Spring Sprint 2: AR.Drone SVE Architecture, and Autonomous Docking

Individual Lab Report #7

Job Bedford

Team C: Column Robotics
Eric Sjoberg, Rohan Thakker, Cole Gulino

ILR6
2/11/16

Individual Progress

Architecture:

These past 2 weeks I designed the software architecture that will be used for the AR.Drone SVE Demo. As a team we had a design review to critique my design add any further detail the team deemed nesseary. This Architecture is expandable onto the IRIS+ drone to an extent and will be used to coordinate our software schemes and planning.

The architecture will be a state machine. The State are as followed:

1. **Setup and Tornado Search**

Once the demo environment is set up, the Drone will start from an initial position and will be manually operated to a 'Flight Plane' height, where it will wait for the demos initiation. Once initiated the drone will enter the Tornado search state where it will generate an autonomous cone flight path from a function (given a cone angle defined by the field of view of the front carmea. A rough sketch is shown below in figure 2. This flight path runs a loop of trajectories to the 'Tum_Ardrone' in increments of .5 meters with a half second delay.

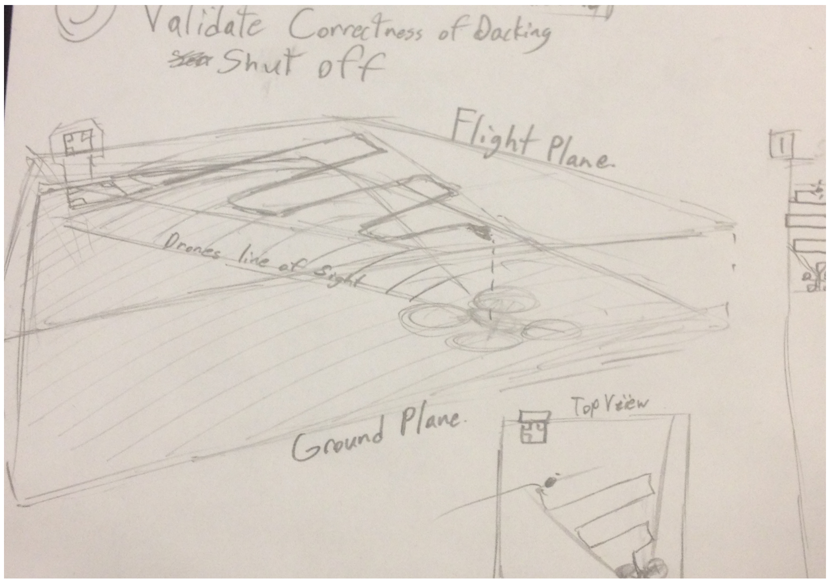

Figure 2: Sketch of Tornado Search.

The state is broken if the April tag detection node sends an 'identify' tag message. If the tag is lost in the succeeding state, the system will return to this state and continue the tornado search. If system reached 7 meter parameter limit while in this state, it will return to origin and land.

2. **WellHead Alignment**

Once Wellhead tag is detected, the system utilizes April tag poses and tum_ardrone state estimation in a PD loop (inside mover node) to position drone 1 meter in front of Wellhead at the appropriate Flight Plane height. Once system is in proper position it will switch to the downward facing camera, breaking out of it's current state. If PD control is compatible with tum_ardrone framework, will utilize the tum_ardrone's pose estimation and take a series of flight steps to approach it's target.

3. **Dock Alignment**

System will utilized april tag detection with downward facing camera to run PD Loop (small cmd_vel's) to hover over dock for alignment. State will be broken when tag is within +/- .3meters from center of cameras view. A depiction of this state can be seen in figure 5.

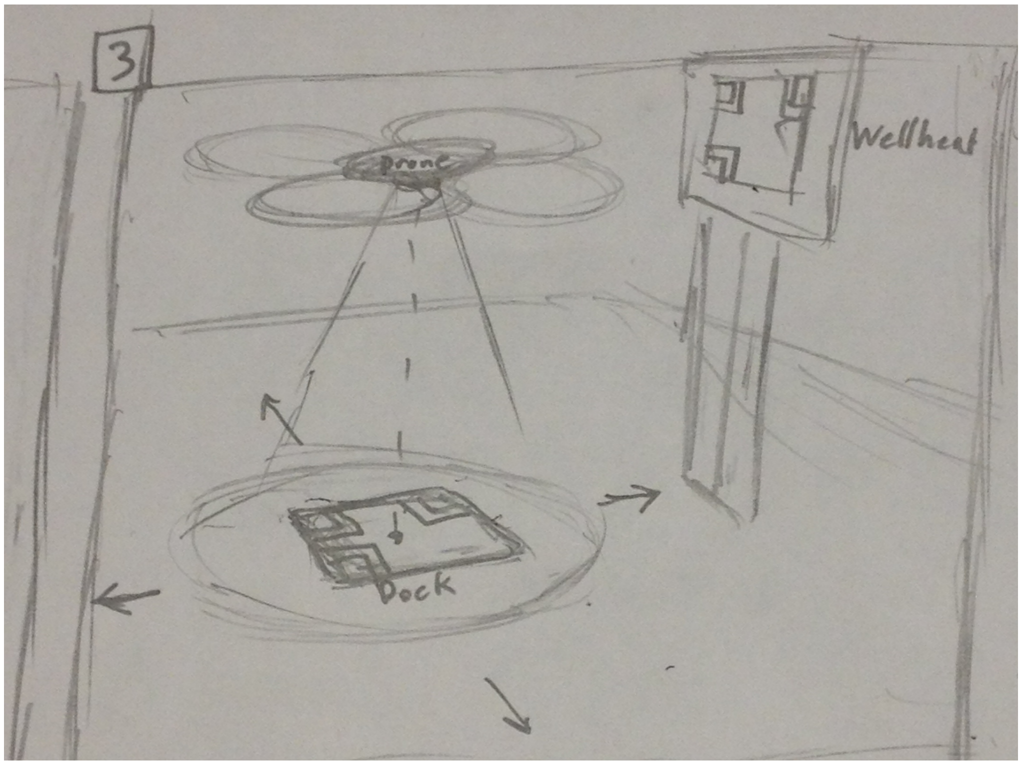

Figure 5: Dock Alignment State

4. **Autonomous Docking**

System will descend in .5 meter increments until 1 meter above target. This descent is to the lowest allowable height given ground effects, camera view and tum ardrone limitation. System will readjust once more (using previous PD loop) Finally system publishes to /land topic for hard landing. Process is shown in figure 6.

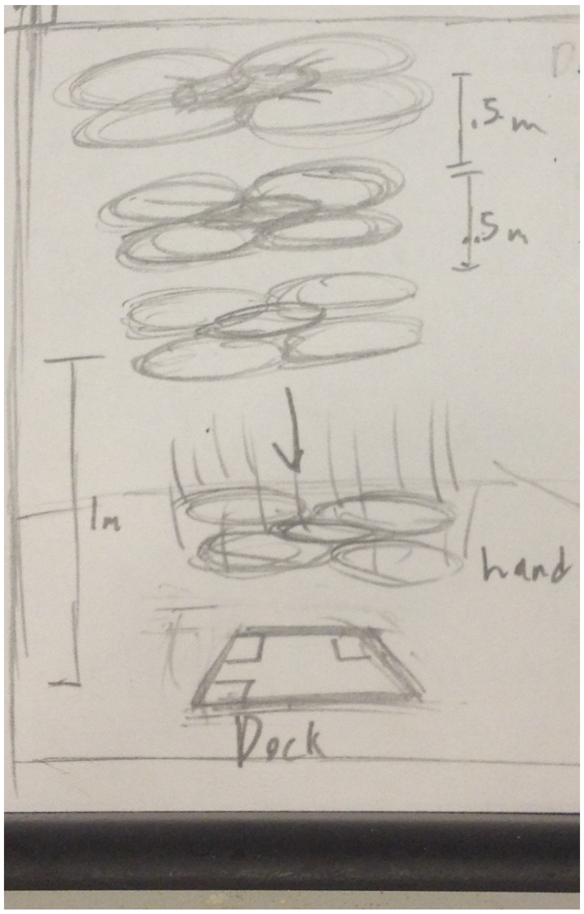

Figure 6: Autonoumous Docking State

5. **Validation and Shut Off**

System will turn off. Validation will proceed based on drone landing gear locations relative to circular markers, as seen in figure 7. This concludes the demo

Autonomous Docking with forward facing camera:

This week my demo showcase the ardone autonomously aligning with a well head tag and landing in front of it. The demo primarily used the april tag detection node and the mover node from last sprint.

In the demo, the system determines the drones relative distance from target in 3-space. Then repositioned itself to be .25 meters in front of target. There existed error in this estimate. The longitudinal distance, the drone's forward direction, estimate was precise, while the latitudinal distance, the drone's right and left, was imprecise. During the flight approach, if tag left cameras field of view and drone is lost. The inaccuracies were proportional with distance. The further the drone was away from the tag the more error existed in the measurement. Step approach was used to close the distance between drone and target. The drone would take a half distance first step, recalculate, take a full-distance second step, recalculate, then precisely hone onto the tag in the third step, and end with landing. Landing is a near-straight drop, and thus not requiring the précising of downward facing camera for accuracy. A PD loop was not used since tum_ardrone framework offered reliable pose estimation.

Due to team's success with the IRIS+, I will no longer be working on the ar.drone, and will work on autonomous docking for the IRIS+ this upcoming sprint.

Challenges

I originally claimed to dock using downward facing camera in the last ILR., but the downward facing camera was incompatible with April tag detection node package. Figure 8 show one of the debugging tests. After many attempts to debug and get this problem resolved, I decided to switch to using the ardrone tag that came with the robot. The ardrone is already programed to recognize this tag and just has to have its driver configured properly.
Unfortunately the downward facing camera and tag configuration had terribly documentation, and search around various forum, I realize other had the same issue I was have. I started to construct my own RGB blob detection node to be used as a back, but due to time constraint and priorities of the mission choose to re-scope to autonomous landing with the forward facing camera. Since the drone's landing is nearly a straight-drop. Using the downward facing camera proved unnecessary.

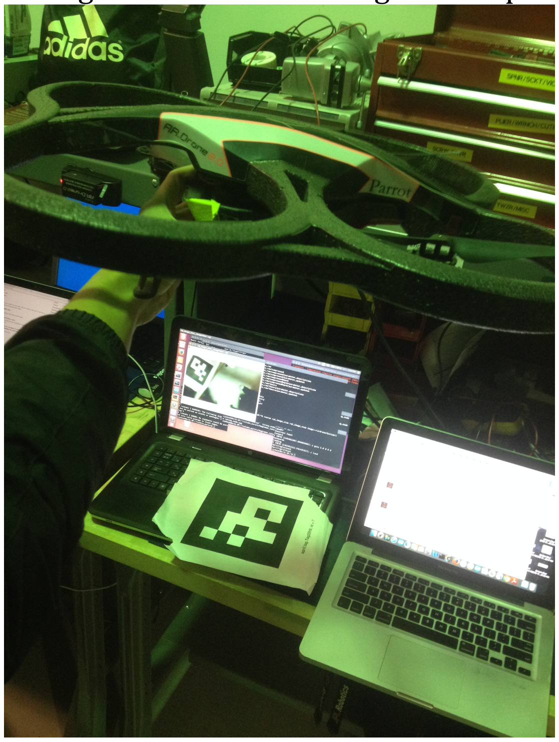

Figure 8: Debugging Downward Facing Camera

Teamwork

The team divided up into three groups this sprint.

My mission was ardrone autonomous docking.

Eric's mission was to validate the perception capabilities with the ros package rtab mapping. This included checking feasibility of running the package on the Odroid as well as measuring the computational cost and update rates. Eric spent most of his time installing the right softwares and drivers to make the package compatible with the Odroid's ARM processor. He was successful in this endeavor and the rtab real-time mapping and localization package looks quite promising.

Cole and Rohan set out on autonomous hovering with the Iris+. The Pixhawk comes with an autonomous flight mode that the MAVROS package can command. Due to the small flight area of the MRSD net they were unable manually fly the quad long enough to switch to the autonomous flight mode. Eric and I are the only ones can properly pilot a quadrotor on the team, so Cole and Rohan's work was sometimes bottle necked by the availability of a pilot. After switch to a bigger flight net in Wean Hall, we were able to achieve autonomous hovering right away.

Upcoming Week

Autonomous docking of the Iris+ is the goal for this upcoming sprint. Team C will break up into two subteams that will converge on the same goal from two different directions. The 1st week one team will work on precision hovering utilizing april tag dectection. The other team will test and quantized the iris+'s open loop landing capabilities. The 2nd week both teams will share information and code, then focus on autonomously docking. I will be starting on the precision hover task, since I have the most experience working with April tags on the team.