Team 10: Carlos Jones, Aiden Deady, Manas Pise

Professor Rosendo

RBE3002 D24

May 1, 2024

RBE 3002 Final Report

Introduction:

The goal of the final project was to use the skills learned from the labs over the course of the term to navigate and map an unfamiliar maze using a Turtlebot3 Burger. Specific objectives included localizing the robot within the maze, exploring the environment while avoiding obstacles, identifying points of interest for map expansion, and efficiently navigating back to the starting point. Algorithms were developed to recognize when the mapping process was complete and to plan optimal paths to explore unexplored areas effectively. This project served to demonstrate knowledge on robotic exploration and navigation. The final product consisted of launch files that allowed the robot to complete these tasks.

Methodology:

Phase 1: Mapping The Environment Using GMapping

The project begins by implementing GMapping in ROS, connecting it with our TurtleBots' Lidar systems. This setup enables real-time creation of an occupancy map reflecting the TurtleBot's perspective, essential for navigating the unknown maze. To account for the robots' actuation error, we expand the configuration space (CSpace) around walls on the map.

This process involves enhancing the identified occupied spaces to ensure safe navigation clearance for the robot.

Once the map is updated, the next task is to identify frontiers, which are the boundaries between explored and unexplored areas. These frontiers are detected by examining changes in the occupancy values across the grid. We then calculate the centroid of each frontier, grouping nearby edges to simplify the process. This approach helps in managing load and improves the efficiency of the mapping process. Frontiers that are too small are ignored, as they are likely errors in Lidar readings or mapping, and this allows the robot to fully navigate the map in an efficient timeframe.

With the frontiers identified, we then determine the nearest centroid using Euclidean distance from the robot's current position. The robot then uses this information to navigate towards unexplored areas. The navigation is facilitated by the plan_path() service (created in Lab 3), which utilizes an A* algorithm modified to minimize turns and avoid close proximity to walls, enhancing the robot’s safety and efficiency in pathfinding.

Phase 2: Navigating Back to the Starting Position

After the map is fully explored, the robot's new task is to return to its starting position. This is achieved by utilizing the A* algorithm to chart the most direct and safe return path. The initial pose of the robot, recorded at the beginning, serves as the endpoint for this phase. Since the map is now fully known, navigating back involves fewer risks of collision, allowing the robot to follow the planned path without needing constant adjustments.

Phase 3: Navigating to a Specific Goal

The final phase requires implementing the AMCL package within ROS for precise localization, which is crucial as the robot navigates to a specific goal set by the SA's in the lab. Once the goal position is identified on Rviz, the robot begins localizing its position with higher accuracy. This process involves the AMCL node tracking the array of possible robot poses, converging on the most likely actual position.

During localization, the robot may perform varied-speed rotations to refine its position estimate, with the process continuing until the robot has a high confidence in its location (as indicated by a low covariance value). Once localized, the robot employs the previously developed path planning algorithms to navigate to the goal. This phase tests the robot's ability to apply its mapping and navigation capabilities to reach a designated point efficiently.

By following these structured phases, the project aims to demonstrate practical applications of robotic navigation and mapping in unknown environments.

Results:

Our team was able to successfully complete the first Phase of the lab, despite our robot colliding with the walls of the course a few times throughout its frontier exploration. Our robot performed better during practice run, however during the day of the demo, our team-specific robot (Rosalina), would not work, and our team was forced to use another team's robot. As a result, our previously tested and defined parameters did not work as well on the new robot. We believe that this led to some of the collisions our robot experienced during the demo. In spite of this, our robot successfully completed Phase 1.

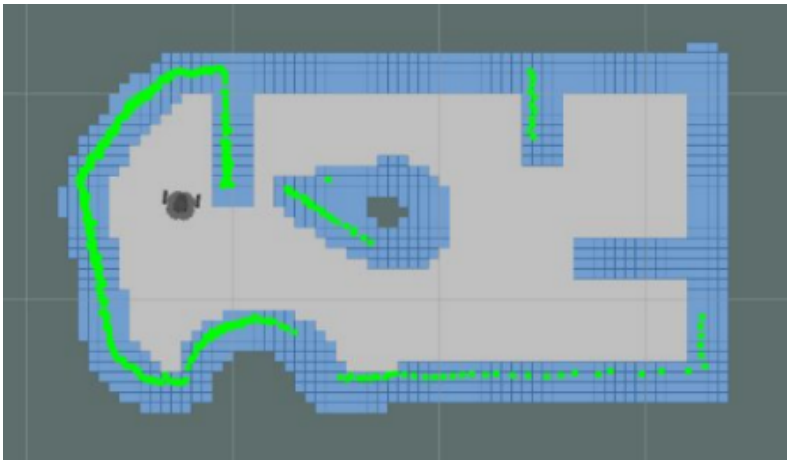

Figure 1: Simulation of the robot during Phase 1, searching for frontiers.

Figure 1 shows an image of the simulation our team underwent to test the first Phase of the lab. When the robot first enters the space, it is able to detect obstacles, as seen in green. Using this, the robot was able to calculate new frontiers needed to explore and create a CSpace from the detected obstacles. From there, it used our A* algorithm, created in Lab 3 (and re-designed in Lab 4), to find the most optimal to the highest priority frontier space. It would continue this to maneuver through the map until there were no more unexplored areas of the map.

For Phase 2, our robot was able to successfully navigate back to its initial starting position using the most efficient path. However, our CSpace was not quite large and our robot did collide with the wall on its return to the starting position. In spite of this, we still successfully completed Phase 2.

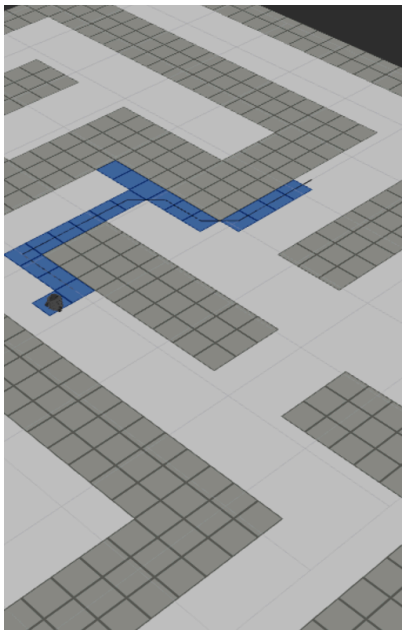

Figure 2: Robot generates C-space to avoid walls during Phase 2 (image in simulation).

For Phase 3, our robot was unable to complete the entirety of the phase. In both simulation and with the physical robot, our robot successfully localized itself using AMCL. It also successfully used the A* algorithm to plan the most efficient path to whatever point was determined with the 2D Nav Goal in Rviz. However, in both the simulation and in real life, our robot encountered an accumulation of small errors that prevented it from successfully traveling to the chosen point.

Discussion:

This lab aimed to build on our previous learning by programming a Turtlebot3 to autonomously navigate and map unknown areas. The robot needed to continuously understand its location within the arena, map the environment, and return to its starting point. Additionally,

we were tasked with setting a 2D navigation goal in Rviz for the robot, which it then had to physically reach in the real world.

One of the primary challenges encountered during Phase 1 was the robot's tendency to collide with the maze's corners while navigating. This often resulted in the robot losing its bearings regarding its precise position and orientation on the map. The problem was particularly pronounced when navigating around corners, where the robot would frequently cut the corner too closely and collide with the wall. In response, the team increased the configuration space (C-space) to provide a larger buffer between the robot and the walls. While this adjustment helped reduce the frequency of collisions by expanding the clearance area, it inadvertently caused the C-space to cover the entire available path in narrower sections of the maze, restricting the robot's movement. To address the limitations of the enlarged C-space, we implemented a secondary strategy involving a modification to the pathfinding algorithm. We added a heuristic to the A* algorithm that prioritized grid cells located further from the walls. This approach effectively reduced the risk of collisions by encouraging a safer distance from potential obstacles. However, it also introduced excessive maneuvering in the robot's path, leading to an increased number of turns, which can be suboptimal for path efficiency. To optimize the robot's trajectory, we modified the path to focus on driving along collinear segments where possible. This strategy aimed to reduce the cumulative error associated with frequent directional changes. Nonetheless, it introduced a significant issue with drift, especially over longer straight paths. Small initial errors in heading became amplified over distance, leading to noticeable differences between the goal and reality.

In an attempt to mitigate the drift and streamline the navigation process, we further optimized the path by removing alternate poses. This reduction in path complexity decreased the

number of necessary turns, enhancing overall navigation smoothness. However, this simplification sometimes caused the robot to clip corners during turns. This issue showed the balance between path simplicity and the precision of navigational commands necessary to maintain an accurate trajectory within the maze.

Phase 2 of our project focused on enabling the robot to autonomously return to its starting position after successfully navigating the maze. This capability was tested extensively in both simulated environments and real-world conditions, where the robot demonstrated perfect accuracy in retracing its path back to the origin. The success in these trials can be attributed to the pathfinding algorithms and the precision of the robot's localization and mapping systems, which consistently maintained an updated and accurate map of the environment. However, an unforeseen challenge arose shortly before our final demonstration. The original robot encountered issues that necessitated its replacement with a new Turtlebot. This last-minute switch introduced significant variability in the system's performance due to differences in the variable calibration and sensor sensitivity between the two robots. While we attempted to make quick changes, this last minute issue could not be fixed, and the discrepancies in variable calibration between two different robots became clear. Without being able to use the Turtlebot3 that we had calibrated throughout the entirety of the term, we noticed more performance errors in our demonstration. This showed the importance of parameter tuning in robotic applications, which can be critical when unforeseen issues arise.

In Phase 3 of our project, the robot was tasked with navigating from a predetermined starting corner of the maze to a specific destination, termed the "evil location," as decided by Professor or the SA. This phase was designed to evaluate the robot's ability to execute a complete navigation run under time constraints, with the added pressure of achieving the goal

within a limited duration of five minutes. Initially, the robot's performance in simulations and controlled environments was promising, demonstrating accurate localization and efficient path planning from the start position. However, significant challenges arose when translating these results to real-world conditions. As the robot moved through the maze, discrepancies between its calculated position and actual location became increasingly problematic. Despite the A* algorithm effectively identifying the optimal route, the robot struggled with accurate localization, which was critical for following the planned trajectory. As a result, the team was unable to complete Phase 3 successfully. The repeated localization errors prevented the robot from reaching the designated "evil location" within the allotted time, thereby failing to meet the criteria for phase completion and sign-off. This experience highlighted critical limitations in our current robotic system, particularly in adapting to dynamic and complex environments.

Through these modifications, we gained insights into the complexities of autonomous navigation in constrained environments and the trade-offs involved in different robotic path planning strategies.

Conclusion:

In the final project for this course, our team had the chance to put into practice everything we've learned from lectures and prior labs. This project focused on path planning, obstacle avoidance, and localization, giving us a look at how real-world conditions can affect outcomes in ways that simulations might not show. Although we didn't manage to complete Phase 3, the challenges we faced and overcame deepened our understanding of robotic systems, in particular the Turtlebot3 and implementing it using ROS. Along the way, we identified several potential

improvements that could be implemented if we were to do the project again. This experience has prepared us to tackle future challenges in the field of robotics with greater expertise.

References:

Works Cited

"Robotise." *Manual*, emanual.robotis.com/docs/en/platform/turtlebot3/slam/. Accessed 1 May 2024.

"ROS AMCL Wiki." *Ros.Org*, wiki.ros.org/amcl. Accessed 1 May 2024.

"Ros Frontier_Exploration Wiki." *Ros.Org*, wiki.ros.org/frontier_exploration. Accessed 1 May 2024.

"ROS GMapping Wiki." *Ros.Org*, wiki.ros.org/gmapping. Accessed 1 May 2024.

"Turtlebot Navigation Stack Wiki." Ros.Org, wiki.ros.org/navigation/Tutorials/RobotSetup#Costmap_Configuration_.28local_costmap.29_.26_.28global_costmap.29. Accessed 1 May 2024.