Harikrishnan Suresh

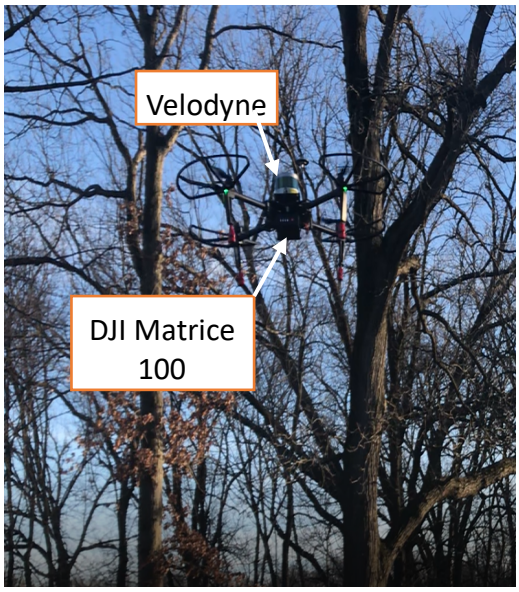
Team C: Fly Sense
Teammates: Shivang Baveja, Nicholas Crispie, Joao Fonseca, Sai Nihar Tadichetty
ILR08
March 1, 2018

Individual Progress

I have been working on porting Joao's point cloud coloring and sound warning algorithms to C++ and ROS. After a few productive discussions on the algorithms, it was decided to implement and test them first before fine tuning it for our application. The first version of the ROS C++ code has been written. The aim now is to feed dummy data in the form of ROS point cloud sensor messages and compare the results with the MATLAB results. The code has to be optimized to improve performance and robustness so that it can be deployed real time on the actual quadcopter system. Once this is complete, the code will be integrated with the flysense software stack.

In order to ensure minimum pre-flight wait time, we have decided to launch the code on power-up of the Jetson. I looked into the 'robot upstart' package which sets up a 'config' file to run on power up. The main flysense onboard launch file was successfully linked to the package and tested on my laptop. The challenge now is to set up the Wi fi connection between the Epson and the Jetson, by specifying the IP address and setting up the ROS master.

The first flight test with the Velodyne onboard was completed, and live point cloud data was streamed from the Jetson. The point cloud data and the Bird's eye view image was visualized in Rviz. The point cloud data along with state data from the DJI was recorded to a bag file, so that I can be used to test the algorithms offline.



We also had our first pilot workshop at Near Earth Autonomy. The main objectives were to gather feedback about the final HUD user interface on the Epson along with the key elements projected and the SVE tests. The major takeaways from the pilot workshop are:

1. The major elements in the HUD are speed and altitude, yaw is optional.
2. Birds Eye View (BV) –
    - It is important to clearly communicate what the heading direction is, using a triangle or arrow.
    - The size and position of the BV should be customizable, if possible
    - One of the ellipsoids used as thresholds for coloring must be explicitly represented using a light ghost ring to be used as reference for the pilot
    - The obstacles in red should be blinking to convey the urgency, and the blinking frequency must be aligned with the beeps in sound warnings
3. Sound warnings
    - It is more convenient for the pilot to have flat sound if the special 3D sound is not possible. Having stereo left/right sound warnings does not add lot of impact because of the numerous orientations possible for a helicopter
    - Preferred warnings setup – One single long beep for the first level, two beeps of shorter duration repeating with a slight delay for the second level and high frequency beeps for the high level of danger
4. The pilot must be informed that his control has been over ridden, by indicating some message on the HUD
5. The tests have to be redesigned as the pilots are not comfortable moving between containers. The final test scenario will be discussed and confirmed soon.

Challenges faced

1. Since majority of my work this semester has been in collaboration with other teammates, the varying course schedule and associated commitments have made it difficult to find free time to work together.
2. The plan was to do multiple tests with the Velodyne onboard. However, due to some hardware issues, we had to stop after the first test.
3. While setting up the Jetson TX-2 software, CUDA 9 was installed on my laptop. CUDA 9 reattaches the OpenCV bindings from the normal source to

itself. This uninstalled the ROS OpenCV interface, and the FVE code stopped working on my laptop. I spent a whole day trying to solve this, and finally succeeded. I had to uninstall CUDA 9.0, install OpenCV 3 over OpenCV 2 and enable ROS support for openCV 3. I had to also install additional support packages in ROS for OpenCV 3 support.

Team work

Team Member	Contribution
Shivang Baveja	• Conducted flight tests with the complete system onboard. • Completed the first version of the pilot control override code. The next step is to test the algorithm in simulation.
Nick Crispie	• Completed all the major hardware tasks including Velodyne cable modification, quadcopter wiring and power module. • Assisted in all the flight tests
Joao Fonseca	• Implemented and tested the algorithm to color obstacle point clouds and sound warnings in MATLAB • Completed calibration of the quadcopter model based on the flight data
Nihar Tadichetty	• Setting up the environment for image segmentation, sorting all the compatibility issues between OpenCV and CUDA 9 • Setup code to read FPV camera videos. Next step is to project the FPV video on the Epson.

Tasks

Team goals

- Complete the hardware setup and make it more robust.
- Conduct more flight tests to collect data and improve the algorithms. Test FPV video during the flight tests.
- Test the pilot control override code in simulation

My goals

- Complete the obstacle coloring and sound warnings code in ROS and test it with the MATLAB output
- Optimize the code to ensure real time performance
- Integrate the code with the flysense software stack