Cloud Controlled Robotics and Submarine Design

Jacob Samuels
Electrical and Computer Engineering
George Mason University
Fairfax, Virginia
jsamuel3@gmu.edu

***Abstract*—This project expands on the previous project of cloud-controlled robotics with servoing and odometry to combine all functions into one package that installs via bash on a SBC.**

I. Introduction

In this project, I designed a cloud-controlled water-resistant small robot. It featured odometry using optical encoders and servoing to cardinal directions using an inertial measurement unit. This was accomplished by using differential drive mechanics to approximate the distance traveled with the optical IR sensors.

II. Implementation

A. Design

The goal for the design of this robot was to keep the entire bot within a 10cm cube, make the robot have encoders on each motor, and the overall design to be IP68 waterproof. IP68 waterproofing states that no dust can enter the device, so while my design is an enclosed box with the robot's electronics enclosed. It is important to note that sealant must be used on all joints to prevent leaking. In creating the robot, I designed a platform for the wheels to sit on. My motors were housed in an enclosed shell that the motors slid through and were held secure by a set screw. With the motors mounted, I designed the wheels to be 50mm in diameter to apply more torque to the robot. Motors and wheels are attached, now I modeled a small caster, so the robot has 3 points of contact with the ground to make servoing easier.

B. Servoing

My first goal with this project was to get servoing working correctly. When given a cardinal direction the robot would turn in place until facing the direction required. Firstly, I attained the ability to face in the cardinal directions by using the IMU to check its heading and apply a proportional loop. The important note here is that the IMU loops from 360 to 0 when crossing across north. This poses to be an issue when dealing with error loops for servoing. This complication was fixed by mapping the values of 360 and 330 to 0 and 30. Allowing for a smaller error when the robot is close to the location. With cardinal direction working, I translated commands for left and right turns into a desired heading and the closed loop can handle servoing to magnetic headings. The theoretical heading was maintained by IR feedback with each tick of the motor. This was used in conjunction with the IMU for the closed loop for accuracy

C. Closed Loop Driving

To operate driving in a closed loop we needed optical feedback from the encoders to register distance. This was accomplished with simple analytical methods that have measure the change and x-direction and change in y-direction with a tick of either motor encoder. One tick of either motor encoder multiplied a delta-x and a delta-y by the sine and cosine of the current heading to figure out the actual delta-x and delta-y for the robot assuming distance the motor moved and the heading. With distance maintained, we can calculate velocity of each motor by checking the distance traveled over an increment of time. With closed loop velocity maintained we can change the target velocity and have multiple velocity steps.

D. Cloud-Control

This robot uses UDP communication over access point mode off the Arduino device. The Raspberry Pi Zero W connects to the Arduino's access point and sends the bot UDP packets over port 4242. The received packets on the Arduino contain a wanted velocity, heading, and mode. The mode determines if the bot is servoing, driving, stopping, or reporting. When the robot reports, it sends back two arrays. One has odometry information of x and y global distances. The other has compass data off the IMU. This is compared to the odometry information for accuracy.

III. Conclusion

Cloud-controlled robotics and odometry is explored in the scope of this project. I created a small robot that worked with odometry and servoing to determine its placement in the world about its starting point and magnetic north. Overall most functions were working, however, for the demo, servoing to north did not function properly, and speed control with closed loop did not function properly. I did end up getting it to work, however, the speeds were too slow to register to the strengths of the motor.