# Autonomous Robot Navigation Using SLAM, RRT, and Real-Time Path Planning

Altaaf Ally (2424551)
Hamzah Mia (2430188)
Rayhaan Hanslod (2430979)
Hamdullah Dadabhoy (2441030)

## I. Introduction

Autonomous navigation in robotics involves the integration of various algorithms and techniques to allow a robot to move through an environment without human intervention. This project demonstrates the use of SLAM for mapping, RRT for path planning, and real-time navigation to achieve autonomous robot movement.

## II. Manual Control of the Robot

The first part of our project involved manually controlling the robot to explore the environment. We used a Python script to publish velocity commands to the robot based on keyboard inputs.

## III. Generating the Map using RViz and GMapping

To generate a map of the environment, we utilized the GMapping package in ROS, which is an implementation of a SLAM algorithm. We visualized and created this map using RViz which is illustrated in Figure 1.

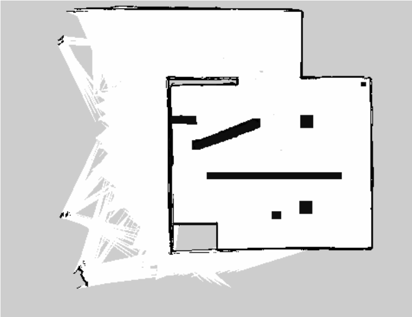

Fig. 1. Mapping process using RViz and GMapping.

To move the robot around for mapping, we used a python script file, allowing us to manually control the robot's movement using keyboard inputs.

## IV. Path Planning Using RRT

Once the map was generated, we used the Rapidly-exploring Random Trees (RRT) algorithm for path planning. The goal was to find a collision-free path from the robot's current position to a specified goal.

Our implementation involved loading the saved map image, converting it into a binary format, and identifying obstacle-free spaces. We then generated random coordinates within the free space and applied the RRT algorithm to find the shortest path.

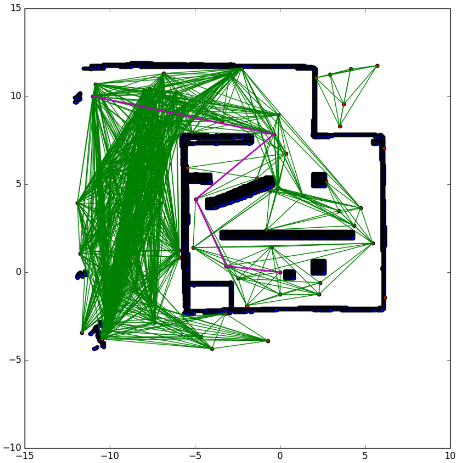

Fig. 2. Path generated by the RRT algorithm. The figure shows all possible paths explored by the RRT algorithm, with the best path highlighted.

## V. Real-Time Navigation

The final step was to navigate the robot in real-time along the path generated by the RRT algorithm. The 'Turtlebot-Mover' class handled the movement of the robot, subscribing to odometry data and publishing velocity commands.

We used a PID controller to ensure smooth and accurate movement, adjusting the robot's orientation and velocity to reach each goal point. The robot continuously updated its path based on real-time sensor data. The user inputs the (x, y) coordinate as the goal, and the robot navigates towards it. The robot avoids obstacles and dynamically adjusts its path to ensure it reaches the specified coordinates efficiently.

## VI. Conclusion

This project successfully demonstrated the integration of manual control, SLAM using RViz and GMapping, RRT path planning, and real-time navigation to achieve autonomous robot movement. The robot was able to navigate a mapped environment, avoiding obstacles and reaching specified goals efficiently.