Rohan Thakker

Team C: Column Robotics

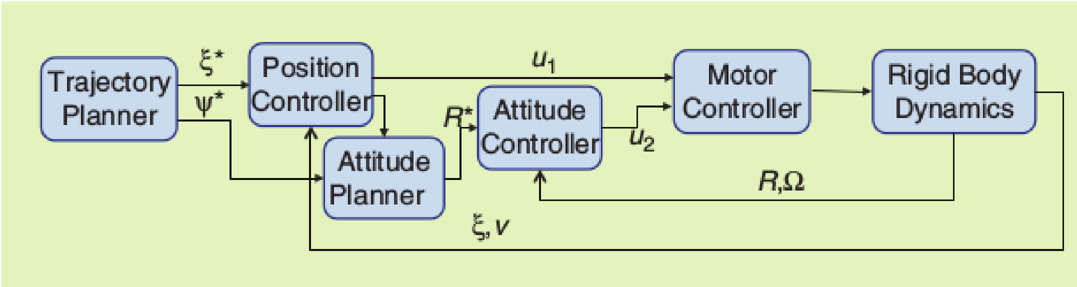
Teammates: Job Bedford, Cole Gulino and Erik Sjoberg

ILR09

March 17, 2016

1) Individual Progress

For this sprint, our main goal was to complete autonomous docking using the feedback from the APRIL tag. I was responsible for implementing a position controller in ROS on the ODROID XU-4 Single Board Computer (SBC). Figure 1, shows the architecture of the cascaded controller which is used on most quadrotors. The reference trajectory is generated for the x, y, z position and yaw of the quadrotor. The position controller calculates the reference attitude and sends it to the attitude planner. The attitude planner generates a smooth trajectory to reach that reference attitude. The attitude controller follows this trajectory by running a PID loop.



Mahony, Robert, Vijay Kumar, and Peter Corke. "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor." *IEEE Robotics & amp amp Automation Magazine* 19 (2012): 20-32.

Figure 1: Control Architecture

During our last sprint, we implemented this architecture by running the position and attitude controller on the PIXHAWK. We used the RF remote control to test the position controller on the PIXHAWK. During this sprint, we tried to send reference positions using MAVROS from the ODROID to the PIXHAWK over UART interface. However, gains of the PID loop running on the PX4 stack were too aggressive and we were not able to get stable flight. During our last sprint we had already tested velocity control of the IRIS+ by sending MAVROS messages from the ODROID. Hence, instead of investing time to debug the controller which was running on the PX4 stack, we decided to implement our own proportional controller on the ODROID in ROS. By the end of the sprint, I had successfully implemented and tested the position controller which was running on the ODROID.

In the previous sprint, we had implemented the APRIL tag detection on the ODROID. However, we were getting the position of the APRIL tag with respect to the camera represented in the camera frame. Hence, the position of the APRIL tag was a function of the orientation of the drone. I helped Cole in resolving this issue by suggesting that we represent the position of the camera with respect to the APRIL in the APRIL tag frame. Cole implemented this using the inverse of the transform of the position of the APRIL tag with respect to the camera represented in the camera frame.

2) Challenges

- Testing the drone was a major challenge for me because I did not know how to fly it. Hence, I had to wait to for Erik or Job to help me test my code. During this sprint, I learnt how to fly a drone by buying a cheap $40 micro-drone. The bandwidth of the dynamics of the micro-drone was much higher. This made it was more difficult to fly. Finally, after learning to fly the micro-drone, I was easily able to control the IRIS+.
- To implement the position controller, I had to verify convention for the orientation of the frame in which the reference velocities were represented. I did this experimentally by removing the propellers of the quadrotor and measure the speeds of the BLDC motors using a tachometer.
- While running the APRIL tag nodes, the estimates of the orientation were not very accurate. This was mainly because we were using the default intrinsic matrix of the camera for the APRIL tag detection library. Cole fixed this issue by calibrating the camera using a checkerboard pattern and experimentally deriving the intrinsic matrix.
- We were using a Wi-Fi router placed on our desk to SSH into the drone. Hence, to perform tests, we always needed to shift the router to the third room where the net was located. Erik resolved this issue by placing the Wi-Fi router in the second room. Hence, we can now connect to the drones from the entire lab without shifting the Wi-Fi router.
- Small net made it very difficult to fly the quadrotor for testing. Erik was responsible to follow with the progress of the setup of the net. This week the net was set up in hallway of the basement. Hence, we will now be able to test our search algorithm on the IRIS+.

3) Teamwork

As stated earlier, I was responsible to implement the position controller on the ODROID and Cole was responsible to get rotation invariant position estimates from the APRIL tag. Erik initially worked on getting testing the performance of the position controller of the PX4 stack. He noticed that the performance of the controller was not good, hence I worked on implementing our own position controller. Later, Erik along with Job, worked on integration of the APRIL tag position updates with the position controller. Job also helped me to implement and test the position controller.

4) Plan

This sprint we have achieved autonomous landing over the APRIL tag with the IRIS+. This was our final deadline to decide between the IRIS+ and ARDRONE as our final platform. We have already shown that we can achieve all the functionalities that are possible with the ARDRONE. Moving forward, it will be equally difficult to meet our requirement on both platforms. Hence, we have decided to stick to the IRIS+ because this allows us to use the RGB-D camera which is required for our desired non-functional requirements.

For the coming sprint, our goal is to achieve autonomous search and landing with the downward facing camera. To do this, Erik will be working on implementing the search of APRIL tag with the downward facing camera. Job will be working on improving the precision of landing. I and Cole will be working on fusing the position updates from the APRIL tag node into the Extended Kalman Filter.

Here is a description of our demo for the next PR:

1. Set up the mock docking station (sheet of paper with APRIL Tag stuck on it) on the ground inside the net
2. Take off the IRIS+ under manual control and hover to a known location where the APRIL tag is not in the field of view of the camera.
3. Switch to off-board mode and observe the IRIS+ autonomously search for the APRIL tag.
4. Observe that the IRIS+ hovers above the APRIL tag and performs autonomous landing.

5) References

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

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