MAPF and Trajectory Optimization with Drones

Ashna Khemani, ashna.khemani@gmail.com
ACT Lab
American High School, Class of 2022
USC Viterbi Department of Computer Science, SHINE 2021

Introduction

Multiagent Pathfinding (MAPF) aims to facilitate efficient coordination between multiple mobile robots in a finite closed environment. These robot "agents" must navigate from their starting locations to their respective goals using the most efficient route that results in no conflicts with other agents or obstacles.

Objective & Impact of Research

The ACT Lab is developing algorithms for the coordination of multiple robots to complete automated tasks more efficiently. This would help make such teams of robots more commonplace in industry and possibly even in everyday life.

Currently, a popular application of MAPF is in warehousing, such as in Amazon's Colorado facility, where human workers load packages onto robots, along with the corresponding destination within the warehouse. Each of the robots must find the shortest path to their destinations while predicting and avoiding any possible conflicts with other robots or obstacles.[1]

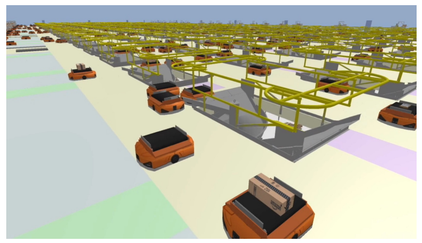

Fig. 1: *Simulation of coordination between robots at an Amazon warehouse*
PC: Amazon

Having an effective MAPF algorithm allows the human workers to rely more on these robots, reducing the physical and mental strain that often comes with the repetitive and physically demanding nature of such jobs.

[1] Simon, Matt. "Your First Look Inside Amazon's Robot Warehouse of Tomorrow." *Wired*, Conde Nast, 5 June 2019

Methods

- Before exploring MAPF, I first implemented Single-Agent Pathfinding (SAPF) algorithms like A* using Python and illustrated the results with MatPlotLib (Fig. 2).
- Though A* is a SAPF algorithm, it can be extended to MAPF by combining it with the Conflict-Based Search (CBS) algorithm. This fusion allows multiple agents to use A* to get to their respective goals while coordinating to avoid conflicts with one another (Fig. 3).
- One challenge with the solution in Fig. 3 is the sharp turns. These would be difficult to execute, especially if the agents are drones. Instead, it is much easier for drones to follow rounded curves, which can be created using Bézier curves. This is essentially like "pulling" on the lines surrounding these sharp corners in different directions to form a smooth curve.
- Implementation of Bézier curves to "round" the paths formed by CBS is shown in Fig. 4.
- These methods were then tested in a more complex instance, with a larger environment and more agents (Fig. 5).

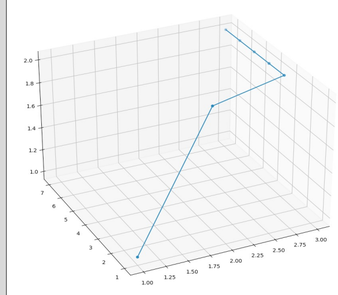

Fig. 2: *A* working for a single agent*
PC: Ashna Khemani

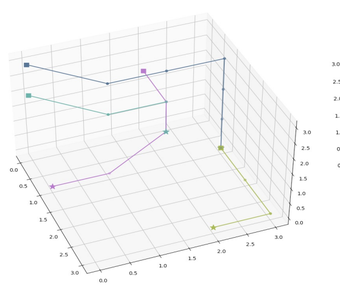

Fig. 3: *CBS allowing coordination between four agents*
PC: Ashna Khemani

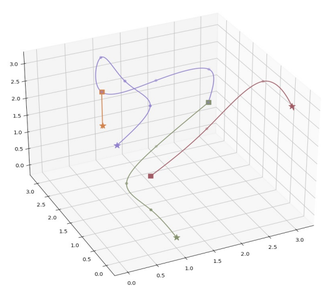

Fig. 4: *Using Bézier curves to make softer, rounder turns.*
PC: Ashna Khemani

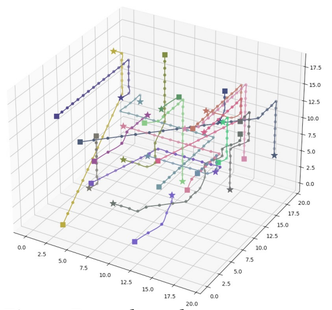

Fig. 5: *Extending these methods to a larger environment with more agents*
PC: Ashna Khemani

Runtime Analysis

I analyzed the relationship between the number of agents and the algorithm runtime. I steadily increased the number of agents from 2 to 10, ran CBS fifty times for each set, and recorded the average runtime for each set (Fig. 6). I used two SciKitLearn modules, PolynomialFeatures and LinearRegression, to find a line of best fit for this data (Fig. 7). A quadratic curve fit the data best, showing this is a $\Theta(n^2)$ algorithm.

# of Agents	Avg. Runtime (ms)
2	20.742
3	223.914
4	426.549
5	432.280
6	1838.574
7	2043.750
8	1844.476
9	3438.979
10	3644.972

Fig. 6: *Runtime data collected and used to find the line of best fit*
PC: Ashna Khemani

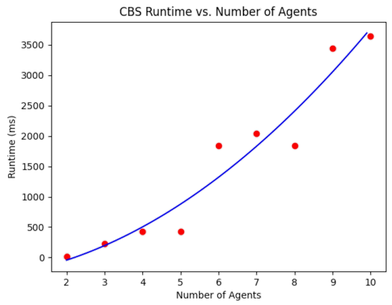

Fig. 7: *Scatter plot of data from Fig. 6 and its line of best fit.*
PC: Ashna Khemani

Skills Gained

- Reading and understanding scientific literature
- Capturing, analyzing, and presenting data
- Learning about the different systematic approaches to MAPF
- Understanding derivation and methods behind Bézier curves
- Effectively creating and using classes in Python
- Using various data structures such as arrays, sets, queues, stacks, and more

Future Research

- Simulating a more continuous space that allows more direct trajectories
- Setting a "preference" for gentle turns over 90º turns
- Implementing obstacle avoidance and "narrow corridor" navigation
- Incorporating physical properties and behaviors of real drones in the algorithm
- Adding efficiency metrics beyond path length (i.e., fuel efficiency)
- Optimize growth of algorithm's runtime, which is currently $\Theta(n^2)$

Acknowledgements

I would like to thank Professor Nora Ayanian for welcoming me into the ACT Lab and giving me this learning opportunity. A big thank you to my mentor, Eric Ewing, for explaining MAPF theory and guiding me throughout this project. Finally, thank you to Dr. Katie Mills, Monica Lopez, Cassandra Jeon, and the rest of the SHINE team and cohort for making this such an enriching and fun-filled experience!