Voronoi-Based Coverage Control for Distributed Multi-Robot System

Shule Hou[1, a *, †], Zijian Wang[2, b, †], Xiang Xu[3, c, †]

[1]Faculty of Science and Engineering University of Liverpool, United Kingdom

[2]Jacobs School of Engineering University of California, San Diego La Jolla, United States

[3]School of Precision Instrument and Opto-electronics Engineering Tianjin University Tianjin, China

[*]Corresponding author: [a]Sgshou2@liverpool.ac.uk, [b]ziw065@ucsd.edu, [c]philxu_520@tju.edu.cn

[†]These authors contributed equally.

Keywords: Coverage control; Voronoi-based methods; Lloyd’s algorithm; multi-robot system.

Abstract: Coverage control is one of the fundamental research fields in the study of the Multi-Robot System. The majority of coverage control algorithms aims at improving the efficiency of robots in exploring the environment while minimizing the information loss rate. This paper presents a control algorithm based on the Voronoi diagram for a distributed multi-robot system. Our method is based on Voronoi diagrams, which consists of polygons bounded by vertical bisectors in the plane. Our method can generate the optimal deployment location and optimal task space for each robot in the team by applying Lloyd’s algorithm to find the centroid of polygons and get the Centroidal Voronoi Partitions. We also verify our method by combining the open-source Lloyd’s Algorithm simulations with the distributed multi-robot system in a non-convex environment.

1. Introduction

Since the 1980s, researchers have focused on designing and building the Multi-Robot System (MRS), which is generally defined as a groups of robots working together when they are assigned tasks [1]. A Multi-robot system has always been a significant concept in the robotic domain. According to Avinash Gautam and Sudeept Mohan’s research [2], nowadays there are more complex tasks such as mapping, search and rescue, manufacturing, robot distribution, exploration, etc., These tasks require cooperation between multiple robots, while an MRS can solve it through "dispatching smaller sub-problems to individual robots in a group and allowing them to interact with each other to find solutions to complex problems. [2]" Furthermore, compared with a single-robot system, a multi-robot system has better robustness and redundancy (an instance would be environmental monitoring including seismic surveillance and topographic mapping.) Attributed to the great number of terminals, the spatial distribution of MRS is larger than a single-robot system [3]. MRSs can be widely deployed in many categories of industries including electronic manufacture, medical service, geological survey, etc.

Although the multi-robot system has obvious advantages, there still exist limitations that need to be resolved and further improved. One of the most critical research areas is solving the coordination problem among robots. According to [4], MRS can be divided into six categories based on coordination and system dimensions:1) Unaware Systems, 2) Aware, Not Coordinated Systems, 3) Weakly Coordinated Systems, 4) Strongly Coordinated, Strongly Centralized Systems, 5) Strongly Coordinated, Weakly Centralized Systems, and 6) Strongly Coordinated Distributed Systems. The research [4] shows that, with the improvement of the technique on cooperation between robots, the demands on multi-robot cooperation and communication have been essentially increased. For an instance, space sharing is a common type of issue in a Multi-Robot System. Researchers are required to plan multi-robot motion to avoid collision, congestion and deadlock problems [4]. Another typical difficulty occurs in MRS is resource conflict problem, such as mutual exclusion problems [5]. In this paper, we are concerned with a specific coordination problem in the multi-robot exploration field, which is finding a coverage control strategy for MRS in a non-convex environment.

The coverage control problem aims to define a coordination strategy for MRS in the exploration field. This strategy is designated to locate the target message or object in the shortest possible time. Researchers have proposed several approaches to solve the coverage control problem, such as Particle Swarm Optimization (PSO) [6] [7] and Koordinierte Multi-Robot Exploration [8]. Multi-robot coverage control provides a solution to determine a path through all points of a region avoiding obstacles. In this paper, we use one specific multi-robot coverage control method to solve this problem.

Up to now, there have been numerous studies on multi-robot coverage control. By traditional coverage control strategies [9] [10], each robot plays the role of a mobile tunable sensor, in which scenario, robots are required to be in motion for maintaining the desired coverage level. A similar approach is presented in [11], the authors utilize a strategy to optimize robot position in a convex environment. In [12], authors solve the problem of covering a nonconvex environment with heterogeneous robots equipped with sensors. The Voronoi-based method is among the most successful algorithms for coverage control [13], which can be naturally combined with multiple target tracking algorithms for coverage information gathering [14-16]. By combining classical Voronoi coverage with the Lloyd algorithm and the local path planning algorithm TangentBug, the authors compute the motion of the robots around obstacles and corners in [13]. In our paper, inspired by their literatures, we introduce a coverage control algorithm which can be implemented in a distributed manner based on Lloyd's algorithm. In the following parts, we demonstrate the efficacy of this method using MATLAB simulations.

2. Problem formulation

2.1 Environment Description

In this project, we suppose that the environment we deployed our MRS is non-convex. We define exploration set Q as the global area of the environment. Point i in the exploration set can be written as $p_i \in Q$. To define the number of the information at point p_i, we introduce a map $\phi: Q \rightarrow R$ represents the information density function of the exploration area, where R is the set of positive natural numbers. According to these formulas, $\phi(p_i)$ represents the information density at point p_i. The high $\phi(p_i)$ indicates a point contains a higher concentration of information compared with other low information density points.

2.2 Coverage Control Problem

The basis of the coverage control problem is to find a coordination method for MRS, and helps it to find/cover the target message/environment in the shortest possible time. This project uses a parallel strategy. We divide the searching area into n regions, where n equals the number of robots, and assign a robot to each region. Every single robot/terminal is only responsible to search the region assigned to it. In our scenario, we consider the possibility that the robots miss the message while scanning the entire region. The loss possibility is positively correlated with the detection distance. Hence, we use $f(x)$ to describe detection probability, where x is the detection distance.

2.3 Voronoi Diagram

The Voronoi diagram is a famous data structure of computational geometry, widely used in Geometry, Architecture, Geography, Meteorology, etc. Geometrically, the Voronoi diagram is a subdivision of the plane space with several sites, which contains continuous polygons consisting of a set of vertical bisectors of two adjacent sites. In the case of two sites, the boundary is their vertical bisector, which divides the full plane into two half-planes. In this case, the distance between any point in the plane and the base station in this half-plane is smaller than the distance to any other base station. When the number of sites is multiple, the full plane will be divided into multiple areas including only one site, and any point in the area is closest to the site in that area.

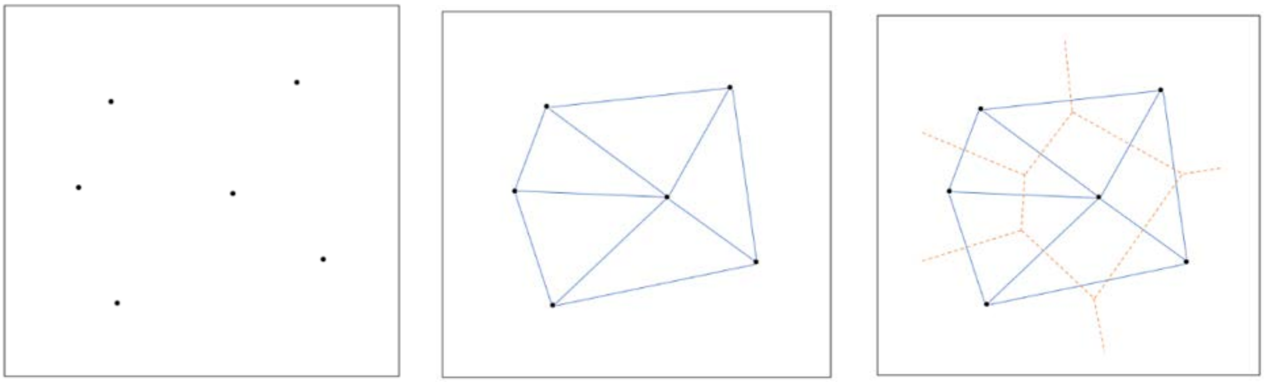
Voronoi diagram's properties are as follows. Firstly, there is one site inside each polygon. Secondly, spots within each polygon have a shorter distance to this site than to any other sites. Thirdly, spots on the polygon boundary are equidistant from the sites that generated this boundary. Lastly, the vertices inside the polygon have 3 edges connected to them, and these edges are the public edges of three polygons next to each other.

The Euclidean distance between any two points p and q, denoted by $dist(p,q)$, can be written as (1).

$$dist\ (p,q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2} \tag{1}$$

Let $p_i = \{p_1, \dots, p_n\}$ be any spot on the plane and each spot is different from others. In our project, we take each spot as a site. According to the definition, these sites can generate a Voronoi diagram. In this diagram, the whole plane is divided into n Voronoi cells $R = \{R_1, \dots, R_n\}$, which have properties as follows.

1) Point q is located in the cell corresponding to point p_i,



(a) (b) (c).

Figure 1. The steps to create a Voronoi diagram, the points in figure (a) correspond to the initial position of the robots, and we use these points to generate a triangular network which is shown in (b). Figure (c) shows the vertical bisectors of each edge of Delaunay triangulation, and this will form the final Voronoi diagram we want.

2) If and only if $p_j \in R_j, i \neq j$ always have $dist(q,p_i) < dist(q,p_j)$.

3) Point q is located on the boundary of two cells corresponding to point p_j and p_i, always has $dist(q,p_i) = dist(q,p_j)$.

The key point to establish the Voronoi diagram algorithm is connecting discrete point into a Delaunay triangulation network [17]. The steps to create a Voronoi diagram shows in Figure 1. Initially, according to the constraints of Delaunay, we need to generate a triangular network. Then, we make vertical bisectors of each edge of Delaunay triangulation. Finally, we need to connect vertices on vertical bisectors.

2.4 Centroidal Voronoi Partitions

In the previous section we introduce the Voronoi Diagram, and this diagram describes a set of regions that include all of the closest points from the generator of each Voronoi cell.

By applying this method to the exploration set Q, we produce a set of Voronoi cells belonging to Q, and we use $\{V_i\}_{i=1}^{k} \in Q$ to represent k Voronoi cells in Q. If the positions of exploration robots are fixed and already known, the optimal detecting area for each robot is the Voronoi Diagram generated from their location. In a real-life scenario, we start this method by locating the optimal position in Q to deploy the robot and detect as much information as possible. To get an optimal solution, we need to apply the functions we defined before, information density $\phi(p)$ and detection probability $f(x)$. Please be aware that in this paper we are only concerned with the collaborative strategy for the MRS, therefore we assume that the information density of the environment $\phi(p)$ is already given. For the coverage control problem, we need to be consisted with the rule that the higher $\phi(p)$ (information density) the $f(x)$ higher detection probability. This means in the high information density area, the system should assign more resources to prevent detection missing, just as [18] describes: "we need to assign more guards to the expensive paints." In this paper, we use to represent the performance of the control method we use. In (2), x_p is the distance between the robot in Voronoi cell V_i and point p.

$$H\left(\phi_{V_i}, f(x_p)\right) = \sum_{V_1}^{V_i} \int f(x_p)_{v_i} \phi(p) dp \tag{2}$$

Though (2) we can find the higher H function indicates the system allocates more resource at the area with high information density, and the Voronoi Partitions can be considered as the optimal solution with unit $\phi(p)$ and fixed robot position.

2.5 Lloyd’s Algorithm

Another important algorithm we plan to utilize in our research is Lloyd’s Algorithm. This Algorithm has been named after Stuart P. Lloyd. Lloyd’s Algorithm is also called Voronoi iteration, which is a more advanced algorithm developed from the Voronoi algorithm. The Centroidal Voronoi Partitions mentioned in the previous paragraph was evolved based on this Lloyd’s Algorithm. This was originally used as a technique for pulse-code modulation after being proposed in 1957 [19]. The main function of Lloyds' Algorithm is to find the centroid of the area divided by the Voronoi algorithm then use the Voronoi diagram to re-partition the area based on the centroid and generate a new centroid for the next iteration. When relaxation is performed, the distribution of mass tends to be more even. Through the iterations, eventually, the centroid will tend to move to a stable position. Therefore, Lloyd’s algorithm is guaranteed to converge.

In our project, we are only concerned about the application of Lloyd’s Algorithm in a two-dimensional region. We are looking for the mass centroid of this region. The mathematical explanation is listed below [21]. If we are given a density function of ρ in the designated Voronoi region V, we can find the mass centroid *z* of this Voronoi region V by

$$z = \frac{\int_V y\rho(y)dy}{\int_V \rho(y)dy}. \tag{3}$$

We could define the Voronoi diagram produced in each iteration $V_i, i = 1,2,3,4,\dots,k$ and the mass centroid from iteration can be presented as$z_i = 1,2,3,\dots,k$. The final z_k would infinitely approach the mass centroid of the exploration field. We refer to [21] for a more detailed computation and calculation of Lloyd’s Algorithm. In our project, we use this algorithm to find the maximum value of (2).

3. Method

In this part, we show the method for working out the optimal solution (max value of (2) in part 2) by applying Lloyd's algorithm in a multi-robot system.

3.1 Robot Model Assumptions

As part 2 said, we produced a team of n robots searching message in a map (exploration set) *Q*. Every robot has the same functions and they are capable to detect information within a radius *R* around themselves. With the increase of detection range, the detection capability of the sensor will be diminished. This indicates $f(x)$ of the robot sensor will decrease when *x* increases. Furthermore, in our experimental model, the instantaneous status of each robot is the most crucial message of the whole system, because the system is designed to run based on the robot's real-time location. To satisfy this requirement we define a group of navigation robots that have following modules, and the design specifications are shown in table 1.

Table 1. Robot model Design specifications.

Modul name	Specification
Location module	To get global location information of itself
Movement module	To control robot's movement and move the robot to the target location
Working region	The Voronoi cell of itself
Sensor performance	To produce the sensor performance f at the given location
Lloyd's algorithm model	Discussed in section C

3.2 Environment Assumptions

In this project, we suppose the information density *ϕ (p)* at point *p* has been provided and the environment for navigation is a close map.

Table 2. Environment design specifications.

Contents	Specification
information density ϕ(p)	information density at point p
Map region	Describe the boundary of the map
Robot start position	The start position of the robot is fixed and the location is randomly generated in the first run.

3.3 Lloyd's Algorithm Model

In the previous section we discuss Lloyd's algorithm. The key point of this algorithm is each robot in the system must be able to calculate its own Voronoi cell. The steps of Lloyd's algorithm are as follows.

1) Step one is to select *n* starting points in the data set.
2) Step two is to calculate the Voronoi cells of these *n* starting points.
3) Step three is to calculate the centroid of these Voronoi cells.
4) Step four is to move the starting point of each Voronoi cell to overlap each center of mass.

In our project, our expectation is to move each robot to the center of mass of each Voronoi cell. As the robots move, their Voronoi cell also changed. According to Lloyd's algorithm, robots must communicate the weight of the local area in the environment with a neighbor robot. For this purpose, each robot is able to store previous location information and share current location information with other neighbor robots. Figure 2 and Algorithm 1 show this process.

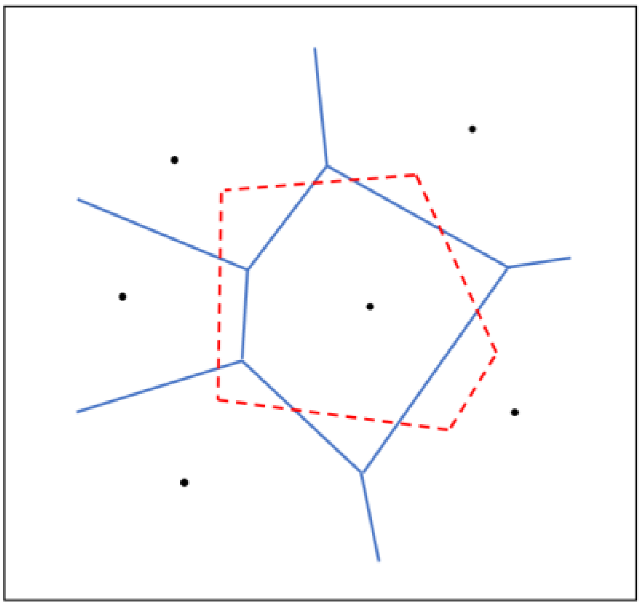


Figure 2. An example of 6 robots (donated by 6 black dots), the areas divided by blue lines are current Voronoi cells, the area surrounded by red dotted lines is a previous Voronoi cell.

Table 3. Algorithm 1 Distribute Voronoi cells.

Message input: 1) information density $\phi(p)$ 2) region of each Voronoi cell	
1:	Robot i shares location information with neighbours N
2:	Compute each Voronoi cells $R^t = \{R_1^t, R_2^t, \ldots, R_i^t, \ldots, R_j^t\}$
3:	Robot i shares location and Voronoi cells information with neighbours.
4:	for $i \in N$ do
5:	Compute $R_{ij} = R_i^{t-1} \cap R_j^t$
6:	Send location information to robot i.
7:	end for

Each robot calculates the intersection of its previous Voronoi cell R^{t-1} and current Voronoi cell R^t of neighbor robots, then transfer the weight of the intersection area to their neighbors.

3.4 Distribute Coverage Controller

The distribution coverage controller of the project is shown in Algorithm 2.

At first, the controller loads the map and defines each point's information density ϕ (p). Next, robots are randomly deployed in the environment, and locations of robots by time are recorded in the R^t. The following steps are to keep applying Lloyd's Algorithm until finds the centroidal-Voronoi partitions of the map. Finally, the controller checks the $H\left(\phi_{V_i}, f(x_p)\right)$, if it is larger than any of its previous values, then we define $H\left(\phi_{V_i}, f(x_p)\right)$ with $i = k$ as the optimal solution.

Table 4. Algorithm 2 Distribute Coverage Controller.

1:	Generate map and set the information density $\phi(p)$
2:	Randomly set the start position of the robots R^t
3:	Compute each Voronoi cells $R^t = \{R_1^t, R_2^t, \ldots, R_i^t, \ldots, R_j^t\}$, set $H_{previous} = H\left(\phi_{V_i}, f(x_p)\right)$
4:	Robot i shares location and Voronoi cells information with neighbours.
5:	for $i \in N$ do
6:	Compute $R_{ij} = R_i^{t-1} \cap R_j^t$
7:	Send location information to roboti. Set $H\left(\phi_{V_i}, f(x_p)\right) = H_{previous}$
8:	end for
9:	Calculate function $H_{new} = H\left(\phi_{V_i}, f(x_p)\right)$
10:	if $H_{new} > H_{all\ previous}$
11:	Move to the location

4. Simulation

To demonstrate the feasibility of our control algorithm, we conduct a set of simulation ex periments based on MATLAB. The environment is a 10×10-meter area with no obstacles, an d the maximum speed of each robot is limited to 1m/s. 20 robots are equipped with sensors that can detect the environment around. These terminals are capable to exchange informatio n between them. Our simulation is conducted by using the source code https: //www.mathwo rks.com/matlabcentral/fileexchange/41507-lloydsalgorithm-px-py-crs-numiterations-showplot.

The first step of our simulation is to generate and place n stationary robots. At the beginning of our experiment, each robot is given a starting position in the environment, and each starting position is contained in a small grid in the lower-left corner of the environment. Then we will iteratively apply Llyod's algorithm to make robots close to the expected position.

In this process, we use the function '$PolyCentroid$' to compute the coordinates for the centroid of the polygon with verticesX, Y. The centroid of a non-self-intersecting closed polygon defined by n vertices$(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ is the point(C_x, C_y). And another function '$VoronoiBounded$' computes the Voronoi cells about the points (x, y) inside the bounding box. The final result is shown in Figure 3.

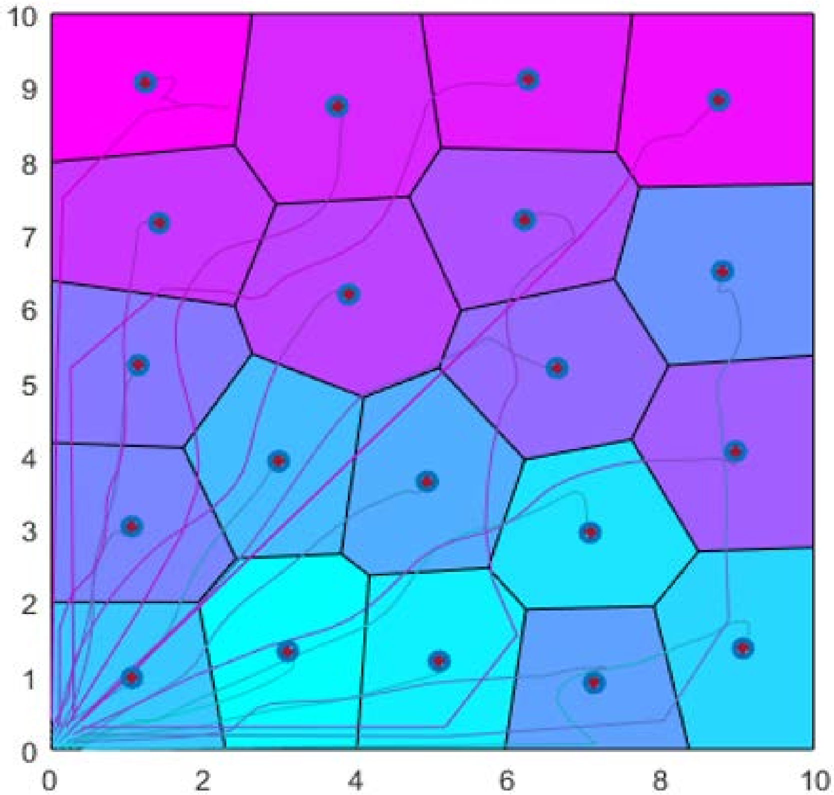


Figure 3. A simulation result of cooperation of 20 robots in a 10X10m area using Lloyd's algorithm. Each dot is a robot. The polygons in different colours represent the Voronoi cells of different robots. The lines are the movement trajectory of the robots.

As shown in figures, these is a dynamic process. At the time equals 3, 6, 15, 30, the distribution of robots is shown in Figure 4.

As we mentioned in the previous section, the whole operation process is the process of iteration of Lloyd's algorithm. Lloyd's algorithm starts with an initial distribution of samples and consists repeatedly executing one relaxation step:

1) The Voronoi diagram of all the points is computed.

2) Each cell of the Voronoi diagram is integrated and the centroid is computed.

3) Each point is then moved to the centroid of its Voronoi cell.

As we continue the iteration, the robots will eventually move to the ideal position, and the centroid of the polygon will gradually coincide with robots' current position. Taking one of our examples (20 robots, 10×10m map), the robots will find their final position at the time of 45.

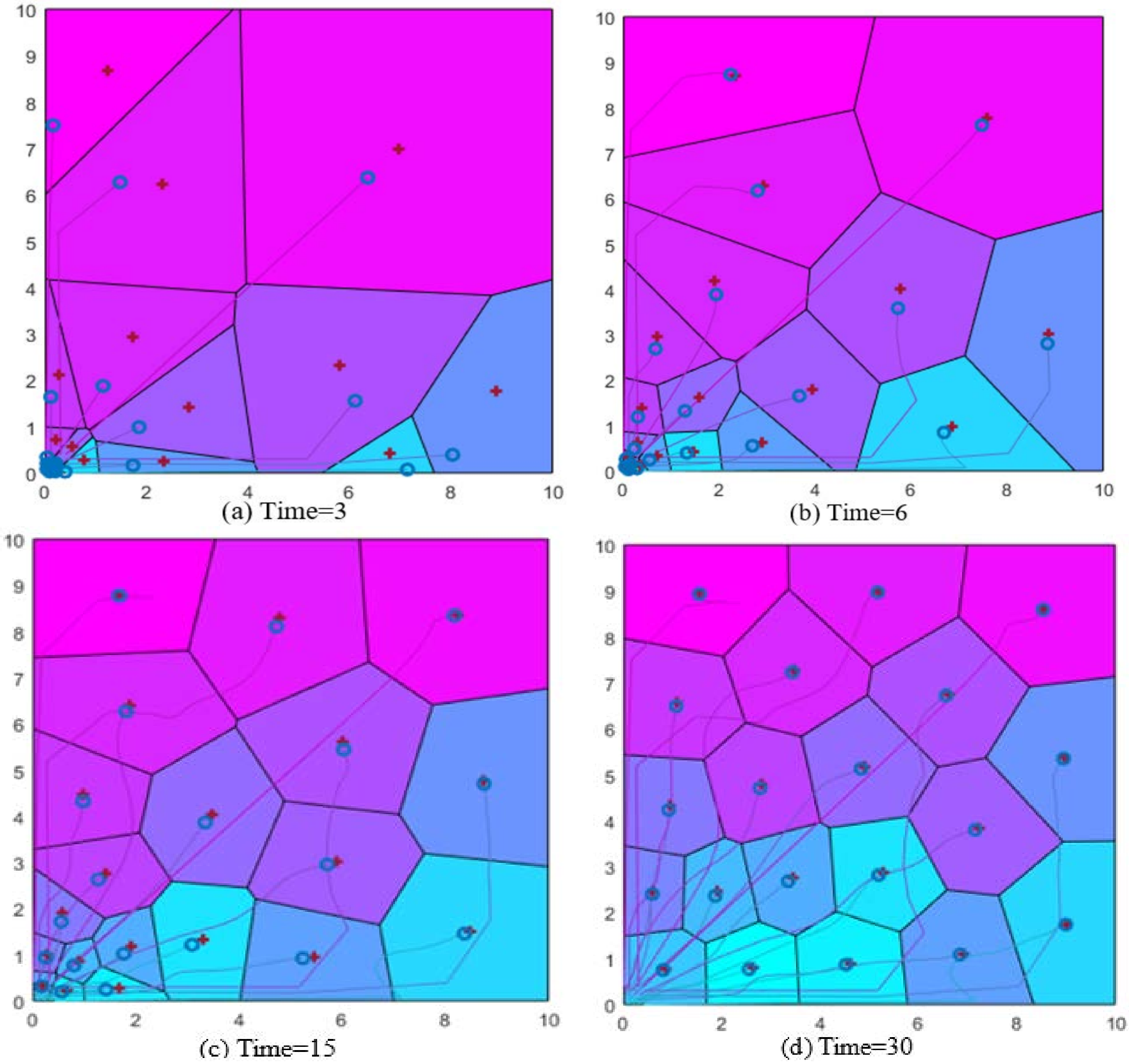


Figure 4. Iteratively apply Llyod's algorithm. Top to bottom left to right, the operation time is 3, 6, 15, and 30. Each dot is a robot. Each plus sign represents the goal of each robot (also the centroid of the current polygon). The polygons in different colours represent the Voronoi cells of different robots. And the lines are the movement trajectory of the robots.

5. Conclusion

In this paper, we proposed to use a distributed algorithm to coverage control multi-robot in an unknown environment. There are two main components: 1) a Voronoi-based method and 2) Lloyd's algorithm strategy. We utilize the Voronoi diagram to divide the experimental area, as preparatory work for robot distribution. Also, Centroidal Voronoi Partitions are introduced. In this process, the centroidal of each Voronoi cell is calculated to show the ideal positions of robots. To track robots in a moving environment, we use Lloyd's algorithm. Combining the methods mentioned above, this algorithm is the guideline of the distribution of robots in the simplest environment and allows us to observe robots' movement at the same time. Finally, our paper solves the problem of the multi-robot distribution in the simplest environment, demonstrated the feasibility of the Voronoi diagram and Lloyd's algorithm, which have great potential in the future.

Future work will continue to explore the distributed algorithm on multi-robot coverage control in various environments. We will also look at the adaptability of the method under different situations and select the most suitable one in modern environments.

References

[1] R. N. Darmanin and M. K. Bugeja, “A review on multi-robot systems categorised by application domain,” 2017 25th Mediterranean Conference on Control and Automation (MED), 2017, pp. 701-706, doi: 10.1109/MED.2017.7984200.

[2] A. Gautam and S. Mohan, "A review of research in multi-robot systems," 2012 IEEE 7th International Conference on Industrial and Information Systems (ICIIS), 2012, pp. 1-5, doi: 10.1109/ICIInfS.2012.6304778.

[3] Z. Yan, N. Jouandeau and A. A. Cherif, “A survey and analysis of Multi-Robot coordination”, International Journal of Advanced Robotic Systems, vol. 10, no. 12, pp. 399, 2013.

[4] A. Farinelli, L. Iocchi and D. Nardi, “Multirobot systems: a classification focused on coordination,” in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 34, no. 5, pp. 2015-2028, Oct. 2004, doi: 10.1109/TSMCB.2004.832155.

[5] E. Dijkstra, “Cooperating sequential processes,” In Genuys, F., editor, Programming Languages: NATO Advanced Study Institute, pp. 43–112. Academic Press, 1968.

[6] R. Eberhart and J. Kennedy, “A new optimizer using particle swarm theory” Micro Machine and Human Science, Proceedings of the Sixth International Symposium on, 4-6 Oct, pp. 39-43,1995.

[7] R. Eberhart and J. Kennedy, “Particle swarm optimization” Neural Networks, Proc. of IEEE International Conference on, Nov/Dec, pp. 1942-1948, 1995.

[8] W. Burgard, M. Moors, C. Stachniss and F. E. Schneider, “Coordinated multi-robot exploration,” in IEEE Transactions on Robotics, vol. 21, no. 3, pp. 376-386, June 2005, doi: 10.1109/TRO.2004.839232.

[9] J. Cortes, S. Martinez, T. Karatas and F. Bullo, "Coverage control for mobile sensing networks", IEEE Trans. Robot. Autom., vol. 20, no. 2, pp. 243-255, 2004.

[10] Y. Diaz-Mercado, S. G. Lee and M. Egerstedt, “Distributed dynamic density coverage for human-swarm interactions,” Amer. Control Conf., pp. 353-358, 2015.

[11] M. Schwager, J. McLurkin and D. Rus, “Distributed Coverage Control with Sensory Feedback for Networked Robots,” in Proceedings of Robotics: Science and Systems, Philadelphia, USA, 2006.

[12] L. Pimenta, V. Kumar, R.C Mesquita and G. Pereira, “Sensing and coverage for a network of heterogeneous robots,” in 47th IEEE Conference on Decision and Control, Cancun, Mexico, 2008.

[13] J. Chen and P. Dames, “Distributed and Collision-Free Coverage Control of a Team of Mobile Sensors Using the Convex Uncertain Voronoi Diagram,” 2020 American Control Conference (ACC), 2020, pp. 5307-5313, doi: 10.23919/ACC45564.2020.9147359.

[14] J. Chen and P. Dames, “Collision-Free Distributed Multi-Target Tracking Using Teams of Mobile Robots with Localization Uncertainty,” 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020, pp. 6968-6974, doi: 10.1109/IROS45743.2020.9341126.

[15] J. Chen, and P. Dames, “Distributed Multi-Target Search and Tracking Using a Coordinated Team of Ground and Aerial Robots.” In Robotics science and systems, 2020

[16] J. Chen and P. Dames, "Distributed Multi-Target Tracking for Heterogeneous Mobile Sensing Networks with Limited Field of Views," in 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021, pp. 9058-9064, doi: 10.1109/ICRA48506.2021.9561888.

[17] A. Breitenmoser, M. Schwager, J. Metzger, R. Siegwart and D. Rus, "Voronoi Coverage of Non-Convex Environments with a Group of Networked Robots," in IEEE International Conference on Robotics and Automation (ICRA), Anchorage, USA, 2010.

[18] T. Shirakawa, A. Adamatzk, Y. Gunji, et al. “On simultaneous construction of Voronoi diagram and Delaunay triangulation by Physarum polycephalum”. International Journal of Bifurcation and Chaos, 2009, 19(09): 3109-3117.

[19] S. Bhattacharya, R. Ghrist, and V. Kumar, “Multi-robot coverage and exploration on Riemannian manifolds with boundaries”, The International Journal of Robotics Research, vol. 33, no. 1, pp. 113–137, 2014.

[20] S. Loyd, “Least squares quantization in PCM. IEEE Transactions on Information Theory,” 28(2), 1982 pp. 129–137. https://doi.org/10.1109/tit.1982.1056489

[21] Q. Du, et al. “Centroidal Voronoi Tessellations: Applications and Algorithms.” SIAM Review, vol. 41, no. 4, Society for Industrial and Applied Mathematics, pp. 637–76, 1999, http://www.jstor.org/stable/2653198.