Multi-Robot Path Planning System for Effective Navigation

Andres Ibarra, Jake Olson, and Jason Pak

***Abstract*— For our final project, our group expanded on the concept of path planning to tackle the challenge of multiple robots operating within the same environment. By decoupling the path planning process, each robot could independently plan its path while considering the presence of other robots and potential collision scenarios. Additionally, to handle the configuration of multiple robots, the project involved developing custom launch files and communication mechanisms to facilitate coordination among the robots. In these ways, this project applies the concepts and techniques learned in single robot path planning to the more complex scenario of coordinating and planning paths for multiple robots within the same environment, with an aim of enhancing the efficiency and safety of the entire operation.**

I. Introduction

Multi-robot path planning has been an active area of research due to the potential benefits and applications of multi-robot systems in the real world. Whether it be domains such as warehouse automation, food delivery, swarm robotics, surveillance, or disaster response, the ability to enable multiple robots to work collaboratively allows for tasks to be accomplished more quickly and effectively. Moreover, multi-robot path planning contributes to resource optimization, as robots can cooperate in sharing information, avoiding duplication of effort, and minimizing energy consumption. As a result, this ability to handle simultaneous path planning for multiple robots is crucial for systems that require robots to work together in navigating through complex environments, ensuring smooth and efficient movement that don't interfere with one another.

In our course material, path planning was covered as one of the fundamental concepts of robotics. Yet, we only worked within a single-robot system when implementing strategies of global planning. For these reasons, our group chose to extend upon the material covered in class and focus on implementing a multi-robot path planning system for our final project. Our objective was to implement path planning for two or more robots – given each robot's start and goal locations, the system would output the best route for each robot to follow within the environment. By decoupling the path planning process, each robot could independently plan its path while considering the presence of other robots and potential collision scenarios. Additionally, to handle the configuration of multiple robots, the project involved developing custom launch files and communication mechanisms to facilitate coordination among the robots. These custom launch files played a crucial role in configuring the initialization and synchronization of multiple robot nodes, expanding upon the single robot systems explored through course material. In these ways, this project applies the concepts of path planning for a single robot system to the more complex scenario of coordinating and planning collision-free paths for multiple robots in a given environment.

II. Related Work

The field of multi-robot systems has numerous studies focused on addressing optimal path planning. One approach involves combining the optimization capabilities of the A* algorithm, along with the search capabilities of co-evolutionary algorithms, resulting in a set of routes that are collision-free [1]. Another example is the artificial potential field method, which makes use of a virtual potential field around the robot [2]. The potential field combines attractive forces, which pull the robot towards the desired goal, as well as repulsive forces, which repels the robot from obstacles. Neural network and reinforcement learning methods can also be used to train robots to optimally navigate through an environment [3]. Another example is using path coding, where a path is represented as a sequence of discrete "genes", or actions for a robot to perform [4]. The ant colony algorithm can also be applied to path planning, in which "ants" compute paths based on probabilistic decision-making based on pheromone trails and heuristic information [5]. Current implementation strategies for multi-robot path planning make use of one or more of such methods, offering a wide variety of implementation strategies to consider in implementing a multi-robot path planning system.

When generalizing these strategies, we can divide them into two main categories: centralized and decoupled. Centralized and decoupled approaches represent two different paradigms in multi-robot programming, each with distinct advantages and considerations. In the centralized approach, all the robots in the system are combined into a larger, singular body to determine an optimal strategy [6], [7]. While a centralized approach may find the optimal path, however, problems with this approach arises in systems with many robots. Combining multiple robots into one body result in the cumulative robot to have an increased number of degrees of freedom. This is undesirable for the time complexity of the exponentially increasing path finding algorithms, making the centralized approach computationally inefficient for large multi-robot systems.

In contrast, the decoupled approach emphasizes individual robot autonomy and independence. In this approach, a path for each independent robot is calculated, before considering the interactions between different robots [8], [9], [10], [11]. While the decoupled approach is not complete and may not produce the ideal solution, it is computationally much faster than the centralized approach. As a result, for our project, we chose the decoupled approach to multi-robot programming due to prioritizing time efficiency over working with a complete algorithm. Scalability was a crucial factor for us, as we wanted the system to handle a variable number of robots efficiently. The decoupled approach offered the flexibility of adding many robots without causing major impacts on the entire system's performance. Even if the calculated paths are not necessarily the fastest routes, the decoupled approach was fitting since the aim of our project was more focused towards quickly generating smooth and efficient movement strategies that don't interfere with one another.

III. Problem Definition

Given the map environment, the objective is to design a multi-robot path planning system that efficiently finds routes for each of the robots in the environment. The system will consider the following inputs:

- Representation of the environment provided as an *OccupancyGrid*
- Number of robots within the environment
- Both the start and goal locations for each robot within the system

Given this input information, the system should produce an output that represents the path for each robot to navigate through the map. The path will be provided as a sequence of poses for a given robot to follow through. The solution should prevent the collision of robots, whether it is with the boundaries in the environment or with other robots of the system.

In terms of assumptions, the system will assume that the given environment is static and finite, without changes during path planning, aside from the robots that will be moving through the environment. The system will also assume that once the optimal paths are computed, the robots will appropriately follow the recommended sequences of poses provided to them. Additionally, the system assumes that all robots within the environment are the same type of robot, in terms of size and velocity ranges.

IV. Method

The implementation of our multi-robot path planning system can be broken down into three major components.

A. Mapping

As specified earlier, a representation of the environment is provided as an input. This component is responsible for appropriately processing this occupancy grid message and storing it as a reference for later use. For the project, we implemented a class that represents the grid using the matrix data structure. Within the grid class, we wrote methods to handle conversion of global coordinates to grid indices and handled factors such as dealing with the grid resolution. Additionally, we implemented a wall expansion strategy to create a buffer zone around the obstacles and to prevent the robots from colliding with the walls when in motion. The size of the buffer zone is calculated using the robot's size and the resolution of the map.

B. Robot Initialization

As a multi-robot system, there are multiple robots that will need to be initialized, each with its respective communication channels. This component is responsible for maintaining such information about a given robot. For the project, we implemented a class that represents a single robot. Each instance of the robot class is initialized with a unique number/ID and has its own channels for publishing Twist messages and pose sequences. Within the class, we wrote methods to handle robot movement — the robot can move forward or rotate for a desired distance or angle.

C. Path Planning

This component is responsible for calculating the routes for each robot to follow to effectively navigate its way around the environment to its goal destination. Given the start and end goal of a robot, the Breadth-First Search algorithm is used to calculate an optimal path. BFS was chosen as it is 1) complete since the environment is finite and 2) optimal since the cost of moving to a different cell are all equal to 1. Having calculated a path, the cells that lie within that path are then marked in the occupancy grid as being occupied. As a result, collisions are prevented in any following paths that are calculated for other robots. Having calculated a sequence of cells for the robot to move through, this sequence is converted to a sequence of poses for the given robot to follow through.

```
for all r ∈ ROBOTS do
  start.val ← r.start,   goal.val ← r.goal
  start.back ← null,   goal.back ← null

  goal ← r.goal
  frontier = queue()
  explored = set()
  frontier.enqueue(start)
  explored.add(start.val)

  while frontier is not empty do
    curr ← frontier.dequeue()

    if curr = goal then
      path ← backtrack(curr)
      grid ← mark all cell ∈ path as occupied
      r.path ← path
      continue to next robot

    for child ∈ curr do
      if child not in explored then
        explored.add(child)
        child_node.val ← child
        child_node.back ← curr
        frontier.add(child_node)

  r.path ← "no path exists"

return ROBOTS [ now with filled "path" parameters ]
```

Figure 1. Pseudocode for Path Planning Algorithm

In terms of the architecture and how the different components work together, the multirobot path planning system consists of a global planner that maintains the map data and references to all the individual robot objects of the environment. The global planner is responsible for calculating the optimal paths and communicating these paths to each individual robot. Within each robot instance, the robot object is responsible for processing the optimal path sent by the global planner and executing the planned path on the robot by sending proper Twist messages.

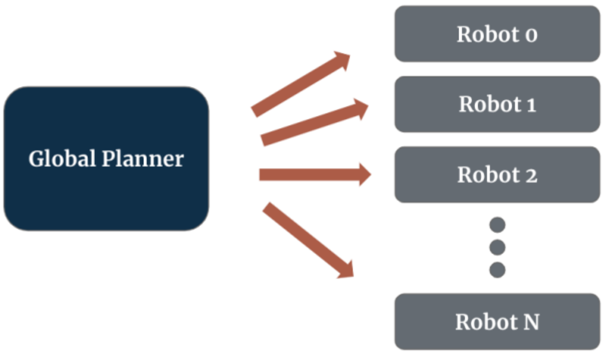

Figure 2. An architecture diagram showing the heirarchy of the system. A global planner maintains references to all the individual robot objects and sends the calculated optimal paths to each robot for individual execution.

Additionally, to facilitate and expedite execution of the code, a launch file was created. In particular, the launch file runs the ".world" and ".yml" files, and creates static transform publishers for each robot in the simulation. This lowers the number of terminal windows required and increased the speed of production. While launch files were never officially introduced or explained in class, we successfully utilized our resources, and with the help of Ravi, a TA, and Professor Quattrini Li, we were able to make it accomplish all our goals.

V. EXPERIMENTS

Due to the limited availability of real robots within the lab that were shared among the entire class, we decided the most feasible method of experimentation was through virtual simulations. By leveraging the simulation capabilities provided by ROS, we were able to create virtual environments that closely mimicked real-world scenarios of a multi-robot system. We designed and implemented the multi-robot path planning algorithms within this simulated environment, allowing us to test and validate the system's performance, evaluate the efficiency of the paths generated, and verify collision avoidance mechanisms.

To ensure thorough testing and evaluation of our path planning system, we loaded a map environment with sufficient space while incorporating an abundance of obstacles and walls. This approach allowed us to create realistic and challenging scenarios to assess the effectiveness and robustness of our path planning algorithms.

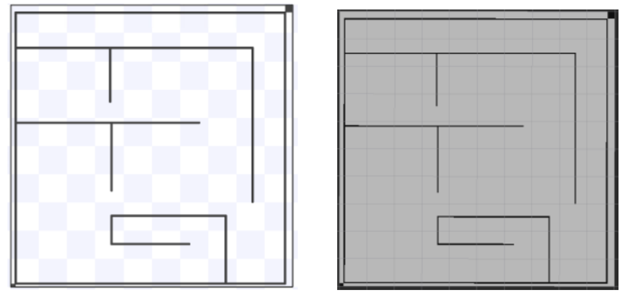

Figure 3. Example of a map environment. On the left is a map we used visualized within the simulation. On the right is the visualization of the grid in rviz, having processed the OccupancyGrid message.

To initially explore the capabilities of our multi-robot path planning system, we started with a two-robot system and conducted simulations using the traditional BFS algorithm. This allowed us to test and evaluate the system's performance by considering varying start and goal locations for each robot. To visualize the output of the path planning algorithm, we had the robots publish the calculated paths to a topic and subscribed to these topics on rviz.

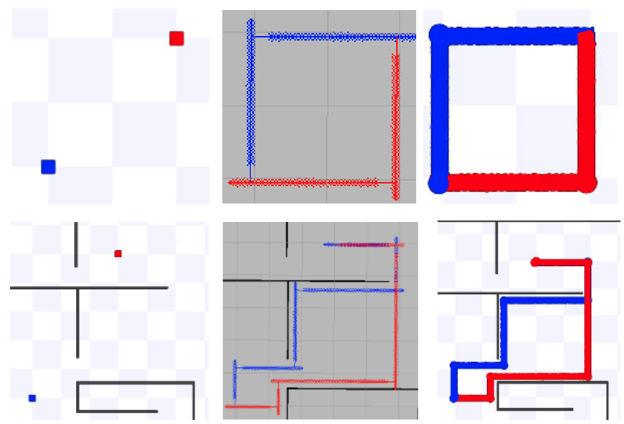

Figure 4. Two different simulations of a two-robot system with robots programmed to move to the other robot's starting location. On each row, the first picture is the initial positions of the robots, the second picture is the paths visualized in rviz, and the final picture is the output of the simulation of moving the robots. In the second simulation, we see that despite a collision of paths, the robots don't collide in the simulation since they don't occupy the same cells at the same time.

Within the two-robot system, we designed tests where the robots were required to swap their starting positions, essentially moving towards each other in the environment. This setup was intentionally designed to force potential collisions between the robots and evaluate the effectiveness of using just the traditional BFS algorithm. From this test, we observed that even if the planned paths for the two robots had potential collisions, collisions would only occur if the robots attempted to pass through the same cell at the exact same time. Despite this fact, there still was a compelling need to handle potential collisions, as other experiments showed instances of robots crashing into one another (such as Figure 5).

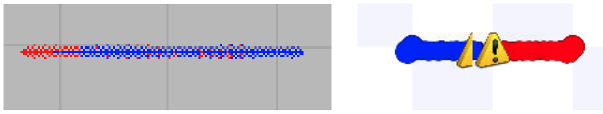

Figure 5. Example of a collision scenario, which occurs since robots attempt to pass through the same cell at the same time.

To improve upon this potential collision scenario, we implemented the marking strategy outlined in the method section of this paper. After calculating a path for one robot, the cells that lie within that path are marked in the occupancy grid as being occupied. As a result, the algorithm ended up producing a list of paths that have absolutely no overlap, preventing collisions in any paths that are calculated for the robots.

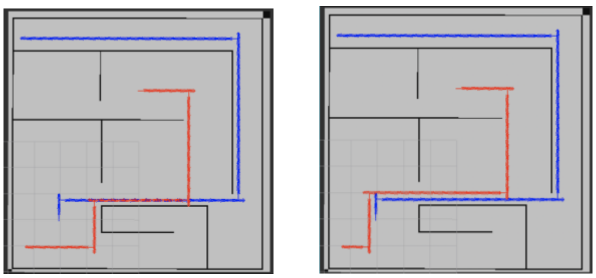

Figure 6. Comparison of the path planning algorithm output without vs. with the marking strategy. As shown right, new paths are calculated without interfering from formerly calculated paths.

Having implemented the marking strategy, we ran the simulations again on various scenarios. While the act of marking cells prevented the paths from overlapping within the grid cells, we observed that collisions between the robots still occurred during the simulation. This unexpected outcome highlighted a crucial oversight in the algorithm: it did not account for the actual widths of the robots. Even if the robots were not on the same grid cell, they could still come into contact due to the dimensions of their bodies. To mitigate this issue and ensure collision-free movements, we recognized the need to incorporate the physical dimensions of the robots into the path planning algorithm.

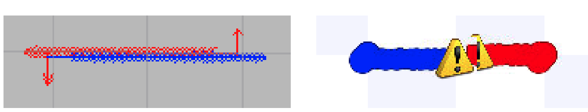

Figure 7. Running the scenario shown in Figure 5, but with the marking strategy. On the left, we can see that the path planning algorithm reintroduces a new path that avoids occupied cells. Yet, as shown right, the robots still crash during simulation due to the dimensions of their bodies.

Drawing upon the idea of wall expansion, we introduced a mechanism to expand the cells that needed to be marked as boundaries. Instead of solely marking the occupied cells from the robot's planned paths, we expanded the marked cells in both horizontal and vertical directions to cover the width of a robot. This expansion ensured that the robot's physical properties were accounted for when calculating the optimal paths. By expanding the marked cells, we created a safety margin around each robot's planned path, providing an extra layer of protection to prevent unintended collisions.

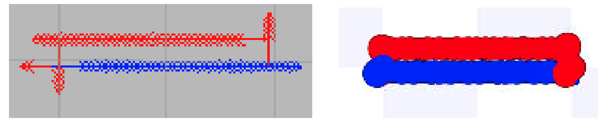

Figure 8. Running the scenario again but now with the expansion strategy of marking cells. As shown left, the path planning algorithm now calculates a path that factors in the width of the robots. As shown right, the robots succesfully move to their desired locations and correctly swap spots during the simulation, free from collisions.

After testing and verifying the performance of our path planning system with a two-robot system, we proceeded to extend its capabilities to accommodate a variable number of robots.

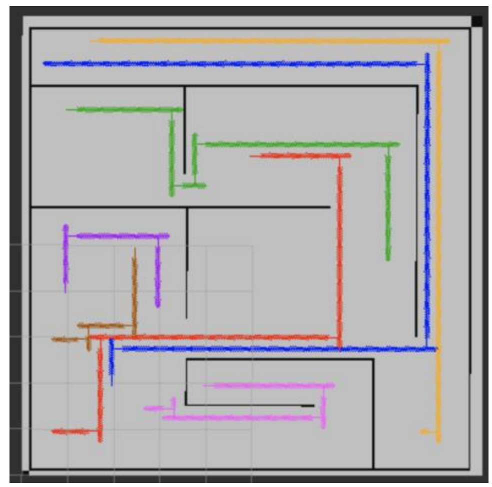

Figure 9. Visualization of the path planning algorithm output for a scenario with seven robots in the same environment.

Overall, the system performed well in generating paths for multiple robots, demonstrating its capability to coordinate and plan paths for a variable number of robots. The robots successfully navigated within the environment, avoiding collisions, and reaching their respective goals. This provided promising results in terms of achieving efficient and safe multi-robot coordination.

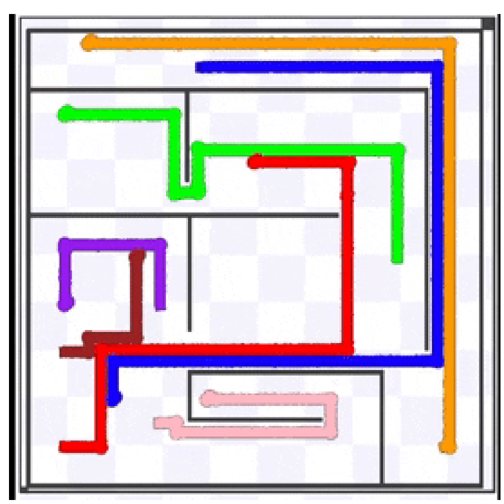

Figure 10. Simulation of the path planning algorithm output for a scenario with seven robots in the same environment.

However, it is important to acknowledge the limitations that our current implementation imposes. While the conservative approach of cell marking helped prevent potential collisions, it introduced some limitations and trade-offs that need to be considered. By marking previously calculated paths as boundaries, we aimed to ensure that robots would avoid traversing through cells that were already occupied by other robots' planned paths. This strategy aimed to minimize the risk of collisions by maintaining a clear separation between the robots' trajectories. However, this conservative approach may lead to certain drawbacks. Firstly, it assumes that all cells within a previously planned path are completely off-limits for other robots. This might result in unnecessary restrictions and limitations on the movements of other robots, as a cell might be free if a robot has not yet reached it. In situations where paths intersect or share common cells, the conservative marking of paths as boundaries might overly restrict the robots' movements and limit their ability to reach certain locations.

Additionally, marking path cells as boundaries can create a prioritization bias based on the order in which robots have BFS performed on them. If one robot's path is calculated before another, its path cells will be marked as boundaries, potentially blocking subsequent robots from accessing certain areas. This order-based prioritization may introduce inequities and limitations, as some robots might be unfairly constrained in their navigation options. To address these limitations, future improvements to the system can focus on dynamic updates of the path planning algorithm. This would allow for adaptive adjustments to the marked boundaries based on real-time robot positions and enable more flexible and efficient path planning. By dynamically evaluating the occupancy of cells and considering the progress of each robot, the program can optimize paths while still ensuring collision avoidance within a multi-robot system.

VI. Ethics

As contributors towards this project, our group recognizes that a multi-robot path planning system can give rise to ethical dilemmas that need to be carefully considered. While a system with multiple robots working efficiently together has positive use cases, potential negative use cases also arise from improved collaboration, as the speed at which malicious robots work together could be increased.

One major application is the use of path planning algorithms in military or defense applications to guide autonomous weapons or drones that can cause harm to humans in conflict situations. The ethical dilemma lies in the potential for these autonomous weapons to make lethal decisions without human intervention, raising concerns about accountability and the risk of civilian casualties. Additionally, path planning algorithms can be utilized to guide autonomous surveillance systems, such as drones or robots, to gather intelligence data. The ethical dilemma centers on concerns of privacy invasion, potential misuse of collected data, and the implications of widespread surveillance on civil liberties.

Supporting views argue that autonomous weapons and surveillance systems reduce casualties and the physical and psychological risks faced by military personnel. By employing autonomous systems, human operators can be kept at a safer distance from potentially dangerous or hostile environments. Additionally, the robots may exhibit faster decision-making and reaction times compared to humans, potentially leading to more effective responses in certain combat scenarios. By employing advanced sensors and targeting systems, a multi-robot system is likelier to achieve higher precision and accuracy in engagement than human operators.

Opposing views argue that by removing direct human involvement in the use of weapons or surveillance systems, it is difficult to assign accountability for the actions made by the robots, as responsibility becomes ambiguous in the absence of direct human control. The development and deployment of autonomous weapons may also contribute to an arms race, heightening concerns about their misuse or falling into the wrong hands. Additionally, the use of autonomous surveillance systems raises major concerns about privacy rights, as widespread monitoring and data collection can infringe upon individuals' privacy without their consent or knowledge.

Using a deontological approach, the establishment of clear ethical guidelines and principles that adhere to international humanitarian laws will help ensure that the development, deployment, and use of autonomous systems align with moral principles. Additionally, multi-robot path planning systems should still maintain human oversight and control over critical decision-making processes. The systems should ensure that humans are responsible for the ultimate authorization of the use of force and can intervene or override the autonomous system's actions if necessary. By establishing mechanisms to ensure accountability for the actions of autonomous systems, multi-robot systems will be able to hold individuals, organizations, or governing bodies responsible for any ethical violations or unintended harm caused by these systems.

Using a utilitarian approach, we can restrict the use of autonomous systems in defense applications and instead allocate such efforts toward humanitarian purposes, such as disaster response, search and rescue operations, or humanitarian aid delivery. By doing so, we can utilize the capabilities of multi-robot path planning to save lives, alleviate suffering, and support the well-being of affected populations. By prioritizing overall societal welfare and minimizing harm toward humans, we can strive to achieve the greatest net benefit while mitigating potential negative consequences of multi-robot path planning systems.

VII. Conclusion

In conclusion, our final project successfully expanded on the concept of path planning by addressing the challenge of multiple robots operating in the same environment. By adopting a decoupled approach that extends from path planning algorithms covered in class, we enabled each robot to independently plan its path while considering the presence of other robots and potential collision scenarios. This approach not only improved the efficiency of the system but also enhanced safety by minimizing the risk of collisions.

However, the algorithm still has some limitations that need improvement. Currently, all paths are restricted to have absolutely no overlap, which can be overly conservative. In some cases, a cell may be free if a robot has not reached it yet. To address this limitation, future work should focus on incorporating dynamic updates to the path planning algorithm, allowing for adaptive and efficient path adjustments based on real-time robot positions. By refining the algorithm to handle dynamic environments and considering the possibility of temporary overlap in paths, we can further enhance the performance and flexibility of the multi-robot path planning system.

Furthermore, our group developed custom launch files and communication mechanisms to effectively handle the coordination and communication among the multiple robots. By applying the concepts and techniques learned in single robot path planning, we were able to tackle the more complex task of coordinating and planning paths for multiple robots. This project highlights the importance of scalability and adaptability in real-world scenarios and demonstrates how path planning techniques can be extended to enhance the efficiency and safety of multi-robot systems.

Acknowledgments

We acknowledge the following contributions towards developing the final project:

- **Andres**: Launch files, grid processing
- **Jake**: Robot initialization, execution of path sequence
- **Jason**: Path planning, collision prevention algorithms
- **Ravi & Professor Quattrini Li:** Assistance on launch files

References

[1] E. García, J. R. Villar, Q. Tan, J. Sedano, and C. Chira, "An efficient multi-robot path planning solution using A* and coevolutionary algorithms," *Integrated Computer-Aided Engineering*, *30*(1), pp. 41–52, 2022.

[2] P. Shi and J. N. Hua, "Mobile Robot Dynamic Path planning based on artificial potential field approach," *Advanced Materials Research*, pp. 490–495, 994–998, 2012.

[3] Y. Yang, J. Li, and L. Peng, "Multi-robot path planning based on a deep reinforcement learning DQN algorithm," *CAAI Transactions on Intelligence Technology*, 5(3), pp. 177–183, 2020.

[4] H. Jun and Z. Qingbao, "Multi-objective mobile robot path planning based on improved genetic algorithm," *2010 International Conference on Intelligent Computation Technology and Automation*, 2010.

[5] J. Zhao and X. Fu, "Improved ant colony optimization algorithm and its application on path planning of Mobile Robot," *Journal of Computers*, *7*(8), 2012.

[6] S. Carpin and E. Pagello, "On parallel RRTs for multi-robot systems," *Proc. Italian Association for Artificial Intelligence*, pp. 834–841, 2002.

[7] R. W. Ghrist and D. E. Koditschek, "Safe cooperative robot dynamics on graphs," *SIAM Journal on Control and Optimization*, 40(5), 2002.

[8] M. Saha and P. Isto, "Multi-robot motion planning by incremental coordination," *Intelligent Robots and Systems*, pp. 5960–5963, 2006.

[9] M. Cáp, P. Novák, A. Kleiner, and M. Selecký, "Prioritized Planning Algorithms for Trajectory Coordination of Multiple Mobile Robots", *IEEE Transactions on Automation Science and Engineering*, 12(3), pp. 835–849, 2015.

[10] S. D. Han and J. Yu, "DDM: Fast Near-Optimal Multi-Robot Path Planning Using Diversified-Path and Optimal Sub-Problem Solution Database Heuristics," *IEEE Robotics and Automation Letters*, 1(1), pp. 99, 2020.

[11] E. Masehian and A. Hassannejad, "A Hierarchical Decoupled Approach for Multi Robot Motion Planning on Trees," *IEEE International Conference on Robotics and Automation*, 2010.