Multi-agent Motion Planning for Non-Holonomic Mobile Robots via Heuristic Optimization

Om Vinayak Gaikwad
Robotics Engineering Department
Worcester Polytechnic Institute
Worcester, MA
ogaikwad@wpi.edu

Vishrut Bohara
Robotics Engineering Department
Worcester Polytechnic Institute
Worcester, MA
vbohara@wpi.edu

Aadesh Surendra Varude
Robotics Engineering Department
Worcester Polytechnic Institute
Worcester, MA
avarude@wpi.edu

***Abstract*—[Youtube Link] The objective of this project is to propose a motion planning methodology that can generate trajectories for multiple non-holonomic mobile robots, which are safe, dynamically feasible, and nearly optimal. The state-of-the-art [4] literature divide the path search of the agents based on priority, which is further used for planning the path using Enhanced Conflict Based Search (ECBS). However, prioritized optimization may lead to infeasible subproblems for lower-priority robots due to lack of consideration by higher-priority robots. Hence, we propose a heiristic based motion planning approach that generates safe, dynamically feasible and near-optimal trajectories for multiple non-holonomic mobile robots. Our current implementation is divided into two parts, enhanced conflict-based search (ECBS) [1]is leveraged as the multi-robot discrete path planner. These paths are subject to constraints to ensure obstacle avoidance and further these paths are parsed through a Los (Line of sight) control checking algorithm [7] to generate a nonholonomic feasible path. Our current implementation involves A* based prioritized MAPF algorithm with feasible motion primitives, which adhere to the non-holonomic constraints of differential drive robot. We also propose heuristic based A* MAPF algorithm with feasible motion primitives, which imply the non holonomic constraints. We test our implementation on the DiffusionBot in ARGoS. Additionally, we have developed a custom visualizer using OpenCV to evaluate our algorithm's performance.**

***Index Terms*—Multi Agent Path Finding (MAPF), Conflict-Based Search (CBS), Motion primitives, heuristic, Non-holonomic.**

I. Introduction

Multi-robot systems due to their capacity to provide more varied usefulness and efficiency when compared to single-robot systems, have grown in popularity within the industry. This system allows Mobile robots to navigate through complex environments while coordinating with one another. This coordination involves generating collision-free trajectories connecting the initial and final positions of each robot, which is known as the labeled multi-robot trajectory planning problem.

In previous research, the problem of generating feasible near to optimal trajectories for a group of robots is addressed by using various trajectory generation methods [1]. Although, these techniques do not guarantee optimality and work only for fewer obstacles.

In environments with many obstacles, the trajectory planning problem is commonly solved by following a two-stage pipeline, pathfinding and trajectory optimization, which involves generating a geometric path and then optimizing it for smooth and dynamically feasible trajectories. This pipeline can be applied in multi-robot trajectory planning, where collision-free discrete paths are generated for all robots, followed by formulating a quadratic program problem to optimize the trajectory of each robot. These multi-robot trajectory planning algorithms using the two-stage pipeline guarantee completeness and also significantly improve computational efficiency due to the pathfinding stage providing a good initial guess for the trajectory optimization.

However, these trajectory planning algorithms are designed for robots with linear dynamics, which can be formulated as a convex quadratic programming problem. In contrast, most modern mobile robots are differential-drive and subject to nonlinear dynamics, which makes the existing multi-robot trajectory planning methods inapplicable since trajectory optimization in this case is a general nonconvex nonlinear programming problem. Therefore, motion planning for multiple differential-drive robots is typically tackled via discrete formulations. However, these planned piecewise linear paths include corner turns that are dynamically infeasible for differential-drive robots, making them difficult for the robots to execute.

II. Literature Review

The paper ”Efficient Trajectory Planning for Multiple Non-holonomic Mobile Robots via Prirotizied Trajectory Optimisation” [4] presents an approach for planning efficient trajectories for multiple non-holonomic mobile robots in cluttered environments. The paper proposes a prioritized trajectory framework to overcome the computational complexity in multi-robot trajectory planning. The problem focused here is to solve issues associated with multi-robot path planning by finding optimal trajectories for multiple robots while avoiding collisions with the surrounding obstacles and other robots as well. As explained in this paper this becomes more challenging when the robots are non-holonomic considering the constraints.

To solve the challenge of multi-robot path planning the authors have devised the following solution: First, using a multi-robot path planner to generate solutions to an initial problem by finding the path from the start to the goal. This is done by first implementing a search method to find the shortest and collision-free path for all non-holonomic robots.

Implementing a safe corridor around each robot's path which models the safe space of the robot along with a prioritized optimization method increases the computational efficiency while decoupling the multi-robot trajectory optimization problem. In this work Enhanced Conflict Based Search (ECBS) method is used to take advantage of the multi-robot path planner. ECBS works in a two-fold method, first, at a high level, a binary constraint tree is formed to resolve the detected conflicts, and at the low-level optimal paths for individual robots are planned. In the trajectory optimization process the line segment of the path is divided into h equal parts and the method is followed in two parts: First, the differences between the two control inputs are penalized for the smoothness of trajectories, and in second part since the obtained solution is already feasible to keep this feasibility the deviation between the optimal trajectory and the references trajectory is penalized.

Lastly, exhaustive evaluations of the proposed approach by performing simulations and performing real-world experiments. In this paper the authors have tested their algorithms using OctoMap to visualise the occupancy map and IPOPT to solve the trajectory optimisation problem. In the real world, the multi-robot navigation is done using 3 robots and one of the robots is equipped with LiDAR which is used to map the environment.

In conclusion, this paper has presented an efficient trajectory planning algorithm for multiple non-holonomic mobile robots in a cluttered environment. A prioritized trajectory optimization method is introduced to plan paths based on priority and later perform trajectory optimization methods. Finally, the algorithm is tested on both simulation and real-world setups.

The paper "Suboptimal Variants of the Conflict-Based Search Algorithm for the Multi-Agent Pathfinding Problem" [1] proposes several modifications to the Conflict-Based Search (CBS) algorithm to improve its runtime performance while still maintaining near-optimal solutions for the Multi-Agent Pathfinding Problem (MAPF). The authors start by introducing the MAPF problem, which involves finding collision-free paths for multiple agents in a shared environment. They then present the CBS algorithm, which is a two-level approach that first generates a conflict-based search tree and then solves the conflicts using various techniques. The authors proposed methods are Enhanced CBS, Bounded Suboptimal CBS, and Greedy-CBS (GCBS): Suboptimal CBS. The Greedy CBS (GCBS) algorithm employs the same framework as CBS, but offers greater flexibility in both the high-level and low-level searches, prioritizing the expansion of nodes that have a higher likelihood of quickly producing a valid solution, even if it is suboptimal. For the BCBS variant, the authors suggest applying both levels of CBS as focal search, in the high level we can apply the focal search to the nodes based on the cost of the node and the number of hops towards the goal as a heuristic. The low level focuses more on applying the focal search on a consistent single agent path, based on the A* heuristic f(n)=g(n)+h(n) and a number of hops towards the partial path up to a node as heuristic. The ECBS algorithm generates a CT node and the low-level search returns two values to the high level search the cost of node and the minimum lower bound of all nodes in the OPEN list of the high-level search. ECBS has an advantage over BCBS in that it offers greater flexibility in the high-level search, while allowing the low-level search to have the same level of flexibility as BCBS. Specifically, ECBS provides extra flexibility in the high-level search when the low-level search produces low-cost solutions, which is indicated by the lower bound value being close to the node's actual cost.

The paper "Efficient Multi-Agent Trajectory Planning with Feasibility Guarantee [2] using Relative Bernstein Polynomial" proposes a new approach for solving the Multi-Agent Trajectory Planning (MATP) problem using Relative Bernstein Polynomials (RBP). The authors start by introducing the MATP problem, which involves finding collision-free trajectories for multiple agents in a shared environment. They then present the RBP method, which is used to generate smooth and continuous trajectories that satisfy velocity and acceleration constraints. For the trajectory formation of each quadrotor with the given start and goal point, the authors use Bernstein polynomial to represent it in M segment form that contains all the control points of the quadrotor , as shown in equation below

$$p^i(t) = \begin{cases} \sum_{k=0}^{n} c_{1,k}^i B_{k,n}(\tau_1) & t \in [T_0, T_1] \\ \sum_{k=0}^{n} c_{2,k}^i B_{k,n}(\tau_2) & t \in [T_1, T_2] \\ \vdots & \vdots \\ \sum_{k=0}^{n} c_{M,k}^i B_{k,n}(\tau_M) & t \in [T_{M-1}, T_M] \end{cases}$$

The jerks are minimized based on the objective function, and the constraints for the maximum velocity and acceleration are define by the convex constraints. THe authors deal with the obstacle avoidance by satisfying the condition by Minkowski sum of the obstacle configuration space and the control points of the each quadrotor (As shown in the below equation).

$$p^i(t) \oplus \mathcal{C}_{obs}^i \subset \mathcal{F}, \quad t \in [T_0, T_M]$$

The author generates a initial trajectory using the ECBS based variant of the MAPF family. After the initial trajecotry is generated then they build a safety flight corridor for all the quadrotors using RBP's convex hull property for control points of each quadrotor. Then they optimize the trajectory by an efficient sequential optimization method by using dummy agents. The authors proposed algorithm performs better than the SCP based methods with 100% collision avoidance guarantee and in computation and conversion.

"Multi-Agent Path Finding with Mutex Propagation" [3] is a research paper that proposes a new approach for solving the Multi-Agent Path Finding (MAPF) problem using Mutex Propagation. The paper starts by introducing the MAPF problem and discussing its importance in various real-world applications. It then presents the existing approaches to solving the MAPF problem and highlights their limitations. The authors then propose the use of Mutex Propagation as a new approach to solving the MAPF problem. The authors divide the given goal and path graph into an MDD tree structure and then find the mutex node in order to check the cardinal conflict condition, later these nodes are used as constraints, to obtain an optimal solution.

The paper "Risk-DTRRT-Based Optimal Motion Planning Algorithm for Mobile Robots" proposes a method to improve the quality of the path generated using time-based RRT in a dynamic environment. The algorithm is proven to return a dynamically feasible homotopy optimal path for the RRT-generated path. The algorithm is divided into two major components, a LoS control checking algorithm which calculated the dynamically feasible line of sight path and a rewiring algorithm which uses the LoS algorithm to calculate the homotopy optimal path. We have used a similar algorithm to find the homotopy optimal path for the A* path calculated using CBS. This algorithm is fast and can be used in place of optimization algorithms.

III. Implemenations

A. Prioritized CBS based A* using LoS

Firstly, we use a conflict-based search (CBS) algorithm as a multi-robot discrete path planner. This algorithm is capable of considering the conflicting trajectories of multiple robots and generating paths that can avoid collisions with obstacles and other robots.

Secondly, we enforce additional constraints on the generated paths to ensure non-holonomic feasibility, which means that the paths adhere to the motion constraints of non-holonomic mobile robots, such as differential drive robots. To achieve this, we apply a line of sight (LOS) control checking algorithm that ensures the generated paths can be realistically followed by the non-holonomic robots without violating their motion constraints.

One way to satisfy non-holonomic and heuristic constraints in a given problem is by utilizing smoothing algorithms such as the Biezer curve and B Spline approach. These algorithms can help to reduce the roughness of the solution trajectory while maintaining the required constraints. The Biezer curve is a mathematical function that smoothly connects two or more points in a curve, while the B Spline approach uses a set of basis functions to approximate the shape of the curve.

B. Prioritized CBS based A* with motion primitives.

Our objective is to propose a motion planning strategy that can generate optimal trajectories for multiple mobile robots while taking into account their safety and dynamic feasibility. Our approach relies on the implementation of a conflict-based search (CBS) algorithm using A* search. Specifically, we prioritize the checking of the validity of each node with the path of the other agents, while accounting for time constraints.

At each iteration of the CBS algorithm, we select the next node based on eight motion primitives, which are the different types of movement that the robot can perform, and take into consideration the non-holonomic constraints. We then calculate the cost of the child node and parent node and push the result into the queue.

Overall, this approach aims to find the optimal trajectory for each mobile robot while ensuring its safety and avoiding any potential collisions with other robots or obstacles. By considering the dynamic feasibility of each motion, we can generate trajectories that are not only safe but also efficient and effective in achieving the goals of the robots.

Algorithm 1 Prioritized CBS based A* using LoS

Input: The map, $start$ nodes, and end nodes of all the robots in priority order
Output: Non-holonomic constraint feasible trajectories
$computed\_paths$ = []
for i in all robots **do**
$initial\_path = CBS\_A^*\_path(start[i], end[i], computed\_paths)$
$possible\_start$ = $start[i]$
$next\_end$ = 0
L_{last} = 0
while $end[i]$ not reached **do**
  L = $LoS\_path(possible\_start, initial\_path[next\_end])$
  if L is collision free **then**
    $next\_end$ = $initial\_path[next\_end + 1]$
    $L_{last} = L$
  else
    $actual\_path$.append(L_{last})
    $possible\_start$ = $initial\_path[nextend - 1]$
  end if
end while
$actual\_path$.append(L_{last})
$computed\_paths$.append($actual\_path$)
end for
Return $computed\_paths$

Algorithm 2 Prioritized CBS based A* with motion primitives

Input: The map, $start$ nodes, end nodes of all the robots in priority order, and motion $primitives$
Output: Non-holonomic constraint feasible trajectories
$computed\_paths$ = []
for i in all robots **do**
$path = get\_path(start[i], end[i], computed\_paths, primitives)$
$computed\_paths$.append($path$)
end for
Return $computed\_paths$

C. Heuristic CBS with motion primitives

This algorithm is a variation of a multi-agent path planning algorithm, which aims to find collision-free paths for multiple agents (robots) to reach their respective goals. The first step is to discretize the configuration space by dividing the environment or workspace into a set of discrete points. This is typically done to simplify the problem and reduce the complexity of the search space. This can be done in various ways, such as using a grid or Voronoi diagram where each point in the set represents a possible location for an agent to occupy. The next step is to calculate the path of costs: For each agent, the algorithm then calculates the cost of traveling from each possible location (discrete point) to its goal. This is done using a breadth-first search (BFS) algorithm, which explores all possible paths from the starting location to the goal, while keeping track of the cost of each path. In addition to BFS, motion primitives may be used to calculate the path costs, which are essentially pre-defined sequences of movements that an agent can perform. The last step is to do Greedy Path selection which involves the calculation of path costs have been calculated for all agents, the algorithm then selects a path for each agent that minimizes the sum of BFS cost and distance heuristic with all the other agents. The distance heuristic is a measure of the distance between the current location of an agent and its goal. The aim is to find a path that is both collision-free and efficient in terms of time and distance.

Algorithm 3 Heuristic CBS with motion primitives

Input: The map, $start$ nodes, end nodes of all the robots, and motion $primitives$
Output: Non-holonomic constraint feasible trajectories
$goal\_distance = []$
for i in all robots **do**
$distance\_map = BFS\_costs(end[i], map, primitives)$
$goal\_distance.append(distance\_map)$
end for
$heuristic\_function = (\frac{8}{distance})^3$
$reached = False$
$computed\_paths$ = []
$current\_pose = start$
while *not reached* **do**
  $priority = goal\_distance[current\_pose]$
  $robot\_order = argsort(priority)$
  for i in $robot\_order$ **do**
  $next\_pose = Greedy\_NH(i, current\_pose, Heuristic)$
  $current\_pose[i] = next\_pose$
  end for
  $reached = check\_reached(end, current\_pose)$
  $computed\_paths$.append($actual\_path$)
end while
Return $computed\_paths$

IV. Experimental Results

In this section we have implemented a baseline CBS algorithm for multi-agent path planning from scratch, videos for the implementation could be found on our youtube presentation [**Youtube Link**]. We have attached images (Fig1,Fig2,Fig3) of the multi-agent robot path that we obtain from our code.

Further, we have implemented CBS based A* algorithm with motion primitives that satisfy the non-holonomic constraints for multi-agents (Fig4,Fig5,Fig6).

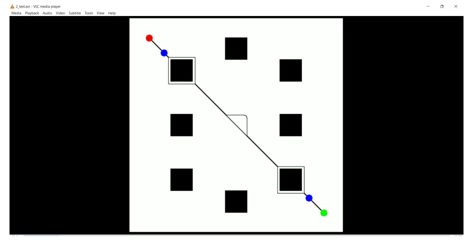

Fig. 1: Visualization of the path for 2 agents.

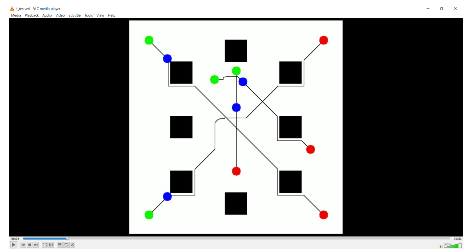

Fig. 2: Visualization of the path for 4 agents.

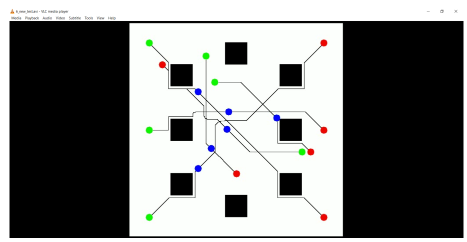

Fig. 3: Visualization of the path for 6 agents.

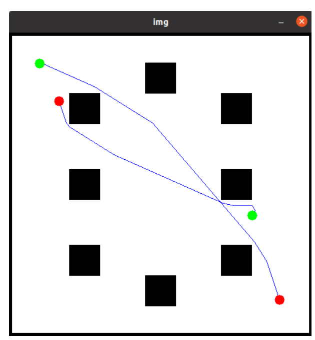

Fig. 4: Non holonomic path for 2 agents.

Lastly, we have implemented Heuristic based online CBS using Motion Primitives for non-holonomic mobile robots Fig7,Fig8).

We use the ARGoS Fig9 as experimental platform to visualize our results. It is a multi-robot physics simulator that is lightweight and user-friendly, making it easy to analyze algorithms on multiple robots. The platform allows users to model the number and type of robots, simulation world, and sensors using an XML file. The robot controller and loop function file contains the control algorithms that produce the trajectory and set the respective velocities for each robot.

Our evaluation for the path lengths Fig10 conveys that

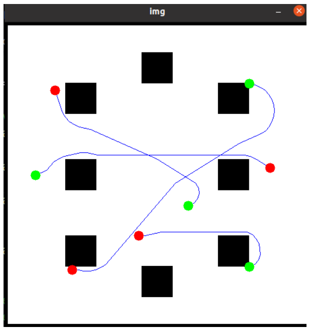

Fig. 5: Non holonomic path for 4 agents.

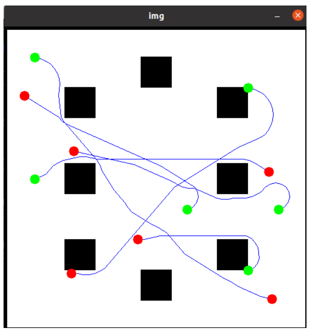

Fig. 6: Non holonomic path for 6 agents.

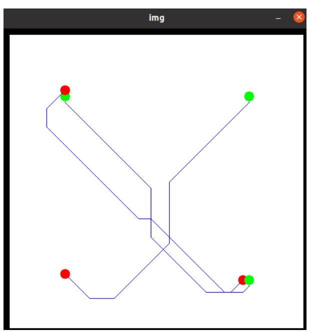

Fig. 7: Non holonomic hueristic based path for 3 agents.

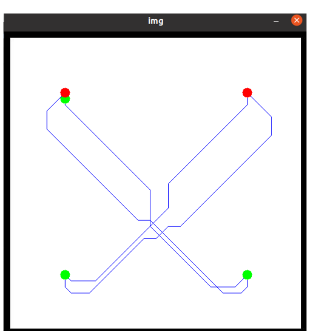

Fig. 8: Non holonomic hueristic based path for 4 agents.

the heuristic paths are heavily dependent on the heuristic choice and tuning, and can outperform the priority algorithm. Heuristic algorithm takes more computation time than the prioritized algorithms as shown in Fig11, the growth of the

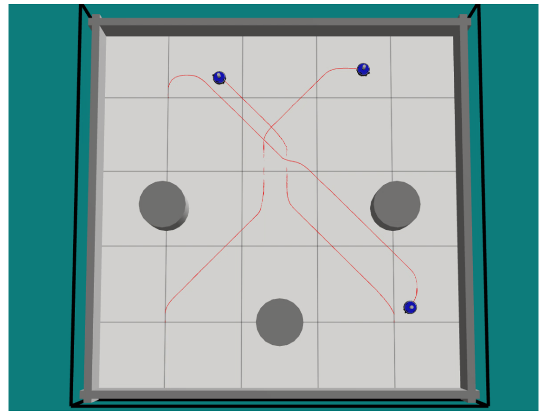

Fig. 9: ARGoS simulation setup.

difference in computation time is dependent on the map size.

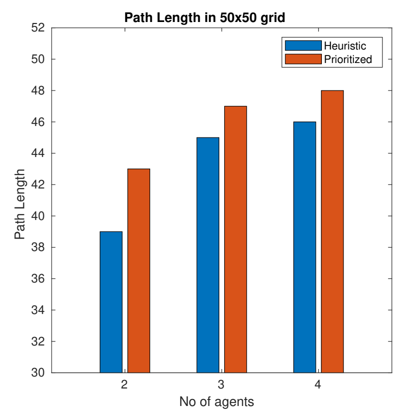

Fig. 10: Path length comparison plot.

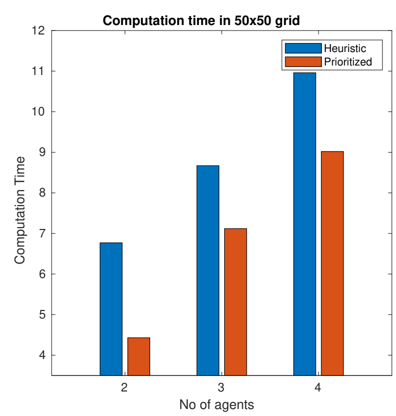

Fig. 11: Time computation plots.

V. Conclusion and Limitations

We have successfully implemented the A* based prioritized algorithm with motion primitives for 2,4,6 agents. We have also presented the implementation of A* based heuristic algorithm for multiple agents. The Heuristic-based algorithm serves as evidence of the limitations of the current state-of-the-art (SOTA) priority-based algorithm. While the priority-based algorithm is scalable, it falls short in terms of optimality. On the other hand, the Heuristic-based algorithm, while scalable, cannot guarantee completeness, and is only approximately optimal. These shortcomings have highlighted the need for continued research and development to create an algorithm that is both scalable and optimal, while also guaranteeing completeness.

VI. Schedule (Task Division and Timeline)

In the following table we have added our current work distribution and estimate of future work and distribution (Refer TABLE I).

TABLE I

Timeline	Task	Task allocation
Week 1	Literature Review and concept understanding.	Aadesh, Om, Vishrut
Week 2	Literature Review and State of the art simulation setup and code study.	Aadesh, Om, Vishrut
Week 3	State of the art setup error issues and new gazebo simulation setup.	Om
Week 3	Code implementation and visualizer setup	Vishrut, Aadesh
Week 4	Proposal. slides, Video creation	Aadesh, Om, Vishrut
Week 5	Spring break	
Week 6	Exploration of the optimization and smoothening methods.	Aadesh, Om, Vishrut
Week 6	Understanding motion primitives.	Aadesh
Week 7	Debugging the ROS simulation errors	Om,Aadesh
Week 7	Exploration of the optimization and smoothening method	Aadesh, Vishrut
Week 6/7	Setting up environment in ARGoS	Om, Vishrut
Week 8/9	A* implementation with motion primitives for MAPF.	Aadesh
Week 8/9	ARGoS simulation robot implementation	Om
Week 8/9	Implementation of optimization technique.	Vishrut
Week 9	Working and designing the heuristic function.	Om, Vishrut
Week 10/11	ARGoS code implementation with the new smoothen curve for 3 Robots	Aadesh, Vishrut, Om
Week 12/13	Final testing and simulation implementation	Aadesh, Vishrut, Om
Week 14	Final report and Documentation	Aadesh, Vishrut, Om

References

[1] Barer, Max, et al. "Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem." Proceedings of the International Symposium on Combinatorial Search. Vol. 5. No. 1. 2014. 434-440, doi: 10.1109/ICRA40945.2020.9197162.

[2] J. Park, J. Kim, I. Jang and H. J. Kim, "Efficient Multi-Agent Trajectory Planning with Feasibility Guarantee using Relative Bernstein Polynomial," 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 2020, pp. 434-440, doi: 10.1109/ICRA40945.2020.9197162.

[3] Zhang, H., Li, J., Surynek, P., Kumar, T. S., Koenig, S. (2022). Multi-agent path finding with mutex propagation. Artificial Intelligence, 311, 103766.

[4] J. Li, M. Ran and L. Xie, "Efficient Trajectory Planning for Multiple Non-Holonomic Mobile Robots via Prioritized Trajectory Optimization," in IEEE Robotics and Automation Letters, vol. 6, no. 2, pp. 405-412, April 2021, doi: 10.1109/LRA.2020.3044834.

[5] W. Ding, W. Gao, K. Wang and S. Shen, "An Efficient B-Spline-Based Kinodynamic Replanning Framework for Quadrotors," in IEEE Transactions on Robotics, vol. 35, no. 6, pp. 1287-1306, Dec. 2019, doi: 10.1109/TRO.2019.2926390.

[6] Choi, J. W., Curry, R., Elkaim, G. (2008, October). Path planning based on bézier curve for autonomous ground vehicles. In Advances in Electrical and Electronics Engineering-IAENG Special Edition of the World Congress on Engineering and Computer Science 2008 (pp. 158-166). IEEE.

[7] W. Chi, C. Wang, J. Wang and M. Q. . -H. Meng, "Risk-DTRRT-Based Optimal Motion Planning Algorithm for Mobile Robots," in IEEE Transactions on Automation Science and Engineering, vol. 16, no. 3, pp. 1271-1288, July 2019, doi: 10.1109/TASE.2018.2877963.