Project II Final Report

Ole Kjørholt & Nadira Mahamane – UCA Department of Physics & Astronomy

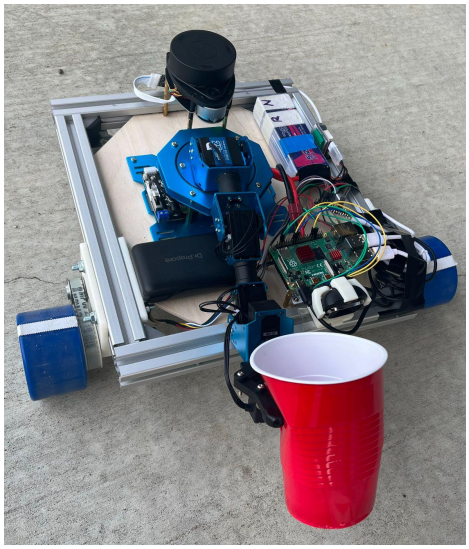
May 5, 2022

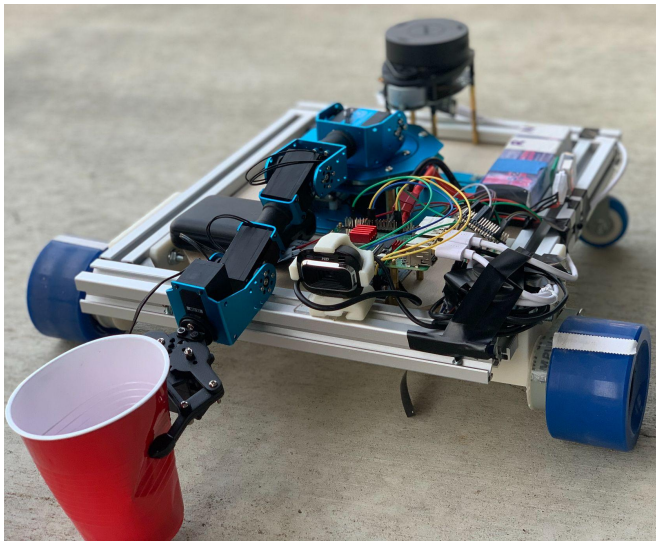
Introduction

The main purpose of the semester activities is to design and build a differential drive robot using navigation and environment awareness. The main goals of this second project was to incorporate a LIDAR, a camera and a robot arm in order to autonomously create and navigate through a mapped space. In real life this could be applied to many sectors such as cave or exoplanet exploration.

Design

Since our last project report, most of the robot design has remained unchanged, however there are a few changes. First of all we decided to move from using the PiCamera to using a simple USB web camera due to a potentially broken camera and considering time restraint. In addition to that, we mounted the robot arm in the middle of our robot, where it will be able to be manipulated in the forward, left and right direction as needed. The last thing we have done is make all of our different parts securely and permanently connected by making sure that our wires are taped down so that our robot will be able to move freely without risking any damage to any of our parts. Below, in Figure 1, you can see the final setup of our robot:





Analysis:

Mapping

As mentioned earlier, our first goal for this project was to be able to create a sensible map of a given delimited space where the robot is positioned. In order to do this, one of the first steps is to create a simulated mobile robot using the Universal Robot Description Format (URDF), which is the standard ROS format for robot modeling. The URDF file is an XML-file which describes the physical attributes of the robot. It contains the complete physical description of the robot such as the wheels, different links, and sensors.

After getting the URDF file done, our next step was to set up our odometry. Odometry is about using data from different sensors to estimate the change in a robot's velocity, position, and orientation over time relative to a given point. The odometry is obtained from the wheel encoders reading through an arduino nano board and a LIDAR. To publish the odometry we used we wrote a python script file containing 3 fundamentals functions. The first two are the velsub_Callback and the actvel_callback timer functions publish at a frequency of 100Hz. The velsub_Callback is responsible for getting the target velocities via the teleop keyboard control input by the user. The actvel_callback read the count per seconds of each wheel via the encoder to determine the actual robot velocities. The error between the actual and target speed are used for the proportional control of the robot. Lastly, the odom_callback function is used toset the position of the robot, increment the change and publish the odometry in the odom frame using the package tf2 transformation. Tf2 determines the relation between the robot base and its different sensors which are the lidar and the camera in our case. The coordinate frame in ROS that is most commonly used for odometry is the odom frame. The odom frame is essentially the starting point where the robot first starts moving from.

As written in the official documentation, the two most common packages used for localization are the nav2_amcl package and the slam_toolbox. Both these packages publish the map → odom coordinate transformation that is necessary for a robot to localize on a map. In our case, we used the slam_toolbox.

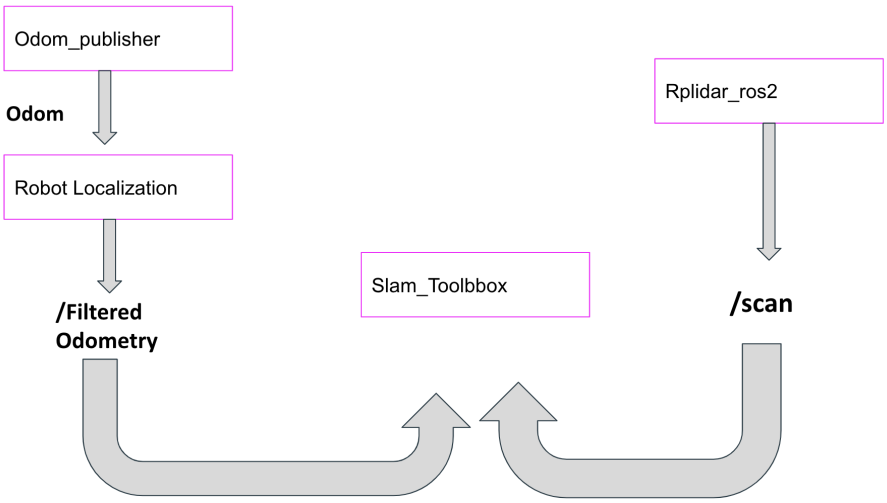


Figure 2. Mapping flowchart

Object Detection

The camera was tested using the OpenCV tutorial opencv. The object detection package was mainly made by cloning the tensorflow repository. The training model was successful at recognizing the water cup we set in our mapping zone.

Conclusions

To conclude, we can say that the project as a whole was somewhat of a success. We did not achieve autonomous navigation within a map and we did not manage to get the camera topic to display our video simultaneously in Rviz.

However, we managed to create accurate and reliable maps, at a reasonably fast speed. This thereby means that our robot's odometry/proportional drive controller is able to be successfully published. In addition, we managed to incorporate the use of OpenCV in order to have our camera detect and distinguish between different objects. While we did not achieve all of our goals for the project, we still came a long way and expanded our robotics knowledge along the way.

Appendix:

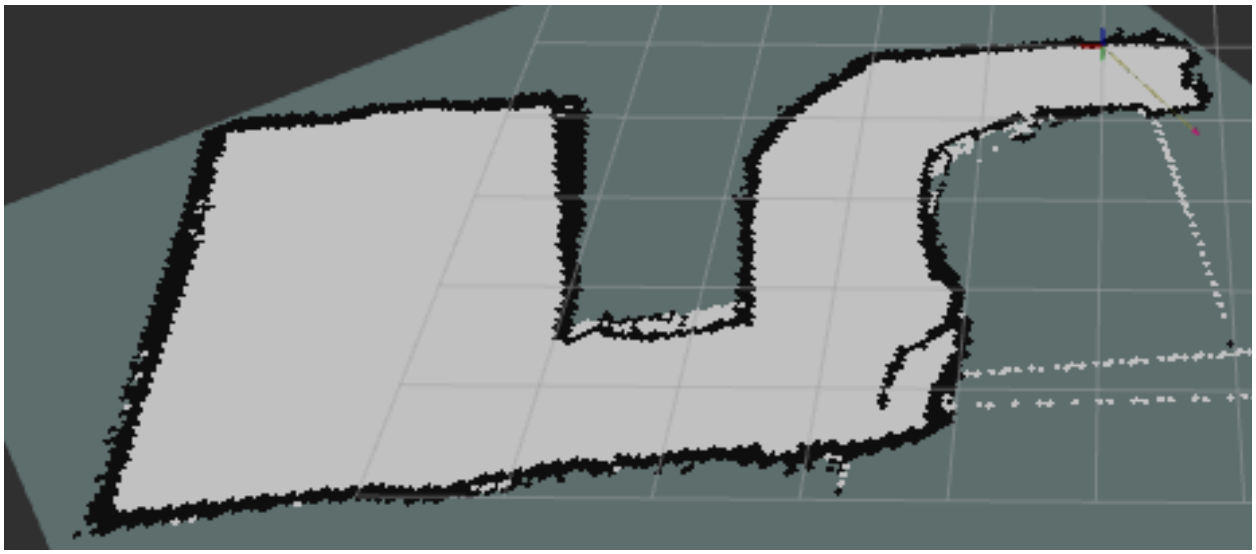


Figure 2. Screenshot of the final map build in rviz using /scan topic.

Sources:

- https://docs.ros.org/en/foxy/Tutorials/Tf2/Introduction-To-Tf2.html
- https://automaticaddison.com/getting-started-with-opencv-in-ros-2-foxy-fitzroy-python/
- https://navigation.ros.org/
- https://github.com/tensorflow/examples.git