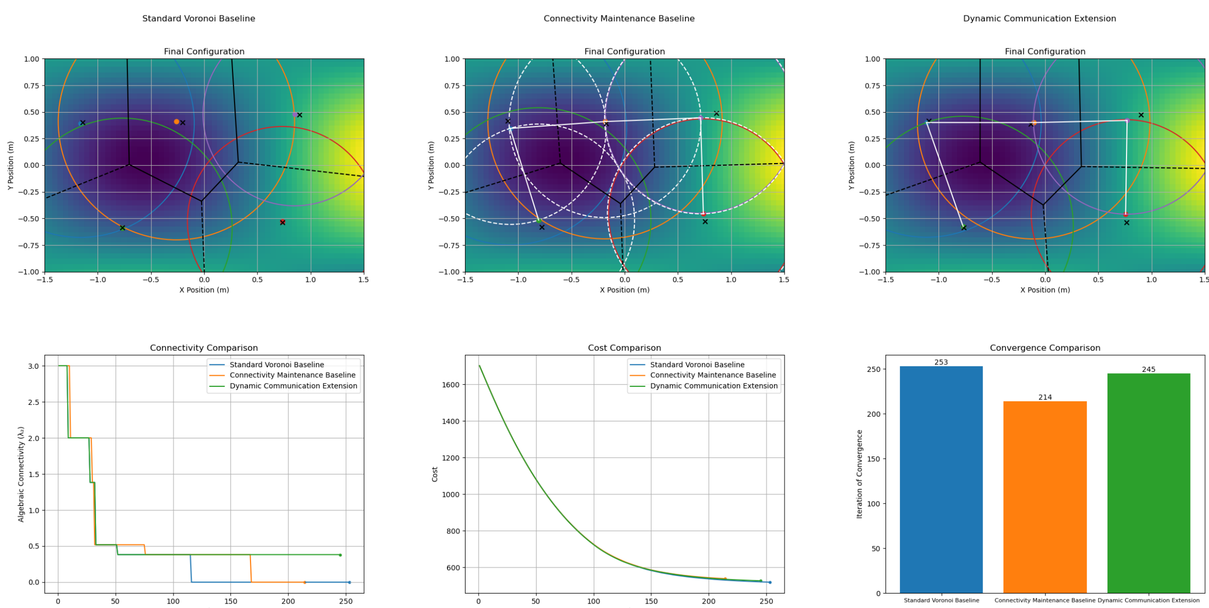

Fig. 4. Experimental Results for the 2nd Random distribution function.