FlySense

"Augmented Reality Assisted FPV navigation for aerial vehicles"

CRITICAL DESIGN REVIEW REPORT

Team C
Shivang Baveja
Nicholas Crispie
João Fonseca Reis
Harikrishnan Suresh
Sai Nihar Tadichetty

Sponsor: Near Earth Autonomy

14th December 2017

Abstract

This document includes details on Critical Design Review of the project undertaken by Team C under the MRSD Project 1 course conducted during Fall 2017. MRSD Project course is a core element of the MRSD program at Carnegie Mellon University, and allows students to work in teams towards a common goal of developing a robotic system which can be used in an identified problem scenario. This document summarizes the activities carried out by Team C – FlySense during the Fall semester to develop their Augmented Reality based assistive technology aiding aerial navigation.

The report begins with a basic description of the project, highlighting the need for the assistive system, followed by a use case that clearly depicts how the system will be used. A list of system level requirements, both performance and non-functional are presented. The approach is graphically represented in functional and cyber physical architecture.

A detailed description of the current system status is also provided, starting with design requirements that were targeted during the fall semester, the underlying implementation to fulfill each of those requirements and how they were evaluated in the Fall Validation Experiment. The report then shifts to the project management section, comprising of the Work Breakdown Structure, schedule for spring semester highlighting the major milestones for each progress review and the final Spring Validation Experiment tests. The project management section details the budget and an update on risk management. The report concludes with the key takeaways from the fall semester and an outline of key activities for the spring.

Table of Contents

1. Project Description

Helicopter pilots have one of the toughest jobs in the world. Their jobs are usually task and sensory saturated, with limited ability to process new information and many different controls to be used in an instant. However, there aren't many aids for helicopter pilots that present useful information in a relevant way. The U.S. military has invested millions of dollars in state-of-the-art headsets for conveying all sorts of information to fighter pilots in real-time, but nothing close to that technology has been introduced in the commercial domain given the current price point and the focus on assisting firing and targeting systems.

Helicopter pilots face difficulties in different phases of flight and mission types. Some of these are low-altitude flights, landing in tight spaces with fixed structures and navigation in low-visibility scenarios. Out of the listed flight stages, one of the most critical is a flight at an altitude below 200ft AGL (Above Ground Level) where, unlike commercial airplanes, there are no autonomous piloting features in place to aid with the landing. Helicopter pilots resort to their instruments, but above all look for visual landmarks to understand their environment and judge how far they are from obstacles. This can be even more difficult when flying in unfamiliar environments, like in areas where the landscape is monotonous (e.g. desert, or a grass field) or in situations where it's hard to judge obstacles that can cause a crash (e.g. a pole near the tail rotor).

Through this project, we aim to develop a pilot assistance system using Augmented Reality, that gives the pilots enhanced situational awareness in the least intrusive way. With our FlySense system, the pilot will be better equipped to handle the difficult flight scenarios mentioned above as he will rely on the visual and audio warnings informing him about the possibility of collisions. FlySense will offer a high level of assistance through mapping of surrounding obstacles, feasible trajectory to the destination avoiding those obstacles and low-level autonomy to override bad decisions by the pilot.

2. Use Case

Lori is an EMS helicopter pilot operating out of the University of Pittsburgh medical center in Oakland. She's an experienced pilot who has been doing this job for over 10 years, doing hundreds of flights every year to rural areas of Western Pennsylvania bringing patients to get treatment at the world-renowned UPMC Hospital. On this day, she arrives for her shift bright and early at 6 am ready to go at a moment's notice. As soon as she's grabbed a cup of coffee, an alert comes in from the dispatcher. "We have multiple severe injuries in a multiple vehicle collision on highway 30 near Clinton. I'm sending in the GPS coordinates now, we need an EMS helicopter for the victims as soon as possible!"

Lori grabs her gear and heads out to the helicopter. Upon reaching the helicopter, she does a quick walk around and proceeds to the cockpit for his preflight checks. She grabs her FlySense visor and turns it on. Within seconds the headset boots up. After a 10 second calibration procedure, a couple options pop up. With a quick verbal command, Lori selects a trip planning view and enters in the GPS coordinates on the flight computer.

Figure 1: EMS helicopter pilot at the ready

As she goes through her checklist, the FlySense headset gives pointers on the current state of the helicopter. Today, the helicopter appears to be in fully working condition: it just came back from its regular maintenance and there are no problems reported. All told, the assistive system integrated into the helicopter allowed Lori to complete her complete checklist in about half the time it would take here without it, even with her years of experience.

As she starts to take off, a member of the grounds crew sets off her obstacle alarms as he runs across the helipad. She couldn't even see him herself from her vantage point, so it was a good thing she had full coverage visually and with sound warnings from the Bird's eye view, which automatically popped up in the takeoff sequence.

Once Lori ascends to 200 feet, she engages the autopilot. This makes her job a lot easier, but her FlySense display is still active on "Heads Up Display" view. It gives her constant updates on her attitude, altitude, and a view of the horizon, even as she goes through a couple low hanging fog banks. It also shows her location of flying vehicles around her and what path to take to reach the destination all within the HUD mode. The flight was going smoothly until she reached the destination and had to find a good landing spot (**Figure 2**).

The accident she is responding too has swarms of EMS vehicles around the crash, and a set of power lines and multiple trees are surrounding the area, making it a little tricky to come in. Lori tells the computer "Guide me in the open area between the police car and the fire truck." Within moments, a guided trajectory comes up in her view. As she descends, she toggles briefly to the Bird's eye view to check her distance to the power lines, but those are only yellow, without any sound warnings, so she knows she is clear but to stay careful.

Figure 2: Lori trying to land on at a congested highway accident scene

As soon as she touches down, her medical crew springs to action and gets 2 patients aboard the helicopter in minutes. They need to make it back to UPMC as soon as they can since both patients have lost a lot of blood and they only have limited medical resources to cope with that on the helicopter. Lori guides the helicopter backwards outside of the area of trees and power lines, relying heavily on her bird's eye view to safely extricate her helicopter from the tricky situation, even as the wind starts picking up dramatically. Once safely up and preventing a dangerous crash, Lori guides the helicopter and the patients back to the medical center.

As Lori gets closer to the medical center, there is much more air traffic in the area. She gets warnings and recommended adjustments to her route through the map and path planning interface, with vectors around the display showing the relative positions of other aircraft nearby using ADSB signaling. Lori adjusts the trajectory, slowing down and hovering to allow another helicopter clear out of her approach path at the hospital. It seems like she's not the only one taking patients to the hospital for treatment today. Now with a clear path, she can make her approach smoothly and safely, getting the patients directly to the helipad, where medics rush out to give patients the needed treatment.

3. System Level Requirements

To accomplish the design, development and ultimately deployment of the system, the team is following a systems engineering approach. The system requirements were defined at the beginning of the project after careful analysis, research and deliberations among the team and stakeholders. All the effort of the team is directed towards fulfilling these requirements.

The requirements presented during the Preliminary Design Review(PDR) have been updated based on the feedback received from various stakeholders and observed system's performance during the Fall Validation Experiment. Specifically, some of the requirements have been made more clear and specific in terms of what the pilot is supposed to see and how pilot performance improves with the FlySense system over a non-aided pilot. A couple of requirements have been changed from Desired to the Mandatory category. None of the requirements have been deleted.

Each requirement is mapped to a subsystem (Aerial or User). Also, the requirements marked with an asterisk (*) are the ones modified or added and justification for these modifications is given in the adjacent column.

3.1. Mandatory performance requirements

Subsystem		Description	Justification
Aerial	M.P.1	Receive, and process point cloud data from one Velodyne VLP-16	
User	M.P.2*	Recognize 5 voice commands with an accuracy of 90% without noise and 70% with noise	Accuracy numbers increased and an attempt to make it work in noisy environment
Aerial	M.P.3	Detect obstacles in the flight envelope projected 5 seconds into future	
Aerial	M.P.4	Detect obstacles of size greater than 2m x 2m located at distances less than 10m	

Aerial	M.P.5*	Generate Bird's eye view image in vehicle frame at a rate of at least 10Hz	Frame rate decreased from 25Hz to 10Hz, due to communication limitations
Aerial	M.P.6*	Color obstacles in bird's eye view (red, yellow, green) based on pilot inputs and time to impact. Red corresponds to the lowest time to impact, then yellow and then green.	No major change, few details added. This requirement was earlier written as "Segment flight envelope into safe and unsafe".
Aerial	M.P.7*	Recommend feasible trajectory to goal maintaining clearance of at least 1m from all obstacles, and display in AR interface	Added requirement based on inputs from stakeholders. It was in the Desired category earlier.
Aerial	M.P.8*	Override pilot commands to stop the aerial system at least 1m before the obstacle	Added requirement based on inputs from stakeholders. It was in the Desired category earlier.
User	M.P.9*	Render all modes on the AR interface at refresh rate of at least 10Hz	Frame rate decreased from 25Hz to 10Hz, due to communication bandwidth considerations.
User	M.P.10	Generate binary audio, left or right based on the obstacle with latency less than 1 sec	Added details

3.2. Mandatory non-functional requirements

Subsystem		**Description**	**Justification**
User	M.N.1*	Easily set up (within 1 minute) by a single operator	Removed the requirement for the system to be able to integrate with NEA flight system. This is done keeping in mind time constraints.
User	M.N.2	Feel natural to the pilot, i.e. Project images at focal distance up to 20 meters	
User	M.N.3	Wearable like normal glass	
User	M.N.4	Comfortable to wear headwear for long periods of time, i.e. should weigh less than 1 pound.	
User	M.N.5	Displays information in a clear and simple manner.	
User	M.N.6	Be non-intrusive to the pilot, i.e. pilot should be able to see through the projected images.	
User	M.N.7	Be non-distracting for the pilot, i.e. pilot should be able to engage or dis-engage the system as and when	

		desired and with simple voice commands	
Aerial	M.N.8	Solution hardware is more affordable than available solutions (Cost below 5000 USD)	

3.3. Desired performance requirements

Subsystem		Description	Justification
User	D.P.1*	Voice commands personalized to 3 users	New requirement. Added to increase the accuracy of voice command recognition
Aerial	D.P.2*	Override pilot commands to maneuver around obstacle maintaining radial clearance of at least 2m	New requirement. Helicopter pilots are interested in override features to increase safety.
User	D.P.3*	First Person View (FPV) video overlay on the AR interface at frame rate greater than 10Hz	New requirement. Added to be able to validate other requirements. This would allow the pilot to fly with and without the FlySense system.
Aerial	D.P.4*	Segment obstacles into 2 categories (Trees or building)	New requirement. This was suggested by one of the stakeholders, to reduce cognitive load on the pilot.

3.4. Desired non-functional requirements

Subsystem		Description	Justification
User	D.N.1*	Easily customizable to include more features and widgets	New requirement. Not aiming for spring.
User	D.N.2*	Ability to integrate with flight simulators to train pilots	New requirement. Not aiming for spring.

4. Functional Architecture

The updated system requirements were used to update the functional architecture, which is shown below in **Figure 3**. The complete system operation is depicted in a block diagram capturing functions and overall flow of information. Here, we have divided the system into 3 stages which are happening continuously and concurrently. These are:

a. Pilot/flight:
   i. Pilot voice commands: These are received directly and are used as a way for the pilot to interface with the system. As pilots are saturated with switches and controls, voice commands provide an intuitive way to interact with the system.
   ii. Flight system: It provides sensor data from IMU, GPS and LIDAR and the pilot commands. It also provides a way to override pilot, if an obstacle is in the immediate path of the flight system.

b. Process/Plan:
   i. Received sensor data and video and publish.
   ii. The sensor data is also used to detect obstacles in the flight path.
   iii. The state of the vehicle and detected obstacles are used to find a feasible path.
   iv. The obstacles and feasible trajectory is incorporated as a bird's eye view image
   v. Override pilot commands if obstacle in the immediate path of flight system.
c. Output/Convey:
   i. The sensor information and video are rendered in the User system
   ii. The detected obstacles are used to generate sound warnings
   iii. The feasible trajectory and feasible path is rendered as bird's eye view

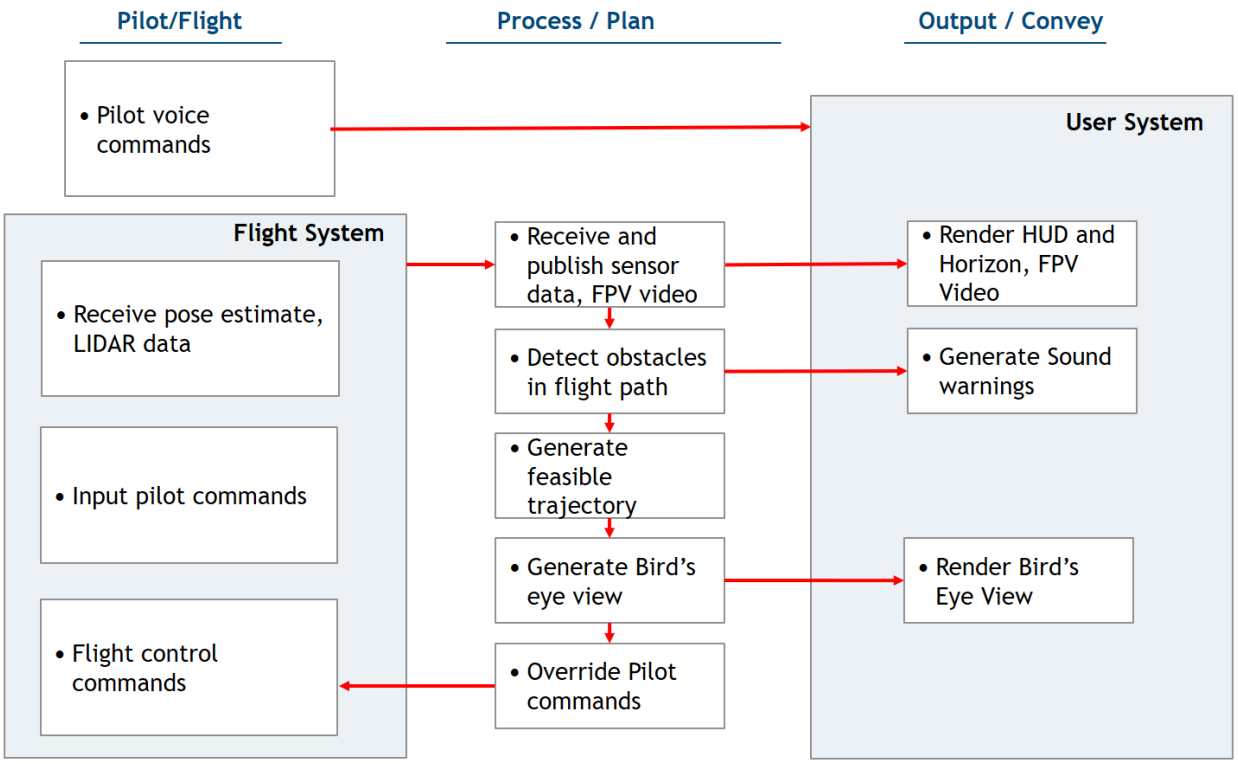


Figure 3: Functional Architecture

5. Cyber-Physical Architecture

The functional architecture was used to update cyber physical architecture which is shown below in **Figure 4**. The cyber-physical architecture delineates the functions among different subsystems and goes into details of implementation on a higher level. It also explains the decisions taken based on the trade-studies to identify components, algorithms, etc.

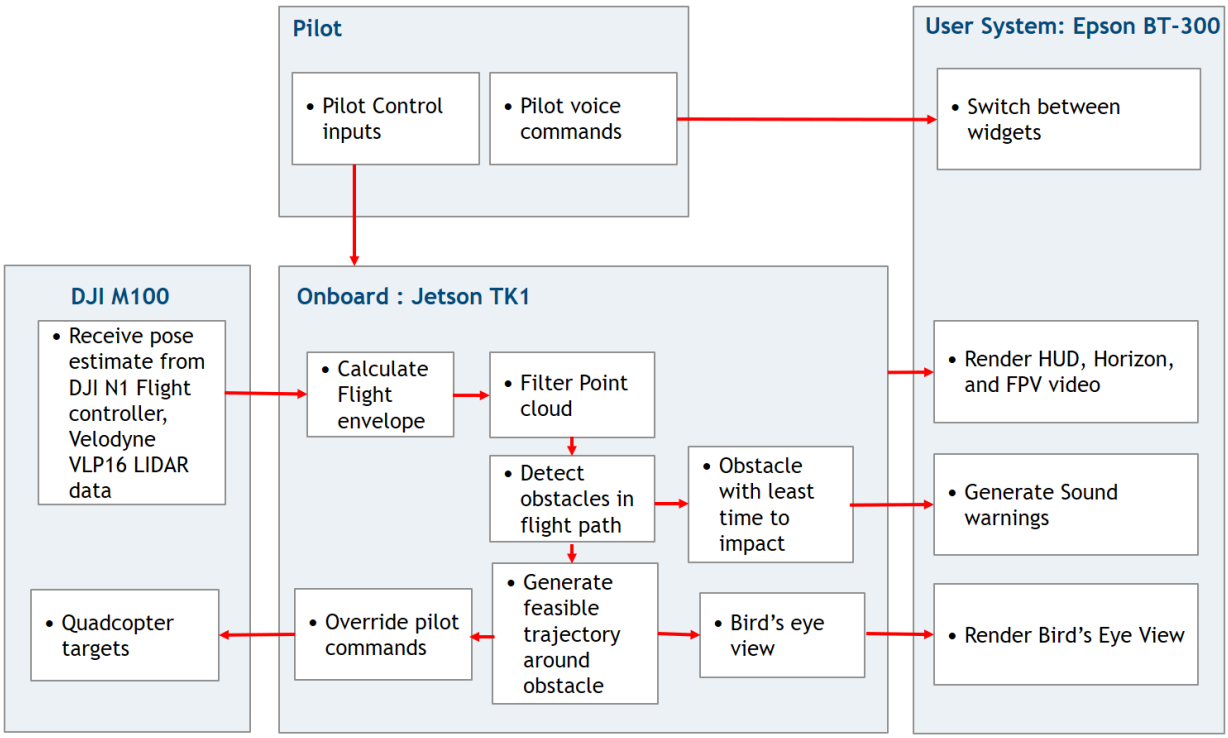


Figure 4: Cyber-physical Architecture

Our system has 2 subsystems (Aerial and User). The aerial subsystem is further divided into 3 key components. These are DJI M100 quadcopter, Jetson TK1 computer, Pilot. Each of these component/subsystems is described below:

a. Pilot: Pilot is responsible for two main functions.
   i. Flying the quadcopter
   ii. Giving the voice commands to switch between available assistance modes in the AR headset
b. DJI M100 Quadcopter: This is the flight system selected for capability demonstration:
   i. To publish the pose estimates, LIDAR data as ROS messages.
   ii. Receive commands to override pilot inputs
c. Onboard computer Jetson TK1:
   i. Calculate Flight Envelope: The flight envelope is calculated from the received pose estimate and pilot inputs. This is the addressable area around aircraft where aircraft can reach in 5 seconds. This does not include sudden malfunction/crash.
   ii. Filter Point Cloud: The flight envelope calculated is used to get rid of the extra point cloud data. This is done to reduce required onboard processing.
   iii. Detect obstacles: The filtered point cloud is then used to identify the obstacles in the flight path.
   iv. The obstacle with least time to impact: Among the obstacles detected, obstacles with least time to impact is calculated and used to generate sound warnings.

v. Generate feasible trajectory: The obstacles detected are used to generate a feasible trajectory.
vi. Bird's eye view: The obstacles detected, and feasible trajectory are combined to generate a bird's eye view image.
vii. Override pilot commands: The pilot commands are modified if the detected obstacles are in the immediate path of the flight system.

d. User System: Epson BT300 AR headset has been selected after conducting trade studies.
i. Switch between widgets: Pilot voice commands are recognized by the AR headset to switch between widgets (assistive modes).
ii. Render HUD, Horizon and FPV video: The sensor information and video received from the onboard computer are rendered.
iii. Generate sound warnings: The sound warnings are given to the pilot as beeps.
iv. Render Bird's eye view: The bird's eye view generated by the onboard computer is rendered.

6. Current System Status – Target Requirements

For the fall semester, a subset of the system requirements was targeted. These are shown in **Figure 5**. The aim was to show the two subsystems working together as one unit with the all the basic features deployed. These basic features were also the most critical one in the FlySense system, as these were picked by helicopter pilots based on their flying experience. All the requirements presented during the Preliminary design review were met by the system. The quadcopter was mounted on a rolling cart for Fall semester. This was done to reduce the burden of flight testing and to allow the team to setup the complete software framework without worrying about integration and testing related problems.

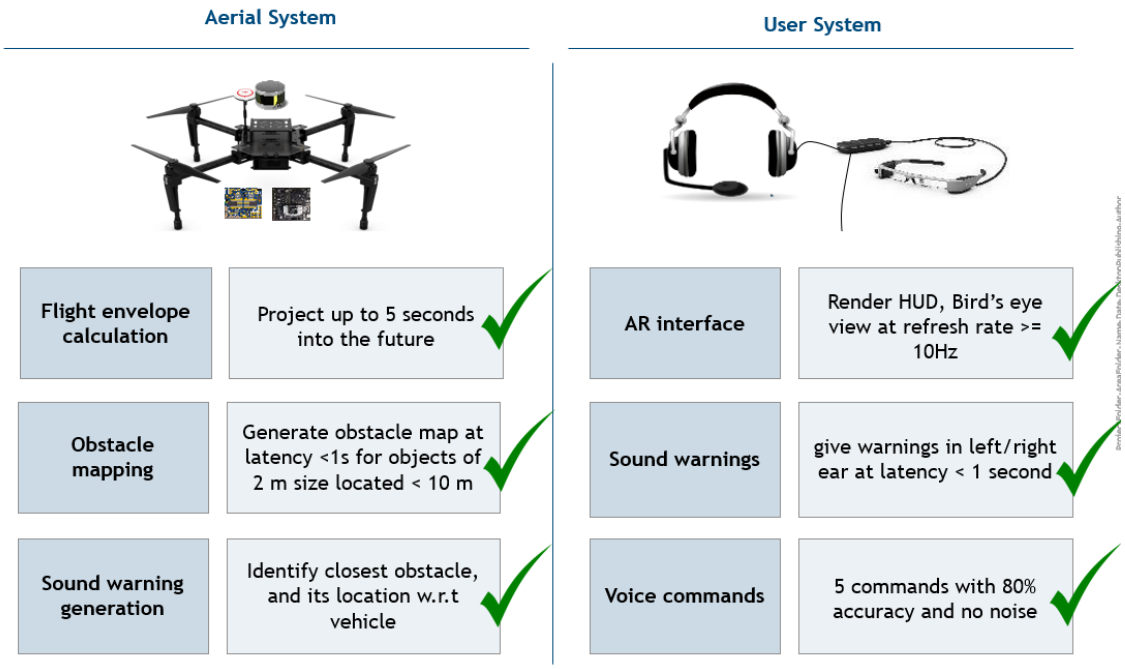


Figure 5: Targeted System Requirements (Fall)

7. Current System Status - Subsystem Descriptions

Our final system consists of 3 major subsystems as shown in **Figure 5**. These are:

- Aerial subsystem – A DJI Matrice 100 mounted with Velodyne VLP-16, FPV camera, onboard computer Jetson TK1 and our custom power distribution board.
- User subsystem – The Augmented Reality headset Epson BT 300 and another audio headset for sound warnings and voice commands.
- Communication subsystem (COTS)– A 5 GHz Dual Radio Base Station with MIMO technology. No development required in this sub-system.

Figure 6: System diagram

The implementation and status of each of these subsystems is described now keeping in mind the requirements outlined in section 6. Also, the green ticks in the images indicate that the requirement has been completely met, yellow indicates that the work is in progress.

7.1. Aerial subsystem

The Aerial subsystem encompasses multiple software and hardware components.

The software architecture for the aerial subsystem is shown in **Figure 6.** The software architecture has been developed keeping in mind the testing and safety constraints associated with flying an aerial system. We have setup a way to test the system by feeding in data from a recorded bag file. This allowed us to record the data once in our mission scenario and use it continuously to fix minor bugs which would be difficult to fix if we were to move on to live operation directly. It also helped us as a team as an individual developer were able to test software components with minimum dependencies.

The architecture is shown in **Figure 6** also includes the software components that will be completed during the spring semester, namely, pilot override and recommending the feasible path to the pilot. The software architecture would have to updated to incorporate a way to do flight simulation to test the pilot override feature offline in a safe virtual environment.

The key hardware components that were developed were:

1. Power Distribution Board PCB
2. Rolling cart setup for Fall Validation Experiment

The key elements of the aerial subsystem are described in detail now:

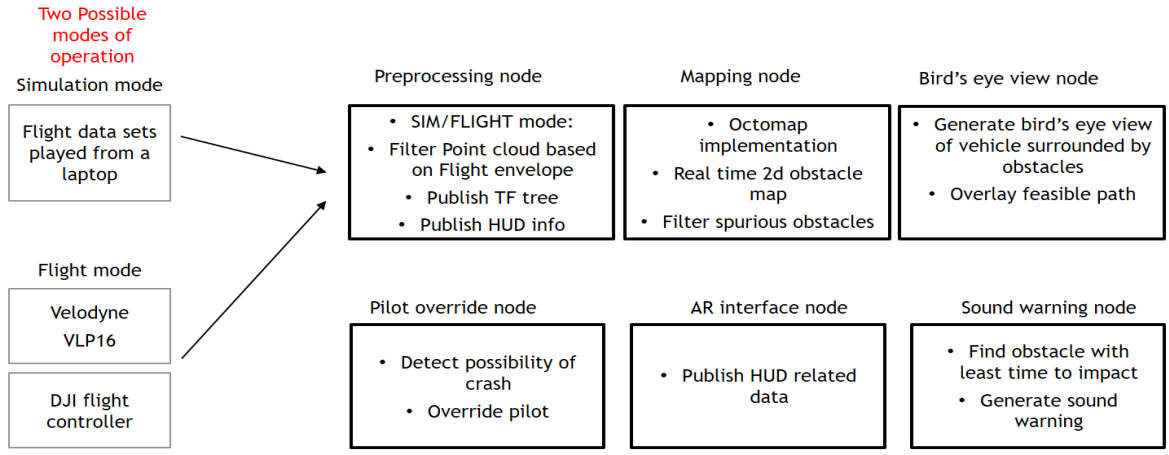


Figure 7: Aerial Subsystem Software Architecture

7.1.1. Onboard computer interface to DJI Matrice 100

The DJI Matrice 100 provides a way to interface to external computers via a serial port. This interface was utilized to setup a robust communication interface between the quadcopter and the Jetson TK1. The goal for the fall semester was to be able to publish a tf tree which is required by other software routines working with LIDAR point cloud data. We also wanted to publish basic telemetry information to the AR headset so that it can be shown live in the Headset as part of the Heads-up-display(HUD) widget. **Figure 7** shows an overview and status of this interface.

The attitude information received from the DJI quadcopter followed different frame conventions compared to what was required for the AR headset. The sensor information was therefore converted from ENU(East-North-Up) frame to NED(North-East-Down) frame and then published to the AR headset.

All the requirements mentioned above were satisfied. Pilot override is a new requirement and will be addressed in the spring semester.

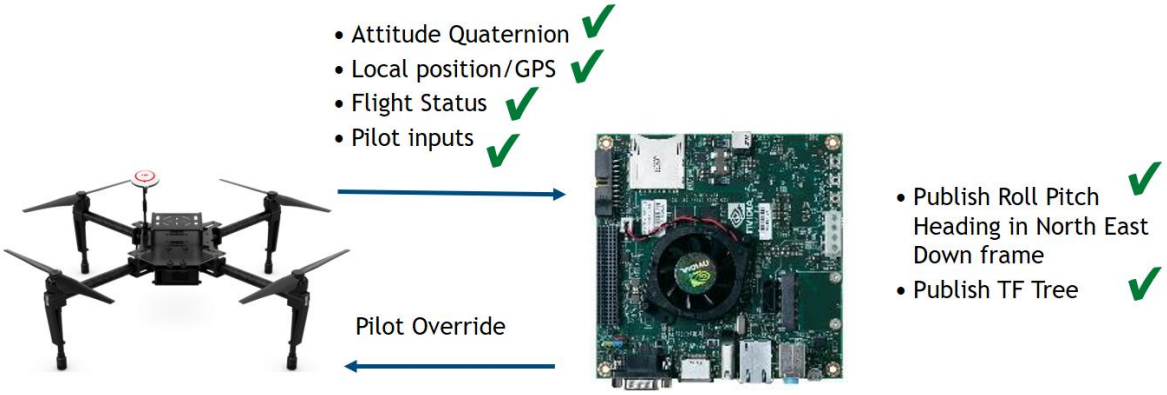


Figure 8: Onboard computer overview

7.1.2. Filter point cloud based on flight envelope

To reduce the processing load on the onboard computer, the point cloud received from the Velodyne LIDAR is filtered based on the flight envelope of the aircraft. We have defined flight envelope as the addressable area surrounding the vehicle where aircraft can reach in next 5 seconds.

The flight envelope at a given time instant is estimated based on the current state of the vehicle and the pilot inputs. Assuming a vehicle starting from rest, the addressable region changes from a circular envelope to an elliptical envelope. Taking into consideration a safety clearance and ease of implementation, the flight envelope has been approximated as a cuboidal envelope surrounding the ellipse. The flight envelope algorithm is depicted in **Figure 8.**

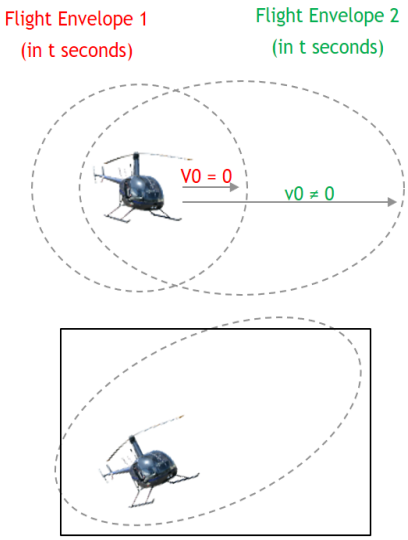


Figure 9: Dynamic flight envelope calculation diagram

The above algorithm was implemented using a cropbox filter from point cloud library. Another cropbox filter was implemented to filter the points captured from parts of the vehicle frame. **Figure 9** shows raw point cloud and filtered point cloud based on flight envelope.

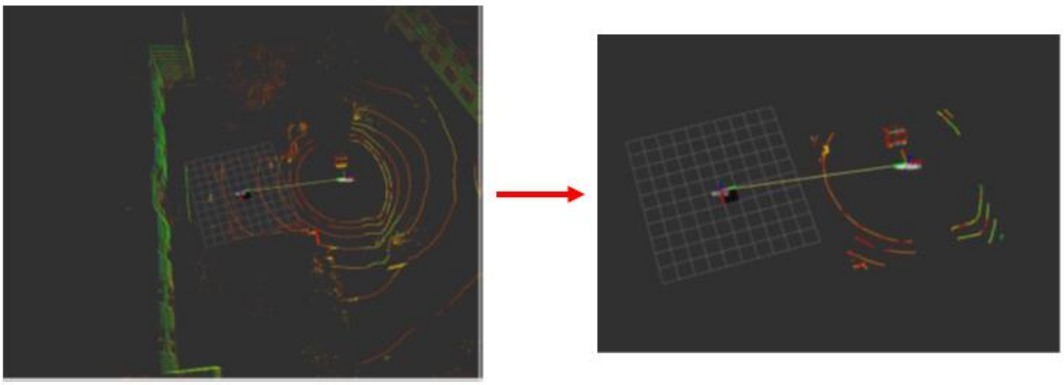


Figure 10: PCL processing before and after

7.1.3. Obstacle mapping and Bird's Eye View generation

The obstacle mapping and bird's eye view generation pipeline is shown in **Figure 10**. The raw point cloud received from Velodyne LIDAR is first filtered based on the flight envelope. This is described in the above section. Then the density of the filtered point cloud is further reduced using voxel down-sampling. The down-sampled point cloud is used for 3 purposes:

a. Sound warning generation Generates the sound warnings by finding the obstacles with least time to impact. It uses the Velodyne heightmap package to convert 3d points into 2d points and then iterates through every point to find if the point is in vehicle path and calculate the time to impact to that point. The point with the least time to impact is used to generate sound warnings. The sound warnings are given in the form of repeated beeps. The frequency of beep is high if time to impact is less, thus conveying the severity of the situation. We have divided the time to impact into 3 levels such that change in beeping frequency is distinguishable.
b. Bird's eye view: The buffered point cloud is transformed from the vehicle frame to the world frame. It is then buffered 3 times and transformed back to the vehicle frame. The top-view of the points is taken as an image.
c. Trajectory planner: The buffered point cloud is fed into the octomap package which finds and keeps tracks of obstacle locations in a 2d map. The goals here is to be able to generate a feasible trajectory which will be recommended to the pilot.

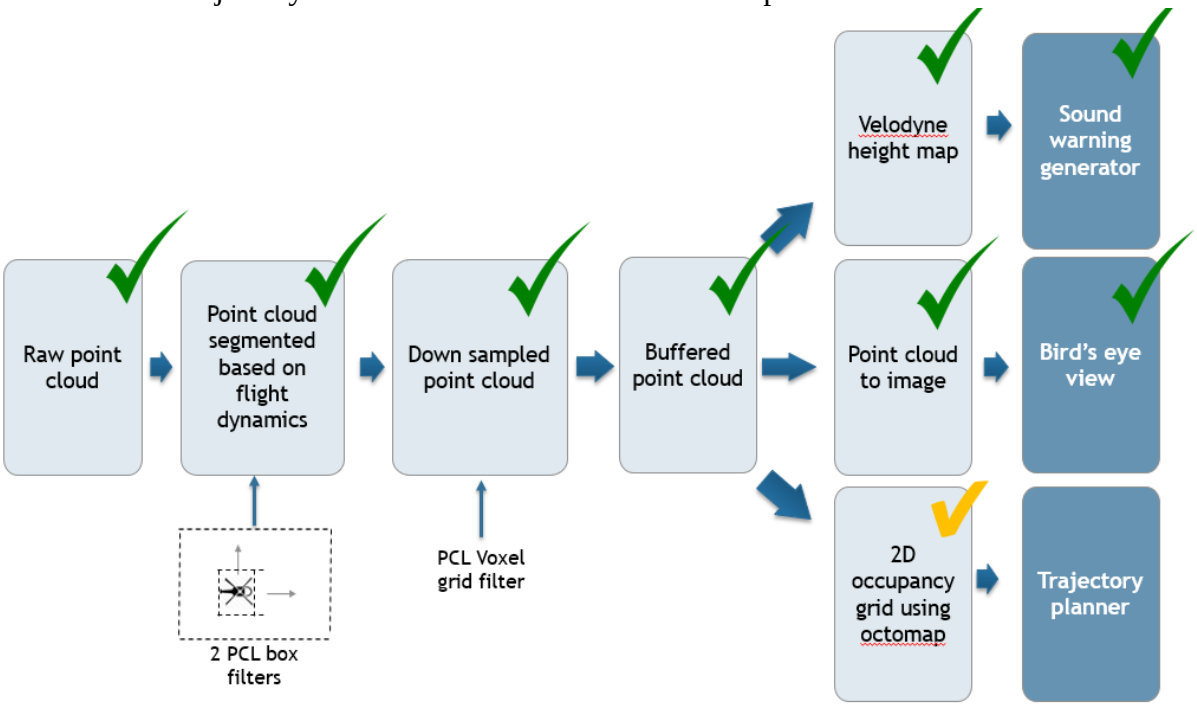


Figure 11: Obstacle Mapping pipeline

Apart from the trajectory planner, everything else was demonstrated during the FVE. The trajectory planner and obstacle occupancy map generation are still in progress and are required for our spring semester goals.

The status of the whole pipeline is shown in **Figure 11**. The figure shows the heightmap which is being used for the sound warning generation, the bird's eye view image with a red blob (aircraft) in the middle and the obstacle occupancy map generated using octomap.

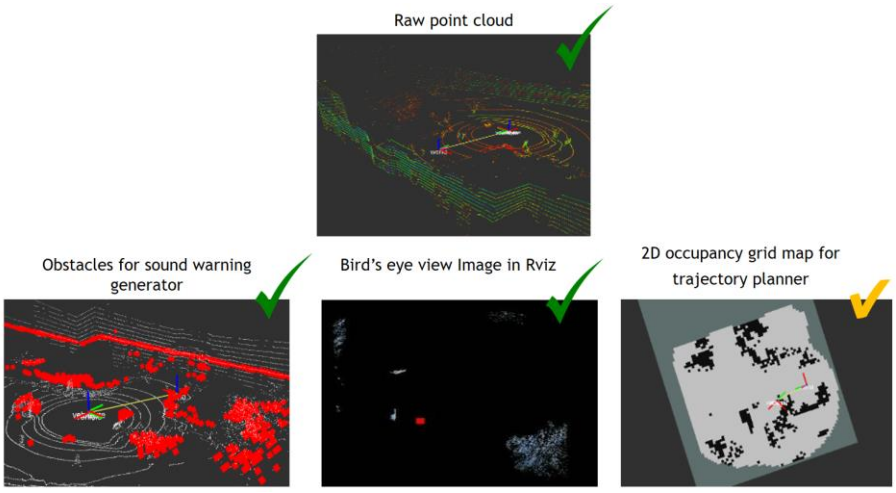


Figure 12: Mapping pipeline status

7.2. User subsystem

The user subsystem consists of the Epson BT 300 AR headset which displays information to the pilot without obstructing his view and an audio headset that provides sound warnings and receives voice commands.

The user interface consists of three widgets (assistive modes) namely, Home screen, Heads-up-display, Bird's eye view. The system gets activated when Pilot gives the voice command "Computer". It then shows the available assistive modes which can be switched during the flight. The pilot can activate the Home screen by saying "Alpha", HUD widget by saying "Bravo" and the bird's eye view by saying "Charlie". By default, the system starts in Home screen.

There are 5 key elements in the User interface which are explained below:

1. Voice commands: To keep our user interface easy-to-use and intuitive, it was important to implement voice commands. This allows pilots to stay busy with flying without getting worried about an extra set of buttons. We have implemented 4 voice commands, which work reliably in the noise-less environment. The goal is to increase the accuracy of voice command recognition in a noisy environment by personalizing it to the pilot's voice. This work will be done in the spring semester.

2. Home screen: As listed in the performance requirements, we have an uncluttered home screen for the user interface with buttons to the corner without interfering with the pilot's vision.
3. Heads-up-display: The heads-up-display gives only the relevant sensor information to the pilot. The information includes the vehicle's orientation in terms of roll, pitch and heading, ground speed, time to impact to the nearest obstacle
4. Bird's eye view: The bird's eye view accurately depicts the environment around the vehicle. Bird's eye view is capable of showing the finer details of the surrounding environment like trees, buildings, etc.
5. Sound warnings: Based on time to impact and location of the obstacle, sound warnings are generated in either left or right ear. If an obstacle is in left vehicle direction warnings are heard in left ear and if an obstacle is in right of the vehicle direction, warnings are heard in the right ear. This feature has been found to work reliably well on the ground. We will do more tests to check the performance in flight.

Figure 12 shows these widgets as they were demonstrated during the FVE. We were able to complete all the requirements that were set out for the user system for Fall semester.

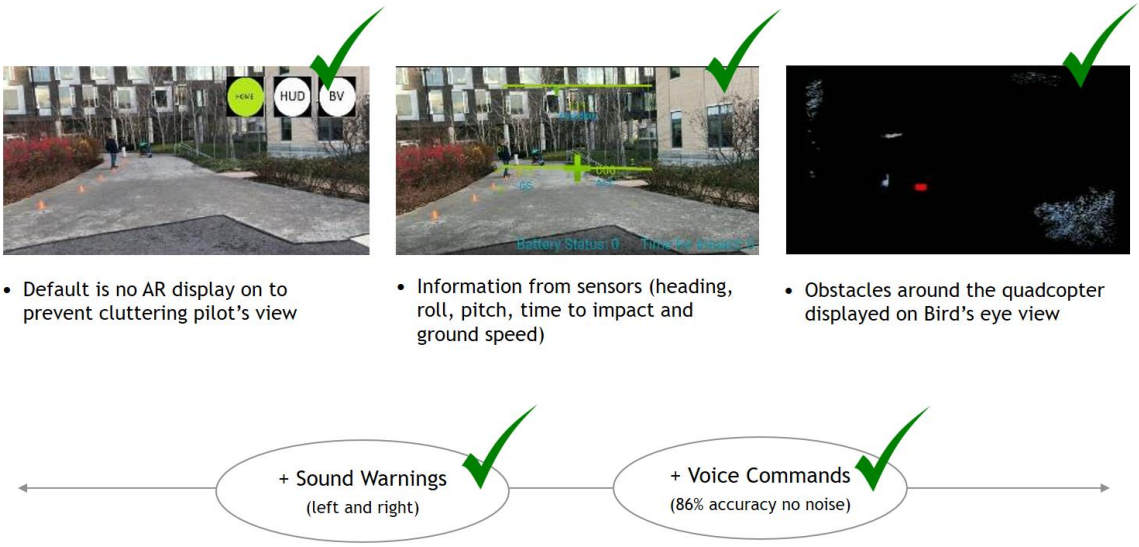


Figure 13: User interface as demonstrated in the Fall Validation Experiment

7.3. Modeling, analysis, and testing

The aerial subsystem and the user subsystem were tested independently and together.

7.3.1. Aerial subsystem

The aerial system was mounted on a cart for the FVE (**Figure 13a**), so the system was modelled to address the issues in a two-dimensional space. The movement of the aerial system provides the AR headset with flight information such as heading, roll, pitch, ground speed and time to impact with the nearest obstacle. It also provides, live view of obstacles as a bird's eye view image and binary sound warnings (Left or Right).

The system was powered by the drone using an onboard power distribution board (**Figure 13b**).

Figure 13a: Mechanical test setup

Figure 14b: Power Distribution board

The tests were conducted outside the NSH building which involved:

1. Pushing the cart around obstacles to check the accuracy of sound warnings and the bird's eye view image
2. Tilt and turn the cart to check if the heading, roll and pitch information is being updated at 10Hz.

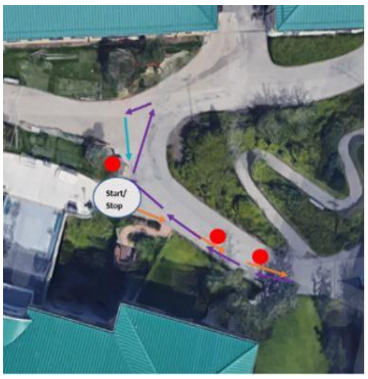


Figure 15: FVE Test Diagram

The map in **Figure 14** shows the path of motion with start/start markers, arrows pointing towards the direction of motion and obstacles in red.

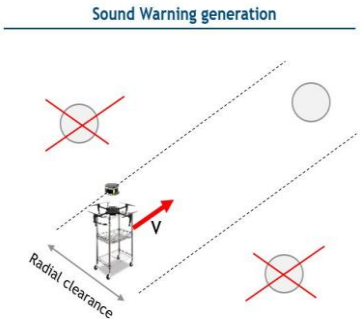


Figure 16: Sound warning generation diagram

The image shown above (**Figure 15**) describes the sound warning generation conditions, that the obstacle should be in the path of the system.

7.3.2. User system

The user system is modelled such that the pilot feels natural interacting with it. In order to achieve the required standards, rigorous testing strategies were employed which helped us in accurately assessing the performance of the system. The team performed tests for 50 iterations in noiseless and noisy environments to check the accuracy of the voice command recognition system.

The system was tested for:

- Recognizing the words and performing suitable action
- Recognizing incorrect word
    - Rejecting Garbage before activation
    - Rejecting Garbage after activation

List of words:

FlySense Dictionary

- Computer: Activation Word
- Alpha: Home Screen Command
- Bravo: HUD Command
- Charlie: Bird's Eye View Command
- Test Garbage
    - Beta, Gamma, Jarvis, Jetson, Delta, Startup (we can test any garbage word)

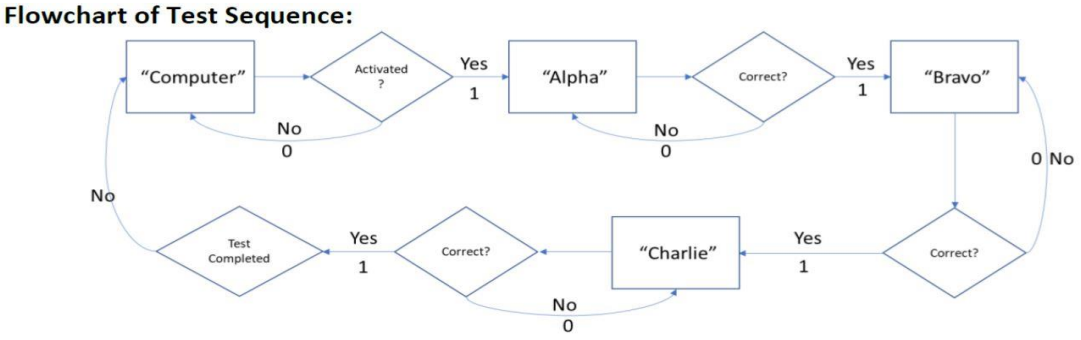


Figure 17: Voice Commands test sequence

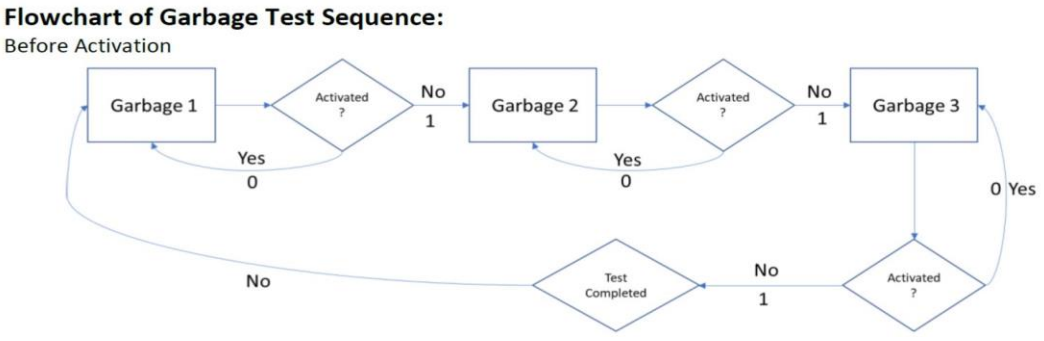


Figure 18: Voice commands test sequence, before activation

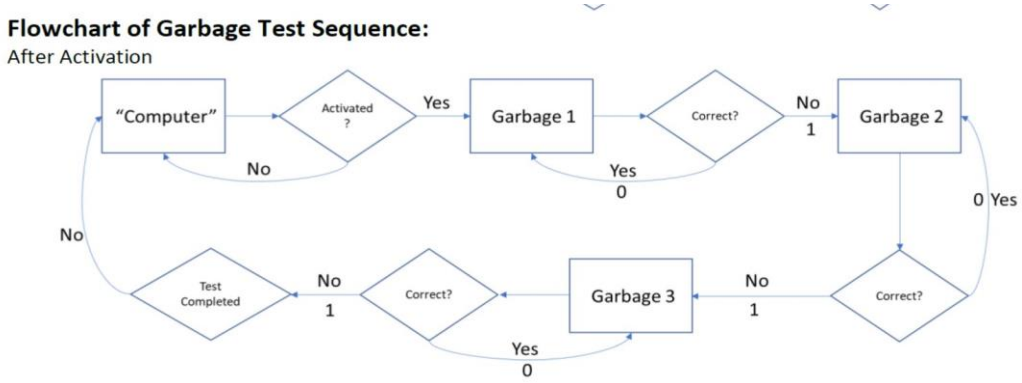


Figure 19: Voice commands test sequence, after activation

Each word is considered a state and only a correct recognition will lead to toggling between the states. Correct toggling between states gives a score of 1 and incorrect toggle or no toggle will give a 0. This directly corresponds to the accuracy of the speech recognition system.

Following the above sequence for 50 iterations gives us the results in **Table 1** and **Table 2.**

Noiseless Environment																			
	FlySense Dictionary				Garbage														
					Before Activation								After Activation						
Sno	Computer	Alpha	Bravo	Charlie	Beta	Gamma	Banana	Jarvis	Delta	Jetson	Startup		Beta	Gamma	Banana	Jarvis	Delta	Jetson	Startup
1	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
2	1	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
3	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
4	0	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
6	1	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
7	0	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
8	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
9	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
10	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
11	0	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
12	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
13	0	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
14	1	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
15	1	1	0	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
16	0	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
17	0	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
18	0	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
19	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
20	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
21	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
22	1	0	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
23	1	1	0	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
24	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
25	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
26	1	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
27	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
28	1	0	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
29	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
30	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
31	1	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
32	1	1	0	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
33	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
34	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
35	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
36	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
37	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
38	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
39	1	0	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
40	1	1	0	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
41	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
42	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
43	1	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
44	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
45	1	0	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
46	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
47	1	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
48	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
49	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
50	1	1	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
Total Score:	**43**	**38**	**46**	**45**	**50**	**50**	**50**	**50**	**50**	**50**	**50**		**0**	**0**	**0**	**0**	**0**	**0**	**0**

Table 1: Results from voice commands tests, no background noise

Results in Noiseless Environment:

• In a noiseless environment, the accuracy of recognizing the activation word is 86%

• The accuracy of recognizing Alpha, Bravo and Charlie is 76%, 92%, and 90% respectively

• The overall mean accuracy is 86%

• Garbage words before “Computer” are completely rejected with an accuracy of 100%

• Garbage words after “Computer” are all accepted with 0% correct results (none are rejected)

Noisy Environment (Simulated Helicopter Noise)																			
	FlySense Dictionary				Garbage														
					Before Activation								After Activation						
Sno	Comput	Alpha	Bravo	Charlie	Beta	Gamma	Banana	Jarvis	Delta	Jetson	Startup		Beta	Gamma	Banana	Jarvis	Delta	Jetson	Startup
1	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
2	0	1	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
3	0	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
5	0	0	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
6	1	1	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
7	0	1	0	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
8	0	0	0	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
9	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
10	0	0	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
11	0	1	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
12	0	1	0	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
13	1	0	0	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
14	0	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
15	0	0	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
16	1	1	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
17	0	1	0	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
18	0	0	0	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
19	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
20	1	0	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
21	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
22	0	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
23	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
24	1	0	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
25	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
26	0	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
27	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
28	1	0	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
29	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
30	0	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
31	1	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
32	1	0	0	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
33	1	1	0	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
34	0	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
35	0	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
36	0	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
37	1	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
38	1	1	0	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
39	0	1	0	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
40	0	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
41	0	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
42	0	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
43	1	1	1	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
44	0	1	0	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
45	0	0	0	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
46	1	1	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
47	1	0	0	0	1	1	1	1	1	1	1		0	0	0	0	0	0	0
48	1	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
49	0	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
50	1	0	1	1	1	1	1	1	1	1	1		0	0	0	0	0	0	0
Total Score:	25	29	37	34	50	50	50	50	50	50	50		0	0	0	0	0	0	0

Table 2: Results from voice commands tests, with background noise

Results in Noisy Environment:

• In a noisy environment, the accuracy of recognition goes down drastically

• The accuracy in recognizing the activation word “Computer” is 50%

• The accuracy of recognizing Alpha, Bravo and Charlie is 58%, 74%, and 68% respectively

• The overall mean accuracy is 62.5%

• Garbage words before “Computer” are completely rejected with an accuracy of 100% in static noise, all bets are off in dynamic noise

• Garbage words after “Computer” are all accepted with 0% correct results (none are rejected)

7.3.3. Complete system testing

Static Lidar Test

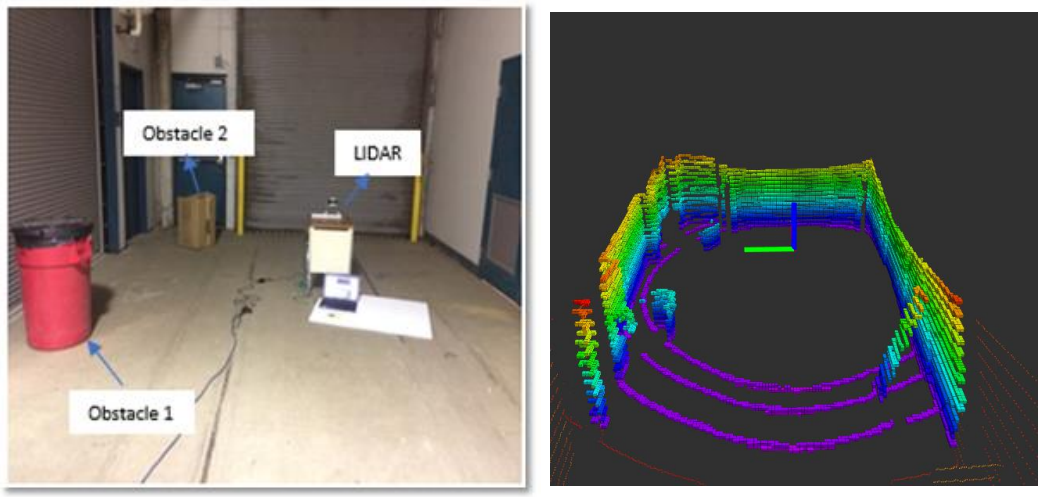


Figure 20: Static Lidar Test setup (left) and results (right)

Moving Lidar Test

Out development included a series of Lidar mapping tests, starting with static Lidar Testing in the lab and progressing to dynamic Lidar tests outside of Newell Simon Hall. Some key takeaways

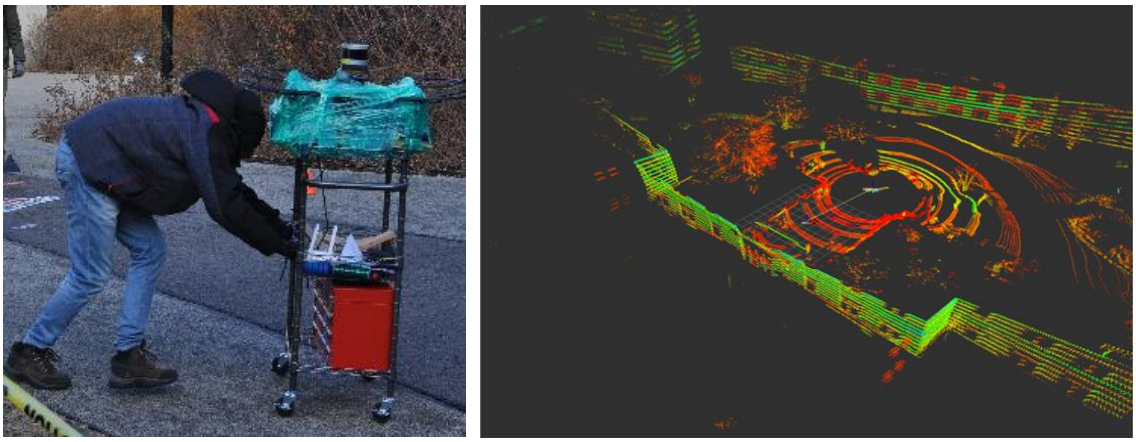


Figure 21: Dynamic Lidar test setup (left) and results (right)

from the initial static Lidar tests were the ground artifacts. Since we were mainly concerned with obstacles surrounding the vehicle sides (at least for these test sequences), we realized that one of our processing steps needed to be a PCL cropbox filter to take out the ground plane and avoid triggering warnings for the ground. In our dynamic testing, we realized that the update rate was crucial: we were getting a significant lag with our initial mapping tests, so we revised the approach and for the Fall validation experiment, implemented a pipeline (described in the subsystem descriptions) that down-sampled enough to reduce the lag to an acceptable level.

The moving lidar test also helped us test the performance of position estimate from DJI M100.

8. Fall Validation Experiment Performance Evaluation

The Fall Validation Experiment was designed such that all the targeted performance requirements and most of the non-functional requirements would be tested. All the tests were aimed to be live demonstrations outside Newell Simon Hall. Unfortunately, the weather played spoilsport on the day of the FVE and we had to change our plans to do a simulation with prerecorded data. The data was processed on the onboard computer and the pilot was able to get both the visual and audio warnings. It was difficult to get the complete experience as the pilot was not aware of what the actual surrounding was, and how our system was assisting him in navigation. Nonetheless, we were fully prepared for a live test on the FVE Encore day, even overcoming the snowfall to demonstrate the capabilities of our system outside Newell Simon Hall, CMU.

A brief description of the major tests conducted during the FVE Encore is given below:

- **Test 1 to evaluate voice commands recognition** - The pilot gives the activation commands along with the toggle word to switch modes on the AR interface. The pilot also gives a series of garbage commands and see that the AR interface does not respond.

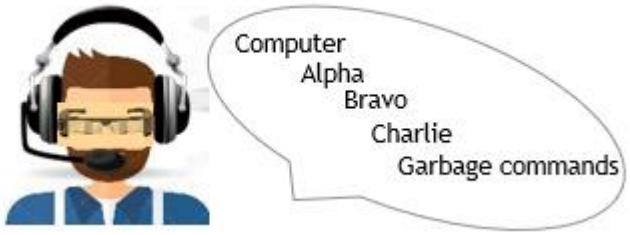


Figure 22: Pilot with voice commands

Result- The system was able to recognize the voice commands with a mean accuracy of 86% in a noiseless environment.

- **Test 2 to evaluate the HUD mode**- The pilot switches to the HUD mode, and sees the telemetry data coming from the vehicle. The operator tilts the flight system changing roll, pitch and heading, and the pilot sees the values update on the HUD.
  Result- The telemetry information was updated on the HUD mode at a refresh rate of 10Hz

- **Test 3 to evaluate the Bird's Eye View**- The pilot switches to the BV mode, and sees the vehicle marked red and only the obstacles around the addressable region. The operator follows a predetermined sequence shown in **Figure 22**, with obstacles appearing on the left and right side at different instances. The vehicle starts at a location greater than 5.5 seconds to impact away from every obstacle, and no warnings are heard. As the vehicle approaches the first obstacle on its left, the sound warnings start in the left ear when time to impact is 5.5 seconds. As the vehicle moves closer, the time to impact decreases and warnings become more frequent. Once the vehicle crosses the obstacle, the warnings stop. The same scenario happens when the vehicle encounters this obstacle (now on its right) during its return course.

- **Result**- All the requirements in this test were successfully completed. The images were rendered at 10Hz, which proved very natural and real time to the pilot. The pilot was able to clearly distinguish through sound the location of the obstacle, and get a better understanding of the environment through the visuals.

Figure 23: Detail of FVE experiment. From left to right: Location and path taken during the test, test as viewed from the ground level, Prof. John Dolan interacting with the user system.

9. Strengths and Weaknesses

The biggest strength of the current FlySense system is the user system. The AR headset is very robust and works with various lighting conditions. The images are very crisp and projected 20m in front of the eye. The visual maps are non-intrusive and provide a clear picture of surroundings.

The sound warnings are very accurate and real time, with the stereo effect working well. Another major advantage is that we have developed a fully integrated system, with the complete software implemented and tested. This puts us in a comfortable position for spring semester when we'll be adapting the same to a flying system while deploying some new features. The software architecture has been designed to incorporate the new features.

There are a few weak points in both the user and aerial system. A major drawback is the existing voice command recognition system which fails in a noisy environment. This limits the system from being deployed in an actual helicopter with background noise. This has motivated us to develop our own voice command recognition network, which can give reasonable accuracy in noisy environments as well. The sound generation code works only in 2 dimensions, which will not work for our aerial system. So, we will be making refinements in our code to incorporate the 3D navigation environment.

Based on the feedback received during FVE, the Bird's Eye View visualization does not convey enough information to identify the type of obstacle. This calls for a need to provide some sort of symbols (For ex: inverted triangle for tree) in the Bird's Eye View to aid the pilots better. Another issue is the drift in the raw point cloud, which affects the quality of the obstacle map and its visuals.

The preprocessing stage of our pipeline does not include point cloud registration, which is another area to work on. The drift in point cloud also causes the occupancy grid map to be very cluttered,

which might cause random errors in the trajectory planner. The update rate of the obstacle map is also slow and must be improved before using it for planning.

10. Project Management

10.1. Work Breakdown Structure

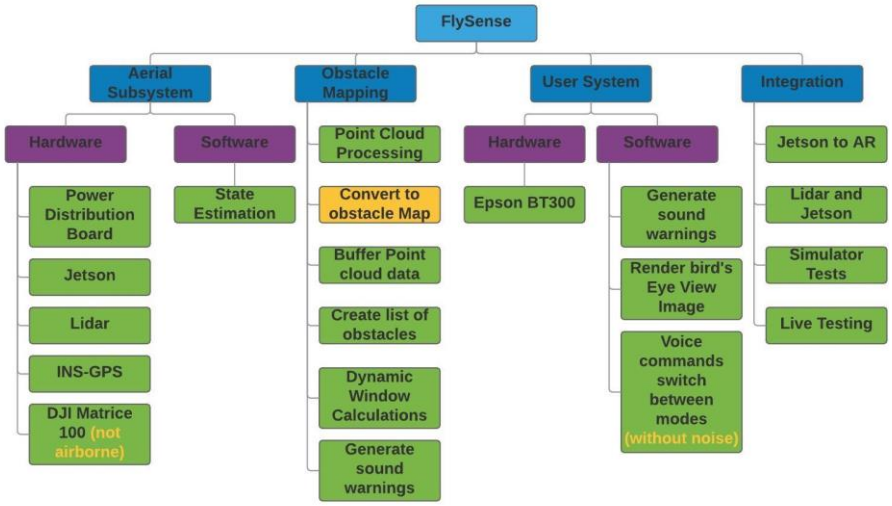


Figure 24: Work breakdown structure for the Fall Semester. Green indicates the work package was completed, while yellow indicates that work is still in progress. The yellow tasks have been remapped to new work packages for the Spring semester.

Work Breakdown Structure: Spring

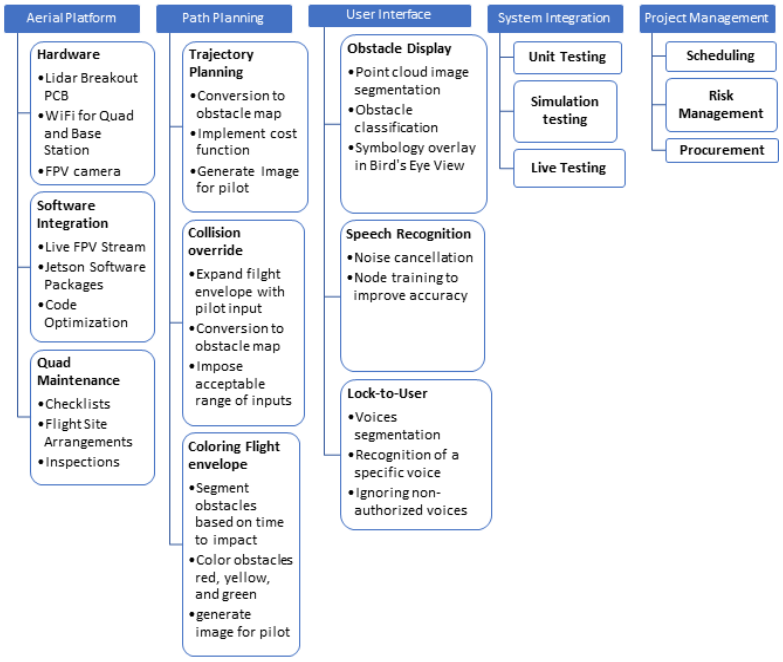
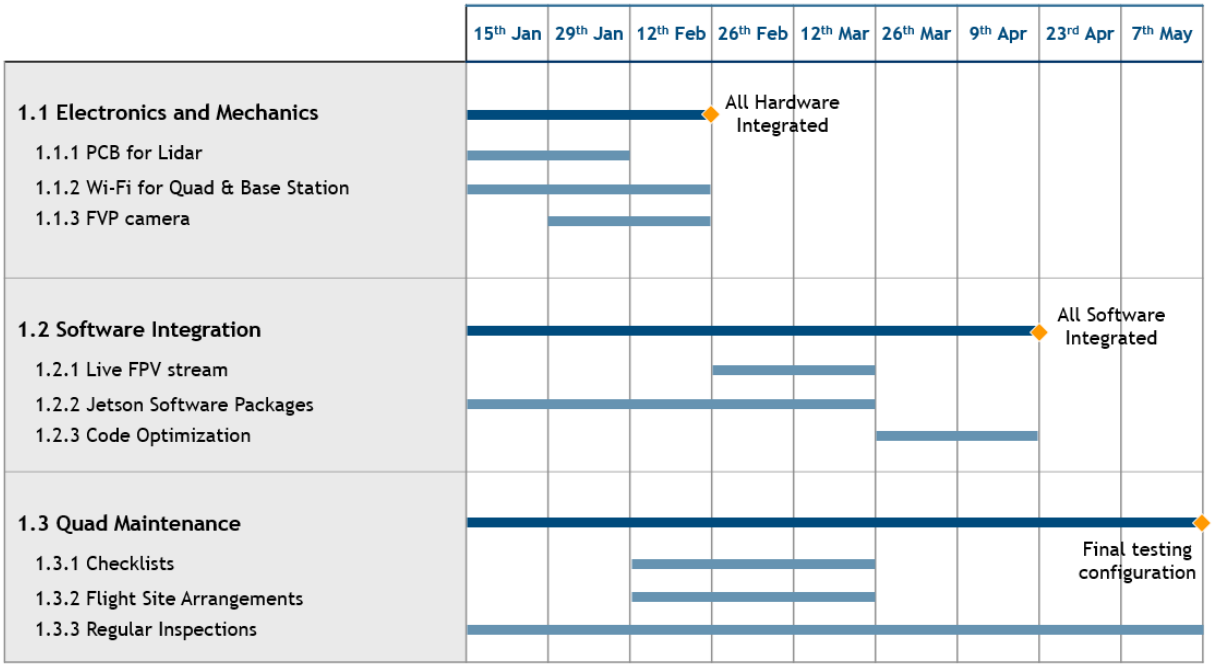
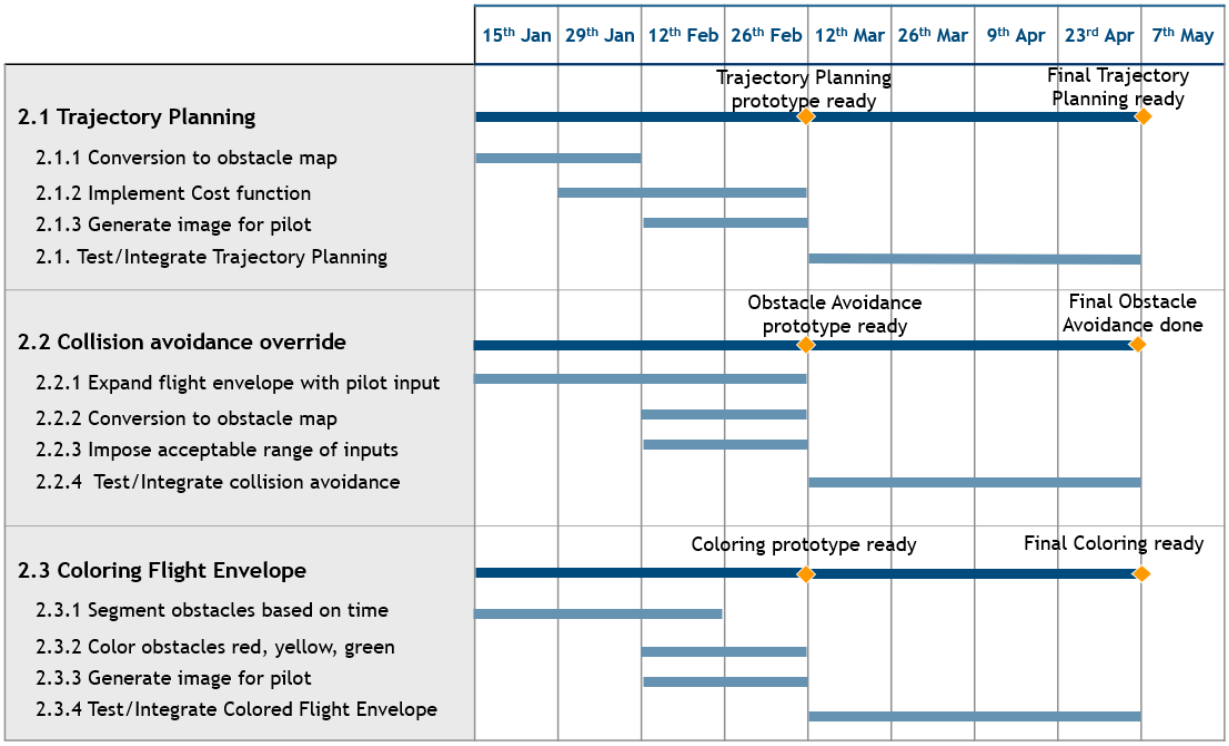


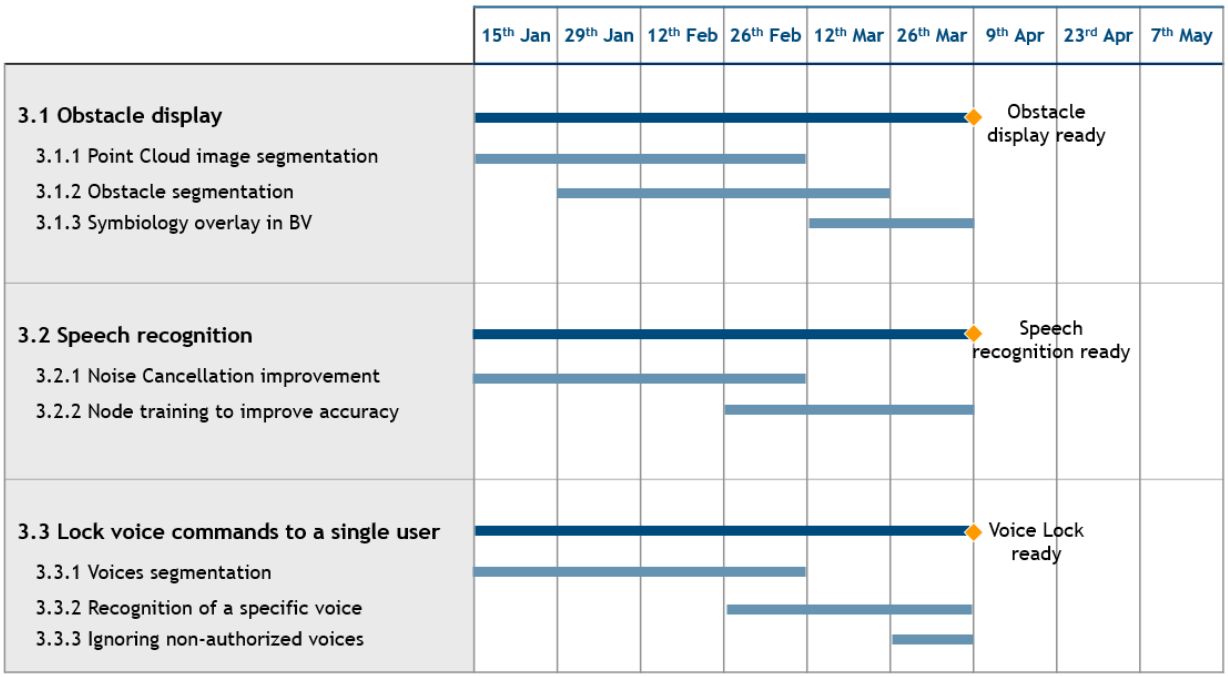
Figure 25: Work Breakdown structure for the Spring semester

10.2. Schedule

Below we have detailed the schedule for the three main work categories we have for the Spring Semester.





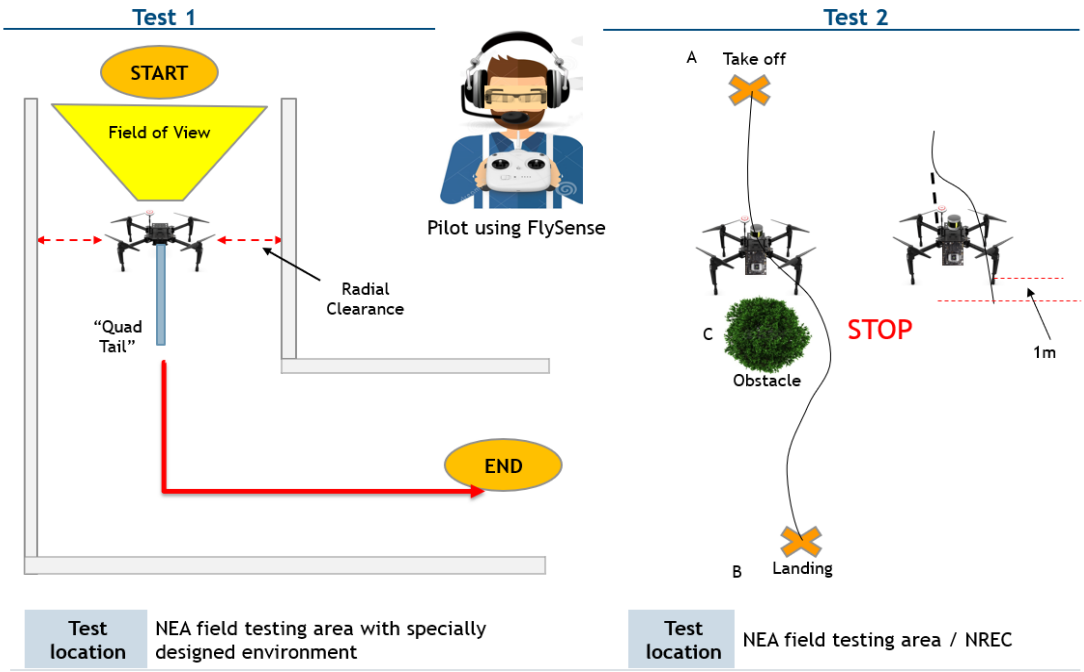


So far, we are on generally on target for the Spring semester. We were able to accomplish all the goals for the Fall experiment. We are a little bit behind in implementing a final mapping solution (we were hoping to have that finalized this semester), but that has become a major priority to get done early in the Spring Semester. We have set major milestones for the middle of February, End of February, the end of March and the end of April to aid us in our development cycle.

10.3. Test plan

Milestone	Desired Functionality	Test Method
PR7	Quad flying with FPV video transmission	Fly quad at NREC Live data transmitted to AR glasses
PR8	Recommended feasible trajectory v1 obstacle avoidance v1	Testing done in simulation
PR9	Quad flying with trajectory generation and obstacle avoidance Personalized Voice command v1	Testing live with aerial platform at NREC Round one of user feedback from focus group
PR10	Trajectory generation v2 Obstacle avoidance v2	Testing live with aerial platform at NREC Round two of user feedback from focus group
PR11	Full System integration with AR	Test at NEA flight testing location Flight testing with AR
PR12	More integration and testing	

Spring Validation Experiment test plan



Test Procedure 1	**Performance Evaluation**
1.Pilot flies the Quadcopter from start to end using only FPV video feed and no FlySense	• Maneuvering time and number of errors (< 1 m away from wall) measured
2.Pilot wears FlySense, gives voice commands in both quiet and noisy environments	• 90% accuracy without noise • 70% accuracy with noise
3.Gives command in RC for Quadcopter to takeoff, switches AR to BV mode	
4.Follows feasible trajectory shown in the AR interface to reach end position	• Trajectory maintains 1m clearance from obstacles • Sound warnings generated in the correct ear • Maneuvering time and number of errors (< 1 m away from wall) reduced by 20%
5.Pilot removes FlySense and gives feedback on the complete system	• Comfort, relevance to reality, extent of assistance

Test Procedure 2	Performance Evaluation
1.Pilot wearing FlySense gives RC command to take off at A	
2.Quadcopter follows the feasible trajectory seen on AR toward B through C	• Trajectory maintains 1m clearance from obstacles
3.Quadcopter reaches closer to C, but still less than 1m	• Obstacle shows color transition (green-yellow-red) based on time to impact • Sound warnings generated in correct ear when time <5.5 seconds
4.Quadcopter diverts from feasible path, tries to get close to the obstacle but stops immediately	• Quadcopter stops at distance less than 1m

10.4. Budget

Type of Budget item	Supplier	Description	Unit cost($)
Borrowed Equipment	NEA	Velodyne VLP16	8,000
Borrowed Equipment	MRSD Lab	DJI Matrice 100	2,847
Confirmed Budget	Amazon	Epson BT 300	799
Confirmed Budget	-	Miscellaneous stuff	656
Confirmed Budget	NVIDIA	Jetson TX2	599
Projected Budget	TBD	Communication system	600
Projected Budget	MRSD Lab	Quadcopter stuff	600
Projected Budget	Amazon	Headset with MIC	300
Projected Budget	Real Flight	Drone flight simulator	100
Projected Budget	E-con systems	FPV camera	250

- Borrowed Equipment (Lidar + DJI M100): $10,847
- Amount spent from MRSD Budget: $2,054
- Projected Budget: $1,850
- Reserve Budget: $ 1,100

10.5. Risk Management

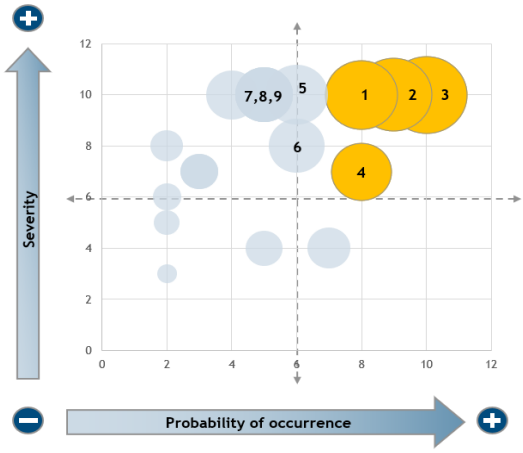


Figure 26: Risk diagram as of Preliminary Design Review

Figure 25 shows the risks that were presented in preliminary design review. The following results came from addressing these risks:

- (1,2,3,4) risks related to mapping implementation (difficulty and performance). We mitigated this by coming up with alternative methods for getting the obstacle display done. We ended up relying on a buffered PCL image to meet our requirements and pushing some of the mapping work to the Spring semester.
- (5) related to delays in the project due to overwork in other classes. We mitigated this by planning on our schedule and supporting one another in common classes, and we were able to complete the tasks as intended.
- (6) related to the robustness of the voice commands. Here we limited our scope to voice commands working only in a noise-free environment and took extensive data to validate so we weren't reliant on a live demo working perfectly in an unknown noise environment
- (7,8,9) risks related to the usability and comfort of the AR headset. We limited the scope of our test to not include wearing the AR headset for a long time but made sure to test the usability consistently throughout the process. We also did a detailed trade study to select the right AR headset early in the development process (and switched approaches midway through after we realized that the Microsoft Holo-lens would not work for our purposes).

We were able to effectively track and mitigate the risk items that we put a high Risk Priority number on but did make some initial miscalculations in evaluating other risks. We did not put effective weight on the communication problems that we ultimately had. However, once it was clear that it could be a major problem, we did a root cause analysis and realized that the DJI controller operating on 2.4 GHz was interfering with our Wi-Fi. To mitigate this risk, we switched to a 5GHz router and had multiple routers available to us for the FVE and development testing prior to that. As a result, the demonstration was smooth and predictable.

As we progress to new challenges, our have been revaluated risks for the Spring Semester, (**Figure 26).**

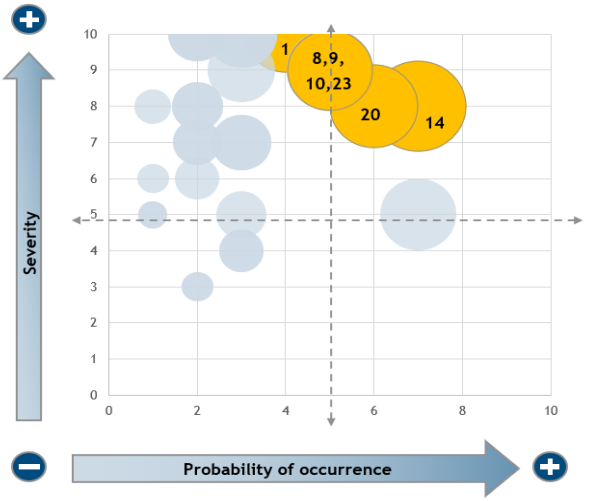


Figure 27: Updated risks

In more detail, the major risks we have identified are:

- (1) Budget constraints: due to the dangerous nature of outdoor testing with drones and expensive hardware, a part failure could cause us to be set back a significant portion of the budget. We are maintaining a reserve of $1000 as a rainy-day fund, but we will be monitoring the budget closely, so we don't have any issues going forward.
- (8) AR Headset usability: AR headsets sometimes give people headaches, and we will be monitoring this in our user tests. Some adjustments we can make are the screen resolution and focal length.
- (9) Data processing constraints: We have a light airborne system, so we are limited to the hardware we can use to power through all the point cloud data we are collecting. These limitations might result in us having a hard time building a detailed enough map, but we will try to mitigate this by down-sampling and segmenting where possible and structuring the codebase to milk the most efficiency out of the hardware we have.
- (10) Drone Communications (Wi-Fi): We had this problem right before FVE, and we anticipate this could rear its head again. The extra challenge here is we need a lightweight system to put on the drone, constraining the solution space. We plan to test multiple solutions early (in January) to come up with the most robust option.
- (14) The weight of the Velodyne Lidar makes flying quad difficult: We don't have a Puck Lite, so we will need to be very careful with our weight budget, and take weight out from other areas. We will test the quad by incrementally adding weight (not the Lidar) and observing the dynamics before putting the most expensive equipment onboard.
- (20) Weather prevents testing: We will schedule multiple test sessions to avoid getting stuck too much and put work into unit testing via simulation.
- (23) Voice commands robustness to noise: We plan to mitigate this by improving our hardware, using multiple microphones, and implementing a learning algorithm to help parse the speech in the noisy environment.

11. Conclusions

Key lessons learned

- Testing a system is much more demanding than testing a single sub-system (e.g. network)
- Designing for a human is substantially different from designing for a robot (e.g. mapping)
- Sometimes the simplest possible solution works well (e.g. direct from LIDAR)
- Time is an extremely scarce resource that needs to be well managed from the beginning
- Cross-functional tasks need to be planned as early as possible to ensure work bandwidth
- Requirement ownership is crucial for success (demand vs “sell them to someone else”)

Key spring activities

- Transfer ground based system to air
- Improve voice commands
- Deploy additional planning features and benchmark safety warning features

12. References

[1] Hornung, Armin, et al. "OctoMap: An efficient probabilistic 3D mapping framework based on octrees." *Autonomous Robots* 34.3 (2013): 189-206.

[2] Asvadi, Alireza, et al. "3D Lidar-based static and moving obstacle detection in driving environments: An approach based on voxels and multi-region ground planes." *Robotics and Autonomous Systems* 83 (2016): 299-311.

[3] Luukkonen, Teppo. "Modelling and control of quadcopter." *Independent research project in applied mathematics, Espoo* (2011).

[4] wiki.ros.org/

[5] pointclouds.org/

[6] medscape.com

[7] nvidia.com

[8] dji.com

[9] velodyne.com