PLANNING FOR AUTONOMOUS ROBOTS (ENPM661)

PROJECT- 5 REPORT

Path Planning for wheelchair navigation in airport layout with dynamic obstacles using A* and RRT*

Team Members: Sharmitha Ganesan (117518931)
Shelvin Pauly (118426556)

Date: 05/08/2022

ABSTRACT

Path planning is a computational problem of translating an object/robot from one position to a desired position. This technique can be used to solve various real-life problems such as autonomous navigation in a known layout (graph). In this project we have used two major algorithms in path planning and applied it to the problem of an autonomous navigation of a wheelchair in an airport environment,

Index Terms:

Path Planning, Search Based Algorithm, Sampling Based Algorithm, A*, RRT*, nodes, backtrack.

1. INTRODUCTION:

Path planning in an environment with dynamic obstacles has been the goal of Autonomous Robots for decades. There are different search-based algorithms as Depth First Search, Breadth First Search, Dijkstra, A* etc., and sampling-based algorithms as RRT (Rapidly exploring Random Tree), RRT*, Fast Marching Tree etc., where all of them focuses on efficient algorithm with fast convergence and low memory space. Search based algorithms computes path over discrete graphs. This technique can provide efficient solutions at the expense of computation time. Sampling based algorithms are fast but can provide inefficient or unusual paths.

2. BACKGROUND:

One of the motivations for this project is traversing airport terminals in wheelchair autonomously. In this environment, the layout is known but it has lot of obstacles as people, trolleys, seating areas etc., To autonomously plan a path in this circumstance, a highly efficient path-planning algorithm which can provide with an optimal solution in minimal time and takes up minimal computational space required.

2.1 LITERARTURE REVIEW:

Based on the limited research on previous works for path planning in dynamic environments, [1] has proposed goal biased RRT and posed it with RRT* for comparison. [2] has given the idea of getting real-time input of the environment as images, localizing the target, and then performing RRT. [3] has proposed fast RRT* approach by combining goal-biased strategy and constraint sampling.

2.2 OBJECTIVE:

a. Generating an airport-like layout with static and dynamic obstacles.
b. Implementation of one search-based algorithm i.e. A* and one sampling-based algorithm i.e., RRT*.
c. Simulation of wheelchair navigation in gazebo with airport-like world map.

3.IMPLEMENTATION:

Firstly, the workspace is generated.

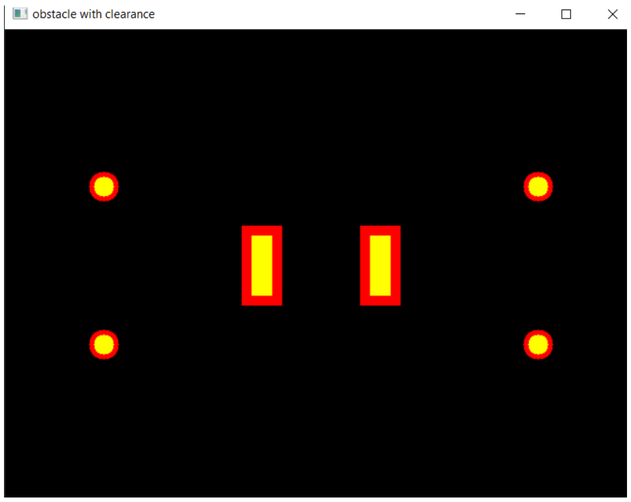

Fig1: Workspace

The map mimics the pillars and seating area of waiting area in an airport. Finally, the implementation of A* and RRT* is performed with static obstacles.

3.1 A* WITH STATIC OBSTACLES:

The layout considered is 640 units in length and 480 units in width. The action set taken is a non-holonomic action set with 6 range of movements as straight_near, straight_far, sharp_left, sharp_right, slight_left, slight_right.

3.1.1 Algorithm:

Input:

Start, Goal, starting orientation, step size.

Method:

While exploring each node in the graph, calculate the cost from the start to the current node (cost to come) and the cost from the current node to the goal node (cost to go).

Add the total and store for each of the node explored. This assures the optimal path with minimal cost to reach the goal node.

Once the goal is found, backtrack the parent node from the child node.

Output: The list of nodes from start to goal.

3.1.2 RESULTS:

The output images of A* implementation with start node (100,100), goal node (305,220), step size 5 and 10 respectively are given below. The green color in the figure represents the explored path and the backtracked path is highlighted in white color.

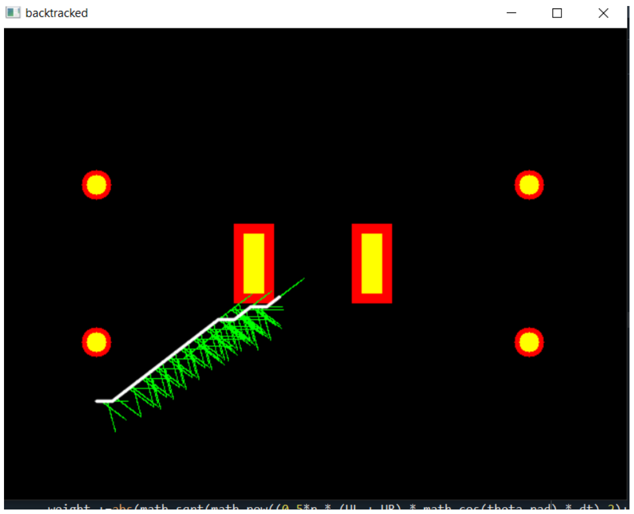

Fig 2: A* with step size 5

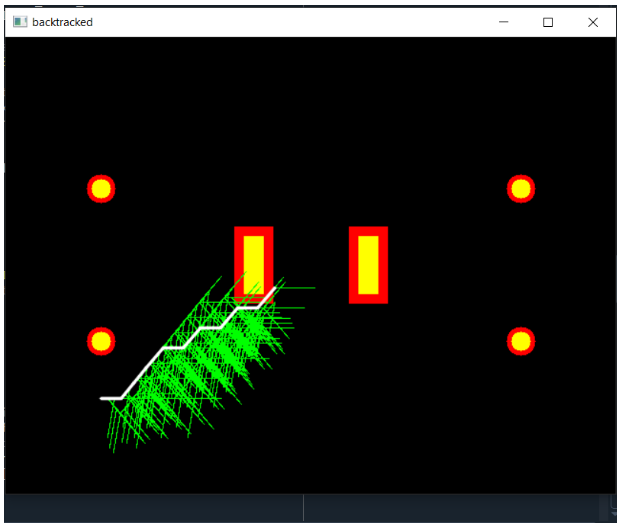

Fig 3: A* with step size 10

3.2 RRT* with static obstacles:

The layout considered is 640 units in length and 480 units in width.

Algorithm:

Input:

Start, Goal, Step size

Method:

Choose a random point in the layout. The next node is in step size 10 with respect to the random point. The nodes are limited with 250 iterations each.

Nearest nodes to the new node are found within neighboring radius of 20 units.

The graph is rewired based on the cost to come of the nearest nodes.

Once the goal is found, backtrack the parent node from the child node.

Output:

The list of nodes from start to goal.

3.2.2 RESULTS:

The output images of RRT* implementation with start node (100,100), goal node (305,220), step size 10 is given below. The green color in the figure represents the visited nodes and the backtracked path is highlighted in purple color.

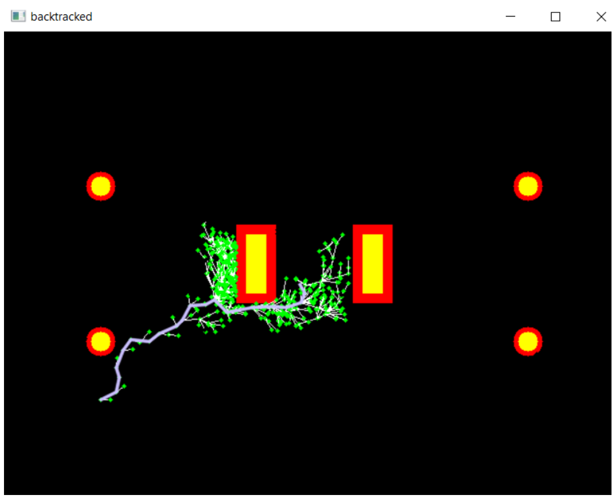

Fig 4: RRT* implementation

3.3 RRT* WITH DYNAMIC OBSTACLES:

The algorithm is same as RRT*. The dynamic obstacles mimicking humans are defined with considerable clearance for the wheelchair to move.

3.3.1 RESULTS

The output images of RRT* implementation with start node (100,100), goal node (305,220), step size 10 is given below. The green color in the figure represents the visited nodes, the dynamic obstacles are the sky-blue dots, and the backtracked path is highlighted in purple color.

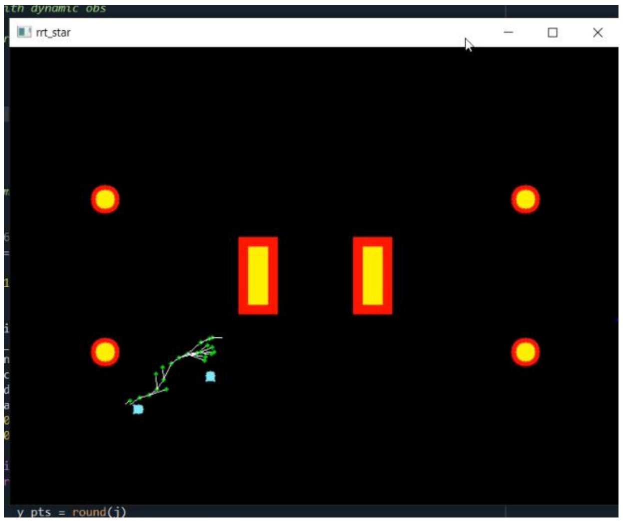

Fig5: Dynamic obstacle avoided by RRT*

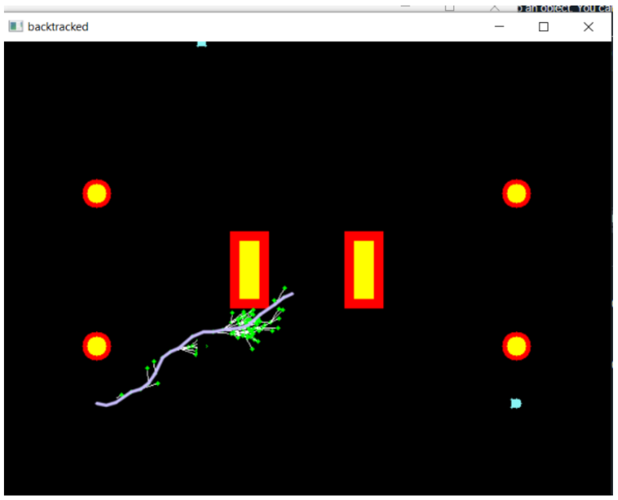

Fig6:Path in RRT* with dynamic obstacles

4.COMPARISON BETWEEN A* and RRT*

As mentioned in the above sections, A* - a search-based algorithm provides with a smoother and more efficient path but takes a lot of time to converge as it explores more nodes over the runtime whereas RRT*, being a sampling-based strategy, converges faster by randomly exploring the map. RRT* is an optimized version of RRT. Here, we specifically used RRT* over RRT to increases the optimality of the path. This means that when the number of nodes approaches to infinity, RRT* delivers the short path. This is only achieved because this algorithm tries to find cheaper alternatives while generating new nodes and tries to rewire the path if it finds a better path. In a real-life scenario as simulated in this project, RRT* can save precious time and provide paths without taking up a lot of the hardware being used.

5. SIMULATION

An airport-like world is created in gazebo. The goal here is to go to the seating area from the entrance of the airport.

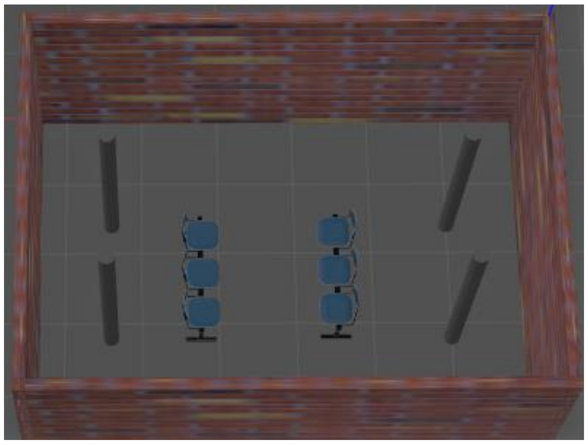

Fig7: Airport-like world in gazebo

It was challenging to incorporate dynamic obstacles in gazebo as it was impossible to mimic the same path of the dynamic points as in the program compilation. So, the path obtained from the RRT* with dynamic obstacle algorithm is used to navigate in the gazebo world without visualizing the dynamic obstacles.

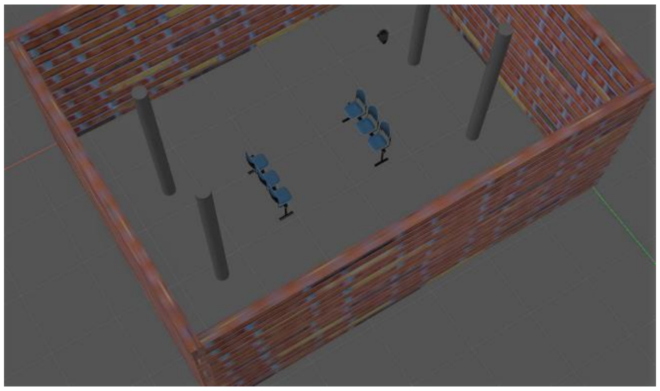

Fig8:Turtlebot launched in the world

The backtracked path is stored and the change in displacement and change in angular displacement between one node to the next node are calculated. The linear velocity is fixed as 0.1 m/s and angular velocity as 2 rad/s. The time to publish the linear and angular velocity are calculated.

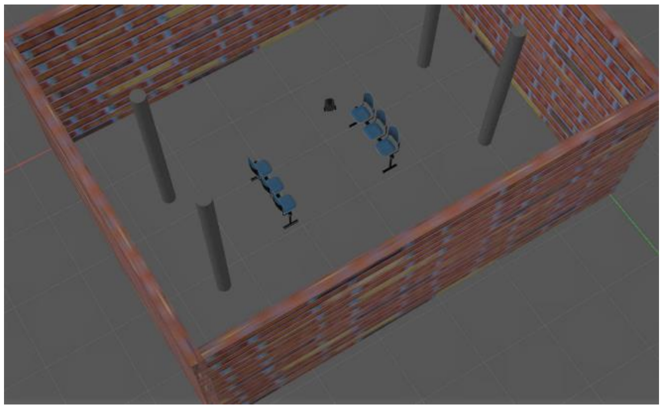

Fig 9 : Reaching the goal

6. CONCLUSION

This project shows the real-life application of using RRT* algorithm for developing a path planner for a wheelchair in an airport setup. To account for inconsistency of the real-life constraints, we simulated the same in environment with static and dynamic obstacles.

There are also many possible avenues that need to be tested before hardware implementation. We still need to account for a busier setup with a lot of moving parts and precise consideration of the hardware capabilities in non-ideal conditions needs to be considered. Furthermore, an interesting topic will be to achieve similar performance as one would with special assistance and getting the same kind of comfort for the person using the wheelchair.

7. REFERENCES

[1] Y. Li, "An RRT-Based Path Planning Strategy in a Dynamic Environment," 2021 7th International Conference on Automation, Robotics and Applications (ICARA), 2021, pp. 1-5, doi: 10.1109/ICARA51699.2021.9376472.

[2] J. Liu, B. Li, T. Li, W. Chi, J. Wang and M. Q. . -H. Meng, "Learning-based Fast Path Planning in Complex Environments," 2021 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2021, pp. 1351-1358, doi: 10.1109/ROBIO54168.2021.9739261.

[3] Qinghua Li, Jihuai Wang and Haiming Li “Fast-RRT *: An Improved Motion Planner for Mobile Robot in Two-Dimensional Space”, http://dx.doi.org/10.1002/tee.23502