William Seto

Team B: Auto Pirates

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

ILR02

October 23, 2015

Individual Progress

For this lab, I worked on developing the functionality to integrate the captured radar data with our ROS system. In addition, I made some improvements to sections in our CoDR, based on the feedback given by the instructors and TAs.

As a brief summarization to the state our project, we are currently able to interface to the radar and visualize data using an open source source chart plotting software named OpenCPN, and a plugin, Navico BR24, which does the actual parsing of the UDP multicast packet data sent by the radar over the ethernet network.

Our current goal is to bring the radar image data, stored as an array of structures in the plugin, into our ROS system so that we can publish it as a topic, and then log it and use it to test later. The first step of progress I made was getting comfortable with compiling the plugin source and reinstalling it into OpenCPN so that we can incorporate code to extract the data we want.

Next, we decided that instead of trying to build the entire plugin as a ROS node, we would create a separate node, and then communicate with the plugin through IPC, and accessing the radar data as shared memory. I spent some time learning about the Boost Interprocess library and how we might utilize it to pass data from the plugin to a ROS node. Currently I have figured out how to do some basic shared memory initialization, as well as using mutexes for synchronization between two processes.

For my progress on refining our CoDR, I focused on adding more detail to the project description and use case. Additionally, I worked with Shiyu to refine our requirements so that they would mesh well with our proposed validation experiments.



*Figure 1: Planned flow of radar data*

Challenges

I faced challenges with figuring out how to compile the plugin and reinstalling it into OpenCPN. Since the plugin source only works with the latest version of OpenCPN, for which there is no package distribution available, we first tried to build OpenCPN from source. However, we ran into several issues which involved having the wrong dependencies and libraries. After a lot of uninstalling and reinstalling of various libraries, we finally figured out the right configuration and were able to see some print statements that we added to the plugin.

On figuring out the interprocess communication, I’m facing the usual challenge of learning to use a new library. The challenge is to make sure that both processes can read and write from the shared memory and that the actions are synchronized so we don’t encounter any race conditions.

Teamwork

Shiyu Dong

Shiyu worked with me to understand and compile OpenCPN and the plugin that we are using to get radar data. He also updated the risk management and spring validation experiments sections from the CoDR.

Bikramjot Hanzra

Bikram worked on frame transformations using the ‘tf’ package provided by ROS. In addition, he updated the website and corrected grammatical errors in the CoDR.

Tae-Hyung Kim

Tae-hyung studied various libraries that can be leveraged for path planning. He did the trade study and suggested we go forward with the SBPL library.

Tushar Chugh

Tushar took the lead and assigned us various tasks in revising the CoDR. He also revised the functional and cyberphysical architecture. Finally, he looked into specific algorithms in the SBPL library.

Plans

I will be continuing to work on passing the radar data from the plugin to the ROS node. If this is completed, we will go to NREC and log some real data. Then we will try to redisplay or visualize the data to make sure that the data was passed correctly. In parallel with this task, I will begin to look at relevant papers to study how we can identify objects in the radar image.