Rohan Thakker

Team C: Column Robotics

Teammates: Job Bedford, Cole Gulino and Erik Sjoberg

ILR10

March 31, 2016

1) Individual Progress

For this sprint, our main goal was to implement the autonomous cone search and improve the precision of landing. Another, stretch goal was to integrate both of these sub-systems together. Throughout the sprint, Cole, Job and I worked on improving the precision of landing. As of last sprint, we were getting rotational invariant updates from the APRIL tag detection node. This was implemented by representing the pose of the quadrotor's body frame in the APRIL tag frame. To achieve this rotation invariance we were using the orientation estimates from the APRIL tag detection node.

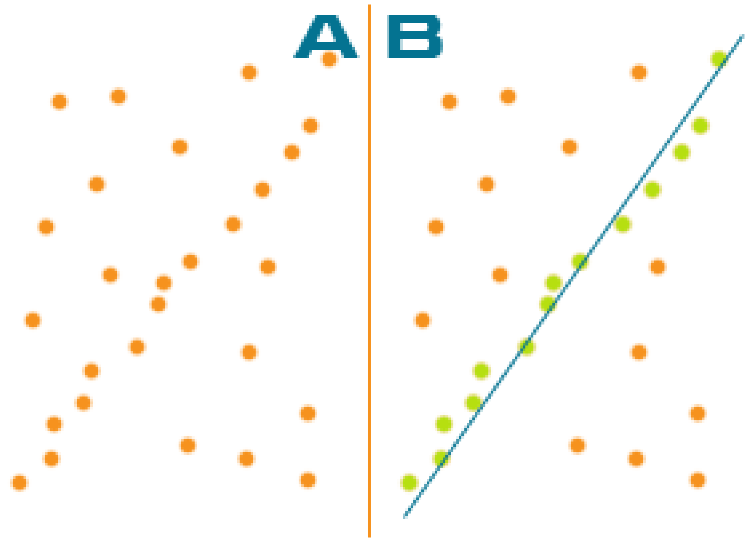

Figure 1: RANSAC Algorithm

We noticed that the rotation invariant pose estimation of the APRIL tag had a high frequency noise. Cole and I were responsible to filter this data. This high frequency noise was mainly due to the presence of outliers. Using a low pass filter such as leaky integrator or moving average would have a bias towards the outliers. Hence, we implemented RANSAC Algorithm (shown in Figure 1) to fix this issue. This was done by picking a random point from a window of 20 data points. The points that lie within a hyper sphere of radius epsilon are declared as inliers. The above process is repeated 10 times. After this, the best estimate of the window is found by taking the average of all the points which are inliers. Further, the update from this filter is used only if there are greater than 70% inliers.

The bottom part of the figure 2 shows the rotation invariant position estimate of the quadrotor with respect to the APRIL tag, represented in the APRIL tag frame. The top part of the figure shows the results after applying our filter. It can be seen that all the outliers are rejected and output is returned only when we have more than 70% inliers.

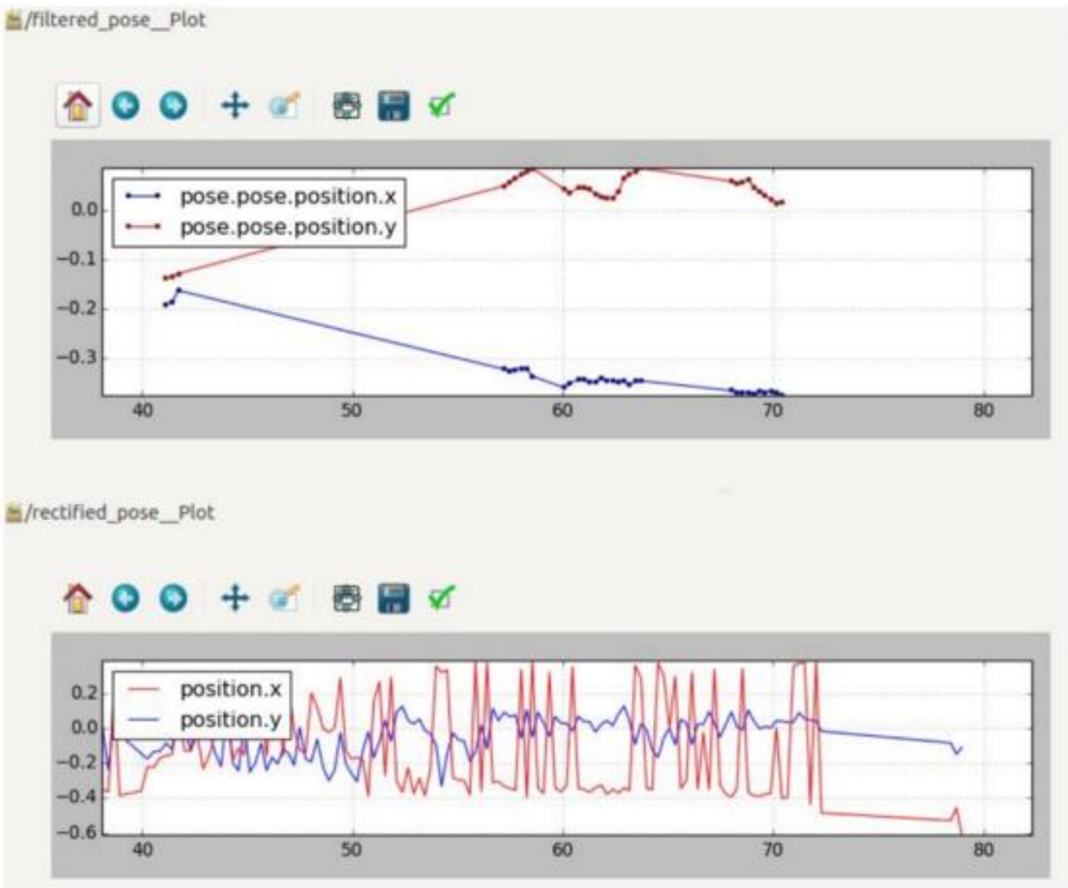

Figure 2 Results with (top) and without (bottom) RANSAC algorithm

2) Challenges

- It was very time consuming to test various algorithms directly on the drone. Hence, we decided to obtain ROS-BAGS for 3 different flights from the quadrotor when we were getting bad performance for servoing over the APRIL tag. We then modified our code and testing on these ROS-BAGS. This strategy helped us save a lot of time in debugging our code. Once our code was returning the expected output on these ROS-BAGS, we were able to get the desired output on our very first test flight.

- We were facing many issues with the visual odometry using optical flow. We noticed that the PX4FLOW sensors worked well in presence of good features that have many edges with corners. Hence, we placed scrap wood at the bottom of the net. This largely improved the accuracy of the visual odometry estimates.

- Further, we noticed that the visual odometry algorithm returned bad estimates because of the shadow of the drone. We resolved this issue by placing a translucent sheet on top of the net to diffuse the light, thus preventing the formation of any shadows on the ground.
- During integration of the landing and searching sub-systems, we faced many issues as the code was spread across different branches of git. Hence, combining them resulted in merge conflicts which had to be re-solved individually. Further, certain branch was using an inconsistent message type for the ROS topic. Hence, during this script we have resolved all these issues and now master is updated with all the changes.

3) Teamwork

As stated earlier, Cole and I were responsible to filter the data of the rotation invariant position estimates of the quadrotor's body frame represented in the APRIL tag frame.

Erik worked on implementing the cone search algorithm. He did a very nice job of implementing this using ROS PARAMS which can be changed externally from the terminal or a python script. Hence we can now change the set-points for our position controller by using an external python script so we don't need to recompile any code. This is similar to the API that was provided by the ARDRONE.

Job worked on improving the precision of landing. He did a lot of testing on our previous algorithm and identified that the readings of the APRIL tag needed to be filtered. He also wrote a python script to integrate the search and land sub-systems.

4) Plan

This sprint we did not completely deliver on the promised demo of the integrated search and land sub-systems. However, we have implemented both these sub-systems and have a nice framework that can help us implement the high level behaviours using python scripts.

For the coming sprint, our goal is to have our complete system integrated for the SVE dress rehearsal. Erik will be working on implementing the behaviour trees using the python script framework that he has developed. He will also be working on integrating the SLAM algorithm with our ROS framework.

Cole, Job and I will be working on closing the loop on landing. Once we have this implemented we will begin testing on the actual docking mechanism. We will also be working on fusing the SLAM updates with the state estimator that is currently running on the drone.

5) References

[1] RANSAC image source: http://www.3dcalifornia.com/

[2] https://pixhawk.org/modules/px4flow

[3] http://www.hardkernel.com/main/products/prdt_info.php