Rohan Thakker

Team C: Column Robotics

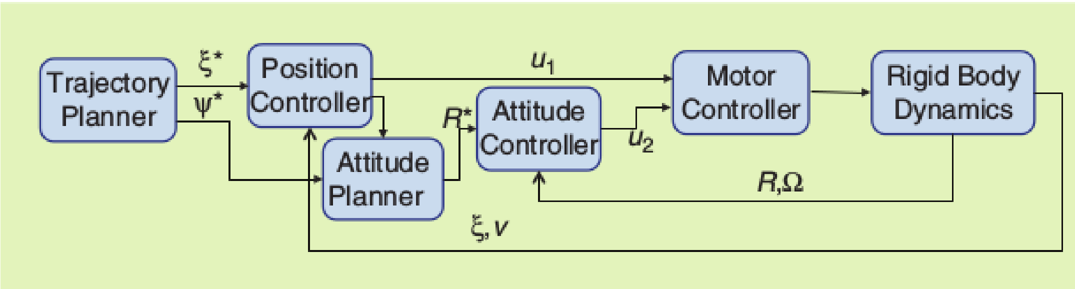
Teammates: Job Bedford, Cole Gulino and Erik Sjoberg

ILR07

Feb. 11, 2016

1) Individual Progress

For this sprint, Cole and I were working on getting the autonomous hovering on the IRIS+ quadrotor. As discussed in the previous ILR, we had multiple ways to implement this. 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 a 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

Our initial plan was to implement the position controller on the OROID and attitude controller on the PIXHAWK. This was done to eliminate the requirement of sensor fusion and time synchronization of the position estimates of state estimator running on PIXHAWK and April Tag data. Accurate global position estimates will be required to land the quadrotor on the docking station.

After going through the PX4 stack we found a feature that supports sending global position updates to the EKF state estimator running on PIXHAWK. Hence, we decided to run the position controller on the PIXHAWK itself. This also allowed us to leverage the implementation of the PX4 which has been tested by several users as opposed to doing our own implementation which could take time to achieve robustness.

To test the position control mode, we took off the quadrotor from the ground under manual control to a height of around 2.5 meters. We mapped a switch on the RF remote to transition from manual control mode to position control mode. Using this switch, we transitioned to position control mode. We verified that the quadrotor was able to hold position in this state. Now, the roll and pitch on the RF remote were mapped to send reference velocity signals. We validated that the quadrotor can achieve stable flight in this configuration.

2) Challenges

- On launching the MAVROS node, the process was exiting with an error. To resolve this issue we found a blog post that stated that the error was related to the ROSCPP package and it recommended reinstalling the ROSCPP package. To do this, we ran "sudo apt-get remove ros-indigo-roscpp", followed by "sudo apt-get autoremove" and "sudo apt-get install ros-indigo-roscpp". This still did not resolve the problem. We had taken a backup of our complete linux image when everything was working on the ODROID. Hence, instead of spending time to debug this issue, we used the backup image file to reset the SD card and resolved the issue.

- We shifted to a new SD card in the process of reverting to a backup linux image. Our earlier issue was resolved, but we noticed speed of Ubuntu had drastically reduced. We suspected this to be a problem related to the new SD card that we started using. Hence we burned the backup linux image to our original SD card and this resolved the low speed issue. We that the slow SD card was taken from a digital camera and hence was not Class 10 (high speed). Hence we ordered Class 10 SD cards.

- We were conducting our initial tests in the net in the MRSD lab. Team E had mounted their mock docking platform to the top portion of the net. Hence, to avoid crashing into the plank, we flew the quadrotor close to the ground. We observed huge errors in the readings from the state estimator. We suspected this to be a result of aerodynamic effects because of being closer to the ground. Hence, we conducted a test in better net at a larger height and this resolved the issue.

3) Teamwork

Our team got a lot of work done this sprint. We followed our plan of working on the things that we feel will be most difficult to implement and leaving the easy things for the later sprints. Cole and I were able to achieve autonomous hovering on the IRIS+ which is an important step to searching and autonomous docking.

Erik had a major breakthrough in getting the RTAB MAP SLAM package working on the ODROID. Due to lack of binaries for ARM architecture, he had to compile the package from source. Further, he individual had to install the correct versions of each decencies and handle conflicts with the versions of dependencies installed with ROS. He put in many hours to get this working. Finally, we are now able to achieve global position estimates at a couple of hertz using SLAM running on-board.

Job continued development on the ARDRONE and was able to achieve autonomous landing. Now, we are more confident about our risk mitigation in case we are not able to achieve our requirements with the IRIS+.

4) Plan

We are moving closer to the deadline of making our final decision of choosing between ARDRONE and IRIS+. We intend to implement autonomous landing by the end of February on the IRIS+. Hence, we have decided that for this sprint, all the four team members will be working implementing autonomous landing on the IRIS+. To achieve this we will divide ourselves into two teams of two members. One team will work on the problem of moving the quadrotor exactly above the APRIL tag (i.e. located on the docking station) and hold position. This will require implementing global position updates from the APRIL tag and use it to close the position controller. The other team will work on implementing autonomous land assuming that we are located exactly above the APRIL tag.

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

1. Set up the mock docking station on the ground (sheet of paper with APRIL Tag stuck on it)
2. Take of the IRIS+ under manual control and hover to bring the APRIL Tag in the field of view of the camera
3. Switch to autonomous mode and observe the autonomous landing of the IRIS+

5) References

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

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