Robotics Final Project Report

Team Members

Aloha Churchill (alch2037@colorado.edu)
Omar Dajani (omda2265@colorado.edu)
Spencer Jackson (spja5809@colorado.edu)

Deliverables

Mapping

We implemented manual mapping using a modified lidar sensor (that was positioned higher on the robot to avoid noise with floor detection). The robot could be manually driven around in this mode and the lidar sensor created a probability map of the obstacle locations.

Localization

Our initial approach to localization was simply using the GPS coordinates which was successfully implemented. Then, we added functionality for localization using odometry. We found the drift in odometry to be unusable for the navigation of our robot and so we used the GPS in the final project. Additionally, we implemented a trilateration scheme. This approach was based initially on EKF-SLAM but ended up being more similar to simply using trilateration. We added multiple cones which acted as set landmarks above the world. We also added two additional lidar sensors that were also positioned above the walls of the world in the same plane as the landmark objects. The idea was that these lidar sensors would detect the predefined landmarks and the distance away from those landmarks (in the range of the lidar). Based on these measurements, we were able to obtain an estimate for the current location. This was done by first creating an artificial noisy GPS value and then using this as an initial guess. Instead of using hardcoded geometrical trilateration formulas, we opted for an optimization algorithm that, given an initial noisy guess, used the detected distances to the landmarks to obtain a position. We incorporated this localization into visualizing our mapping but still used the GPS positions for actually navigating the robot when performing differential drive inverse kinematics.

Navigation

We first used the A* algorithm to navigate the robot. At the end of the project, RRT* was implemented for a smoother course.

Computer Vision

We used the OpenCV mask and found contour functions to detect color blobs. During the mapping phase, blob contours of the yellow goal object color were added to the checkpoint destinations where the robot was stopped (along with user-defined checkpoints). First, the image from the Webots camera was translated into an array that could be used by the OpenCV library. Then, the above functions were used to detect yellow goal objects. MS Paint was used to create the mask values for the yellow color of the goal objects.

Manipulation

Our approach for manipulating the arm was two-pronged. First, we used IKPy to implement rudimentary inverse kinematics for the robot. We struggled to get this library to work for actually grabbing an object and instead used IK as an option for the user to get the arm to go to the default position that did not block the Lidar.

Program Execution

See the GitHub readme for an explanation of the intended flow of execution.

Notes

Individual Contributions

Task	**Completed by**
IKpy manipulation, manipulation controller, navigation, trilateration, main controller execution, debugging	Aloha
RRT* algorithm and path planning, manipulation hardcoding, debugging	Spencer
Object detection and recognition, waypoint hardcoding, debugging	Omar

Goal Objects

This system should be able to collect all goal objects. Ultimately, this project developed a semi-autonomous robot. The user is able to arrest control from the autonomous components at any point in the execution, and objects can be collected using a combined approach of the autonomy and user control of the drive system and manipulation of the robot arm.

Additional Documentation

GitHub

The README.md file in the project description gives an outline of the code structure as well as the resources we constructed in this project. All program and documentation files can be found in the GitHub repository.

https://github.com/omardajani-cu/CSCI3302_Final_Project

Video Link

https://www.youtube.com/watch?v=D_7RKOg03ls