Under Graduate Research Oppurtunity Program

Path Planning for Autonomous Vehicles

Team Members:

G. Akash- AP19110010269

Sai Bhuvanesh- AP19110010216

Hari Prakash- AP19110010210

Sai Praveen – AP19110010215

Mentor:

Muzakkir Hussain Md

Abstract:

In autonomous vehicles, path planning has a great effect on passenger safety, and is at the core of autonomous driving capabilities. One of the challenges of path planning is building an accurate map that responds to changes in the drivable area.The autonomous vehicle consists of perception, decision-making, and control system. Path planning has always been a complex problem, especially in complex environments due to the dynamic environment, safety, smoothness, and real-time requirements, and nonholonomic constraints of the vehicles. To address the problem of travelling in complex environments which consists of lots of obastacles, Dijsktra, RRT, A and A*path planning model is presented in this paper.

Introduction:

The term autonomous car refers to a vehicle that is capable of sensing its environment and operating without human involvement. Simply, a vehicle that gets from one point to another point, without any human interaction.
A human driver isn’t required to take control of the vehicle at any time. In order for autonomous cars to execute software, they need sensors, actuators, algorithms, machine learning systems, and powerful processors. Autonomous cars create and maintain a map of their surroundings with a variety of sensors situated in different parts of the vehicle. Radar sensors detect nearby vehicles and track their positions. Video cameras detect traffic lights, read road signs, track other vehicles, and appearance for

pedestrians. Lidar (light detection and ranging) sensors bounce light pulses off surrounding objects in order to measure distances, identify road edges, and detect lane markings. In order to detect curbs and other vehicles, ultrasonic sensors are built into the wheel.

Path planning is the process through which autonomous vehicles plan their movements and navigate through their surroundings in advanced. Planning an autonomous vehicle's route through a dynamic environment poses a number of challenges:

- Creating an offline coordinate-based map to serve as a foundation for the vehicle's real-world location and intended trajectory.

- Using the map to locate the vehicle's actual position and plan a short route through these points. There can be multiple candidate points for the vehicle's next step. The best option must be chosen based on the positions of impediments detected by the vehicle's sensing modules (like traffic and pedestrians).

- Finding the best heading and acceleration for the vehicle to ensure a safe path possibly also taking comfortability into account (favouring smoother paths with less sudden acceleration). Grid-based offline maps are prevalent, with drivable environment represented as a series of grid cells (called waypoints) with preset placements.

System Architecture of Autonomous Vehicles:

Perception: This stage involves sensing the AVs surrounding through various sensors and also detecting its own position with respect to its environment. In this stage, the AV uses sensors such as RADAR, LIDAR, cameras, real-time kinetics etc. Afterwards, the data from these sensors is sent to the recognition modules that process the information. The AV typically consists of an adaptive detection and recognition framework (ADAF), a control system, LDWS, TSR, and an unknown obstacle recognition module (UOR), as well as a vehicle positioning and localization module (VPL). After that information has been processed, it is fused and passed to the decision and planning stages.

Decision and Planning: In this stage, the AV's motion and behaviour are determined, planned, and controlled with the data gathered during perception. This stage is responsible for making decisions like path planning, action prediction, obstacle avoidance, etc. The decision is based on current and historical information, which includes real-time map information, traffic details and patterns, as well as information supplied by the user. There could be a data logging module that records errors and information for future reference.

Control: The control module receives information from the decision and planning module and performs functions/actions associated to physical control of the AV like steering, braking, accelerating etc.

Control of the automated vehicle:

The automated vehicle can use its knowledge of position and orientation to drive once it understands its location and orientation. The automated vehicle will be able to control speed and heading as it moves around. Its initial purpose is to control its motion along a particular AB line. For this, a Proportional Integral Derivative (PID) is often used to obtain the distance and orientation difference. The goal is to reduce this difference to the smallest amount possible. Using a theoretical model, the automated vehicle outputs a command.

For example, to alter vehicle curvature (angle of trajectory) command may be given to the actuators that drive the vehicle. A PID modifies this command by boosting or dampening the model reaction, which considers previous, present and predicted measurement differences from the desired measurement. In this way, the automated vehicle can be driven toward the AB line with the least amount of overshoot as well as the quickest line acquisition. Tuning the PID is necessary to optimize line acquisition.

Automated vehicle path planning:

When an automated vehicle controls motion along a line, the same principle can be applied to controlling motion from one line to another. Hence, the vehicle's model can create any path with any desired shape. In contrast, Bezier curves can be used to create a path where curvature itself can conform to a curved path, even though the new line could be too small

to be considered a curve. In this way, the vehicle can either follow a preregistered path or choose its own path using decision rules.
The perception modules use raw sensor data to sense parameters such as drivable area and the positions and measured velocities of obstacles.

Offline Map:

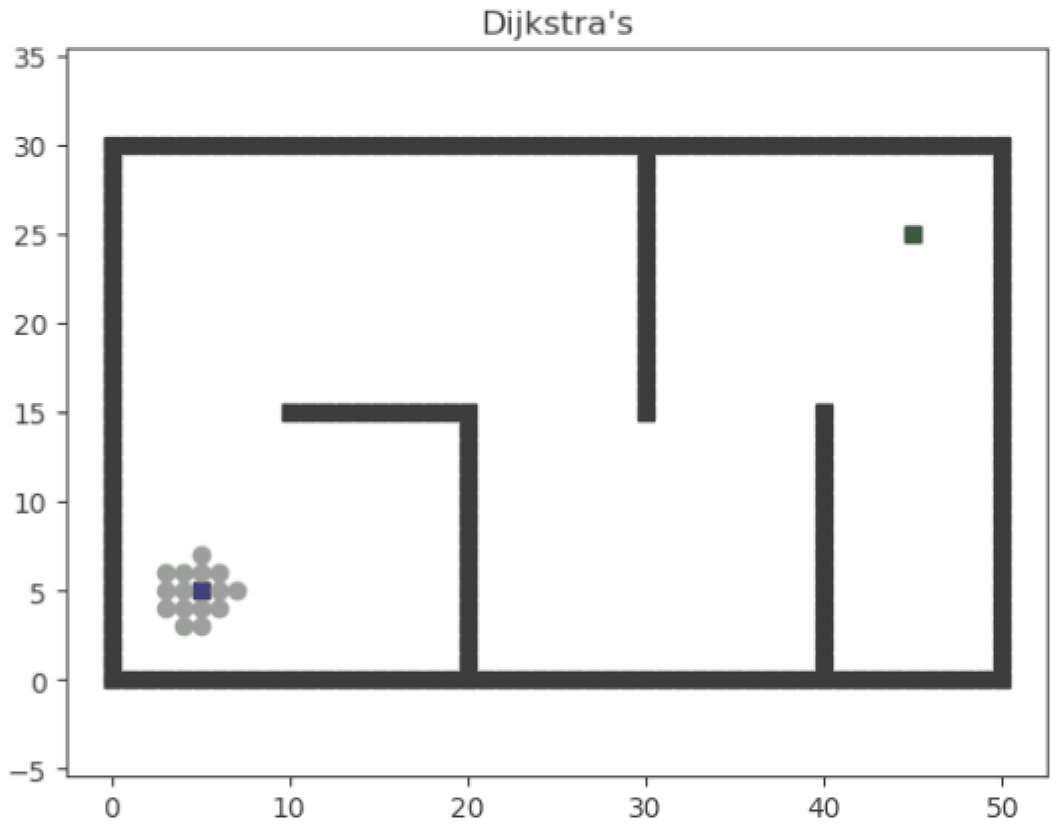
The offline map is a directed graph of GPS coordinates called waypoints. Since the waypoints are connected, they imply a heading. Waypoints are used to indicate the location of a road segment. They should not be used to indicate lanes because the lateral position of the vehicle is independent of the lateral position of the waypoint. Each waypoint is also associated with a maximum driving speed value for this section of the road. Waypoints are loaded from the offline map progressively with a set look-ahead distance starting from the current vehicle position. A point cloud map of the environment is saved locally, and Normal Distributions Transform (NDT) is used to determine the vehicle’s position and velocity within the map from live LiDAR data.

Algorithms used for the Implementation:

Dijkstra Algorithm:

The Dijkstra algorithm is for finding the shortest path between nodes in a graph. This graph data structure is widely used in pathfinding algorithms because it can be applied to cities and roads problems. This Dijkstra algorithm is applied to negative weightless graphs and is used for most problems finding the fastest possible path with the least possible cost. The Dijkstra algorithm is the method of determining the shortest distance by calculating the distance by visiting the neighboring node by looping.

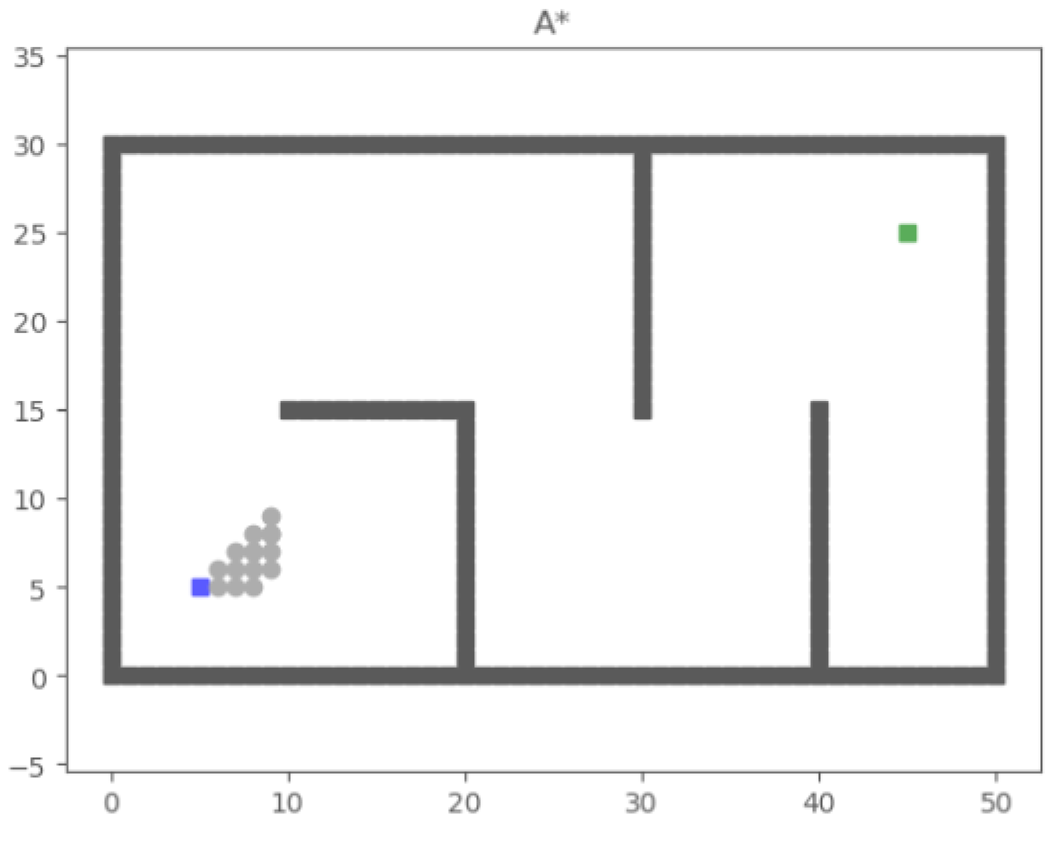
This technique calculates the shortest path between a specified source node in the graph and every other node. It can also be used to find the shortest paths from a single node to a single destination node, with the method halting once the shortest path to the destination node has been identified.Dijkstra's algorithm can be used to find the shortest route between two cities if the nodes of the graph represent cities and the edge path costs represent driving distances between pairs of cities connected by a direct road (for simplicity, ignore red lights, stop signs, toll roads, and other obstructions).Network routing protocols, like IS-IS (Intermediate System to Intermediate System) and Open Shortest Path (OSPF) are a common use of shortest path algorithms . The Dijkstra technique utilizes fully ordered labels that are positive integers or real numbers. It can be expanded to any partially ordered labels as far as the successive labels (which are generated when traversing an edge) are monotonically nondecreasing. The generalised Dijkstra shortest-path algorithm is the name given to this generalisation.

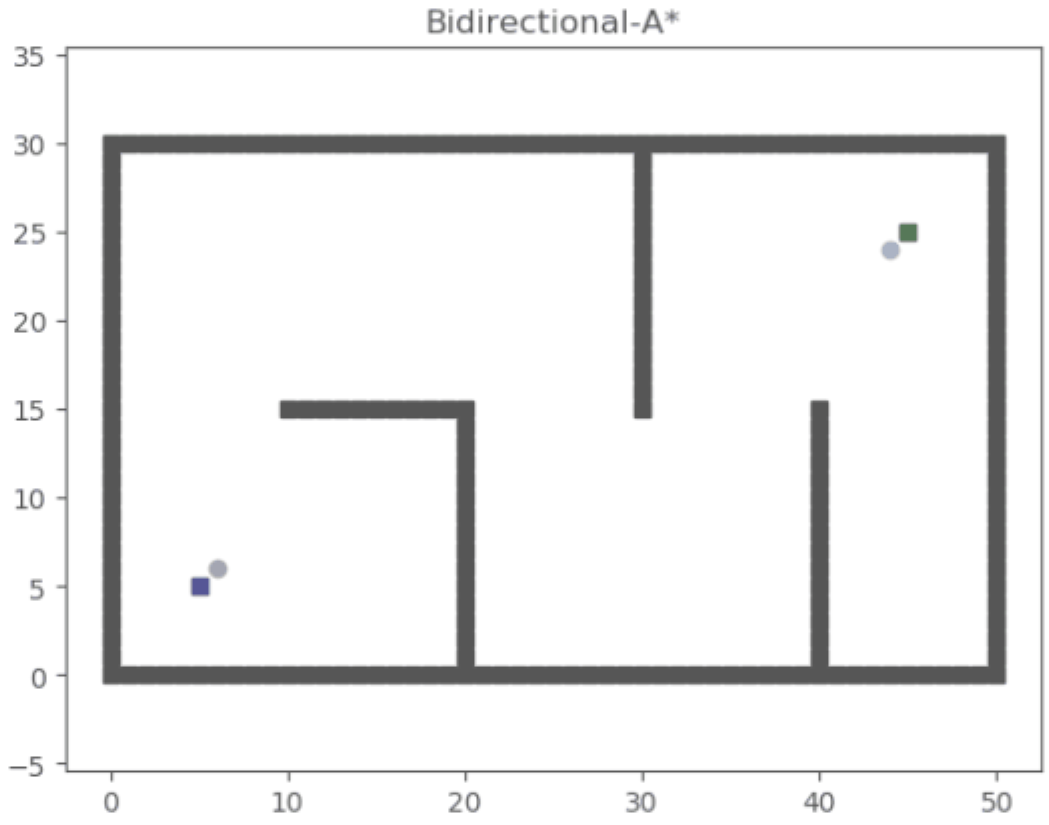


A* Algorithm and A* Variants:

The A* Search algorithm is one of the most widely used path-finding and graph traversal techniques. A* is used in various applications, including maps, to find the shortest path between the initial and final states. Using the A* algorithm, you can plan paths based on heuristic functions.

It calculates the value of the heuristic function at each node on the work area and entails checking too many nearby nodes to get the best solution with no chance of colliding.

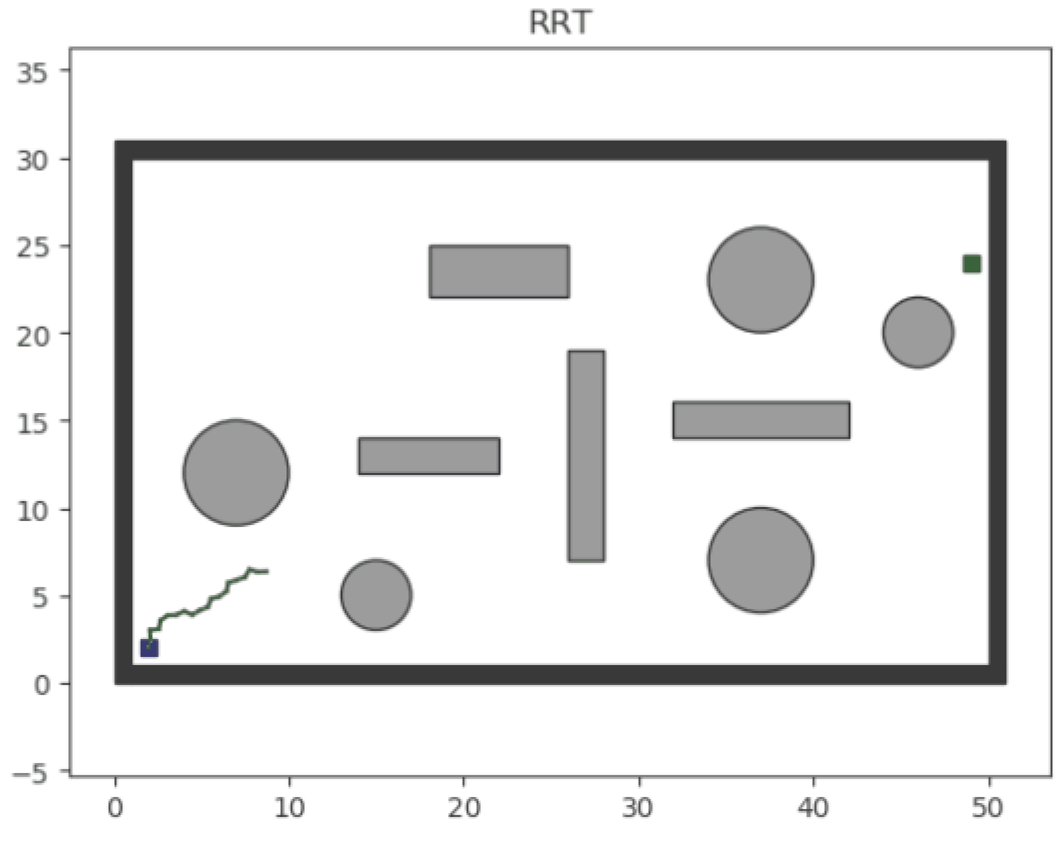




Rapidly-exploring Random Tree (RRT):

A Rapidly-exploring Random Tree (RRT) is a data structure and path planning algorithm that is designed for efficiently searching paths in nonconvex spaces. RRTs are constructed incrementally by expanding the tree to a random sample point in the configuration space while satisfying given constraints. An RRT algorithm can effectively find a feasible path.

Using random samples from the search space, an RRT generates a tree rooted at the beginning configuration. Since each sample is drawn, a link is made between it and the state closest to it in the tree. This results in the adding of the new state to the tree if the link is viable (goes wholly across free space and obeys all constraints). A growth factor frequently limits the length of the connection between the tree and the new state. If the random sample is farther from its nearest state in the tree than this restriction allows, a new state at the maximum distance from the tree along the line to the random sample is utilised instead. The random samples can be thought of as determining the tree's growth direction, while the growth factor regulates its rate. This keeps the RRT's space-filling tendency while also restricting the magnitude of incremental increase.



Conclusion:

With Path planning, Autonomous vehicles will control the motion of the vehicle accordingly. Path planning algorithms are also highly adaptable, which allows them to work on dynamic tasks such as avoiding collisions with obstacles and walls in confined spaces. With Constant Technological growth in Autonomous space we can see more highly advanced algorithms with high efficiency and almost zero chance of collisions. We have used Dijksetra, A* and Its variants for the path planning of Autonomous vehicles. To explore numerous ways of path planning we have also implemented Rapidly Exploring Random tree algorithm.

References:

https://www.sciencedirect.com/topics/engineering/path-planning

https://dzone.com/articles/how-does-path-planning-for-autonomous-vehicles-wor