Nick Crispie
Team C: FlySense
Shivang Baveja
Joao Fonseca
Hari Suresh
Nihar Tadichetty
IRL05
11/10/17

Personal Progress

This week I worked primarily on the PCB and the coloring node that transforms the mapping results into an image that is displayed on the Epson AR headset. We received the PCB parts in class on Thursday last week. We also received a majority of our digikey parts, but were missing a couple of key components for our design topology. Luckily, I was able to find some substitute parts from the stock that we have in the lab and fully populated the board. However, I quickly ran into problems, since the board was not putting out the desired output voltage. I debugged the board and realized a set of capacitors were wired backwards. I was initially under the assumption that the wiring was based on the initial design that I developed, and forgot that I had switched the polarity on the layout in order to accommodate larger traces for the power lines. After I fixed this and replaced the capacitors, the board still was not working properly, but I haven't had time to do any additional extensive debugging since I needed to also work on the onboard sensing coloring node.

As mentioned before, in our design we need a coloring node in order to transform the output from the octomap into something usable for the Epson AR headset to display. This requires building an image from scratch using Open CV, transforming that into a ROS image, then publishing that image on a topic that the AR headset is subscribed to. I built a node and integrated in into the existing codebase that achieved these desired objectives. The coloring node as it exists right now only handles binary occupancy. The node subscribes to a projected_map topic that is published by octomap, which is a 1D array with zeros representing unoccupied, 100 representing occupied, and -1 representing unknown occupancy. The project_map topic also provides some information such as the 2D map height and width, as well as the resolution of the map. In future work, we will have the octomap populate the 2D map with probabilities that have more gradation than 0 or 100, and we can do more than binary coloring. Right now the node colors anything that is an obstacle red, and anything that is not an obstacle greem. The node manually builds 3 separate channels and combines them into one image before translating that into a ros image transport message to be broadcast on a topic that the AR headset is subscribed to. In the PR demo, we weren't able to show this fully working-right now, the image only updates when we run the node on a computer. The first image is received but not updated in the AR headset, but we are working through some connectivity issues on wifi with the Jetson and need to check the fidelity and compression of the image transmission. This is part of our general to do for refining parts and finishing integration, but the core functionality is done.

I also mounted the quad on the rolling cart so that we could have our own test setup. I purchased a rolling cart from Amazon to use as a rolling test platform We were able to get some initial data from the velodyne and the DJI flight controller. I also assisted shivang on testing the DJI pose estimation when he was setting that up last week.

Challenges

I ran into numerous challenges during my work this past week. Chief among them was debugging the Power Distribution Board PCB. As mentioned previously, I didn't have all of the parts I needed, but was able to find them in the lab. After that, I still was not able to get the desired 12V output, which I am still working through a solution for (see the Future Work section of the IRL).

An additional challenge that I ran into was understanding the OpenCV documentation for creating images from scratch. I have very little experience with OpenCV and am still learning a lot of the ins and outs of C++, so reading through a lot of the online documentation for the development of the coloring node was initially challenging, but I was able to find enough resources and test incrementally in order to get a basic output image. After completing this, Shivang and I had some problems when we initially integrated the code to the AR. The image was displaying one image but it was not updating. This might be a case of the wrong compression over a low-fidelity wifi connection (we are trying a new router in the coming days), or some other issue. That is one thing I will be helping out with in addition to refining the coloring node in the ways described in the Future Work section.

As a team, we ran into some challenges with mapping, and are still working on the optimal solution for that. We abandoned the previous mapping method based on the AIRlab code and have dedicated all of our efforts on Octomap. We also initially had some registration problems with the data we obtained through our own test setup, but realized it wasn't really a registration issue and more of an issue of the LIDAR axes being slightly off from what was recommended in the manual, so the pose estimate from the DJI controller was not perfectly matching the frame of the LIDAR.

Teamwork
Joao worked on generating sound warnings from the closest obstacle, refined the dynamic window, and implemented voice commands using the Google API.

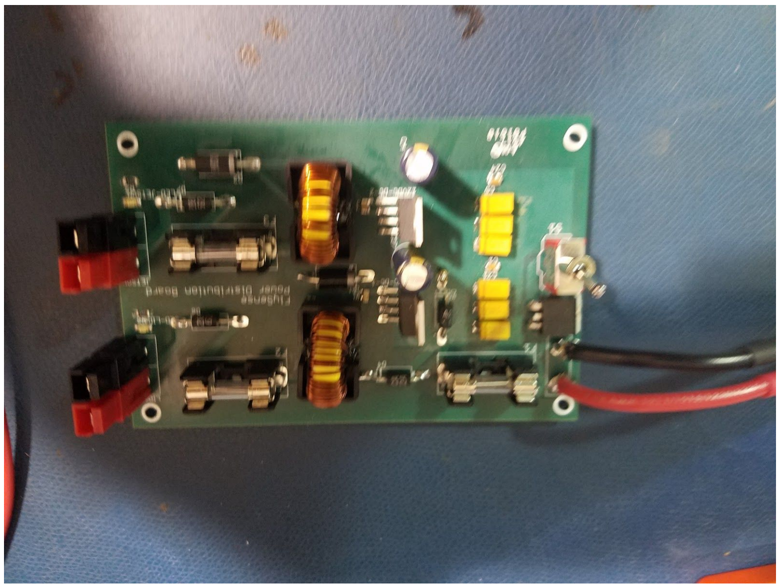
Nihar worked extensively on improving the user interface. He also did work with Joao and Shivang on integrating sound warnings and transitioned the voice commands code Google API that Joao was using to the more robust PocketSphinx API.

Shivang has done lot of work on systems integration and cultivating the system architecture. He also was able to complete the code for the DJI state estimation and did work with Hari on PCL filtering for the point cloud data.

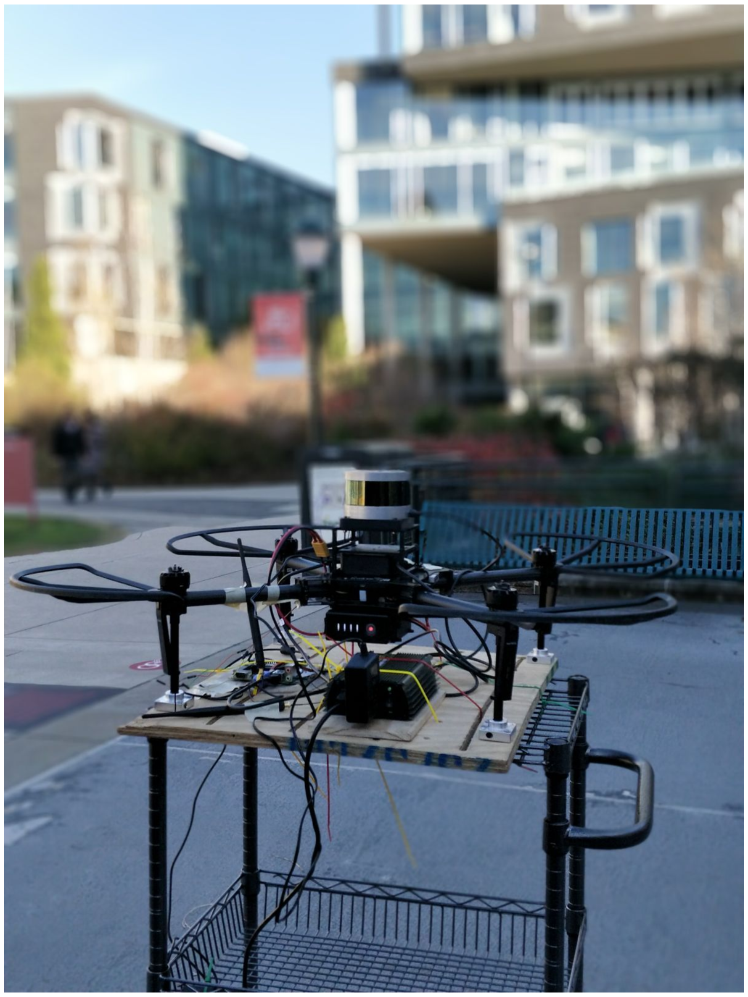
Hari has continued to work on Octomap and PCL filtering for generating the obstacle map. He worked with Shivang to set up and run data experiments with our test setup on the cart.

Figures

Shivang and I worked on calibrating the DJI state estimation



Populated and soldered PCB


Test setup in our FVE location. I designed the setup and did most of the assembly.

Future work:
As a team, we need to refine all the components and finish integrating all the individual parts of the system. Once that is done, we plan on conducting several trial runs of the FVE experiment in order to make sure that is as smooth as possible.

My individual focus in the next week will be to refine coloring node to have a specific fixed size, instead of constantly changing size based on the output from Octomap. This is important since we want to render the image directly from the coloring node on the AR headset screen, and want the size of that image to be predictable so that the user is not surprised nor does he/she get their view obstructed at any point. I'll also be working on integrating the sound warnings based on the nearest obstacle (based on time to impact) in that coloring node from the work that Shivang and Joao have been doing on the time to impact estimation and sound warning generation.

I also need to figure out what is wrong with the power distribution. My best theories involve either ESD damaging the chip or the capacitors not fully charging correctly, or the diodes not allowing current through when they should. I have ordered some extra parts to try out and plan on working through each part individually, though it is hard to figure out exactly what is wrong since the connectivity tests that I've run testing to make sure parts are connected correctly (and not connected when they shouldn't be connected) haven’t come up with any major problems.