Project: Home Service Robot

Introduction

As the technology behind the brains of mobile robots improves, the potential for service robots greatly expands. With better sensors and algorithms, service robots can navigate from point to point with ease, adapting to obstacles such as falling objects or uneven terrain. Such robots can and are being applied in revolutionary solutions from healthcare to manufacturing. At a simpler level, this project aimed to simulate a simple home service robot performing an object retrieval task.

The robot simulates the behavior of traveling to a pickup spot, retrieving the object, and delivering it to a target location. The Turtlebot II robot and a red square visual marker are used for the robot model and object, respectively. The world is simulated in Gazebo to resemble a single floor building with distinct rooms.

Approach

The home service robot needs to localize itself in the environment, receive and identify the pickup and dropoff locations, and navigate to the pickup location and then to the dropoff location. Each task can be implemented by using existing ROS packages. The marker simulating the object needs to appear at the pickup location, disappear from the pickup location when the robot reaches it, and appear at the dropoff location when the robot reaches. These behaviors can be accomplished by subscribing to the robot's odometry topic and publishing markers based on the robot's position.

Implementation

The following ROS packages were used to build this project.

- gmapping: implements SLAM and builds a map of the Gazebo world.
- turtlebot_teleop: allows for user control of the Turtlebot II robot using keyboard input
- turtlebot_rviz_launchers: launches RViz to visualize the robot moving from pickup to the delivery location.
- turtlebot_gazebo: defines and launches the Turtlebot II robot in the custom Gazebo World.

Two additional packages were defined to define the marker positioning and the robot movement.

- pick_objects: defines the pick_objects node to navigate the robot from an initial point to the pickup location and then to the delivery location.
- add_markers: defines two nodes
  - display_markers: publishes marker at pickup location and, after deleting pickup marker, publishes marker at dropoff location
  - add_markers: Creates a subscriber to the robot's odometry topic and a publisher to publish the marker based on the robots position.

Catkin Workspace Structure

src

- **slam_gmapping:** gmapping package
- **turtlebot:** contains turtlebot_teleop package
- **turtlebot_interactions:** contains turtlebot_rviz_launchers package and modified rviz config file, marker.rviz, to display markers when rviz launches
- **turtlebot_simulator:** contains modified turtlebot world launch file to launch turtlebot in the custom gazebo world
- **map:** contains the SLAM generated map.pgm and map.yaml
- scripts:
    - **test_slam.sh**: runs SLAM manually using teleop package
    - **test_navigation.sh**: tests robot's AMCL navigation ability given a navigation goal
    - **pick_objects.sh**: tests pick_objects.cpp in simulated world
    - **add_marker.sh**: tests display_markers.cpp in world
    - **home_service.sh**: runs pick_objects and add_markers nodes in the simulated world
- rvizConfig
- **pick_objects:** contains pick_objects.cpp
- **add_markers:** contains display_markers.cpp and add_markers.cpp