Armon Rahimi

Omar Al Nabulsi

Eric Edelman

CPSC 254

14 May, 2019

Project Summary

Our project goal was to use an IMU (Inertial measurement unit) to create an active tracking system. This was done by using the IMU to provide the direction the object, in our case a directional antenna, is facing based on true magnetic north. We started out with researching how to incorporate software from Rasberry Pi to control a stepper motor and make sure we could get the motor to turn to specific directions. After that, we had to implement the Haversine formula into a python script to get the heading from two sets of GPS coordinates. The haversine of an angle equals one minus its cosine. We needed to implement two functions that would communicate with the ROSbridge to calculate the distance between the 2 coordinates and the radius and distance traveled. We also had to get ROS on the raspberry pi but after some research, we realized that ROS will interfere with the Raspberry Pi’s I2C distorting the IMU data. In order so resolve this issue we had to use a web socket to communicate with ROS between the Rasberry Pi and the Nvidia Jetson TX2 to decode the information from the ROS format to basic Python values.

The motor control was another challenge we ran into. Raspberry Pi pulses needed to select a set of pins to generate the motor movement. The motors themselves were controlled from while loops based on the data produced from the Haversine functions. The active tracking of an antenna to follow the GNSS (Global Navigation Satellite System) through ROS (robot operating system) to receive live updates on location and turn the motor accordingly. It can be controlled remotely in and transmit coordinate data in real time to the user through the terminal on a Linux machine. The operating system we used was a three-clause BSD license, which will prevent us from running into restrictions and distributing the software.

The final accomplishment of this project was successful in creating an active tracking program. We added commands to make the stepper motor move and stop through the Haversine math formulas. We were also able to get the ROS to enable live updates of the correct position, which helped us create a reliable operating system without any delay transmitting data to the ROS through the global navigation satellite system. All in all we met our goal within the time allowed and finished with a working product.