Tour Guide Robot

Natalie Mercer, Justin Santos, and Quyen Tran
Department of Computer Science, The University of Oklahoma

Abstract

Tourism is an important market for both consumers and providers by providing a flow of income to the providers while informing and entertaining the consumers. An interesting extension of this market is the concept of automated tours given by a non-human entity such as a robot. In this project, the usability of this application will be explored by simulating a tour guide Turtlebot using ROS, Gazebo, and RViz. A hybrid deliberative-reactive paradigm based on AuRA (Autonomous Robot Architecture) was used to design this robot. In order to support path planning, navigation, and localization, the ROS navigation package will be implemented within the robot's software to facilitate point-to-point navigation.

Introduction

The primary goal of this project is to judge the efficiency and practicality of a tour guide robot via simulation. The robot used in the simulation is a Turtlebot2 using a Kinect RGB-D camera as its primary sensor. The simulated environment is modelled off of a section of the first floor of Devon Energy Hall at the University of Oklahoma, including the atrium and room 115. The environment also includes several obstacles, doorways, and windows that the Turtlebot must traverse and accommodate for as it is giving a tour. Various interactions will be implemented to better simulate a tour. For example opened and closed doors will be explicitly represented in the robot's world-knowledge, and human-robot interaction could be simulated by allowing a human to open a door for the robot if needed.

Approach

The approach for solving the problem of creating a tour guide robot was heavily impacted by the previous work done in this course. However, the results of these projects were suboptimal as the robot frequently got lost in the simulated world. This tour guide robot implementation improved upon the shortcomings of the previous projects by using the ROS navigation package. This package uses an accurate, three-dimensional voxel mapping representation. The navigation package splits the task of navigation into two parts: global planning and local planning. For global planning, the package uses the A* pathfinding algorithm to find the safest and most efficient path. Local planning facilitates online obstacle avoidance by using the dynamic window approach strategy. It also concurrently navigates the world while localizing by using adaptive Monte Carlo localization, which fulfills a feature that was completely missing from the previous projects.

To utilize the ROS navigation package we needed to have a pre-existing world model. This map was based on a section of the Devon Energy Hall and can be seen in Figure A. We created a map using the gmapping package from RViz. Because we had originally created the map based on this area, we had added some realistic features such as windows and tables. We ran into an issue where the windows were shown as open/unoccupied space. This is because the Turtlebot2's laser scanner was not able to distinguish between open space and glass. To get around this problem, we went in and manually edited the map to classify the areas with windows as occupied space. The map created by RViz can be seen below in Figure B.

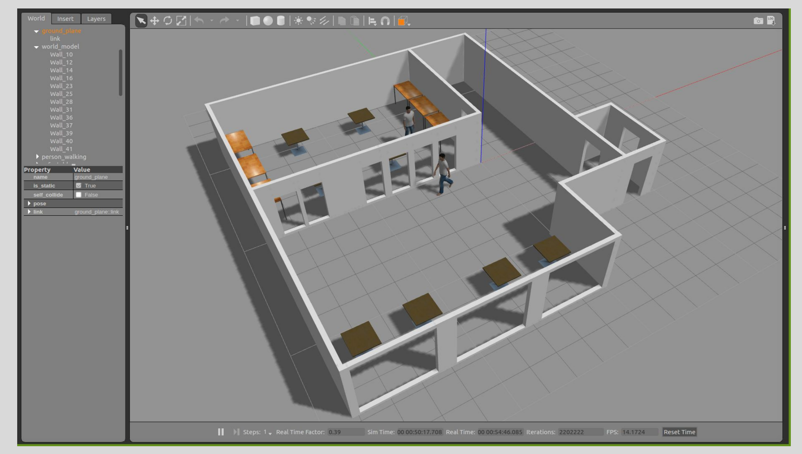

Figure A

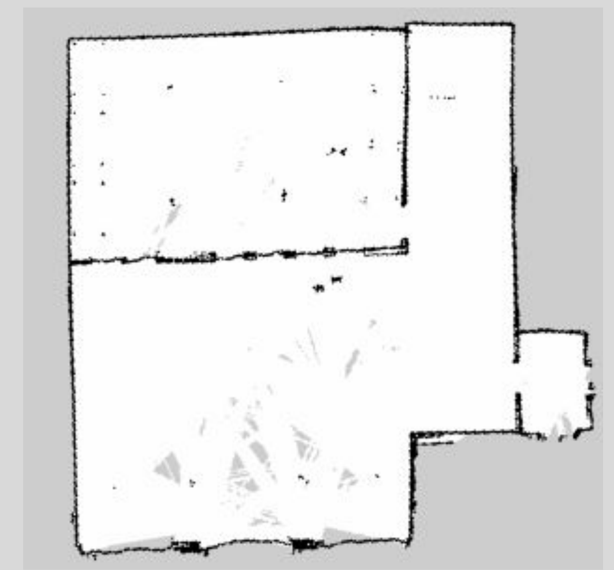

Figure B

We had six tour destinations, two being in the atrium, one being in the entryway, one being in the hallway, and two being in the smaller room. When the robot starts, it creates a path to reach all of these destinations. To add to our project, we implemented a feature to simulate door opening. A physical robot would not be able to open the door for itself, so it would need the help of human interaction to do this. We simulate this by having the robot request access to a room before entering. The user can choose to accept or decline from the command line. If the user chooses to not help the robot, the unreachable tour destination nodes will be removed from the path and a new path will be created to reach the remaining tour destinations from the current position. A visual of the tour destination nodes, doorway nodes, and connecting edges are shown in Figure C. Figure D shows what the graph will look like if the user were to decline to open a door and nodes would be pruned.

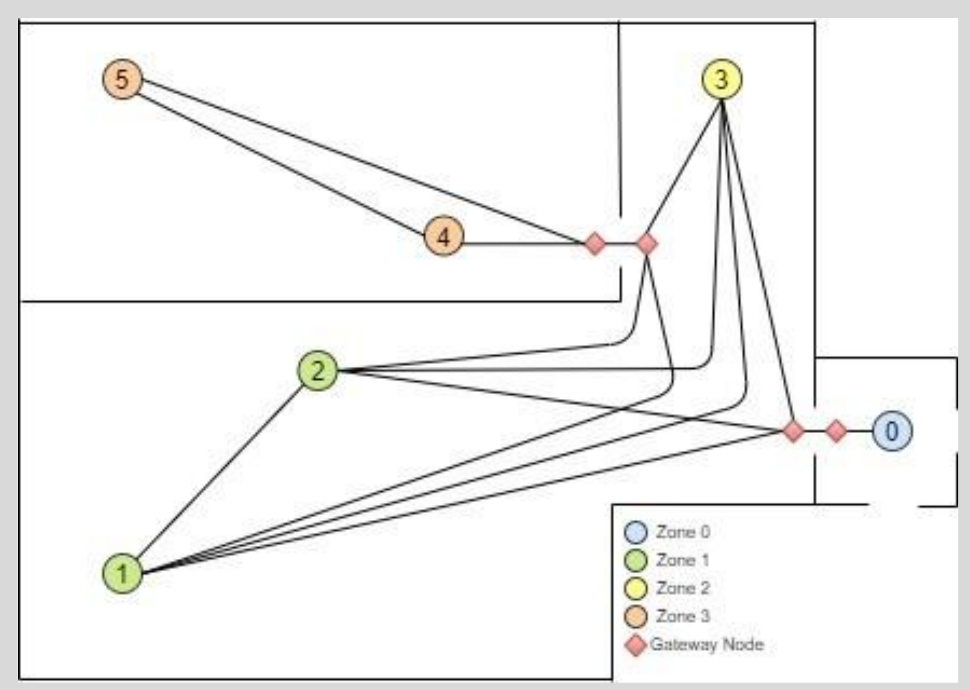

Figure C

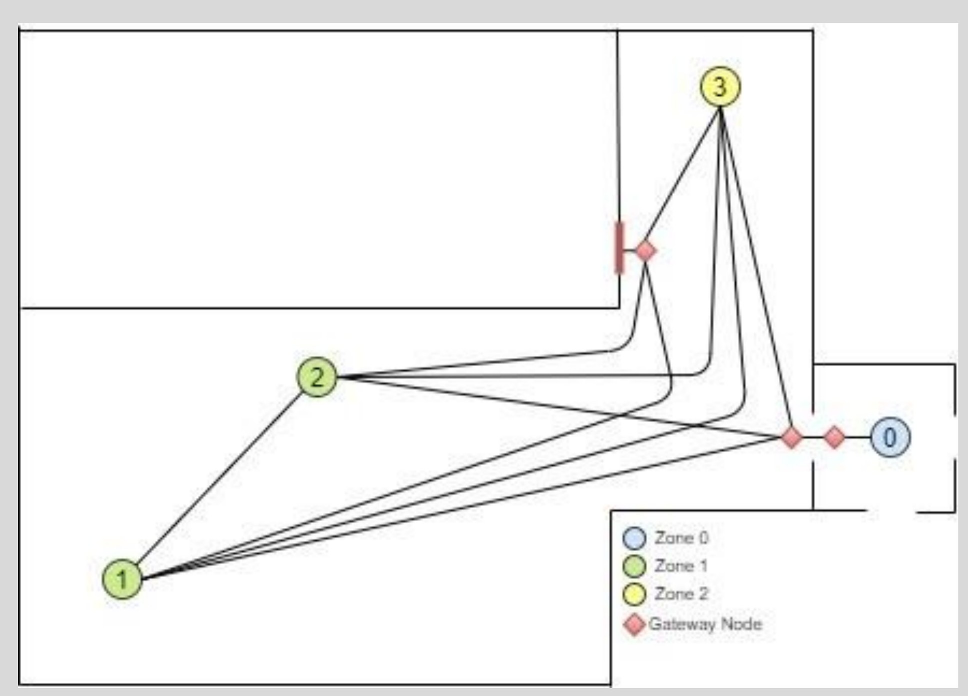

Figure D

When a position is reached on the tour, the robot stops and gives some information about the destination out to the user via the command line. There will then be a pause until the robot moves on to the next available position.

To detect the robot's position and orientation as it moves through the world, adaptive Monte Carlo localization, or particle filter localization, is used. This puts a particle sample, called a particle swarm, around the robot to estimate the position. This can be seen in Figure E. If the estimation becomes too far off from what the robot is detecting the world to be, then a larger particle sample is released to determine the location. By doing this, this localization is efficient and accurate.

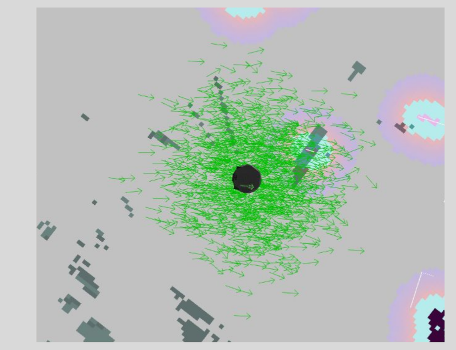

Figure E

Results

We ended our project with a well simulated tour guide robot. Our robot is able to navigate to the different tour destinations and change the route when needed.

We were not able to simulate one aspect of our project that we originally had planned on doing. This is the simulation of a second robot running in our world to allow our tour guide robot to experience dynamic obstacle avoidance. This is an extra feature that could be added in the future.

Conclusion

We were able to complete our goal of simulating a tour guide done with a robot. We found the efficiency of this project to be much better than former projects. This is because of the implementation of the navigation package that allowed for better path planning and localization. The inclusion of door simulation means that this project could be easily be converted into a physical tour guide robot, where it would certainly run into a problem with closed doors in its path.

Future Work

This project is one that can be built upon by adding more features to better simulate a real life tour guide robot, such as the dynamic obstacle avoidance. This could also be transformed in to a physical tour guide robot with the proper equipment and modifications.

References

[1] Arkin, R.C.; Balch, T. (1997). "AuRA: principles and practice in review". Journal of Experimental and Theoretical Artificial Intelligence. 9 (2–3): 175–189.

[2] E. Marder-Eppstein, “navigation,” *ros.org*, 14-Sep-2020. [Online]. Available: http://wiki.ros.org/navigation. [Accessed: 01-Dec-2020].

Appendices

https://github.com/stinsan/tour-guide-robot