Applying Path-Finding Techniques and Swarm Technologies to Package Delivery

Alex Miu
Department of Computer Science
and Electrical Engineering
University of Maryland, Baltimore County
Baltimore, Maryland
Email: ale18@umbc.edu

Josh McCarter
Department of Computer Science
and Electrical Engineering
University of Maryland, Baltimore County
Baltimore, Maryland
Email: jmccar1@umbc.edu

Itay Tamary
Department of Computer Science
and Electrical Engineering
University of Maryland, Baltimore County
Baltimore, Maryland
Email: itay1@umbc.edu

Jack Wang
Department of Computer Science
and Electrical Engineering
University of Maryland, Baltimore County
Baltimore, Maryland
Email: zhennan1@umbc.edu

***Abstract*—The goal of this project is to experiment with different swarm techniques and path-finding algorithms in order to build the most effective independently functioning character that can work with other characters to map out and navigate a maze. To develop this project, the character will perform two functionalities, the first being maze mapping, the second being maze navigation, represented as package delivery to different locations in the maze.**

I. Introduction

Most fields of robotics use several different kinds of artificial intelligence applications. Functionalities such as navigation might implement computer vision and path-finding, while healthcare robots might one day use natural language processing to analyze patient responses. A currently growing field of technology that applies multiple aspects of artificial intelligence is autonomous vehicles. Our motivation for this project was to apply autonomous vehicle technologies to building navigation and package delivery. The environment we chose is a grid-based maze, and there will be multiple agents maneuvering through the grid. Having multiple agents in the maze adds the complexities of collision avoidance and target sharing (discussed later). These individual agents, which we refer to as cars, will navigate the maze independently.

In the first phase, the goal of a car is to map out the maze by discovering unknown grid points. A grid point will be discovered by determining which sides of the grids are walls of the maze, and which sides are open paths. Each independent car in the maze will be able to map out the grid space alone, but in this project they will not have to. When a car discovers a grid point in the maze, they update their own graph of the maze, but also attempt to update the graphs of all other cars. Since multiple cars may be in the maze at once, they will have to coordinate movement so as not to crash into each other, and not to duplicate effort by selecting the same target. We refer to the latter topic as target sharing, which we define as two independent cars planning on completing the same task. An example of target sharing in the first phase is when two cars have selected the same undiscovered grid point as their goal. When this occurs, the cars would decide which one will select a different undiscovered grid point as their target to avoid duplicating mapping effort. The first phase is completed when all reachable grid points in the maze have been discovered

The second phase applies path-finding to the maze by changing the goals of the cars. Rather than mapping the maze, the cars will be tasked with navigating to specific locations in the maze, which we refer to as rooms. Cars will move to a task-assigning room in the maze to receive their target room. The locations of the rooms in the maze are discovered during the first phase, as room locations are predetermined and built into the maze. Once a car receives their target room (we call this picking up a package), it will determine the shortest path to the target room. When a car determines its path, it must also account for the current positions of all other cars in the maze, so as to avoid collisions. If two cars have conflicting paths, they will determine which car needs to move out of the path. An example of this is a room down a single-grid isle in the maze, which only has one entrance path. If the first car is trying to leave the isle while a second car is entering, the two will have to decide how to let one car pass. Cars will not be able to transfer phase two tasks to each other.

To summarize, the overall goal of this project will be to use independently functioning characters (cars) to first map out a maze, then to navigate it by delivering virtual packages to predefined locations (rooms) in the maze. The cars will use swarming technology and techniques to prevent target sharing and collisions. They will also implement path-finding

algorithms to navigate the maze and reach their targets. Different path-finding algorithms and collision avoidance techniques will be tested to find the best combination that builds the most effective car.

II. Resources

A. Languages

For this project we intend to use python to develop the simulation and algorithm framework, and to use python libraries to use the algorithms. When the physical characters are built, they will be controlled by C programming, which will improve control over hardware portions of the characters.

B. Data Sets

1) Maze Set: A set of premade mazes that will used to test and debug algorithms during devlopment.

C. Algorithms

1) A Pathfinding:* A* pathfinding is a best first search algorithm. It takes a node on a weighted graph and gives the shortest possible path from point A to point B. The algorithm will search through the graph and create costs based off of the weights of each of the edges that we assign. For our particular application we can give each edge in the graph a weight of one to start and then if we do some analysis and see that some paths are harder than others we can recalibrate the edge weights for more optimization. This is a very flexible algorithm that will always return the shortest path no matter the graph we pass it.

2) D Lite Pathfinding:* D* Lite operates much like A* pathfinding in that it will find the shortest path from A to B. Not only can this be used for pathfinding, it can be used to map unknown areas on a given map. D* Lite does this through greedy mapping, greedy mapping is when your robot/car is currently in a known cell it will move to the next closest unknown cell. This is very useful in the first phase of our project which would entail the cars moving out and mapping an area/maze before being able to efficiently deliver packages.

3) Physically-embedded Particle Swarm Optimization: This swarm control technique decentralizes processing and decision making to the individual characters in the swarm network. It also ensures collaborative decision making, and synchronized actions, which for the purposes of our project would help prevent collisions and backups in single-lane pathways.

References

[1] Z. Hu and J. Li, *Application and implementation of a* algorithm in picture matching path-finding*, 2010 International Conference on Computer Application and System Modeling (ICCASM 2010) Taiyuan: 2010.

[2] S. Koenig and M. Likhachev, *D*Lite*, In Proceedings of the AAAI Conference of Artificial Intelligence (AAAI) 2002.

[3] M. Couceiro and P. Vargas and R. Rocha and N. Ferreira, *Benchmark of swarm robotics distributed techniques in a search task*, Elsevier BV 2013.