Motion Planning of Mobile Robot in Dynamic Obstacle Environment

Uday Sankar [1] **Gowri Shankar Sai Manikandan** [1] **Upasana Mahanti** [1] **Neha Malaviya** [1]

1. Introduction

About 2,000 deliveries are made by delivery drones in the United States every day. The more deliveries it makes, the more likely it is to crash. The presence of dynamic obstacles in the environment is a big problem. There is a possibility of a collision in the air, especially when multiple UAVs use the same airspace or when delivery drones fly in an enclosed space such as a warehouse. Dynamic path planning is a method for finding a path for a robot or autonomous vehicle to follow in a dynamic environment. An environment is considered dynamic because it can change over time, for example due to moving obstacles or changes in the layout of the environment.

Our project focuses on implementing two approaches based on obstacle movement in dynamic environments, velocity obstacles and nonlinear model predictive control, in an environment similar to a delivery drone.

2. Background Work

There have been several traditional algorithmic approaches to solve a single agent dynamic environment motion planning. There have been several algorithms like variations of traditional motion planning algorithms like A*, RRT in order to work for dynamic obstacle environment. Being approaches mainly designed for static obstacles, they don't consider the motion of the obstacles in dynamic environment, Thus, it leads to lot of complexities like time taken for calculating path, safety issues due to lack of motion sensory inputs etc.

The algorithm used for dynamic path planning must be able to take into account the current state of the environment and plan a path that avoids obstacles while also reaching the goal. The path planned by the algorithm must also be able to adapt to any changes in the environment that occur while the robot is following the path. Many methods have been found to perform dynamic path planning. Two of such methods that we intend to explore are the Velocity Obstacles Approach and the Non-Linear Model Predictive Control.

Firstly, Velocity Obstacles approach involves generating a set of constraints on the velocity of an autonomous agent, so that it can avoid collisions with the objects in the environment. This is done by considering the motion of other objects in the environment as obstacles.This way, a more dynamic and reactive solution can be obtained for path planning.

Model Predictive Control (MPC) is a control strategy that uses mathematical models to predict the future behavior of a dynamic system and uses that information to optimize a control policy. MPC algorithms use an optimizer to determine the best control action that balances conflicting objectives such as tracking a reference signal, ensuring constraints are not violated, and minimizing the use of control inputs.

There have been several work done in Vehicle Obstacles and NMPC, for multi agent vehicles, while we consider for single agent, mainly due to the complexity of the problem, and also, in a realistic drone delivery environment, we are concerned with single agent because of the fact that we are in control of the single drone, rather than controlling all the obstacles that come in the way. As there is not much research for single agent vehicles for these algorithms, we consider the research in MAV to be true for single agent as well.

One of the earliest works on reactive collision avoidance in a group of flying robots with nonlinear model predictive control (NMPC) is work in (1), where distributed NMPC is used to control multiple helicopters in a complex environment. artificial potential field to avoid reactive collision. This approach provides no guarantees and is only evaluated in simulation. In (2), the authors present different algorithms based on the concept of velocity obstacles (VO) to select collision-free trajectories from candidate trajectories. The method was experimentally evaluated with four MAVs flying in close proximity, including humans. However, MAV dynamics is not considered in this method, and decoupling trajectory generation of control has several limitations, as shown in the experimental section of (3). One of the efforts to combine trajectory optimization and con-

[1]Worcester Polytechnic Institute. Correspondence to: Uday Sankar <usankar@wpi.edu>, Gowri Shankar Sai Manikandan <gmanikandan@wpi.edu>, Upasana Mahanti <umahanti@wpi.edu>, Neha Malaviya <nmalaviya@wpi.edu>.

RBE550 Motion Planning Porposal, Worcester Polytechnic Institute.

trol is the work presented in (4). The robot control problem is formulated as a bounded-horizon optimal control problem, and unconstrained optimization is performed at each time step to simultaneously generate time-varying feedback gains and feedforward control inputs. This approach has been successfully implemented in MAV devices and a ball balancing robot.

For multi-robot environment, NMPC and VO might not be effective, because of the fact that both of them are de-centralized approaches. In de-centralized approach, we found that each robot has its own controller and motion planner. Centralized approach like Safe Interval Path Planning (SIPP) are found to work better in a multi robot environment, because of the fact that there is a single central controller that plans the motion of all the robots in the dynamic environment. The concept of Priority Safe Interval Path Planning planning involves assigning a unique priority number to each individual robot being considered for path planning. It then proceeds to generate collision-free paths for each robot based on its priority order. As the planning process iterates through each robot, the paths for each robot are saved, along with the nodes located within a radius of r from the path, in order to avoid potential obstacles in the environment (5). However, as it works on priority, every robot cannot be guaranteed a feasible path, whereas when the robots travel in a well structured and informed environment with not much obstruction to paths of other robots, every robot is guaranteed a feasible path from start to finish (6).

For better collision avoidance, we found that the Conflict-based Search (CBS) algorithm focuses on identifying and managing conflicts between agents' paths. This method involves generating a set of constraints for each agent based on these conflicts, which then breaks down the overall multi-agent path planning problem into numerous combinations of constrained single-agent path planning problems. The CBS algorithm operates on two distinct levels, with the higher level being responsible for detecting conflicts between agents' paths and adding constraints, and the lower level being responsible for planning paths that adhere to the constraints of individual agents (7).

2.1. Application in Industry

There are a lot of top companies like Amazon, Walmart which are leading the way in active drone delivery. UPS even did COVID Vaccine deliveries using peak pandemic. Also, given the current flying zone restrictions, it becomes a primary task to focus on dynamic obstacle path planning for delivery drones. The decentralized approaches work well in an environment where there is a single delivery drone, and all agents are obstacles. In an environment where there are lot of drones, we will have to implement centralized approach due to the central planning approach.

Figure 1. Delivery Drone

3. Method

We implemented two Centralized approaches- NMPC and Velocity Obstacle Algorithm, along with two De-Centralized approaches- Priority Safe Interval Path Planning (SIPP), and Conflict Based Search (CBS) planning. They were entirely done in Python and Numpy. The implementations were referred from (8).

3.1. Non Linear Model Predictive Control

In Non-linear Model Predictive Control, the obstacles are defined in the environment and the robot is spawned in the start position. The end goal position of the robot is also defined. The robot starts a loop that runs for the amount of simulation time that has been previously specified. In the loop, the robot first predicts the future positions of the surrounding obstacles based on the velocity and direction of motion of the obstacles. Based on the current state and the end goal of the robot, a reference trajectory is computed for the robot to follow. In order to get the robot to its end goal, the control algorithm computes an optimal velocity that will enable the robot to avoid all the obstacles and yet reach the end goal.

The control algorithm is basically an optimization problem. The goal is to minimize the two kinds of costs - tracking cost and collision cost. The tracking cost is the difference between the goal position and the current position of the robot, while collision cost is the measure of risk of collision between the robot and the surrounding the robot. Once these costs are found, the sum these costs are taken and then the cost function is minimized using the Sequential Least SQuares Programming optimizer (SLSQP) algorithm. This particular algorithm is used because it is capable of minimizing functions of several variables with any bounds, equality and inequality constraints.

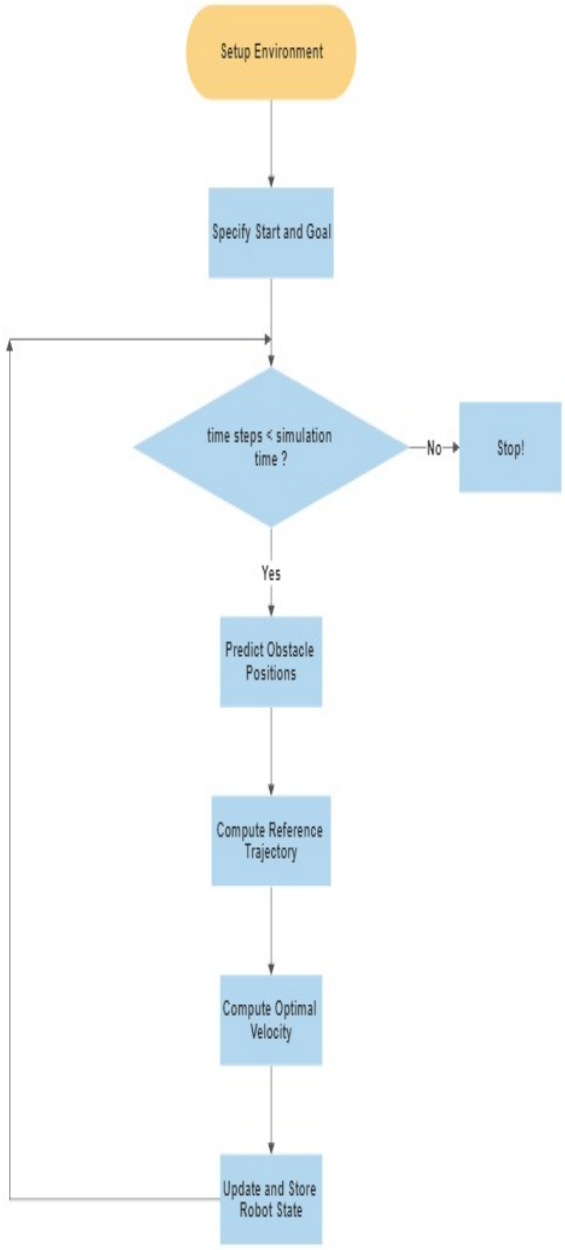

Figure 2. NMPC- Architecture

3.2. Velocity Obstacle

In Velocity Obstacles (VO) path planning, the environment is defined with the obstacles and their motion predetermined. The robot is spawned in its start location and its end goal is also specified. A loop is started which runs for the amount of specified simulation time. Inside the loop, based on the constraints produced by the surrounding obstacles for the robot, an optimal velocity is to be computed. The computed velocity is updated for the robot and is also stored.

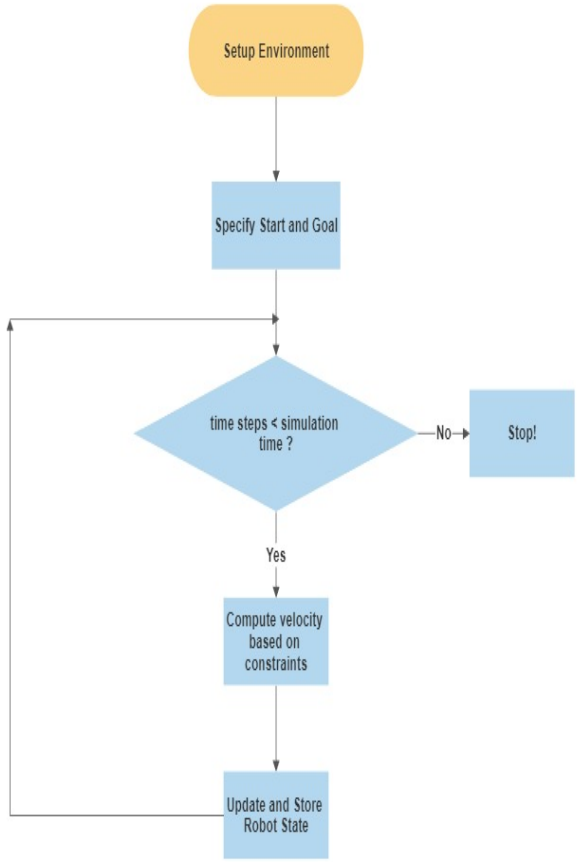

Figure 3. Velocity Obstacle - Architecure

3.3. Prioritized Safe-Interval Path Planning

In its core, Pri-SIPP is an extension of the Safe Interval Path Planning (SIPP) algorithm which plans a path for a single robot amongst dynamic and static obstacles by constructing safe intervals for each cell in a grid map. These safe intervals represent the periods of time during which the cell is not occupied by a dynamic obstacle.

In Pri-SIPP, the robots are prioritized, and the path planning is done one robot at a time according to their priority. When planning for a certain robot, all the previously planned robots are treated as dynamic obstacles, and their planned trajectories are used to calculate the safe intervals. This is done to ensure that the new robot's path does not conflict with the paths of the higher priority robots.

3.4. Conflict-Based Search

Conflict-Based Search (CBS) serves as an effective method for addressing multi-agent pathfinding (MAPF) issues on grid maps. The core concept of CBS is to break down the

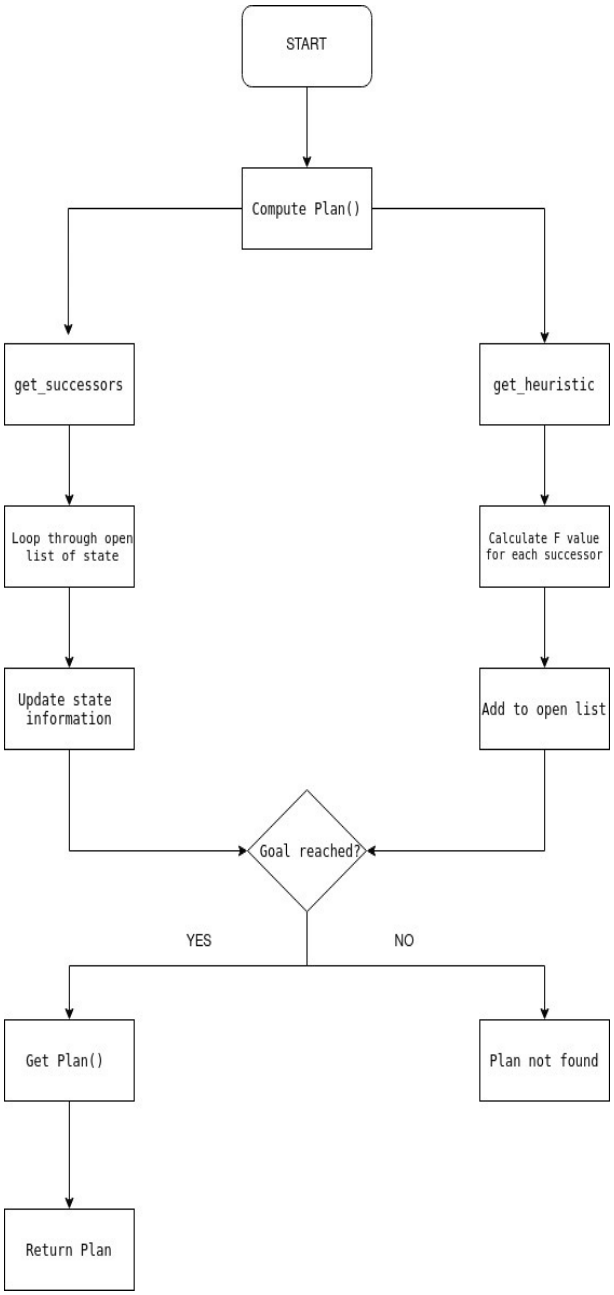

Figure 4. Prioritized Safe-Interval Path Planning - Architecure

task of identifying collision-free paths for each agent into multiple independent subproblems, which are then solved sequentially while considering the conflicts that emerge between agents.

During each iteration, CBS initially produces a high-level plan for every agent, outlining a series of waypoints to traverse while avoiding collisions with others. Subsequently, CBS identifies conflicts present in these high-level plans and integrates corresponding constraints within the search space.

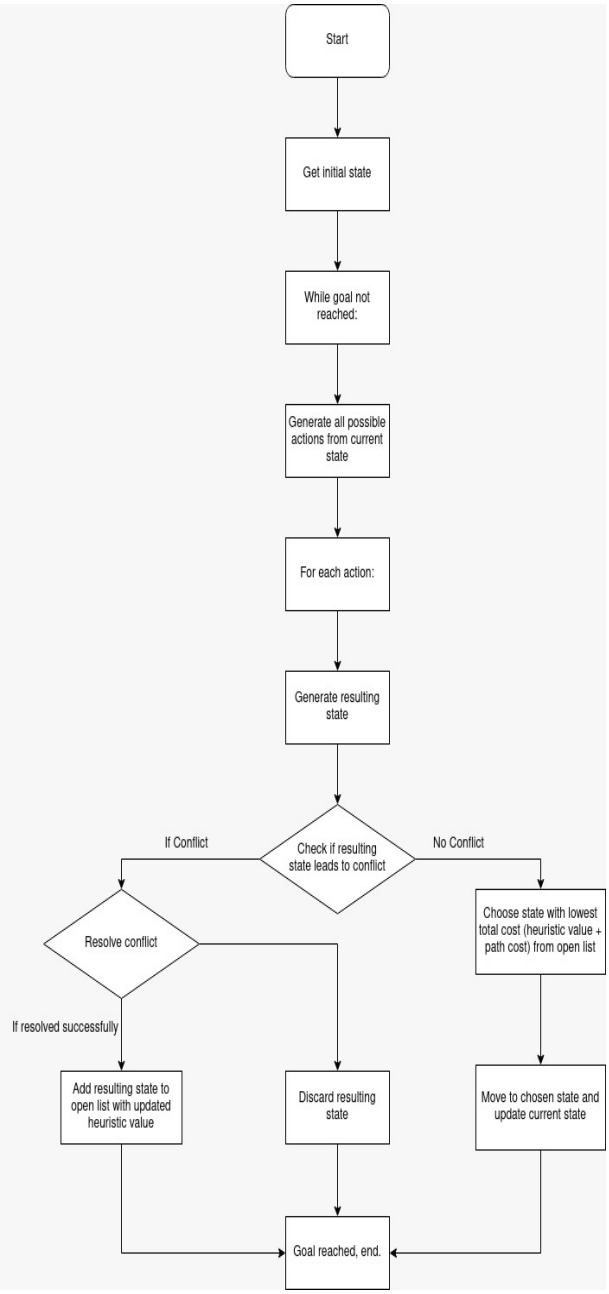

Figure 5. Conflict Based Search - Architecure

4. Challenges

There are some challenges that we have faced till now, and some are possibilities in the future work as well-

- Time–based Simulation System – During our simulation, there are certain test cases for which we could observe that the robot is not moving from the initial position to the goal position. At these times, that robot

might stop navigating even before it reaches the goal position. We are looking into debugging it.

- Model Predictive Control is based on the Vehicle Dynamics of the system. So, a slight change in temperature or moisture in air can give erroneous results. Thus, a strong Vehicle Dynamics Model that considers such conditions needs to be included.

- Our initial goal was to implement MPC and Velocity Obstacle in 3D simulation like Gazebo, with 3 dimensional motion, like that of a drone. However, the main challenge we faced was the conversion from 2D to 3D that adds significant complexity, as our initial implementation was based on 2 axes, and converting it to 3 axes based was implementationally challenging. Thus, we pivoted our end goal to implementing Centralized and Decentralized approach for different dynamic environments as well.

5. Results

A sample path taken by the robot object in dynamic environment, by both the algorithms, NMPC (Figure 6), and Velocity Obstacles (Figure 7) is given below. Along with that, are the paths taken by multi-robots in dynamic environment using Priority Safe Interval Path Planning planning (Figure 8), and Conflict Based Search (Figure 9).

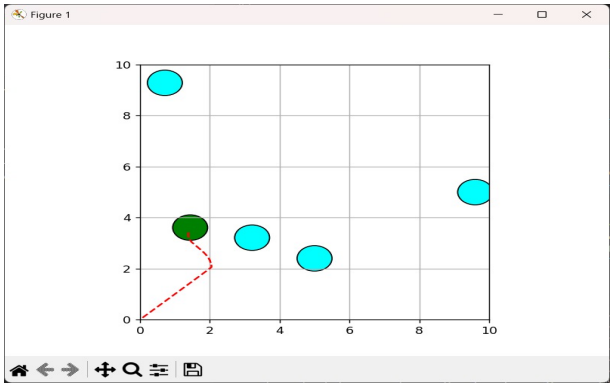

Figure 6. NMPC Sample Output

6. Conclusion and Future Work

In this project, we explored both centralized and decentralized approaches for multi-agent pathfinding. Our results show that each approach has its strengths and weaknesses, and the best choice depends on the specific requirements of the application. Centralized approaches work well with planning a robot in a dynamic environment of obstacles, whereas Decentralized approaches work well in multi robot dynamic environment.

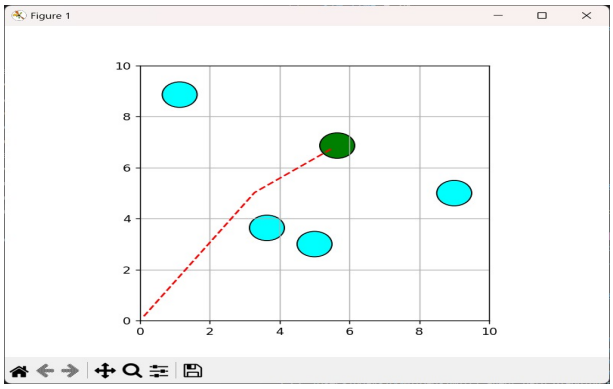

Figure 7. Velocity Obstacle Sample Output

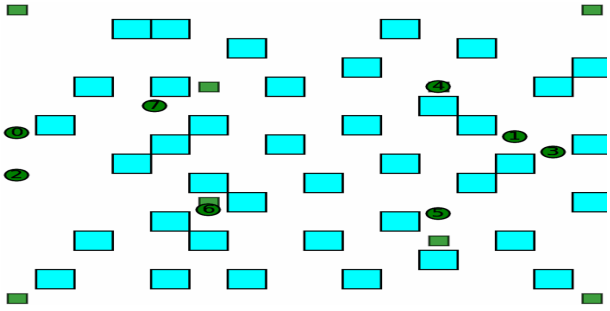

Figure 8. Safe Interval Path Planning

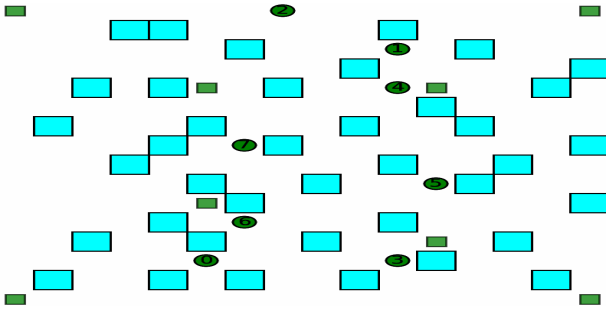

Figure 9. Conflict Based Search

A possible interesting future work can be to explore hybrid approaches that combine the strengths of both centralized and decentralized methods. For example, we could use a centralized method for high-level planning and a decentralized method for low-level control. This could potentially provide the best of both worlds: optimal solutions and scalability.

References

[1] Dong Hyun Shim, Hyo Jin Kim, and Shankar Sastry. Decentralized nonlinear model predictive control of multiple flying robots. In *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, vol-

ume 4, pages 3621–3626. IEEE, 2003.

[2] Javier Alonso-Mora, Tobias Naegeli, Roland Siegwart, and Paul Beardsley. Collision avoidance for aerial vehicles in multi-agent scenarios. *Autonomous Robots*, 39(1):101–121, 2015.

[3] Michael Neunert, Cedric de Crousaz, Fadri Furrer, Mina Kamel, Foad Farshidian, Roland Siegwart, and Jonas Buchli. Fast nonlinear model predictive control for unified trajectory optimization and tracking. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 1398–1404. IEEE, 2016.

[4] Robert Mahony, Vijay Kumar, and Peter Corke. Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robotics Automation Magazine*, 19(3):20–32, 2012.

[5] Brandon Araki, John Strang, Sarah Pohorecky, Celine Qiu, Tobias Naegeli, and Daniela Rus. Multi-robot path planning for a swarm of robots that can both fly and drive. *IEEE Robotics and Automation Letters*, 5(2):1093–1100, 2020.

[6] Michal Cap, Peter Novak, Alexander Kleiner, and Martin Selecky. Prioritized planning algorithms for trajectory coordination of multiple mobile robots. *IEEE Transactions on Automation Science and Engineering*, 12(3):835–849, 2015.

[7] Xiaoxiong Liu, Yuzhan Su, Yan Wu, and Yicong Guo. Multi-conflict-based optimal algorithm for multi-uav cooperative path planning. *Drones*, 7(3), 2023.

[8] Ashwin Bose. Multi-agent path planning. Master's thesis, Ecole Centrale de Nantes, 2022.

A. Appendix

The work has been intended to split into three main parts-

1 . Literature Review: Neha,Upasana,Sai,Uday

2 . Build the environments using NumPy: Sai and Uday

3 . Implement Velocity Obstacle algorithm : Neha and Sai

4 . Implement Model Predictive Control : Uday and Upasana

5 . Implement Prii Safe Interval Path Planning: Neha and Upasana

6 . Implement Conflict Based Search: Sai and Uday

7 . Documentation : Everyone