Progress Review 3

Cole Gulino

Team C / Column Robotics

Teammates: Job Bedford, Erik Sjoberg, Rohan Thakker

ILR # 4

November 13, 2015

Individual Progress

For this project, I mainly focussed on research, systems engineering, and the PCB design.

I did extensive research on four different ROS packages: ardrone_autonomy [1], costmap_2d [2], tum_ardrone [3], and navigation [4].

Figure 1 shows an example of the notes I took on these packages.

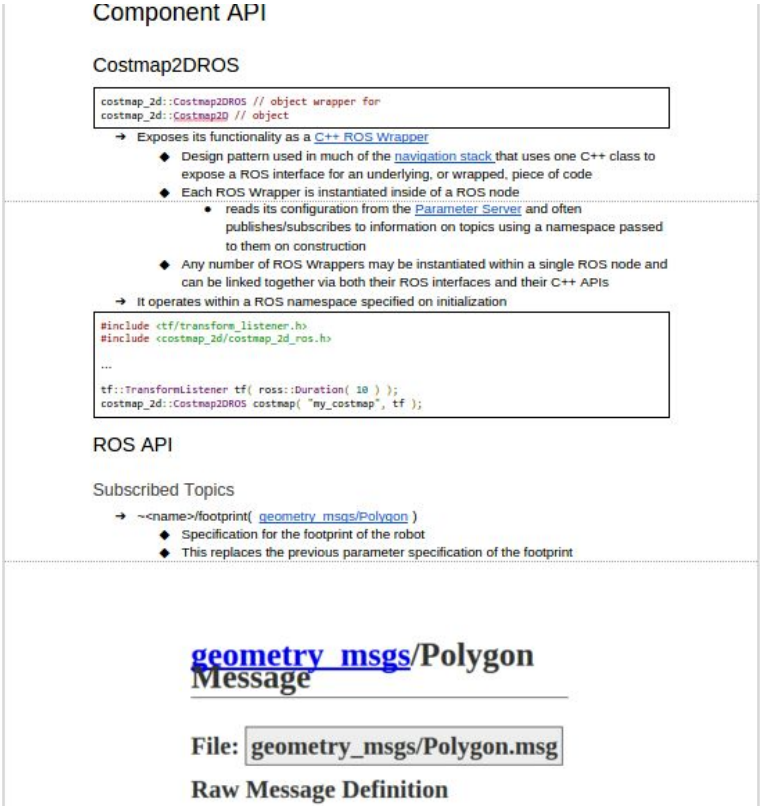

Figure 1) Example of research

The research that I did was done in order to determine which is the best way to set up our navigation stack. Originally, we were looking at using just the navigation stack along with ardrone_autonomy in order to get our position control working.

We ended up able to get tum_ardrone working on the AR.Drone. This is a package written by a student in Germany for his Master's thesis. The package uses Extended Kalman Filtering and optionally PTAM in order to increase the level of control on the AR.Drone. This is the package that we used in our demo.

The package tum_ardrone is very useful for the AR.Drone, but we will need to be able to translate the progress that he made to the Iris+, which is the drone we will be using for the Spring Validation Experiment. In order to facilitate this transition, I began writing code for the navigation stack for the Iris+.

Figure 2 shows a simple piece of code that publishes a target pose (x, y), which can be modified to be provided by the user.

```
#include <ros/ros.h>
#include <geometry_msgs/Pose.h>
#include <iostream>

int main( int argc, char **argv ){
    ros::init( argc, argv, "targetPosePublisher");

    ros::NodeHandle n;
    ros::Publisher targetPosePub = n.advertise<geometry_msgs::Pose>( "targetPose", 1 );
    geometry_msgs::Pose targetPose;

    targetPose.position.x = 2;
    targetPose.position.y = 1;
    targetPosePub.publish( targetPose );

    ros::spin();
    return 0;
}
```

Figure 2) Simple Target Pose Publisher

This simple piece of code just publishes a geometry_msgs::Pose with x and y coordinates. This (x, y) pose could be used to publish a static target pose for the UAV to reach.

In the systems engineering realm, I reworked the functional requirements and the functional architecture.

For the functional requirements, I revamped many of the performance requirements to better fit the direction our project has taken since the functional requirements were originally written.

Table 1 shows the revamped functional requirements.

Table 1) Mandatory Functional Requirements

Mandatory Functional Requirements
MF1. Locate Oil/Gas wellhead infrastructure with known heading in 50 m^2 area ● Changed: Area shrunk due to testing constraints
MF2. Autonomously maneuver to wellhead within 1 hour ● Changed: New performance metric of time deemed to be more valuable
M43. Positively ID as correct wellhead with 90% confidence
MF5. Rigidly dock in 5 DOF ● Changed: 5 DOF more relevant to quadcopters
MF6. Provide status feedback to user of current state at 0.1 Hz

For the functional architecture, I worked to expand some of its blocks into more informative ways. I also added more inputs and outputs (information, material, and energy). Figure 3 shows the original functional architecture and Figure 4 shows the updated functional architecture.

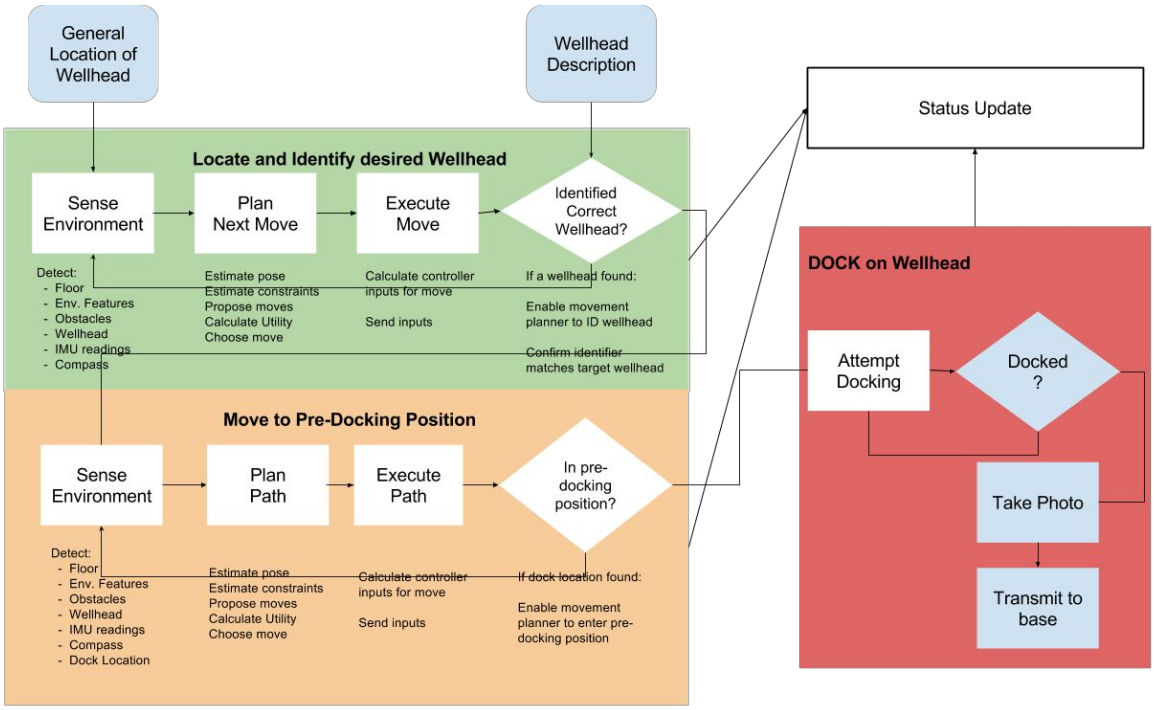

Figure 3) Original Functional Architecture

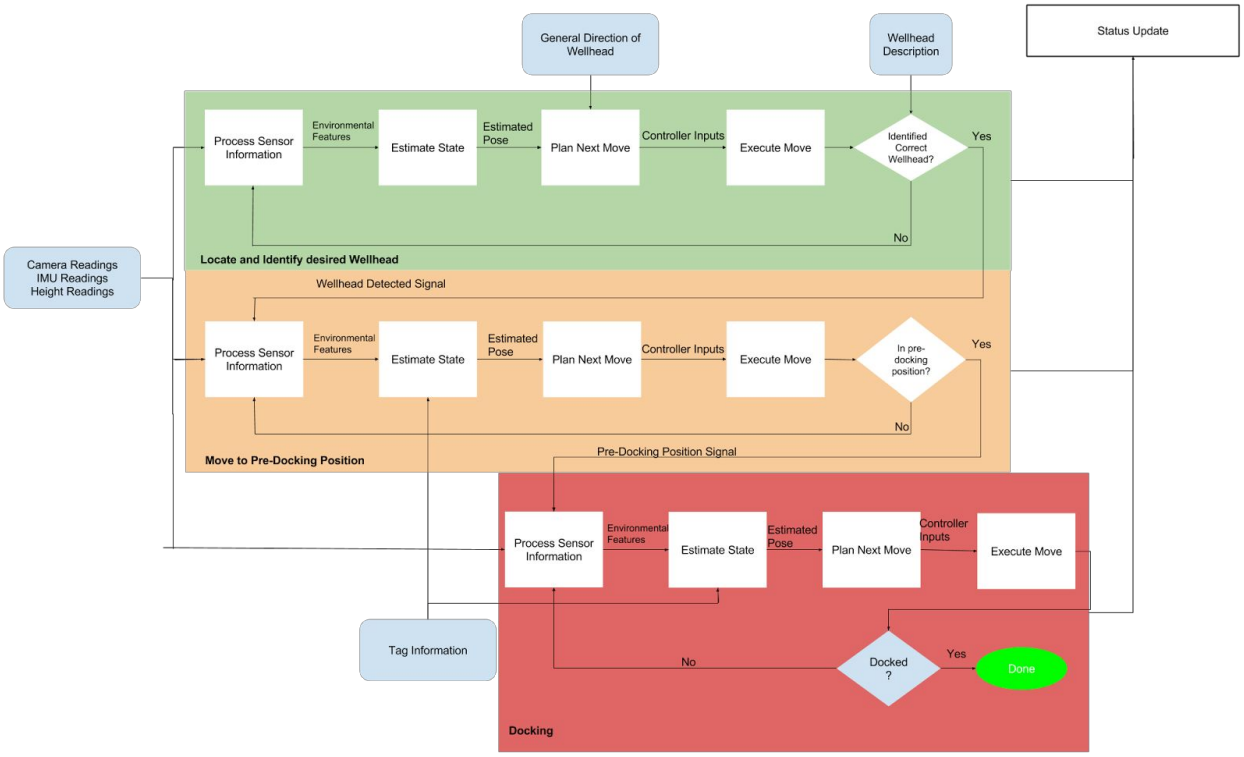

Figure 4) Updated Functional Architecture

One of the main updates are the system, are the expanded blocks in the functional units: Locate and identify desired wellhead, move to pre-docking position, and dock on wellhead. The sense environment mode was expanded to include process sensor information and estimate state. I also included the information that would be provided to each block.

The next main improvement is that I added information that would be inputted to by the user to the system. This is important, because it provides the main distinction between the functional blocks. Each block gets sensor information, but locate and identify desired wellhead is the only block that needs information about the wellhead, while move to pre-docking position and dock to wellhead requires information about the tags we are using and the dock itself.

These two main improvements are very important to the design of our system, and the functional architectures now show this.

I also worked on the PCB schematic, board, and BOM. Figure 5 shows the the final PCB schematic.

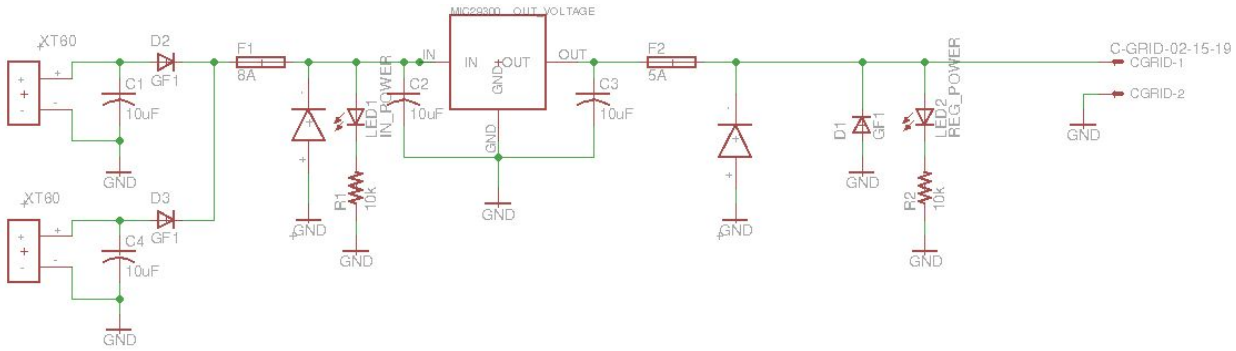

Figure 5) Final PDB Schematic

The schematic has made numerous changes over the course of two weeks. I changed the design to include two inputs from the battery in order to facilitate a hot swap. These include diodes and capacitors to protect the system while hot swapping.

I needed to create a part for the XT-60 connector, because there was nothing in libraries.

I also needed to improve the power protection coming from the battery itself. In general, I needed to change some of the values in the schematic in order to meet some of the parts that are available. For the incoming, I needed to use an 8A fuse and 16V TVS diode. For the outgoing from the voltage regulator, I added a 5A fuse and 5V TVS diode which is not exactly the value that I assumed from the beginning. This was done to facilitate actual parts that I could buy for a reasonable price, but the values are sufficient for the design.

I also redesigned the PDS board, which is shown in Figure 6.

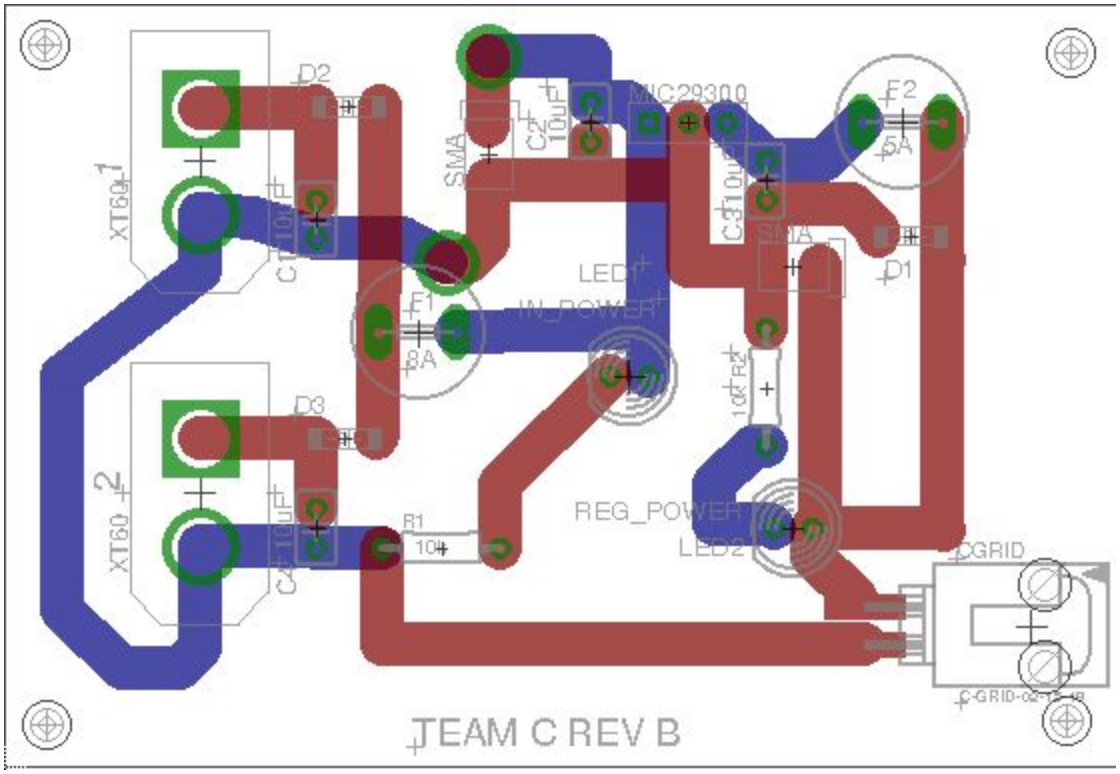

Figure 6) Final PDB Board

In this design of the board, I added more through hole components, because their pads are easier to link up with large track widths. I added larger track widths this time, upon recommendation in order to reduce the heat from the high current and voltage regulator. The board is bigger this time, but the design is more efficient for our purposes.

I also finalized the bill of materials as shown in table 2.

Table 2) Bill of Materials for PCB

Part Type	Value	Part Designators	Quantity	Cost Per Part	Manufacturer	Manufacturer Part #	Supplier Name	Supplier Part #
Through-Hole LED	RED	LED1	3	$0.35	China Young Sun LED Technology Co., LTD.	YSL-R531R3D-D2	sparkfun	COM-09590
Through-Hole LED	GREEN	LED2	3	$0.50	China Young Sun LED Technology Co., LTD.	YSL-R531K3D-D2	sparkfun	COM-09592
Through-Hole Resistor	10k Ohms	R1, R2	6	$0.10	Stackpole Electronics Inc.	CF14JT10K0	Digi-Key	CF14JT10K0CT-ND
Tantalum Capacitor	10uF	C1, C2, C3, C4	12	$0.99	Kemet	T356E106K016AT	Digi-Key	399-3638-ND
Diode	DO-214AC(SMA)	D1, D2, D3	9	$0.48	Diodes Incorporated	SBRT5A50SA-13	Digi-Key	SBRT5A50SA-13DICT-ND
TVS Diode	16V	TVS1	3	$0.45	Littlefuse Inc	P4SMA16A	Digi-Key	P4SMA16ACT-ND
TVS Diode	7.5V	TVS2	3	$0.47	Littlefuse Inc	SMAJ7.5CA	Digi-Key	SMAJ7.5CALFCT-ND
Fuse	8A	F1	3	$0.89	Littlefuse Inc	37418000000	Digi-Key	F5490CT-ND
Fuse	5A	F2	3	$0.70	Littlefuse Inc	37215000001	Digi-Key	WK4263CT-ND
Connector	XT60	XT601, XT602	4	$1.50	sparkfun	PRT-10474	sparkfun	PRT-10474
Connector - Male	CGRID SL	CGRID	3	$0.26	Molex, LLC	50579402	Digi-Key	WM2900-ND
Connector - Female	CGRID SL	CGRID	3	$0.88	Molex, LLC	705430001	Digi-Key	WM4800-ND
Conn Terminal - Female	22-24 AWG	CGRID	3	$0.13	Molex, LLC	16020086	Digi-Key	WM2510CT-ND
Voltage Regulator	5V	MIC29300	3	$3.70	Microchip Technology	MIC29300-5.0WU	Digi-Key	576-1122-ND
Barrel Connector	5.5mmx2.1mm 5V DC	CGRID	3	$3.95	JacobsParts	B00QJ9VMI0	amazon	B00QJ9VMI0

Researching this took quite a bit of time, but Digi-Key and sparkfun were able to provide me with most of my parts. I ordered at least three of everything as recommended in order to protect against parts failure.

Challenges

The main challenge that I faced individually was in the ordering of parts. It was quite a challenge to ensure that every part that I designed and specced could be found somewhere from a manufacturer at a reasonable price. It took quite awhile to find and verify everything.

Another major challenge was researching the navigation packages. It was a lot of work to understand the fundamental nature of everything while also evaluating its merit without directly being able to test it without being able to implement it.

As a group, the main challenge was in deciding what we should do for this sprint. Our requirements had become more focussed to the Fall Validation Experiment, and so we decided that much of the work that we were tracking was not useful for the Fall Validation Experiment. In doing so, we needed to redefine our tasks at a high level before we could decide on our low-level tasks.

Teamwork

This sprint, the group broke up into two subgroups. Me and Erik focussed on the controls of the AR.Drone, while Job and Rohan focussed on getting the hardware and software setup of the Iris+.

We also all worked together on the PDR as a team. We broke up into sections, got our individual work done, and then came together to get our finished product. We worked well as a team and got everything done.

After the PDR, we got much more focused in our preparation for the Fall Validation Experiment. We came together and redefined our higher level tasks. Table 3 shows the redefined high level tasks.

Table 3) New High Level Tasks

High Level Tasks
Open-loop ARDrone Control: Demonstrate takeoff, move, land at push of ROS button
Fall AR.Drone Position X,Y Movement Demo
Hardware and ROS Setup on Iris+
Prototype of dock: Demonstrate one proof of concept, one actual prototype

We were able to finish the first two up to this point.

Future Plans

During the next few weeks, I will be working on extending the package tum_ardrone to be able to work more autonomously. Right now we can only send simple commands in sequence. I want to work on it to be able to run an entire path from a ROS node while also looking for a tag on the ground. This will be something we can extend to the Iris+. This is work for next semester, but it is not part of our Fall Validation Experiment.

Erik will be leading the charge on the Systems Engineering Presentation #3. He will also be running extensive tests and making the Fall Validation Experiment more reliable.

Rohan will be working on finishing the Iris+ hardware and software infrastructure to the level needed for the Fall Validation Experiment.

Job will be working on getting the dock prototype parts ordered and manufactured.

Resources

[1] AR Drone Autonomy
[2] costmap_2d
[3] tum_ardrone
[4] navigation stack