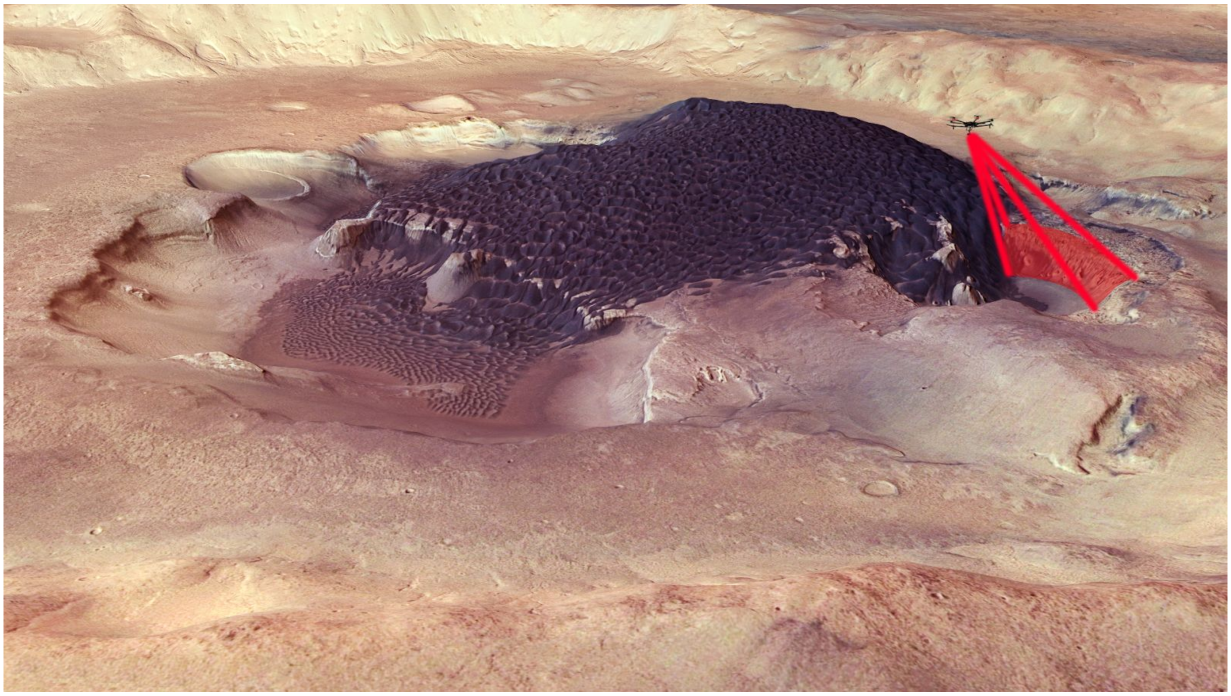


Multimodal Mapping with Aerial Vehicles

MRSD Team B: Arcus

10 May 2017

Team Members:

Clare Cui
Maitreya Naik
Angad Sidhu
Logan Wan

Sponsors:

Wennie Tabib
Nathan Michael

THE ROBOTICS INSTITUTE

1. Abstract

This report details the progress on multimodal mapping project developed during the 2016-2017 academic year. The UAV will be used to test autonomy algorithms that make use of multiple onboard sensors for mapping, localization, and data collection. This application is further explained in the project description and use case sections. The system-level requirements are then defined and accompanied by the functional and cyber-physical architectures that construct the system, fulfilling the requirements. The current system status is explained in detail by focusing on each of the subsystems - software, mechanical, and electrical - which validates specific system requirements given in the spring validation experiment (SVE). Team Arcus has successfully engineered and developed a UAV that generates real-time colorized voxel grid maps with required accuracy in localization. Furthermore, in simulation, autonomous navigation is proved with path planning and online map generation capability. The project management section of this report summarizes the development timeline, budget, and risks. It includes important successes and failures of the team's management techniques. Finally, conclusive remarks are given about the work that has been accomplished and the future work envisioned for this project.

Table of Contents

2. Project Description

NASA is currently in the process of planning an exploratory mission on Mars to probe cave networks for information on their structures and the mineral composition of surface rock layers and sediment. These findings will further inform our scientific understanding of Mars and its suitability as a location for human inhabitation. However, it is extremely costly to send humans to scout and look for desirable landing locations and areas of interest for future scientific missions. What is needed is a UAV (unmanned aerial vehicle) that will generate real-time map data to give scientists and users the critical information necessary to make informed decisions on how to allocate precious resources when on Mars. A major hurdle in the way of full autonomy is the lack of absolute positioning via GPS. The UAV combines LiDAR (light detection and ranging), RGB camera, along with hyperspectral (non-visible light) imaging sensors to create data-rich maps that highlight specific objects of interest fused with spatial geographic information all without the aid of GPS. The final vehicle is designed to fly inside of, discover, and autonomously map fully enclosed, non-uniform, gps-denied environments.

In the scope of the MRSD project, Arcus will be building a remotely piloted UAV that will generate this map information in real-time. It is intended to be used as a rapid prototyping platform for developing navigation and planning algorithms, as well as providing basic software and hardware modularity to accommodate different types of imaging sensors. The platform can navigate by tele-operation or autonomously. The platform and the software provided allow researchers to easily hook into ROS to retrieve point clouds and a textured mesh map describing the environment. Researchers would then develop algorithms which analyze the map and make decisions regarding future behavior. For example, if a branch of a cave seems to tighten in diameter the robot may make the decision to ignore that route to explore because it might be too small to fly through. However, if the entrance to that branch is coated in rich minerals that have not been observed before, the robot might make the decision to traverse down that branch to gather more information. Succinctly, the purpose of the robot is to perform SLAM in a GPS-denied environment and collect structural and RGB color information with low error in both its state estimate and its map.

3. Use Case

Prior to a fully manned mission to Mars, NASA will send out robotic scouting missions to determine both the suitability of habitation and a potential landing spot for a human habitat on the planet. Similar missions are planned for the Moon, where scientists are trying to uncover the actual quantity and location of water around unique geographical landmarks. A mapping UAV will be prepared for the unique requirements of each mission. A moon UAV will likely be propelled by rocket-powered thrusters, whereas the Martian atmosphere would accommodate propeller-driven flight. Sensors such as spectrometers, hyperspectral imagers, and distance ranging equipment would be loaded for the detection of various molecules, water, or precious resources. Mission parameters will be loaded onto the vehicle, such as the general location of interest coordinates and specific features to search for.

Upon delivery as a rocket payload to the terrestrial surface, the UAV will deploy from its base station. A rough map generated from an orbiting observer as well as simulations and calculations will give us a rough estimate of landing location. The UAV will then plan a path from the landing zone to some objective point. While en route, due to communication latency, the UAV will have to make decisions that will optimize its battery life to focus on successfully completing mission objectives. As seen in Figure 1 below, real-time fusion of map structural data with hyperspectral imaging data will allow for the drone to provide a map that would allow the researchers to quickly identify specific features to follow up with higher-resolution data capture. After a predetermined amount of battery consumption, the UAV will return back to the landing site for high-bandwidth data communication back to Earth. Using these high-fidelity data, scientists will then be able to better inform their hypotheses about the characteristics of the celestial body, and much more quickly determine its suitability for human habitation or resource acquisition.

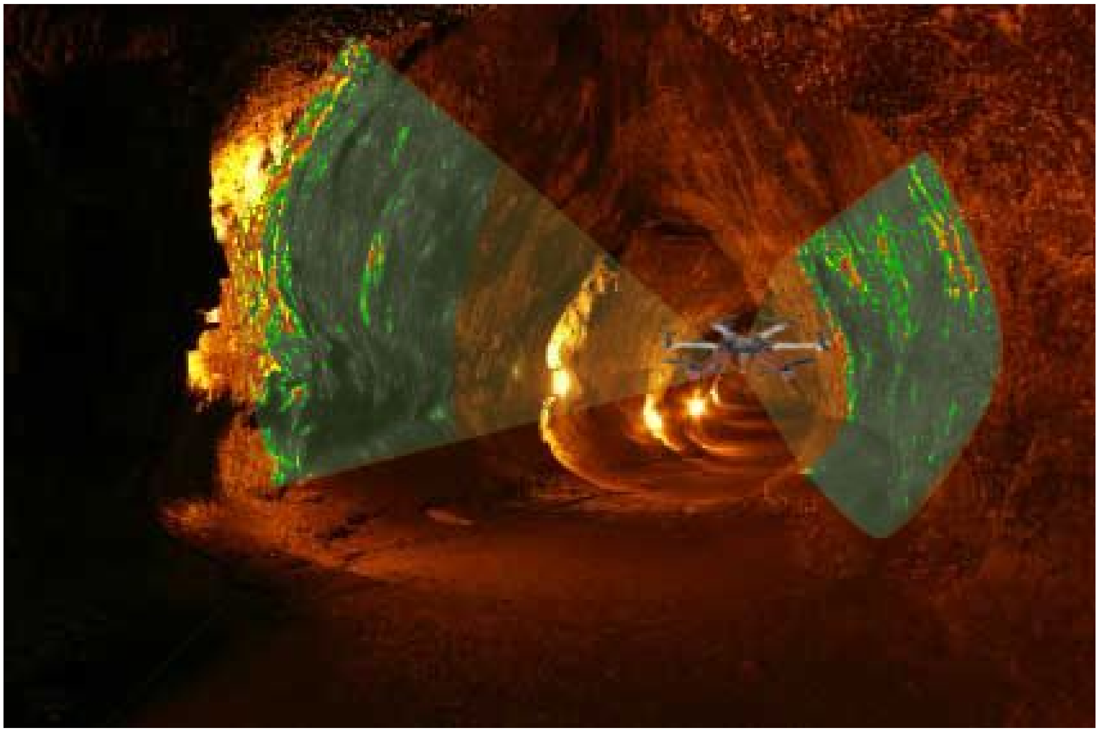


Figure 1: Site operation mock-up

4. System-level Requirements

4.1 Mandatory Performance

MPR1	Fly at a minimum speed of 0.25 m/s
MPR2	Generate a colorized map with a voxel size of at most $(50\ cm)^3$
MPR3	Map 50,000 m^3 in at most 15 minutes
MPR4	Localize accurately so that drift in pose is at most 0.1 meter / meter traveled
MPR5	Update map with fresh sensor data at 1 Hz
MPR6	Provide map back to user at least 0.5 Hz
MPR7	Be tele-operable at a range of at least 20m
MPR8	Localized to less than 3m of error
MPR9	Detect obstacles of 50cm x 50cm x 50cm
MPR10	Plan paths that avoid obstacles by 3m.

4.2 Mandatory Non-Functional

MNF1	Must have enough battery to operate for 15 minutes
MNF2	Additional sensors should be easy to integrate into software.
MNF3	Battery should be easily accessible for hot swapping on successive runs.

4.3 Desirable Performance

DPR1	Safely land and takeoff at 0.3 m/s
DPR2	Wirelessly controllable up to 100m distance from user
DPR3	Wirelessly transmit maps and video data up to 100m distance from user
DPR4	Data point position resolution of at least 15 cm
DPR5	Should be able to fly up to 100m relative altitude

4.4 Desirable Non-Functional

DNF1	Additional perception sensors can be easily mounted.
DNF2	User base station can be easily transported
DNF3	Vehicle should be easily transportable
DNF4	Vehicle operation process should be minimally complex to minimize startup time
DNF5	Vehicle computer should be powered without draining main vehicle battery

5. Functional Architecture

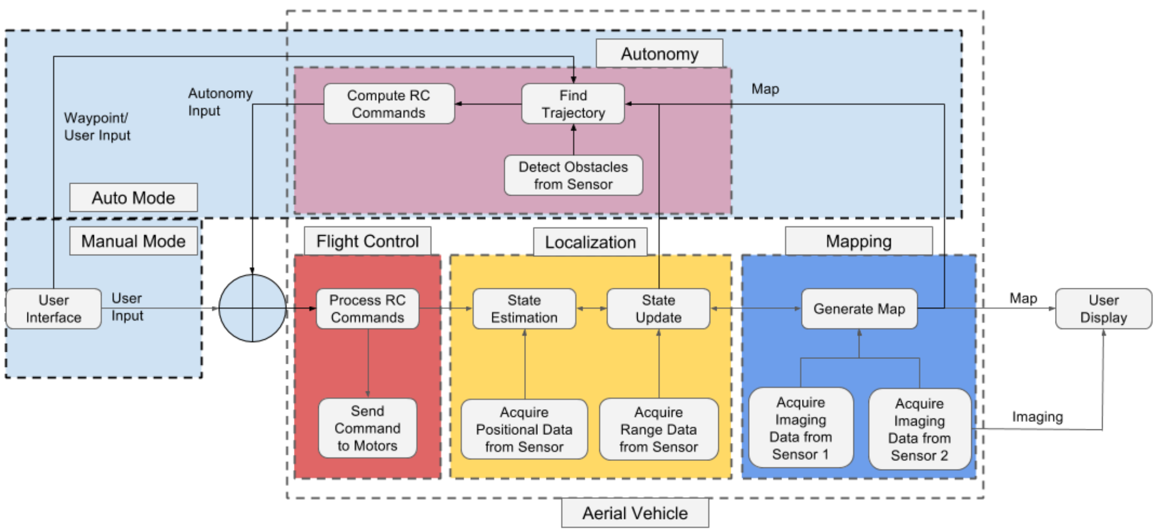


Figure 2: Functional architecture diagram

The functional architecture for this project is visualized in Figure 2. Because the aerial vehicle will be controlled through teleoperation, its sole input will be from the user, who utilizes a physical interface to direct the vehicle. For autonomous navigation, the user input will be a goal position, following which the trajectory generator will produce a path that the planning software will parse into vehicle movements to the flight controller. The expected output will be a real-time mapping of the environment that the vehicle is traversing through. The raw sensor data will also be provided to the user, for later post-processing and further analysis.

Once the vehicle receives remote input from the user (or a waypoint from the autonomy subsystem), it processes the commands, adjusting the speed of the motors to travel at the specified velocity. Simultaneously, the robot conducts state estimation, keeping itself stable in the air from positional data acquired from a sensor. After knowing its approximate state, the robot is then able to take in imaging data from different sensors to perceive its environment and update its state estimate. At a minimum, there will be two mounted sensors whose data will be fused together to provide a 3D map. Additionally, images from one of the RGB cameras will be sent directly to the ground control station to aid the user in manual flight of the UAV. The imaging sensor-fused map will be sent back to the user, updating frequently such that the user is always able to see where the robot is in space.

6. System-Level Trade Studies

Three trade studies were conducted to evaluate three separate components of the system. These were selected based on the requirements to have both an aerial vehicle and multiple sensor modalities for enhancing the point cloud generated by the LIDAR. Each category was selected based on the unique system requirements, and the weights assigned to each category reflect their relative importance towards meeting the system objectives.

6.1 RGB Imaging

A study was conducted to evaluate and compare various RGB cameras used in UAVs and other mapping and surveying UAVs. The RGB cameras below were selected based on their previous use in small form-factor aerial vehicles both in research and industry. As per the requirements, the cameras below were evaluated for their ease of integration into an aerial system, namely: size, weight, interfaces, cost and sensed image quality for generating map texture data. The trade study for this can be seen in Table 1.

Image quality was evaluated based on the camera resolution, lens distortion, global shutter ability, and the dynamic range necessary for high quality texturing of point clouds. Volumetric size and weight were evaluated relative to other components already integrated into UAVs. RGB sensors are typically lower-cost than other imaging modalities; thus cost was considered, but not weighted as greatly as the other categories. Lastly, ease of integration was considered critical to their inclusion in the system. If the camera had unique power, signal, mechanical, or software requirements that made integration less straightforward, they received a lower score. Likewise, higher scores were correlated with their successful documented use in commercial and hobby UAVs.

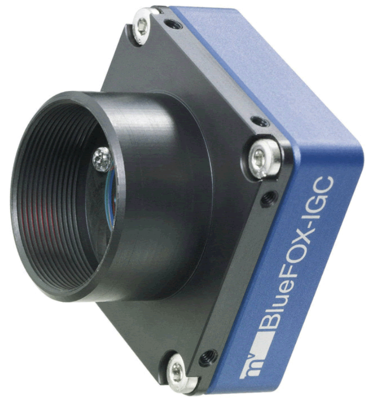


Figure 3: mvBlueFOX RGB camera

Table 1: Trade study for selecting an RGB camera

	Weight	mvBlueFOX [1]	GoPro Hero [2]	See3Cam [3]	Sentera 4k (Multispectral) [4]	Parrot Sequoia (Multispectral) [5]
Image Quality	4	5	4	3	5	4
Cost	3	2	5	4	1	2
Volumetric Size	4	5	4	5	3	4
Weight	4	5	2	5	3	3
Ease of Integration	5	3	1	2	4	4
Weighted Score	100	81	60	74	67	70
Normalized score (5)	**5.00**	**4.05**	**3.00**	**3.70**	**3.35**	**3.50**

As a result of the trade study, it is clear consumer-grade cameras like the GoPro are not suitable for aerial mapping. While the normalized scores of the mvBlueFOX (pictured in Figure 4) and the See3Cam are fairly similar, there remains a tradeoff between cost and image quality that would need to be evaluated. Since the See3Cam is relatively inexpensive, the team will purchase both cameras and should be able to quickly compare their ease of integration as well as image quality to make the final decision.

The Sentera 4k and Parrot Sequoia also present interesting imaging packages that might be worth considering for RGB imaging as well, despite their relatively much higher cost. They combine RGB with multiple hyperspectral sensors, which create a much simpler interface for collecting data from all sources at once. However, the tradeoff is sacrificing RGB image quality for more versatility.

6.2 Hyperspectral Imaging

As the title and main objectives note, in addition to LiDAR and RGB mapping, the team considered other imaging modalities to augment the map data. The lowest-cost and most widely available sensors are typically those used in agricultural imaging (IR, Near-IR, Red-Edge, Green, and Blue), or thermal imaging. These represent hyperspectral light wavelengths that are unable to be seen by the naked eye.

However, as the price of such sensors can quickly reach the five or six digit price range, cost is an important factor in selecting sensors that can provide data which generate valuable insights to end users. Seen in Table 2 is a trade study for determining the best hyperspectral camera for our UAV application. In general, these sensors are larger than typical RGB sensors, so size was slightly deprioritized for an increased price sensitivity. Otherwise, the previous factors in selecting an RGB camera (weight, image quality, ease of integration) all have the same weight when selecting a hyperspectral camera.

Figure 4: Parrot Sequoia Multispectral Camera

Table 2: Trade study for selecting a multispectral camera

	Weight	FLIR Tau 320 [6]	MicaSense Red Edge [7]	Sentera 4k (Multispectral)	Parrot Sequoia (Multispectral)
Image Quality	4	3	4	5	3
Cost	4	3	2	1	3
Volumetric Size	3	4	3	3	4
Weight	4	4	4	4	5
Ease of Integration	5	2	4	4	4
Total Score	100	58	69	69	76
Normalized score (5)	**5.00**	**3.10**	**3.45**	**3.45**	**3.8**

The MicaSense, Sentera, and Sequoia cameras all provide multiple imaging sources from different parts of the light spectrum. These allow for greater flexibility when imaging various points of interest while maintaining a compact package. However, these sensors are typically constrained to the IR, near-IR, and red, blue, green sections of the electromagnetic spectrum. Due to cost, the team is limited to evaluating sensors that operate in the visible light through infrared spectrum. Thermal sensors operate at similar wavelengths in the IR range, but are optimized for sensing heat, with or without infrared. Ultimately, however, in the interest of completing the state objectives of the project, multispectral imaging integration was scoped out of the final vehicle build.

6.3 Vehicle Type

There are multiple vehicle designs and propulsion technologies that are currently used in mapping and surveying UAVs. These include quadrotors, hexrotors, fixed wing aircraft, and helicopters. Each vehicle type has been successfully deployed in research and industry for their respective use cases, and a trade study can be performed to determine the best approach for the vehicle. Because of the team's limited budget and time, the simplicity of the vehicle and its control in addition to physical attributes, including payload, weight, and flight duration must be focused on. Since the main objective of this project is to generate real-time multi-modal maps, the simplicity of the vehicle, as well as its ability to carry the necessary sensor payload to generate the map, was prioritized. Shown in Table 3 is a trade study comparison of several different aerial vehicle designs to select one that best suits our application of planetary exploration.

Figure 5: Sensefly fixed-wing surveying UAV

Table 3: Trade study for selecting vehicle type

	Weight	Quadrotor	Hexrotor	Fixed Wing	Helicopter
Mechanical Simplicity	3	4	5	3	2
Control Simplicity	5	4	4	2	1
Cost	2	3	2	2	3
Weight	2	3	2	5	4
Flight Duration	4	3	2	5	3
Payload Capacity	4	3	5	1	2
Weighted Score	100	68	71	57	45
Normalized score (5)	**5.00**	**3.40**	**3.55**	**2.85**	**2.25**

After an initial evaluation of these vehicle designs, it is clear that fixed wing aircraft and helicopters are ill-suited for this task. While flight duration is important, fixed wing aircraft have a minimum airspeed that may be too fast for the mapping software to keep up with. In addition, precision control of fixed wing aircraft is still very much an ongoing research topic, and certainly out of scope of this project. Helicopter control is also relatively complex compared to quadrotor and hexrotor control, and helicopters don't immediately present any significant advantages over quadrotors and hexrotor vehicles. Quadrotors and hexrotors both have relatively cheaper components due to their commoditization and prevalent use in hobby, industrial, and research areas.

Our team has decided to go forward with developing on a hexrotor platform, which reaps the advantages of sharing components with quadrotor platforms while providing greater payload capacity for more sensor payload flexibility.

7. Cyberphysical Architecture

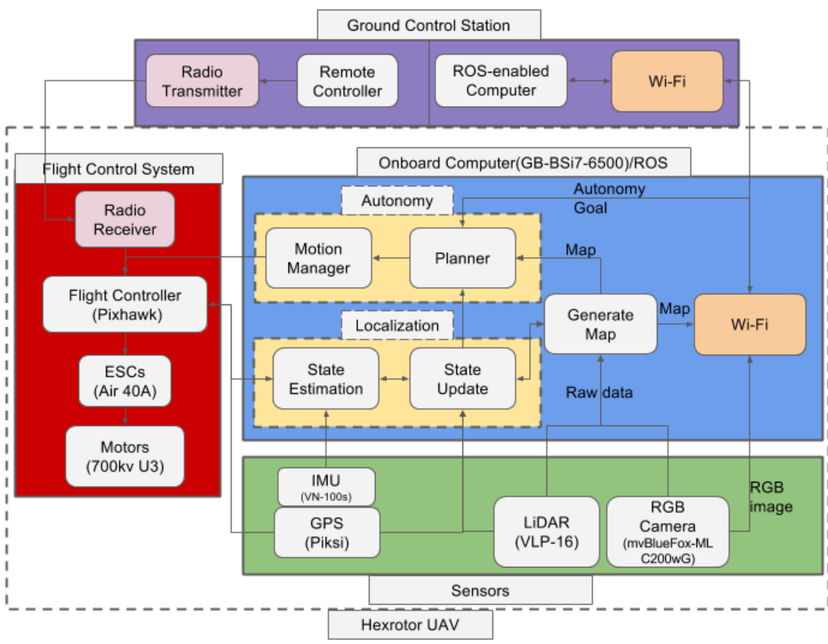


Figure 6: Cyberphysical architecture diagram

Shown in Figure 6 is the cyberphysical architecture diagram for the project. The User Interface from the Functional Architecture changes into a Ground Control Station. The Ground Control Station and the hexrotor are linked on a Wi-Fi ROS network hosted on the hexrotor. A 2.4 GHz remote controller at the Ground Control Station sends commands to the receiver, which communicates with the flight controller, in this case the PX4 Pixracer [8]. A switch on the remote controller helps us switch between "Offboard" and "Manual" flight mode. The autonomy system works only when the Pixracer is in "Offboard" mode. In all other flight modes, the drone is tele-operated. Depending on the manual or autonomy command, the flight controller receives velocity commands for roll, pitch, and yaw either through the RC or the motion manager. The flight controller processes these commands and sends them to electronic speed controllers (ESCs [9]), which engage and control the motors [10] and connected propellers, enabling flight. At the same time, the onboard computer is performing state estimations to determine the pose and orientation of the robot, which is informed by an inertial measurement unit (IMU). The RTK (real time kinematic) GPS is solely for ground truth comparison and is not used for state estimates and updates. The state is updated with information from point clouds generated from the LiDAR (light detection and ranging), which localize the robot in its surroundings. As the robot traverses through the air, it will generate a map, which is a composite of the LiDAR and RGB sensor data. The images from the RGB cameras along with maps that are generated by the onboard computer will be sent through the radio transmitter over 2.4 GHz WiFi network back to the ROS network. This map is displayed to the user at the ground control station. A user-interface with a marker on the ground control station allows the user to send an autonomy goal to the hexrotor when in “Offboard” mode.

8. Current System Status

8.1 Overall System Depiction

The final built vehicle with sensors labeled is shown below in Figure 7. This is the physical vehicle which also contains the below described software, electrical, and mechanical subsystems. Their interaction is described in detail in the cyberphysical architecture above.

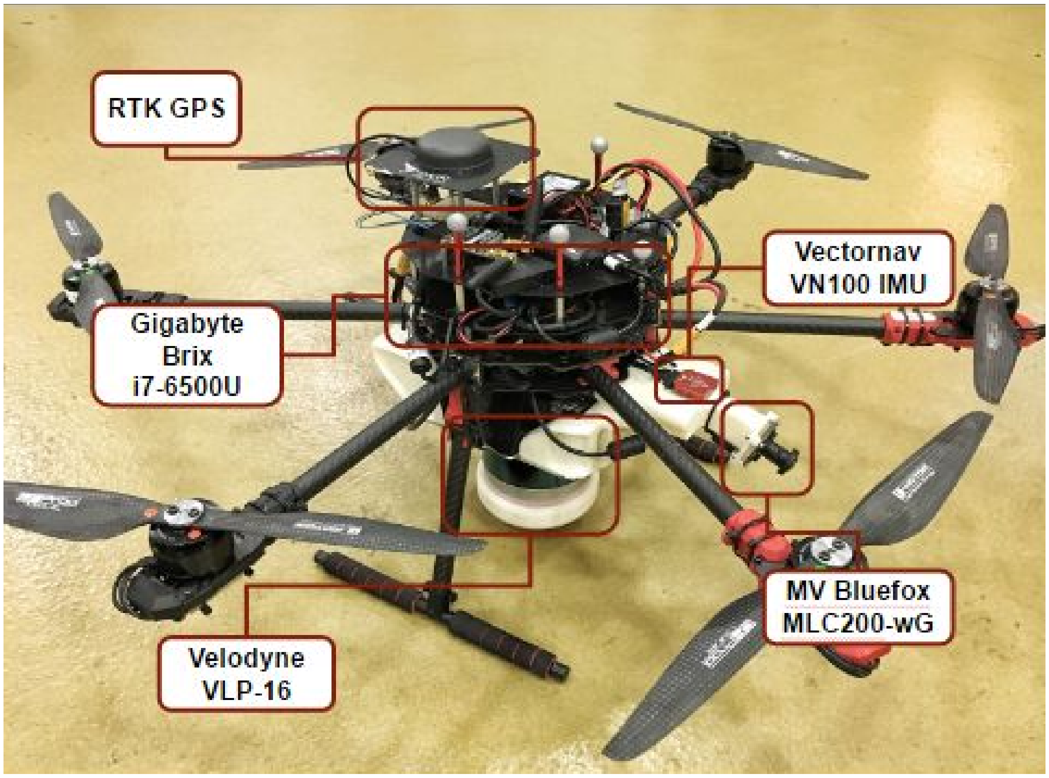


Figure 7: UAV with all sensors and hardware mounted

8.2 Software Subsystem

Figure 8 below describes the flow of data from the sensors to subunits and represents a high level overview of the architecture of the system.

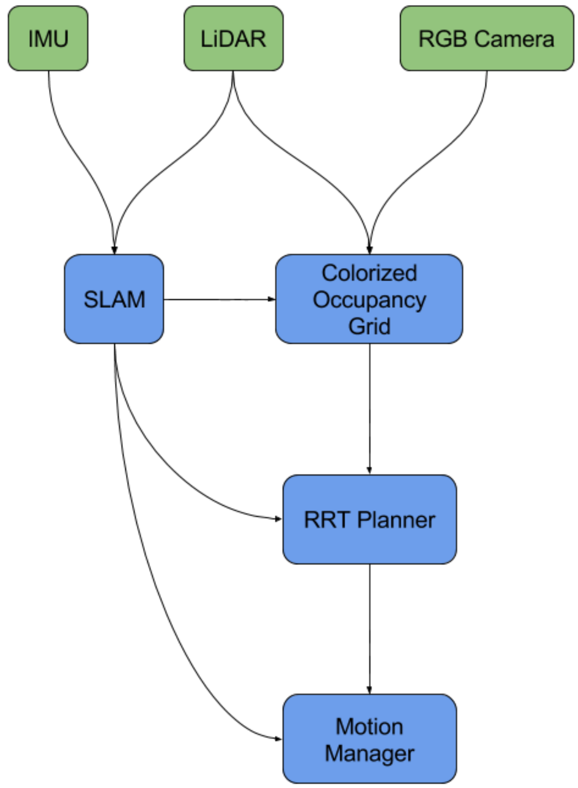


Figure 8: Data flow of sensor data to functional subunits

At the lowest level there are drivers for three sensors: IMU, LiDAR, and RGB Camera. These drivers were acquired online through public repositories. At the next level is the SLAM module and occupancy grid. They each provide a localized state estimate with respect to a dense map and a probabilistic occupancy grid respectively. The SLAM algorithm was also acquired online but heavily modified to include IMU measurements, bug fixes, and various performance improvements. The occupancy grid code was provided by the sponsor of this project and was modified to include multiple imaging modalities. The SLAM algorithms state estimate and the occupancy grid are both fed into the RRT Planner [11] which will generate a sequence of waypoints to a specified goal while avoiding obstacles in the occupancy grid. This planner was written from scratch. The plan and the state estimate are fed into the motion manager which forwards RPM commands to the motors. This was provided by the project sponsor and is used as is.

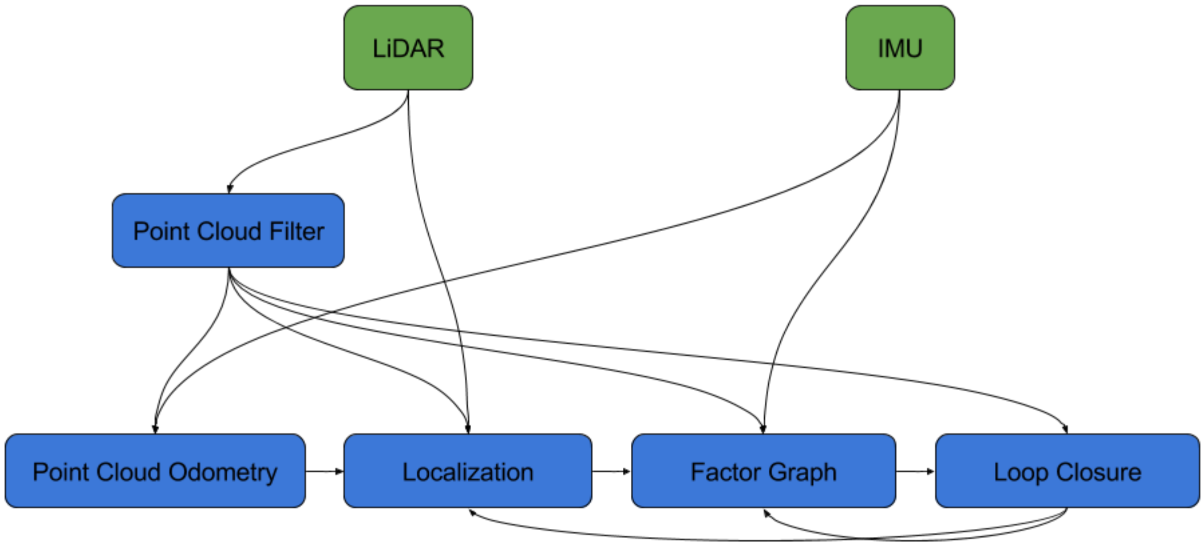


Figure 9. SLAM Functional Subunit

Figure 9 depicts the SLAM functional subunit. This pipeline was initially entirely laser based odometry but was modified for this project to integrate IMU. The algorithm starts with raw point cloud data originating from the LiDAR. These point clouds are throttled through a randomly sampling filter and reduced to 10% of its original size during the point cloud filter stage. This is to reduce processing time for later steps in the algorithm and allows the system to more ably meet the requirement MPR5. The odometry package then performs iterative closest point (ICP) with both the filtered point cloud and an initial orientation estimate from the IMU. This provides a state estimate which is then further refined by performing ICP with a global map which provides our localized state estimate. The full point cloud is then projected into the map given the localized state estimate. The map is currently structured as an Octree where each voxel contains points that have been collected during scans. The filtered point cloud, the localized state estimate, and pre-integrated IMU measurements are then put into a factor and pose graph. This factor graph keeps track of pose velocity and IMU bias. When the factor graph is updated, it is then optimized and solved for the states of the system at all times. The latest estimate is then compared to previous nearby measurements to check for loop closures. This is done by scan matching with ICP. If a match is found with a high fitness score (i.e. scans match very well) then the loop is closed and the map is rebuilt taking into account this loop closure.

The colorized occupancy grid subunit is an additional voxel grid that is built concurrently with the localization map. This occupancy grid stores the probability that any given voxel in explored space is occupied. This is calculated by determining the ratio between how many rays from the LiDAR ended in a voxel and how many rays passed through a voxel. There is then a certain threshold for whether a voxel is occupied or not. This occupancy grid is then colorized by taking a point cloud from the LiDAR and back projecting this into the image plane of the camera. This gives an RGB value for a subset of the points in the scan and the color of the voxels these points lie in.

The RRT Planner subunit takes in a goal pose (X, Y, Z, Yaw) and plans in this dimensional space. The planner is responsible both for planning to a goal and for checking that the plan remains free as the occupancy grid is updated by fresh observations. The RRT algorithm was implemented from scratch and also made use of a KDTree to speed up spatial nearest neighbor queries. The planner generates a path that avoids obstacles by a specified distance according to MPR10. The path is also post processed to reduce length. In order to accommodate replanning on detecting new obstacles, 2 different radii are used for collision detection. As seen in Figure 10, the "Static Danger Zone" is the collision radius used for normal planning. This radius is 1.5 times the "Danger Zone", which is the collision radius used when replanning on detecting a new obstacle in a previously planned path. The "Danger Zone", in this case is 3 times the size of the UAV.

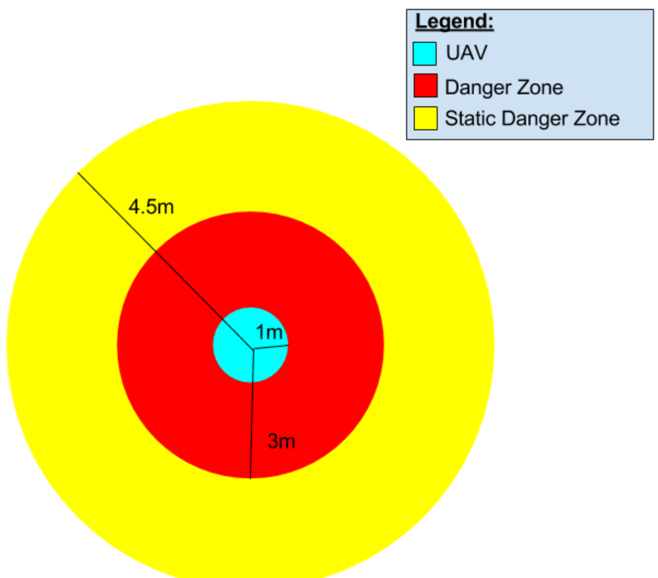


Figure 10: Planner collision radii for static and dynamic obstacles

In order to reduce the bandwidth of transmitting the map, an incrementally built map can be utilized. This means that only the new points added to the map are transmitted, which reduces bandwidth and allows for the system to meet requirement MPR6. Another function depicted is the ability to stream the videos that are captured in real time. Simply switching from RAW image to a Theora video stream reduced network bandwidth from 60 Mbps to 300 kbps.

Aside from the software mentioned here the team has also explored various calibration softwares like Kalibr as well as writing a custom package for LiDAR-Camera calibration. The team was able to get reprojection error for camera intrinsics down to about 1 pixel with a fisheye model. However, remaining calibration results still requires further testing and tweaking to get more accurate values. In the end, the default ROS calibrator was used despite it only providing support for a pinhole model because of ease of use. Because of dependency issues, OpenCV3, which added support for fisheye models, could not be used to rectify images.

In addition to the actual architecture and calibration software, various testing and setup software components have been developed. For example, odometry testing scripts that enabled the team to quickly measure the performance of the system's algorithms were written. Launch scripts were also developed for various different configurations of the sensor array in order to speed up testing and development as the complex manual startup procedure took time time away from actual testing in the field.

8.3 Mechanical Subsystem

The mechanical subsystem has been fully integrated as of end of Fall validation. All sensors (LiDAR, RGB camera, IMU) were mounted underneath the chassis in a custom 3D-printed undercarriage that also held the batteries. The CAD models for this with mounted sensors can be seen in Figure 11 below. It was necessary for the LiDAR to hang at the bottom of the chassis in order to get a full view of the environment. Although the view is partially obstructed by the legs of the UAV, this has not proven to hinder our mapping capabilities. The RGB camera is angled downwards so that its field of view intersects with the LiDAR to make sensor data fusion feasible.

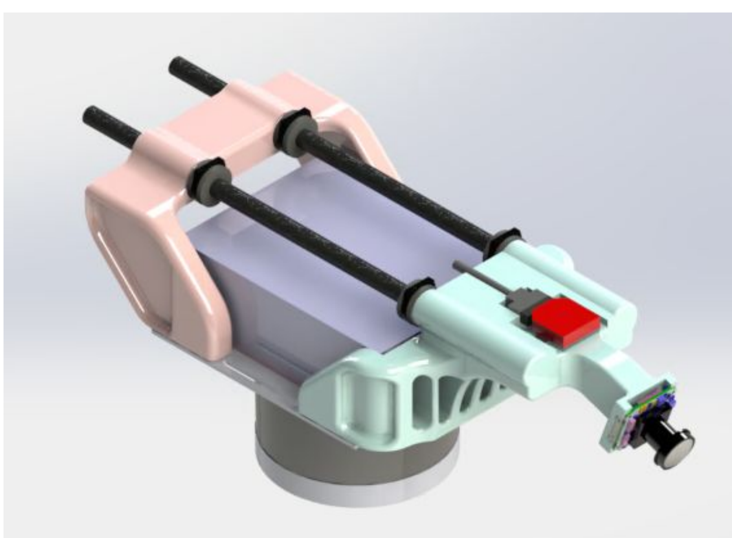
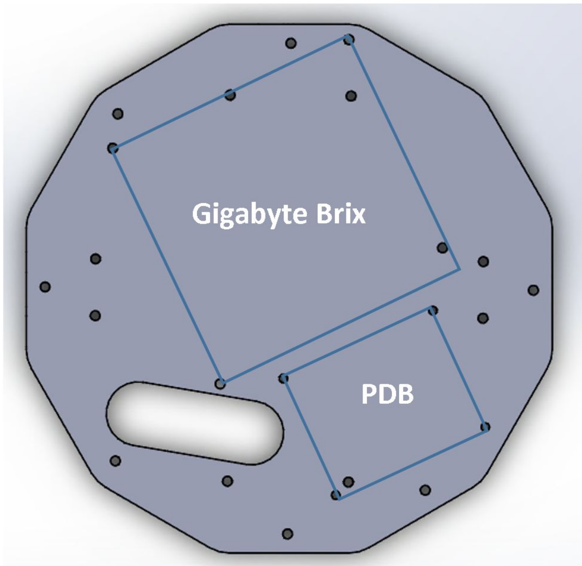


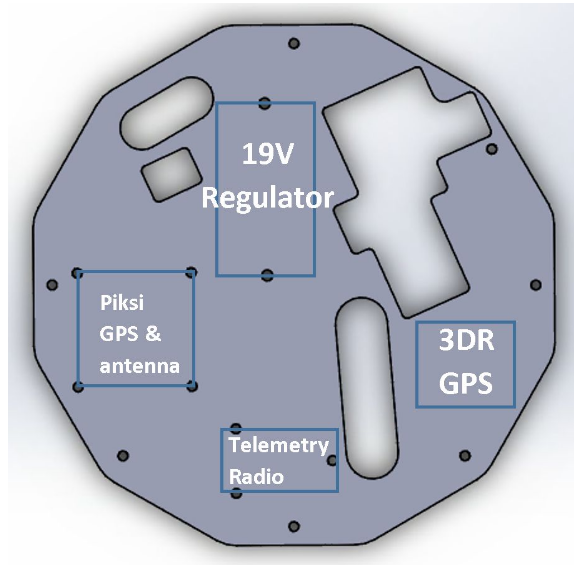
Figure 11: Rendering of the mounted sensors and batteries with the custom mounts.

For mounting the rest of the hardware components, it was necessary to make custom mounting plates that were carefully planned for space efficiency and ease-of-access. There were three levels to the upper chassis. Starting with Platform 1 as the base, the components mounted at each level can be seen in Table 4 below. The laser-cut plates were made of Delrin, and the cutting pattern can be seen in Figure 12. Because the first layer was included with the rest of the chassis, only platforms 2 and 3 needed to be made.

Table 4: Organization of components for each level of the top subassembly.

Platform 1	PixRacer flight controller
	RC receiver
	FPV telemetry radio
	Switch
	Buzzer
Platform 2	Gigabyte Brix
	Power Distribution Board (PDB)
Platform 3	Piksi GPS
	Piksi GPS antenna
	Piksi telemetry radio
	3DR GPS antenna
	19 V voltage regulator





(a) **(b)**

Figure 12: (a) Platform 2 and (b) Platform 3 that show the locations of parts to be mounted.

Since there were several expensive and heavy pieces of hardware being mounted on the UAV, the team decided to do a test flight to ensure the propellers and motors were capable of handling the total system payload. The components not critical for test flight were dismounted and individually weighed. The LiDAR, camera, IMU, and cabling for the bottom half of the chassis weighed 930 g. The top half which included the computer, power distribution board, delrin plates, GPS, and cabling was 750 g. A bottom dummy weight was fabricated from a block of aluminum and the top dummy payload was sourced that equaled these weights within a tolerance of +/- 25 g. After mounting these weights, the total system was 5.35 kg. This system can be seen on the next page in Figure 13. Overall, the flight test was successful and was able to help validate MPR1 by proving that the system was capable of flying with its total payload. The minimum speed was proved out in future test flights. After this, the team was able to move forward with flying the full system with all the mounted sensors and hardware.

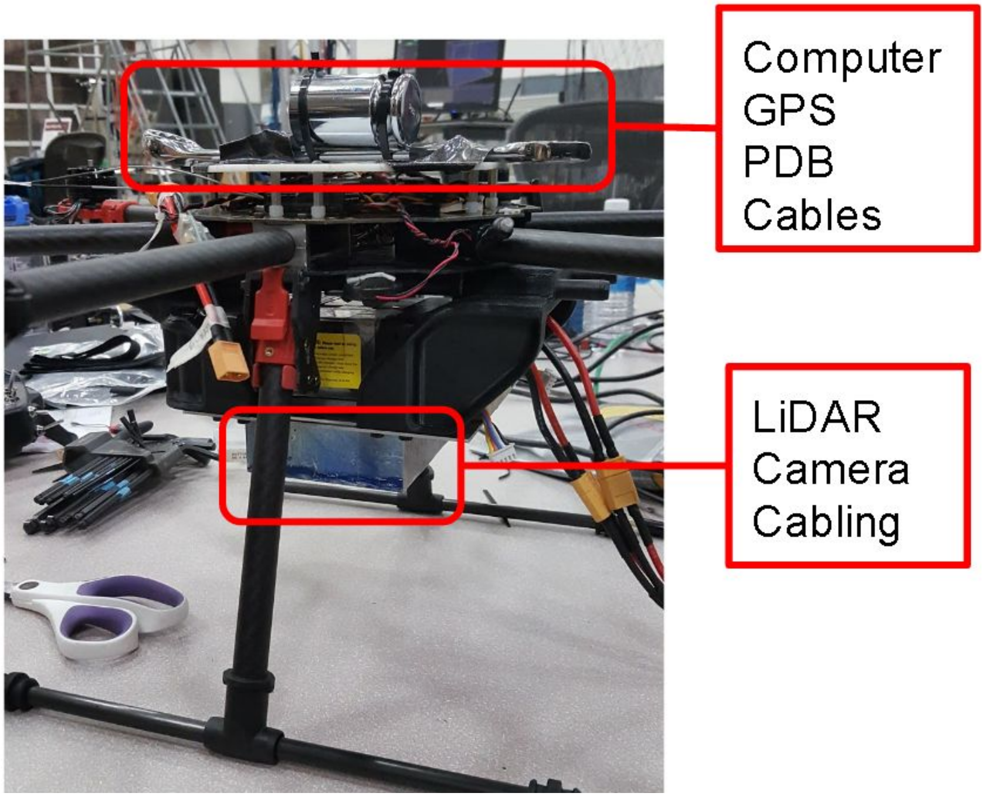


Figure 13: UAV with dummy payload for experimental flight test

8.4 Electrical Subsystem

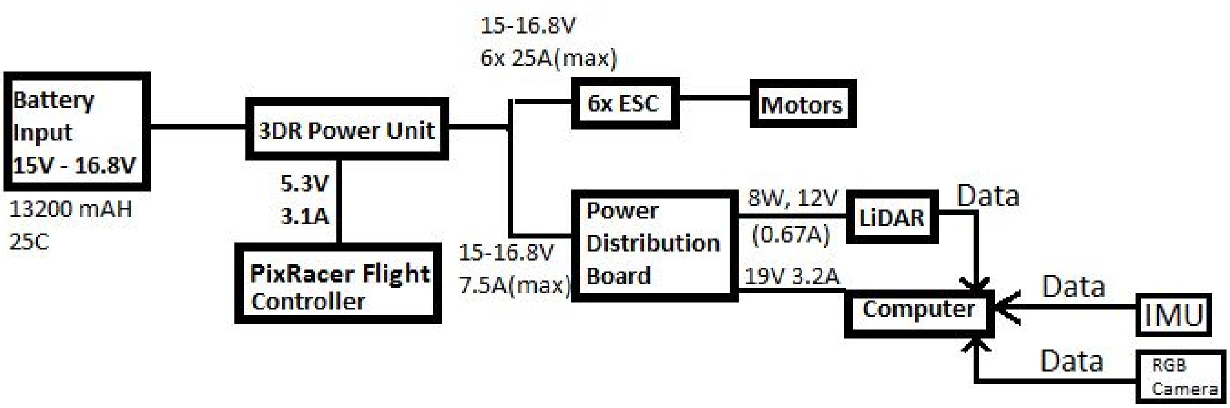


Figure 14: Electrical System

The current electrical system diagram is depicted in Figure 14 above. The Power Distribution Board (PDB) was designed to power the computer and LiDAR off of the battery power and involves voltage regulation and circuit protection elements, such as overcurrent, overvoltage, and reverse voltage situations. A custom cable was made for powering the LiDAR and interfacing the computer. Table 5 below shows the result of the PDB test.

Table 5: Power Distribution Board Test Results

Device	PDB Test Input Voltage (V)	Board Input Current Capacity	Observed Max Input Current	Board Rated Output Voltage	Observed Output Voltage
Gigabyte Brix [12]	15	5A	0.55A (Ideal)	19V	18.8V
			1.7A (Startup)		
			1.7A (OpenCV Compile)		
Velodyne VP-16 [13]	15	2.5A	0.68A	12V	11.65V
Brix + VP16	15	7.5A	2.5A	As above for each device	As above for each device

8.5 Modeling

During the construction of the system the team relied on Solidworks 3D CAD to help lay out and aid with construction of the vehicle. The entire vehicle was constructed and put together in CAD. Figure 15 shows screenshots of the layout of the vehicle. This CAD model also informed the hole patterns of the various stages that were cut as seen in Figure 12.

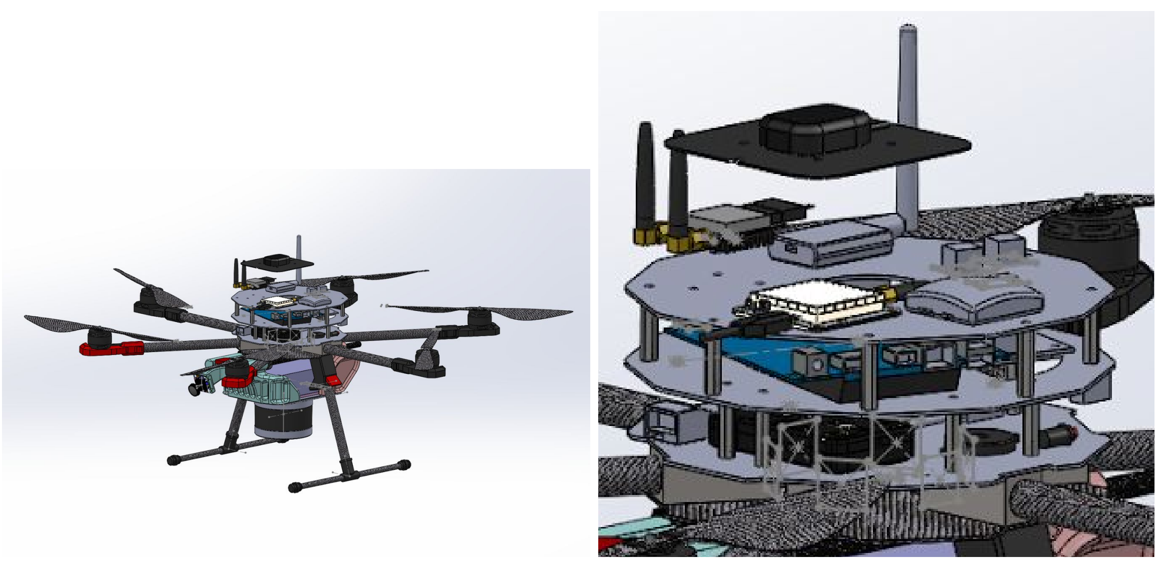


Figure 15. CAD images of drone and close ups on Electronics.

The power distribution board was modeled in EagleCAD and the schematic and board layout can be seen in the Figure 16.

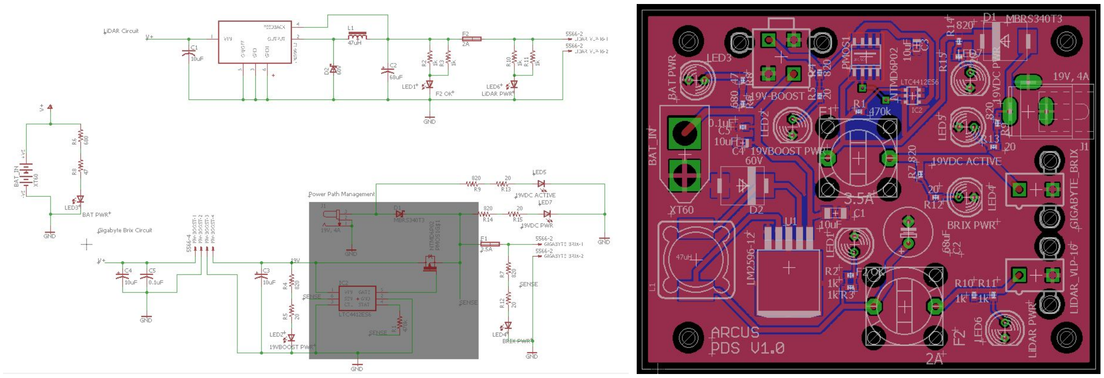


Figure 16. Schematic and Board Layout of Power distribution board.

8.6 Analysis

The primary analysis performed was on the power source configuration. The most important considerations were to meet the thermal, electrical power, voltage, and current requirements of the components described to the right of the battery in Figure 14 above. This was a trade off between mass, cost, and power delivery. The batteries were chosen such that they could provide enough power to supply the motors for flight and to power the sensors and computer that were onboard. The power distribution board(PDB) comprised of 2 regulators and the protection circuitry as seen in the schematic and layout in Figure 16 above. These were a 19V, 76W step-up boost converter, and a 12V regulator. The 12V regulator further required some thermal calculations for an appropriate heat sink, however, it was found that the absence of a heat sink didn't cause too much heat buildup either. Following this design, the minimum input necessary to the PDB came to 15V and hence, a 4S LiPo(14.8-16.8V) would be sufficient. A maximum current capacity of 7.5A was also calculated, which is just more than the maximum input requirement calculated as 5.74A considering 90% power efficiency for the boost converter and 0.71A for the 12V regulator. Although the minimum input voltage requirement is set by the PDB, at 15V, the maximum current requirement is precedented by the motor current draw to 150A as per Figure 14. Note that this is the maximum draw and that the nominal draw is a much lower value (10A*6 = 60A). Adding the current requirements of the 3.1A PixRacer and the 7.5A PDB, we had a maximum input current requirement of 160.6A. Hence, putting 2 of the 6600mAH, 25C batteries in parallel, gives a maximum input current capacity of 13.2*25 = 330A, far beyond the requirement.

8.7 Testing

A large amount of time was spent testing various components of the system to ensure the system was safe to fly. All of the flight tests were conducted in the safety of a flight cage in the PRL Highbay. Flight capability was by attempting to fly without any sensors or computer mounted. This helped in teasing out issues like tuning gains and configuring ESCs without risking any expensive equipment. A payload of the same weight as the sensors and computers was constructed in order to test the system's ability to fly with with a certain payload on it, which can be seen in Figure 17 below.

Figure 17: Shows test setup for dummy payload test.

Next, a series of tests were used to evaluate PCB functionality. First, simple connectivity tests of the unpopulated PDB were conducted. The board was then populated and various components were tested to ensure connectivity and expected values. Next the PDB was tested for correct voltages at correct locations after plugging it into a power supply. Then the PDB was tested to make sure the computer and sensors could be powered on it with input from a power supply. Next, the last two tests were repeated with the PDB integrated into the electronics with our drone and powered off the battery. Figure 18 below shows an example of the test if, with the PDB and computer running, and motors spinning, the power to the computer never faltered.

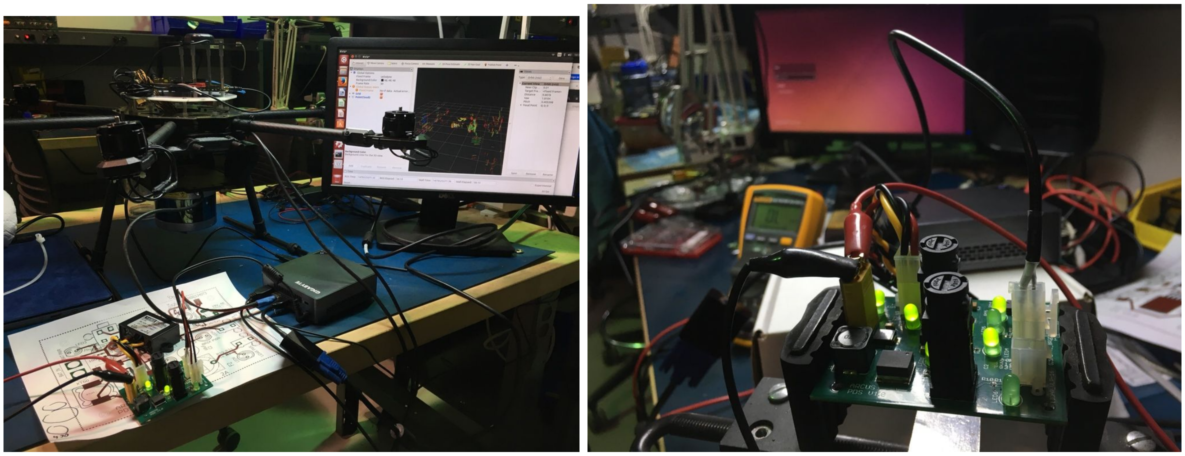


Figure 18: Shows PDB test while spinning the motors.

A simulation was used as a testing method to reduce vehicle setup and teardown time and isolate any hardware issues. A simulator and flight software stack was acquired from the project advisor and modified the occupancy grid and mapping code to support color mapping. Mapping was tested without flight to eliminate movement as a source of error. A full integration test, including color mapping during flight on final hardware, can be seen in Figure 21 in the Performance Evaluation section. The simulator was also used to test and verify the planner.

8.8 Performance Evaluation

The system performed well and met many of its performance requirements. Unfortunately the experiment was changed a number of times before the demonstration due to reprioritization on the side of the sponsor. However many of the requirements that the team set out to fulfill were met. Figure 19 shows that odometry drift error was reduced to less than 0.1m per meter traveled and that the vehicle was localized to within 3 meters of the ground truth position. It shows that by the end of the run the drift over distance traveled is about 0.08m and that the state estimate was never more than 1 meter away from the ground truth. These correspond to MPR4 and MPR8 respectively. This was validated in the flight cage using motion capture data as the ground truth.

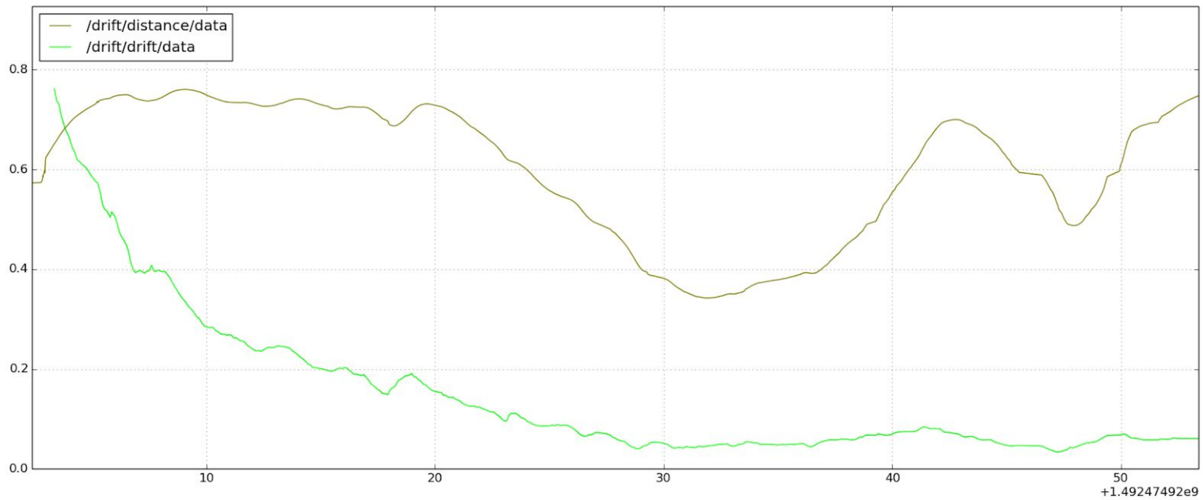


Figure 19: Shows distance from ground truth(m) vs time(s) in brown. In green is the distance from ground truth divided by distance traveled(m/m) vs time(s).

The system also demonstrated that it was capable of planning a path that avoided obstacles by 3 meters. This was proved by checking 50 mm intervals along the path for distance to obstacle and reporting the minimum of those distance checks. This validates the performance requirement MPR10. The system also validates MPR9 by demonstrating it can sense the obstacles. Figure 20 shows the planned path specifically avoiding obstacles and detecting the obstacles.

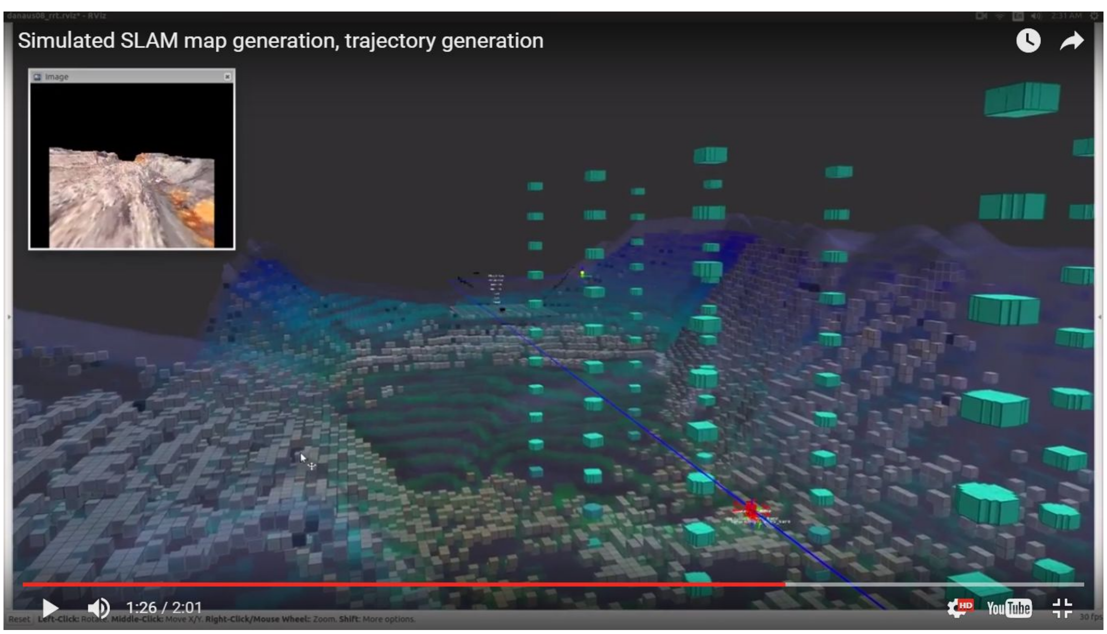


Figure 20: Shows a simulation environment of the drone planning around obstacles. Light teal are obstacles that are place directly in path of drone, blue shows planned path of the drone. See video at https://youtu.be/r5va7mgXalQ.

The system also demonstrated it was capable of providing back to the user a colorized voxel-map. This was demonstrated both in simulation and on the physical platform. This validates the requirement MPR2. Figure 21 shows an example of a color map created in simulation while Figure 22 shows a color map generated on real hardware.

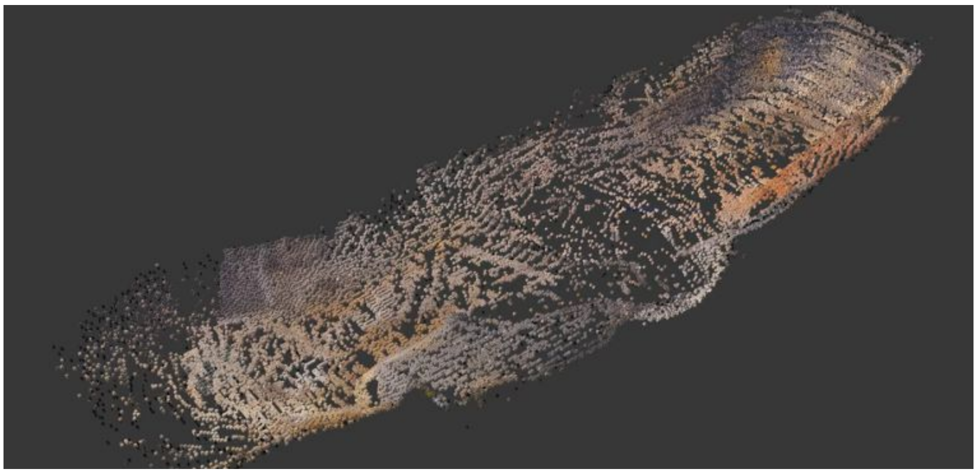


Figure 21: Arcus' map of the LaFarge Quarry in Pittsburgh constructed with range and color modalities

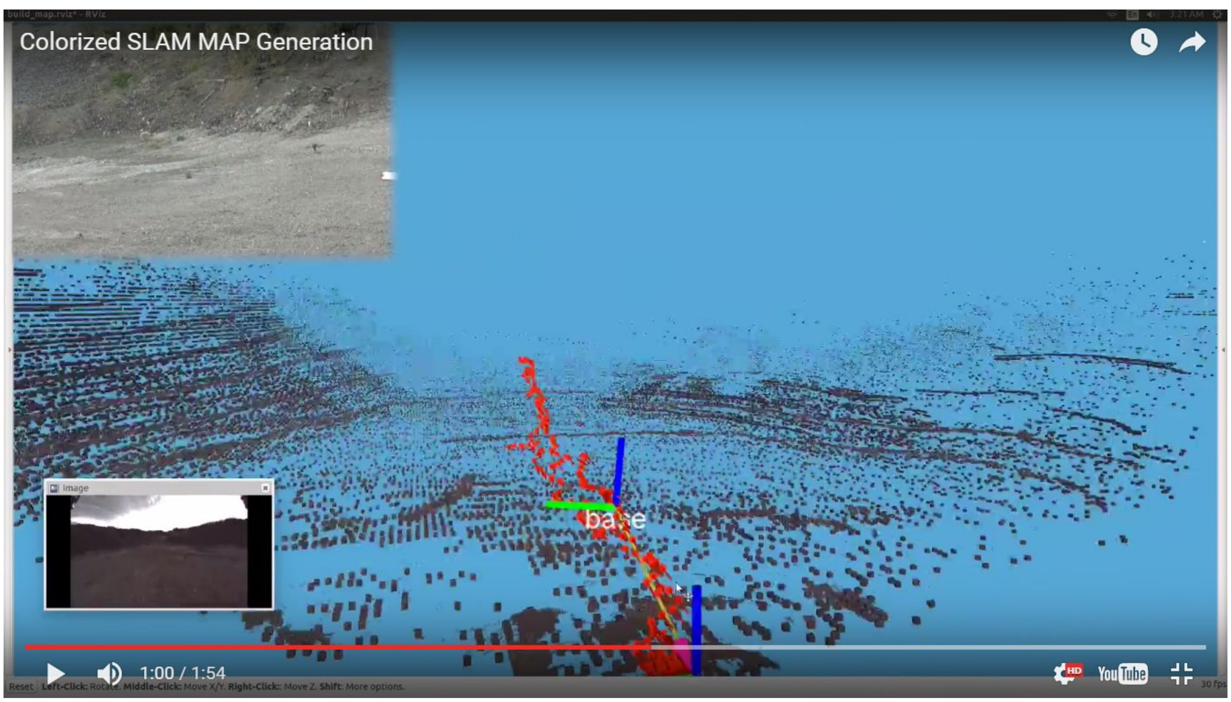


Figure 22. Shows a map of the quarry being generated in real time. Check out the full video at https://youtu.be/eN4vnYmoY2k.

8.9 Strengths and Weaknesses

The biggest strength of the system has been its reliable and efficient SLAM software. It has performed robustly under a variety of conditions and environments. The integration of IMU measurements into it made this even more accurate at the cost of some computational complexity. A simple approach was taken in regards to many aspects of the software architecture and a lot of proven software was repurposed to help make the system robust. The electrical system also proved to be very reliable during the final flight tests. The hardware design was very modular, which allowed for quick recovery after vehicle crashes. Similarly, the team made an effort to mitigate risks by purchasing extra parts and trying to protect the system's most sensitive and expensive equipment. This proved to be very prescient, as these mitigations largely prevented catastrophic damage from occurring.

The system's most immediate weaknesses largely deal with autonomy, fieldwork, and software. The team encountered issues with GPS and were unable to get ground truth for the system out in the field. The team was unable to get a chance to test autonomous flight on real hardware unfortunately, mainly because of lack of testing. In terms of software, some room for improvement would be researching a sparser data structure for storing the occupancy information as the dense grid took up a very large amount of memory for small resolutions. Another area of improvement is rectification of the fisheye camera image. This allows us to get more accurate reprojections and more intelligible maps.

9. Project Management

9.1 Schedule

Fall 2017 consisted of building the sensor and electrical hardware stack on top of the UAV, conducting test flights and bagging preliminary data for visualization, and getting the SLAM pipeline integrated onboard the UAV with raw LiDAR information. The team was able to achieve its goals effectively during this semester and was, as a result, confident in its ability to widen the scope for spring semester. One of the major items that was not achieved during fall semester was mapping, but this goal was migrated over to Spring as it was determined that system integration was a more pertinent issue to take care of before mapping could become a focus. A summary of the Fall schedule can be seen in Figure 23 below.

Figure 23: Team schedule for Fall 2016

The schedule for Spring 2017 consisted of three major pipelines: localization, mapping, and autonomy. The vehicle underwent a hardware refresh that the team accomplished early in the semester. The original schedule with progress statuses and progress review milestones can be seen in Figure 24. Ultimately, the team was able to get the localization and mapping pipelines into a state that satisfied the system requirements but were not able to achieve full autonomy with the UAV as was planned at the end of Fall. Some of the more major unfinished components in the schedule seen include the localization improvements and the testing of the autonomy stack on the mini-quadrotor. The localization improvements were originally scoped to be improving the update frequency, but this was halted in favor of an unscented Kalman filter that would provide much more robust, high frequency state estimations, which was to be provided by the lab. The autonomy stack was not able to physically tested on the quadrotor because there was not enough development able to be done in simulation to confidently test the planner onboard a physical vehicle. Minor issues that were not finished were not crucial to the development or testing of the system or were scoped out due to lack of time.

The team's scheduling process was less effective in Spring compared to Fall for a few different reasons. One large issue was a lack of meetings or in-depth interaction with both our advisor and with just the team, which persisted for a few weeks in the month of March. The reason for a lack of team meetings was due, in part, to members working on their own part of the software, which did not require in-person meetings so much as messaging each other about debugging issues. Additionally, the team became busy with other classes and the individual members did not pressure each other enough to meet. Interactions with the advisor also dropped off steeply, due to the advisor's involvement in other research projects and papers. In general, a lack of communication resulted in a schedule that was not properly scoped out and an inability to reach some of the planned milestones.

Despite this trip-up, the team pulled itself back together for the end of the semester. It was determined that the SVE definitely needed to be descoped, and the team worked to determine exactly what was achievable together, given the limited time left. The team met and properly listed out all the remaining tasks and committed to stand-up meetings every other day. These meetings kept the scope of the tasks fresh in mind and also allowed each member of the team to focus and get help when needed. The change in process allowed the team to scope out work in much smaller two day increments and enabled it to complete a majority of the major tasks left in the project.

Figure 24: Team schedule for Spring 2017

9.2 Parts and Budget

Below in Table 6 is a list of equipment that has either already been acquired by the project sponsor or has been purchased by the team. These are all components in use on the vehicle or have been transformed into functional parts installed on the vehicle.

The budgeting process was mostly a consideration only in the fall as almost all of the hardware in the system was complete in the fall. Initially multiple trade studies were completed and they informed us when making the decision to purchase chassis, motors, batteries, and sensor. The budgeting process was simplified since the project sponsor used a laboratory budget to purchase items instead of using the project course budget. Strengths of the budgeting process include budgeting for adequate spare parts in case of a crash, and very detailed modeling which ensured wiring of the correct length was ordered.

Table 6: Bill of Materials with associated costs and quantities. Team Arcus expenses are bolded.

Type	Part	Cost	Quantity	Subtotal
GPS	Piksi	1000.00	1	1000.00
LIDAR	Velodyne VLP-16	7999.00	1	7999.00
Computer	GB-BSi7-6500	540.00	2	1080.00
RAM	Corsair 2x8GB	160	1	160.00
SSD	SAMSUNG 850 EVO M.2 250GB	330	1	330.00
Computer Power Supply	DCDC-NUC	60	1	60.00
Camera Lens Holder	CMT821	6	2	12.00
Camera Lens	DSL219D-650-F2.0	99	2	198.00
Autopilot Dev Kit	Pixracer	64.99	1	64.99
Frame	Tarot 680Pro	119.99	1	119.99
Motors	700kv U3	98.91	12	1186.92
ESCs	ESC32	17.00	12	204.00
Propeller	CF 13x4.4	65.00	5	325.00
12v Regulator	RMRC 5V/12V BEC	20.00	1	20.00
Battery	6600mAh 4-Cell/4S	180.00	2	360.00
RGB Camera	mvBlueFox-MLC200wG	275	1	275
Sponsor Total Expenses				13,394.9
Electronics, cabling				**(421.17)**
Assembly hardware, fabrication components				**(542.35)**
Testing expenses				**(30.00)**
			Total Expenditure	**(993.52)**

9.3 Risk Management

The risk management table that applied to Spring semester is provided in Table 7 below. For the large part, many of these risks were mitigated. Risk 4 was resolved by achieving loop closure with BLAM; risk 5 was resolved by using the observation_synchronizer package provided by RASL - which was not available to the team during Fall semester; risk 9 was mitigated by successfully integrating BLAM; and risk 10 was no longer a risk since physical autonomous flight was scoped out of SVE. The only issue with risk 6 was that the team did not anticipate the pilot crashing the vehicle for the reasons it occurred, which was due to the harsh wind conditions at the quarry that day. Despite this, the subsequent risk of losing a drone chassis *was* mitigated for, and there were many replacement parts that enabled the team to rebuild the UAV and resume other work in about three days. A Velodyne crash helmet -- a product of Fall risk mitigation -- also ensured that the most expensive sensor on the vehicle was not lost.

In general, the team's risk management process focused mainly on hardware, software, and electrical considerations. To a large extent it was very effective, but there were other risks that were not taken into account that should have been. Risks like not all stakeholders being fully briefed whenever the team was considering changing the project scope were not considered. Similarly, time and resources, particularly the sponsors, being taken away were not considered risks when they did have a large effect on the project's end goal. The risk management techniques employed were effective in saving time in many cases but neglected many of the communication risks that turned out to be just as crucial.

Table 7: Risk Management Table. L stands for 'Likelihood' and 'C' stands for Consequence. Under the 'Type' category, T stands for 'Technical,' S stands for 'Scheduling,' and B stands for 'Budgeting'

ID	Risk	Description	Type	Req.	L	C	Mitigating Actions
4	Unable to form loop closures with BLAM	Despite best efforts, we are unable to close loops in our SLAM package	T	MPR6	2	2	- Find another loop closure library
5	Asynchronous timing between Velodyne LiDAR and RGB camera	LiDAR scans at 10 Hz, RGB scans at 100 Hz (check these numbers), so pixels may be assigned incorrectly, resulting in a warped/inaccurately sensor-fused map	T	SPR3, SPR5	3	2	- Delay RGB colorization until LiDAR point cloud map developed or attempt to predict the UAV's pose in the future based on current velocity/FOV & colorize immediately - Throw out extra RGB and synchronize with LiDAR - Trigger the cameras with

							the LiDAR
6	Pilot crashes UAV during flight	Vehicle damage, bodily, property harm possible	T, S, B	MPR1	3	5	- Get extensive experience flying UAVs for all team members - Emergency landing procedure
9	BLAM cannot be integrated into RASL mapping framework	Mapping framework incompatible or too difficult to integrate with BLAM	T	SPR3, SPR5	3	2	- Find another framework to use - Modify current framework code to make it more compatible with our purposes
10	Robot crashes while flying autonomously	Vehicle damage, bodily, property harm possible	T, S, B	SPR1	3	5	- Testing in simulation - Extensive testing indoors in a safe environment

10. Conclusions

In this project Arcus has demonstrated the functional capability of a teleoperated drone to generate colorized point cloud maps, with extensibility towards alternate imaging modalities. Additionally, the project has shown in simulation that the platform and software stack is extensible towards full autonomous capability. With the validation experiments shown above, the vehicle and system have passed all required validation criteria for this project.

That said, there are multiple challenging factors that complicated the development of the system. One issue, as mentioned before, was a shifting set of requirements which resulted in some time lost chasing unnecessary capability. During the vehicle development process, procedural and bureaucratic steps required to gain access to software resulted in additional latency and lead time for software development. However, this is perhaps countered by a quicker development cycle for a larger project due to better software engineering practices.

The largest issue was simply the overhead necessary in the testing and development of a large, flying hexrotor vehicle. Due to its size and rapidly rotating propellers, the vehicle presented a very real and ever present safety hazard. Thus testing was always delegated to the PRL cage or LaFarge Duquesne quarry, the latter of which would often require 4+ hours per session to test. In order to facilitate quicker and more frequent testing, more time should have been spent piloting smaller quadrotor vehicles to gain experience and minimize dependency on the PRL resources. Finding a closer test site for full system integration and functionality testing would also accelerate the development cycle.

There is a clear path set forward for continued development of the full autonomous functionality of the aerial vehicle. After this initial mapping functionality with teleoperation has been proven, final integration work still needs to be done to test the simulation flight controller with the physical hardware. Full autonomous capability can be developed following this. While the task is seemingly minimal, there is still much safety, system validation, and system verification testing necessary to prove flight safety both in the field and in the PRL cage. Despite this, the team has successfully demonstrated the full functionality of the aerial mapping requirements set forth in the project proposal, and made a significant contribution towards the ultimate goal of extraplanetary discovery and exploration.

11. References

[1] “USB2.0 camera mvBlueFox.” [Online]. Available: https://www.matrix-vision.com/USB2.0-industrial-camera-mvbluefox.html. [Accessed 10-May-2017].

[2] “Hero4 Black” [Online]. Available: https://shop.gopro.com/refurbished/refurbished-hero4-black/CHDNH-B11.html. [Accessed 10-May-2017].

[3] “See3Cam - USB 3.0 Cameras.” [Online]. Available: https://www.e-consystems.com/See3CAM-USB-3-Camera.asp. [Accessed 10-May-2017].

[4] “Sentera Multispectral Double 4k Sensor AG” [Online]. Available: https://sentera.com/product/sentera-multispectral-double-4k-sensor-agriculture/. [Accessed 10-May-2017].

[5] “Parrot Sequoia.” [Online]. Available: https://www.parrot.com/us/Business-solutions/parrot-sequoia#parrot-sequoia-. [Accessed 10-May-2017].

[6] “Tau 2 Uncooled Cores.” [Online]. Available: http://www.flir.com/cores/display/?id=54717. [Accessed 10-May-2017].

[7] “RedSense - MicaEdge.” [Online]. Available: https://www.micasense.com/rededge/. [Accessed 10-May-2017].

[8] P. P. D. Autopilot, “Pixracer · PX4 Developer Guide.” [Online]. Available: https://dev.px4.io/en/flight_controller/pixracer.html. [Accessed: 10-May-2017].

[9] “ESC32 Version 3 | AutoQuad.” [Online]. Available: http://autoquad.org/wiki/wiki/aq-esc32/esc32-version-3/. [Accessed: 10-May-2017].

[10] R. M. Rc, LLC, and the Best Source for All Your FPV Needs, “Tiger Motor - U3 KV700 [U3-KV700] - $109.90 : Ready Made RC LLC, The Leader in All Things FPV, RC, and Beyond.” [Online]. Available: http://www.readymaderc.com/store/index.php?main_page=product_info&cPath=505_506_516_528&products_id=2640. [Accessed: 13-Dec-2016].

[11] “Rapidly-Exploring Random Trees: A New Tool For Path Planning.” [Online]. Available: http://msl.cs.uiuc.edu/~lavalle/papers/Lav98c.pdf. [Accessed: 10-May-2017].

[12] “Mini-PC System (BRIX).” [Online]. Available: http://www.gigabyte.us/Mini-PcSystem/. [Accessed 10-May-2017].

[13] “Puck VLP-16.” [Online]. Available: http://velodynelidar.com/vlp-16.html. [Accessed 10-May-2017].