William Seto

Team B: Auto Pirates

Teammates: Bikramjot Hanzra, Shiyu Dong, Tae-Hyung Kim, Tushar Chugh

ILR03

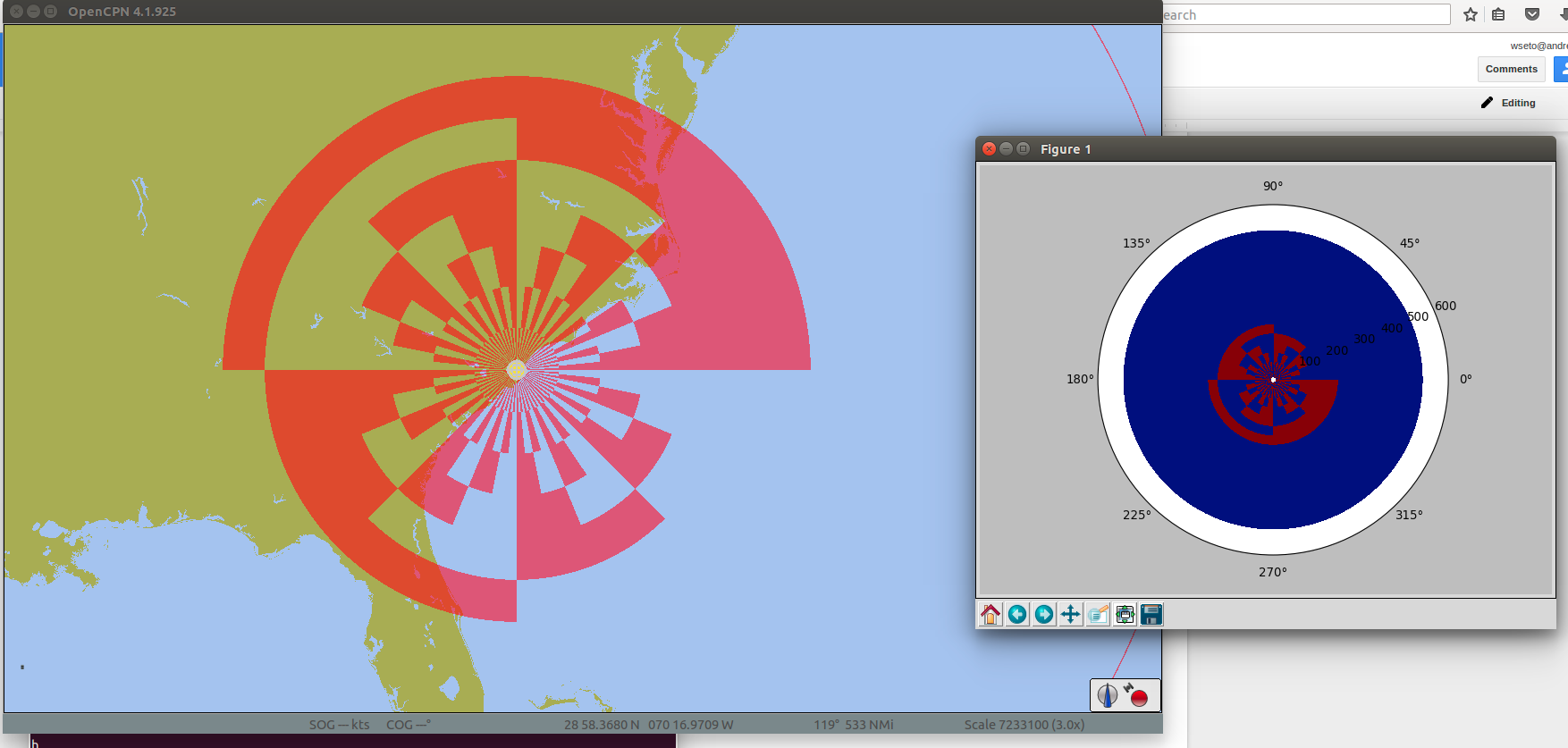
October 30, 2015

Individual Progress

For this lab, I finished the interface between OpenCPN and ROS. In addition, I worked with the team to make more revisions to our CoDR, in which we primarily focused on the FVE, WBS, and critical path.

As mentioned in the last ILR, I was looking into using the Boost IPC library in order to pass the radar image data from OpenCPN to a ROS node. I implemented a basic program which passes data using shared memory with mutex protection. It seems to work well enough, as I was able to publish the received data as a ROS topic, and then wrote a simple subscriber to plot the data published on the ROS topic. Figure 1 below shows the radar image rendered on OpenCPN vs. the replotted data as received through ROS. The plot was OK for our demo, but improvements will need to be made since the plotting library I used, matplotlib, is very slow and cannot plot in real time. Moreover, as seen in the figure, the angles of replotted data are going in the wrong direction as compared in OpenCPN. Finally, it remains to be seen if the synchronization mechanism I used actually works. We had tested with a static image, so when we collect dynamic data, we will see if there are any problems.

For our revisions to the CoDR, I primarily worked on the FVE and the work breakdown for the perception subsystem. In general, our perception goal by the end of the semester is to have some filtering implemented since we have seen that the radar image can be very cluttered. Our hope is to be able to remove the noise and focus on the obstacles of interest.



*Figure 1: Radar image from OpenCPN replotted by ROS node*

Challenges

In implementing the interprocess communication, the primary challenge I faced was understanding the library. Even though the implementation was relatively simple, I had trouble finding good documentation and examples online. The API lists many different ways to create and manage shared memory, but it took some trial and error for me to grasp the basic concepts. Also, I had a lot of debugging, dealing with segmentation faults along the way to successfully passing the image data between processes.

Although it should have been the simplest part of our effort, we spent a whole night just figuring out how to plot the data that was published as a ROS topic. We thought we were working with a simple publisher and subscriber but it somehow ended up that our received message was in a hex string for some reason, and so we needed to do a conversion. This may have been due to the fact that we wrote the publisher in C++, but the subscriber in Python, although this shouldn’t be a problem.

Teamwork

Shiyu Dong

Shiyu worked on understanding the data format of the radar image and did some research on possible filtering techniques. He also worked with me to revise the FVE and risk management sections of the CoDR.

Bikramjot Hanzra

Bikram constructed a model of a boat in SolidWorks. He also spent time learning about Gazebo and Stage to understand their simulation capabilities.

Tae-Hyung Kim

Tae-hyung succeeded in starting a path planning simulation in Stage. He also met with members of the SBPL lab to get some insight on how to apply the library to our project

Tushar Chugh

Tushar spent time understanding the SBPL and experimented with the example simulations. He also worked on the PCB PDS conceptual design and revised the path planning section of the FVE.

Plans

The main goal for the perception team is to collect some real radar data and begin filtering the data. The first thing we will do is collect data with the boat at NREC, and then we will plan for our first field test, which should occur the 2nd or 3rd week of November. As for working with the radar data, we found a paper online, in which researchers successfully developed an autonomous boat using the same radar we have as their primary sensor. Once we have some data, we will try out some of the techniques outlined in the paper.