The Robotics Institute, Carnegie Mellon University

Critical Design Review Report
May 9, 2020

MRSD Project - Pit Navigator

Team G - The Pit Crew
Alex Withers
Awadhut Thube
Justin Morris

Sponsor - Dr. William Whittaker

ABSTRACT

Orbital imagery of the moon has identified several deep pits in the lunar surface, and there is great interest in exploring these pits to determine if they could provide access to subsurface caves where future human habitats could be constructed. A team at CMU proposes to explore one such pit by means of a small, autonomous rover that would circumnavigate the rim of the pit and capture images of the interior walls. Such a mission would require specialized planning and navigation code to keep the rover safe in the treacherous terrain near the pit and manage the process of feeding data back to Earth via a radio-equipped lander module. This project is concerned with designing the software functions necessary for the rover to operate in the pit vicinity, including brinkmanship navigation, path planning, and risk assessment. The scope of the project also includes the development of a rover surrogate and a simulated lunar environment, both used to enable testing of the software that is the focus of the project. This report details progress on the project from its inception to the date of the first demonstration of the major subsystems, and discusses future work planned for the coming year.

Contents

1 Project Description

Studies of the moon's surface have led scientists to hypothesize the existence of a network of sublunarean tubes hundreds of feet across, extending for kilometers below the regolith. If these tubes exist, they could serve as the foundation of future human habitats on the moon. They would provide an enclosed space that would shield occupants from the radiation and extreme temperatures of the moon's surface, and could be sealed and pressurized with breathable air.

Although the existence and extent of these tubes remains a theory, there is direct evidence showing that the moon's surface is studded with large pits, some dozens of meters across. These pits sink deep enough into the moon that they could potentially connect to the tube network. Because of the exciting potential of the sublunarean tube structures, these pits have become a top priority for future research and exploration.

Astrobotic Technology, in cooperation with Carnegie Mellon, proposes to send a small, fast, autonomous rover to explore these moon pits and collect data about their composition and structure. This rover will have between one and two Earth weeks to travel to and collect images of a moon pit near its landing site. This process will take the robot beyond the range of wireless communication with the lander craft, which is its only connection back to Earth. Therefore, the rover must be able to operate autonomously while executing its mission, and return back to the landing site safely.

Any autonomous mission that must operate in the vicinity of a moon pit must have special routines for navigating around the pit edge in a safe and efficient manner. This project proposes to design and implement software that will identify a pre-selected pit from images collected by the rover's cameras, construct a model of the pit's dimensions and location from these images, and generate safe routes to a series of vantage points around the edge of the pit. From these vantage points, the rover will collect images of the opposite wall of the pit, aiming to photograph as much of the pit's circumference as possible over the course of its mission duration.

2 Use Case

A small autonomous rover is exploring the surface of the moon. It arrived aboard a lander, which set down near the known location of a pit. One of the rover's directives is to gather imagery of this pit and transmit that imagery to the lander, which will then forward the collected data on to Earth. The rover was deployed from the lander, and has since been executing various mission objectives, the most recent of which was to autonomously travel to a waypoint set for it by human operators on Earth. This waypoint is in close proximity to the pit, and is outside of the communication range of the lander, meaning that the rover must operate with complete autonomy during its trek to this waypoint.

The pit's location is defined by a map of the area around the landing site that was generated before the mission launched. Prior to the start of the rover's mission, the lander provided the rover with exact coordinates of the lander's location in this map. The rover localizes itself and the pit in relation to these coordinates for the duration of its mission.

Upon arrival at the waypoint, the rover software activates the Pit Navigator routine, which employs specialized behavior designed for successful navigation in the vicinity of the pit. First, the rover trains its cameras in the direction of the pit's expected location. It collects images of its surroundings, and attempts to identify the pit from those images. The rover calculates an estimate of the location and dimensions of the pit, which is incorporated into the rover's map of its surroundings.

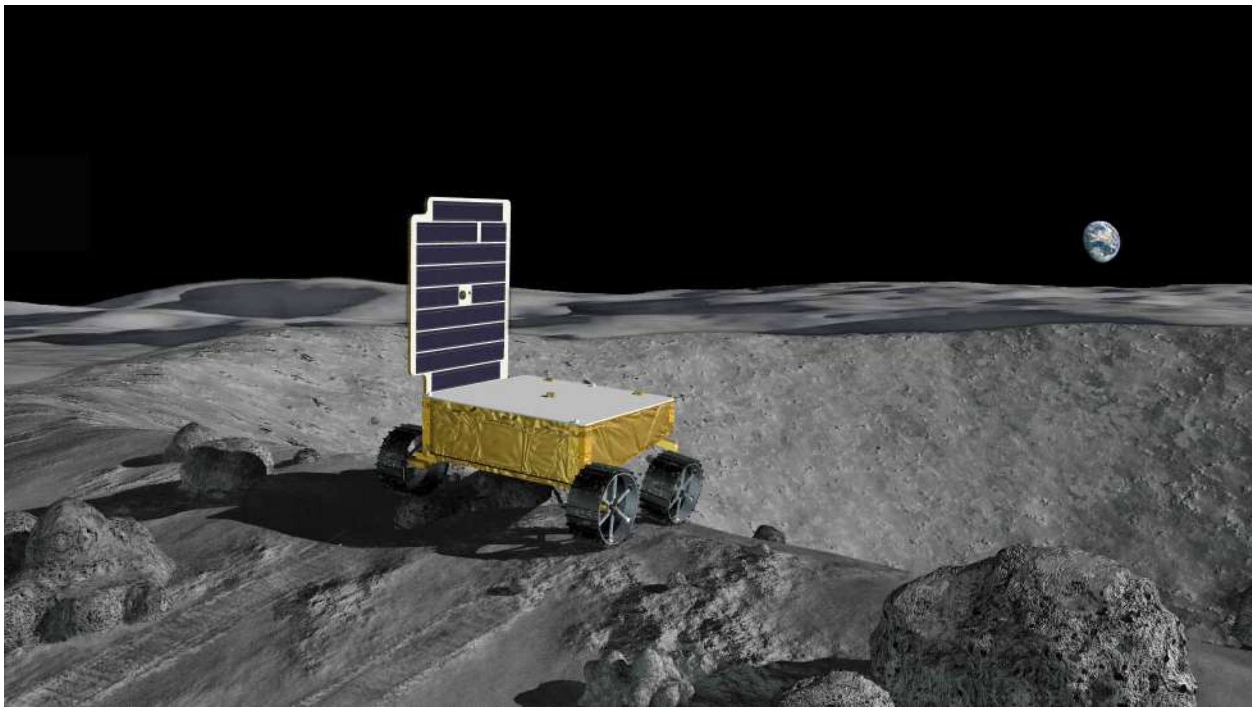


Figure 1: Rover at Pit Edge

As soon as the rover has modeled the pit in its global map, it autonomously selects a new waypoint at the edge of the pit. A local planning algorithm generates a route to this waypoint consisting of multiple small movement steps. The algorithm avoids any obstacles detected in the rover's vicinity and prevents the rover from moving too close to the pit edge, then selects the best route to the destination position with these constraints applied.

The rover begins to execute the planned route to the destination waypoint. After each movement step, the rover uses its navigation cameras to collect information about the area in front of it, and uses its pit imagery camera to capture new images of the pit. Both the pit images and the navigation images are used to update the planned route in order to account for unforeseen obstacles and dangers.

When the rover reaches its destination, it uses the pan and tilt actuators on its pit imagery camera to align the camera such that the camera's field of view is trained on the pit wall opposite the rover's location. The camera pans from side to side, taking images at a range of angles in order to capture an arc of the pit wall. Each of these images is incorporated into the rover's model of the pit. Once sufficient images have been captured, the rover uses its model of the pit to select a new waypoint at a different point on the pit's circumference, and plots a new path to reach that waypoint. This path takes the rover away from the pit edge before moving around the perimeter of the pit, to ensure that the rover will not fall into the pit while navigating to the next waypoint.

The rover will continue to select waypoints around the edge of the pit and take images at each point until the sun sets and the rover runs out of battery power, with the goal of collecting data on as much of the pit's wall area as possible. At all times during its mission, the rover is measuring how much data it has collected. In order to ensure that as much of this data as possible is transmitted safely to Earth, the rover will leave the vicinity of the pit at regular intervals in order to return to within communication range of the lander and deposit all collected data to the lander. When the rover leaves the vicinity of the pit, the Pit Navigator routine ends and control is returned to the standard moon operation subsystems. The data transmitted to the lander includes rover telemetry and navigation images in addition to the pit imagery collected at each of the waypoints. Once data is stored on the lander, the rover will return to the pit once again to collect more imagery for as long as it has sufficient power to continue operating.

3 System Level Requirements

The following requirements are derived from an objectives tree by taking in consideration the mission goals and sponsor expectations. The requirements are categorized as mandatory and desirable requirements. These categories are further divided and the requirements are classified as performance requirements which are functional requirements with an associated performance measure and non-functional requirements.

Our performance requirements are mainly concerned with the amount of data captured and also with the reliability of our system.

3.1 Mandatory Performance Requirements

The system will:

M.P.1 Capture **500 MB** image data on a surface similar to the moon terrain.
This specifies the total data captured during the mission should be more than 500MB. We arrived at the specific number assuming that the camera has very high resolution (assumption is that we capture 4K images).

M.P.2 Capture **75 MB** image data over a single cycle in the mission.
A mission cycle is defined as one complete loop which consists of the data collection activity and the data dumping activity of the rover. We expect to complete 6 such cycles and collect around 75MB of data in each cycle.

M.P.3 Capture **15 MB** image data from specific co-ordinates on the surface of the moon.
Within a single cycle, the rover visits multiple waypoints and gathers data from each of these waypoints. This requirement specifies that it must record 15MB worth of data at each waypoint.

M.P.4 Calculate the relative distance to the pit edge within **2% error**.
While determining the presence of an edge in the surrounding, our system must also estimate it distance to the edge with 2% of error.

M.P.5 Calculate an optimal navigation plan within **20 seconds**.
Before moving to the next waypoint, our system must compute an optimal navigation plan within a time frame of 20 seconds.

M.P.6 Capture images covering **20° angle** of pit circumference from one position.
Once the rover reaches near the edge of a pit, it captures multiple images of its surrounding. This requirement specifies that it must cover about 20° angle of pit circumference from one position.

M.P.7 Operate such that chance of occurrence of a mission ending incident is less than **5:1**
This is one of our most important requirements which emphasizes on the reliability of the system.

Apart from the above mentioned functional requirements, our system should meet the following non-functional requirements. The non-functional requirements stem from the fact that our system is a part of a larger mission and must adhere to the operational standards and interfacing capacity of the larger system.

3.2 Mandatory Non-Functional Requirements

The system shall:

M.N.1 Operate in the vicinity of a pit on the moon.
The system is designed with a very specific use case in practice. However, it may not be feasible to test this use case given the scope of our project. Hence, we have this as a non-functional requirement.

M.N.2 Operate using hardware that meets specifications of overall rover design and mission.
Along with demonstrating the proof of concept, we would like to replicate the mission as closely as possible. Thus, we plan to operate within the specifications of the overall rover design.

M.N.3 Operate within a Linux operating system environment
Initial work in this project was carried out in the linux operating system environment. Hence, operating with the same environment adds as one of our functional requirements.

M.N.4 Be compatible with other software systems running on the rover.
Before we started working on the project, some basic functionalities were set up through previous work on the project. Hence, we aim to utilize those as much as possible and develop our own system accordingly.

M.N.5 Maintain a mission clock.
Maintaining a mission clock is essential for mission planning and this is incorporated as one of our non-functional requirements.

M.N.6 Operate when rover is not experiencing any major subsystem faults.
Finally, our system shall operate efficiently and reliably in the absence of any major subsystem faults.

Note: The requirements with a strike-through have modified performance parameters or have been eliminated from the list.

In addition to the mandatory performance and non-functional requirements, we have also identified certain desirable requirements. These additions are nice to have and extend the project scope in exchange for having a more robust, reliable and valuable system. The desirable requirements are formulated to extract the greatest amount of information even when operating under different conditions.

3.3 Desirable Performance Requirements

The system will:

D.P.1 Operate at a distance of **0.75 meters** from the pit edge 80% of the time.
This requirement is designed to strike a balance between allowing the rover to remain at a safe distance from the pit edge whenever possible, while also keeping the rover near the pit to minimize the travel time between waypoints.

D.P.2 Estimate the shape and size of the pit within **10% error**.
An aspect of the mission that is being developed in parallel to our project is the creation of a 3D model of the pit based on the images of the interior that the rover captures. We hope to be able to use this model to improve the accuracy of our planning map, so that the waypoints chosen are appropriately placed around the rim of the pit.

D.P.3 Capture high resolution images of the pit with each image being **18 MB** in size.
The better the resolution of the images captured, the more valuable those images are to scientific research, and to the creation of the 3D model. However, the rover has limited storage, and the lander's connection to Earth has limited bandwidth, which will reduce the maximum amount of information that can be communicated.

D.P.4 Capture data such that for **80%** of the images **60%** of the image will show the pit.
At each waypoint location, the rover will take images at a range of pan and tilt angles. These angles could be hardcoded, but that creates a risk that some images could capture areas of the sky, particularly if the rover is at an unusual angle when it reaches the waypoint.

3.4 Desirable Non-Functional Requirements

The system shall:

D.N.1 Operate given pits of different sizes and shapes.
The Pit Navigator system should be robust and flexible enough to manage the task of navigating around pits of varying sizes and irregular shapes.

D.N.2 Take rover parameters and state into account during motion planning.
The rover will continuously update its global and local plan as it proceeds with its mission. This recalculation should take into account the state of the rover as it changes. If systems are damaged or unresponsive, that should alter the rover's planned behavior to minimize risk.

4 Functional Architecture

Our functional architecture has covered three main subsystems from the beginning. The architecture outlined below shows these subsystems and the important functions that the system must execute to fulfill the previously mentioned requirements. The functions are derived assuming that the robot is already in the vicinity of the pit.

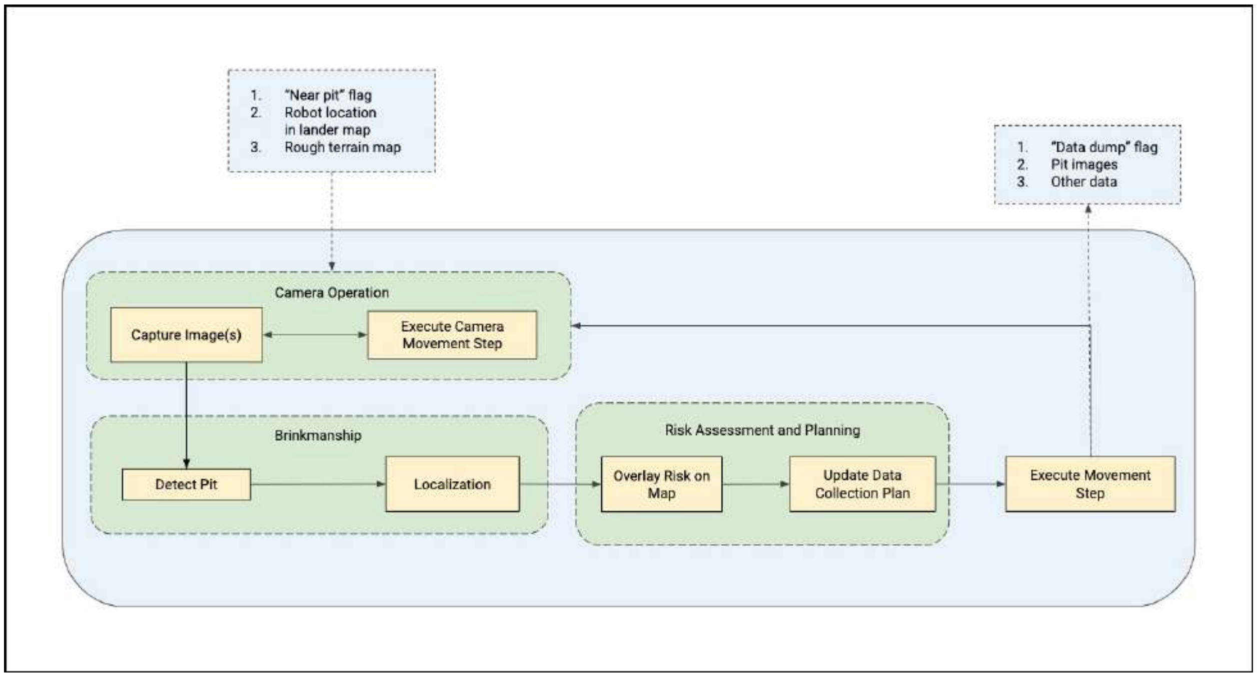


Figure 2: Functional Architecture

The pit exploration phase of the mission consists of the following essential functions:

- **Capture Images:** The robot captures images of its surroundings at regular intervals. During navigation, the camera faces forward and down to capture the terrain in front of the rover. The camera data is used for obstacle avoidance during local path planning. During pit image capturing, the camera captures images at a variety of positions and exposures to maximize the amount of data collected about the pit interior.
- **Execute Camera Movement Step:** This is where the rover will turn the camera to take pictures from different viewpoints/angles. During navigation, this step will be skipped. During pit image capturing, the camera will move to preset angles or align itself with the opposite rim of the pit. The goal of moving the camera is to cover the largest percentage possible of the pit interior wall from each vantage point.
- **Detect Pit:** The rover generates a 3D point cloud of its immediate surroundings using the stereo images returned by the Camera Operation subsystem. It applies one or more heuristics to the point cloud to determine if a brink is within sensor range. Currently, the heuristic simply counts the number of points within the point cloud and indicates a brink if that number drops below a threshold. Future work will include developing more advanced heuristics.

- **Localization:** The rover determines its position with respect to the brink. If the rover's destination position is beyond the brink, the rover will stop short of that position. The rover may update its pre-determined map of the lunar terrain, including the rover's location in that map, based on where it perceives the brink to be. If the pit is not visible from the rover's current position, the rover will still capture data about any obstacles in its surroundings and use that data in planning its next movement step.
- **Overlay Risk in Map:** As the rover proceeds with its mission, it constantly monitors multiple risk factors. These include the amount of data currently stored on the rover, the position of the sun over time, and the rover's position around the circumference of the pit. These risk factors are used to adjust the weights of the map which the rover uses to plan its routes.
- **Update Data Collection Plan:** As the planning map gets updated, the rover must recalculate its planned waypoints. While the risk remains low, the rover will continue to travel to pre-selected waypoints around the rim of the pit, collecting data at each point. If risk is determined to be too high, the rover will stop circumnavigating the pit, and will travel back to the lander to deposit data. After storing data on the lander, it will return to the pit to continue data collection for as long as it has sunlight to power itself.
- **Execute Movement Step:** The rover actuates its motors to execute the next movement step towards its current destination waypoint. The size and speed of this motion may vary depending on the rover's current terrain. For example, near the pit it may be appropriate to move more slowly while approaching the pit edge.
- **Interfacing Operations:** The entirety of the Pit Navigator project will constitute one software subsystem within the PitRanger system, which comprises the whole of the functionality for the moon rover. The Pit Navigator system will control the rover during the period of its mission where it is in the vicinity of the pit, during which time it will interact with other PitRanger subsystems. Functionality like SLAM and low-level motor control, which are needed during all stages of the PitRanger mission, may be implemented by teams working outside of the scope of the Pit Navigator project, and so the Pit Navigator subsystem will communicate to other parts of the software in order to operate these functions.

5 Cyberphysical Architecture

Our cyberphysical architecture has direct correspondence to our functional architecture. It highlights the technology options which will be potentially used to execute the individual functions. The inputs and outputs of individual functions are also included.

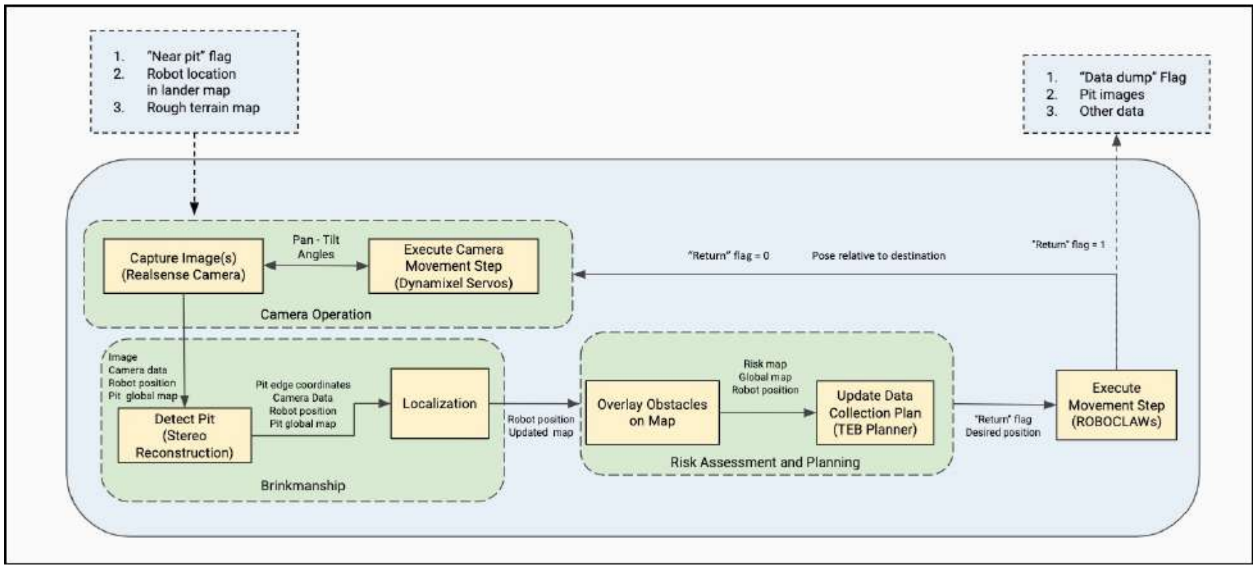


Figure 3: Cyberphysical Architecture

- **Capture Images:** An Intel Realsense camera allows to stream images of the surrounding.
- **Execute Camera Movement Step:** A camera mount having pan and tilt motions will be used. Dynamixel servos are used to provide the pan-tilt motions.
- **Detect Pit:** This will be executed through 3D reconstruction using stereo images.
- **Localization:** We aim to localize the position of the robot with respect to the pit edge using the pointcloud reconstructed from stereo images.
- **Overlay Risk in Map:** A heuristic solution will be employed for estimating the risk involved in continuing the mission.
- **Update Data Collection Plan:** The TEB local planner is used to calculate an optimal path that the rover traverses.
- **Execute Movement Step:** Once the decision is made, the rover either goes back to the starting point or continues to gather more data from different positions.
- **Interfacing Operations:** When the rover reaches the vicinity of the pit, the Pit Navigator system will activate, taking as an input the rough terrain map and robot pose established by the overarching PitRanger system. While the Pit Navigator system runs, it will continually pass collected data including images of the pit to other subsystems within PitRanger.

One loop through our cyberphysical architecture describes one cycle in the pit exploration mission. The robot continuously captures stereo images of its surrounding while navigating at a slow speed. It then constructs a point cloud using the stereo images. The robot also tries to detect (localize) the pit edge using the point cloud. Once the pit edge is detected, the robot will calculate the distance between the pit and the camera frame. Thus, the robot localizes itself with respect to the pit edge. It also updates the global map with the newly computed coordinates of the pit edge.

After the map update is complete, the robot moves in close to the pit edge while continuously measuring its distance from the edge. During this process, the robot estimates the risk of moving close to the edge and decides whether to get closer to acquire better data. Upon getting sufficiently close, the robot captures multiple images covering a large area of the opposite wall of the pit. This will be achieved by having pan and tilt motions for the camera capturing images of the pit. When the required number of images have been captured the robot updates it current state and tracks the mission status. Based on these parameters, it decides whether to navigate to a new waypoint to capture more data or to go back to the starting location and cede control to the standard navigation system for return to the lander.

Subsystem	**Function**	**Inputs**	**Outputs**
Camera Operation	Execute Camera Movement Step	1. Pose Relative to Destination	1. Camera Pose
	Capture Images	1. Capture Flag	1. Images 2. Camera Data
Brinkmanship	Detect Pit	1. Captured Images 2. Camera Data	1. Point Cloud
	Localization	1. Robot Pose in Global Map 2. Global Map	1. Robot Pose Relative to Pit 2. Updated Global Map
Risk Assessment and Planning	Overlay Risk on Map	1. Rover/Mission Parameters 2. Potential Waypoint	1. Updated Risk Map
	Update Data Collection Plan	1. Updated Risk Map	1. New Data Collection Plan 2. Optimal Path

Table 1: Major Inputs and Outputs of Individual Functions

6 Current System Status

6.1 Targeted Requirements

Targeted requirements are focused on and have subsystems built to satisfy those requirements. After reviewing our performance requirements, we decided that requirements M.P.3, M.P.4, M.P.5, M.P.6, and M.P.7 were the most important to work on first. To review, our requirements were:

M.P.1 Capture 500 MB image data on a surface similar to the moon terrain.
M.P.2 Capture 75 MB image data over a single cycle in the mission.
M.P.3 Capture 15 MB image data from specific coordinates on the surface of the moon.
M.P.4 Calculate the relative distance to the pit edge within 2% error.
M.P.5 Calculate an optimal navigation plan within 20 seconds.
M.P.6 Capture images covering 20 angle of pit circumference from one position.
M.P.7 Operate such that chance of occurrence of a mission ending incident is less than 5:1.

For our first spring validation test, we targeted requirements M.P.3 and M.P.6. To verify these requirements we built a pan-tilt camera operation subsystem. This subsystem could pan across 180 degrees of the pit, and tilt 180 degrees to capture the entire vertical of the viewpoint. The subsystem could also take images at a set of locations across the image, and vary the exposure time to account for various lighting conditions. This subsystem allows M.P.3 to be fulfilled once a specific sequence is written to capture the necessary images. However, this subsystem by itself does not fulfill M.P.6 as it assumes the position of the rover is at the edge of the pit and has a line of sight around the pit. Our next test attempts to fulfill that assumption.

During our second spring validation test, we targeted requirements M.P.4 and M.P.7. To verify these we acquired and improved a brinksmanship algorithm that will nose up to the edge of the pit using stereo vision. It would use stereo matching from the two cameras to estimate the edge of the pit and its relative distance. This system can directly fulfill the requirement M.P.4 once a good metric is found to describe the relative distance. Finding a good metric could be tricky when considering all the edge cases. Knowing the distance to the pit, and knowing when to tell the robot to stop moving helps fulfill M.P.7 when approaching the pit but is not sufficient for the mission risk.

During our final spring validation test, we targeted M.P.5 and M.P.7. To verify these requirements, we used an add on to the ROS navigation stack, TEB local planner, to have the robot plan through the environment and develop safe navigation routines around the pit. This navigation subsystem satisfied M.P.5 for now, but there will be more to plan around once the risk assessment algorithm is in place. Keeping a safe distance when navigating around the pit helps fulfill M.P.7 but is not sufficient for mission risk even when combined with the brinksmanship algorithm.

In the next semester, we will target M.P.1, M.P.2, M.P.7. We will target these requirements with system integration and more subsystems. The proposed risk assessment algorithm will target all three of these requirements as it will define how many waypoints to capture pictures before heading back to the lander. It will also define how many trips to make to the lander to ‘save’ the data it has gathered. We will also implement an obstacle avoidance algorithm to target M.P.7.

6.2 Subsystem Descriptions

6.2.1 Camera Operation Subsystem

The camera operation subsystem is one of the simpler subsystems within the Pit Navigator project, but it is crucial to the function of many of the other components. For Blue 2, which uses a single camera for both navigation and data collection, the camera operation subsystem must ensure that the camera is constantly in the proper position and feeding images to other subsystems.

The mechanical components of the camera operation subsystem are a RealSense D435i camera, a PhantomX pan/tilt turret, and an Arbotix-M Robocontroller. The camera is supported on a custom 3D printed mount which is attached to the pan/tilt turret. The Arbotix-M runs an Arduino-based program that allows it to communicate with the main rover computer. Through a ROS package called arbotix-ros, the main computer can send commands to the Arbotix-M to set the angles of the pan and tilt servos. The RealSense camera is mounted to the top of the turret and connected directly to the main computer via a USB-C cable.

During navigation, the camera faces straight ahead of the rover, angled to capture the terrain that the rover is traversing. The rover will use the camera stereo images and resultant point cloud to generate a local plan to navigate through its environment and avoid obstacles. When the rover nears the pit edge, the brinkmanship subsystem will use the point cloud to drive the rover as close to the pit edge as possible without endangering the rover. The rover will then use the pan/tilt turret to capture images of the pit interior at a range of angles and exposures.

This subsystem is effectively complete. The camera and turret have been mechanically and electrically integrated onto Blue 2. Software has been developed to control the exposure of the RealSense camera and transform stereo images into a dense point cloud which can be used by the brinkmanship subsystem. The commands to control the turret position are well-understood and were used to enable the Image Capture and Pit Identification tests during the Spring Validation Demonstration. The Image Capture test in particular applied all portions of the camera operation subsystem, and was used to demonstrate their successful integration.

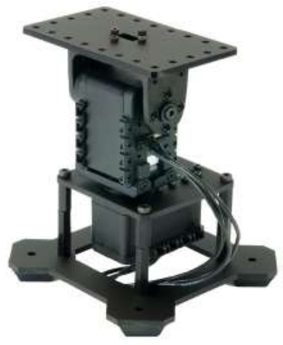


Figure 4: Pan-Tilt Turret

Figure 5: ArbotixM Controller

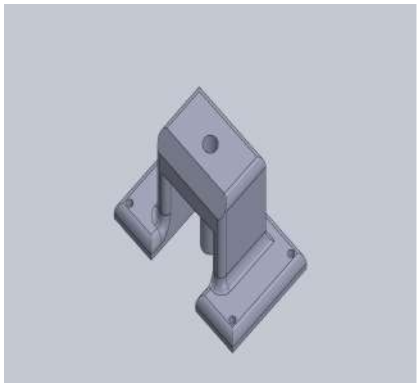


Figure 6: Camera Mount Design

6.2.2 Brinkmanship Subsystem

This subsystem maintains information about the robot's position with respect to the pit. This will involve local brinkmanship checking while the rover is moving to imaging points. The mission plan also includes a plan to assemble and continually update a 3D model of the pit based on images taken by the rover, so this model could be used in the localization process as well.

The brinkmanship system will allow the rover to localize itself on a map which is assumed to be pre-calculated from orbital imagery. The idea is that the rover will be able to move to a predefined location near the pit using this map and then will move close to the pit edge with the help of this subsystem. The brinkmanship system currently uses stereo depth reconstruction to get an estimate of the 3D structure of the surrounding. A pit edge is said to be detected when there is an absence of terrain near the rover which is reflected in the 3D reconstruction. This subsystem will trigger an alert signal in such a case and prevent the rover from getting to close to the pit edge.

In order for this pipeline to assist the rover in navigating its surroundings, we need to identify the characteristics of safe and unsafe terrain, and tune the brinkmanship system so that it can classify the rover's surroundings appropriately. We scouted some locations in Schenley Park which had good examples of safe and unsafe terrain, including small cliffs and slopes of varying steepness. We secured our rover with a rope, to prevent it from being damaged, and then drove the rover over these cliffs and slopes while recording odometry and camera data.

Currently, the brinkmanship subsystem implements a simple heuristic to identify an edge near the rover. The surrounding point cloud is generated from the stereo pair and the number of points in the cloud serve as a metric to trigger the edge alert signal. Once the robot stops near the edge, the camera operation subsystem controls the camera position and captures multiple images of the surrounding. The current operation of the brinkmanship subsystem can be seen in the image shown below.

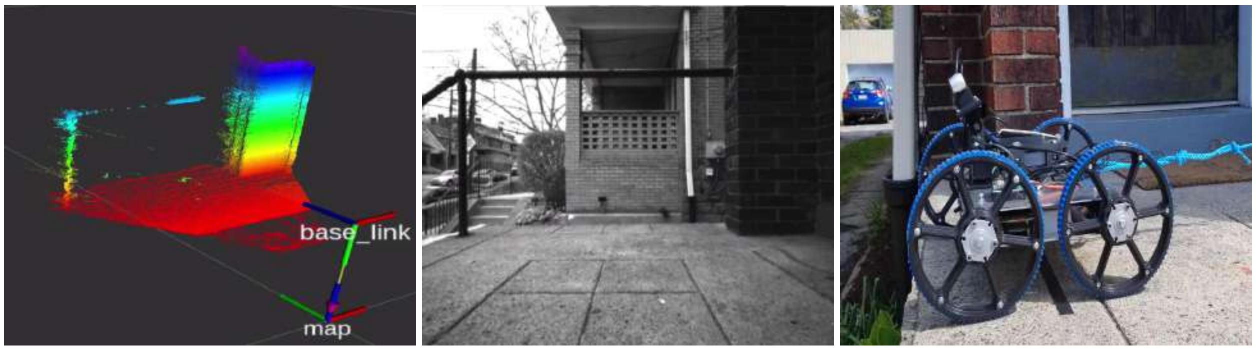


Figure 7: Left: Generated Point Cloud, Center: Rover View, Right: Final Position

The current interdependence and the simultaneous operation of the camera subsystem and the brinkmanship subsystem can be seen through the following flow chart. The rectangular boxes represent the nodes in the system and the elliptical shapes represent the messages.

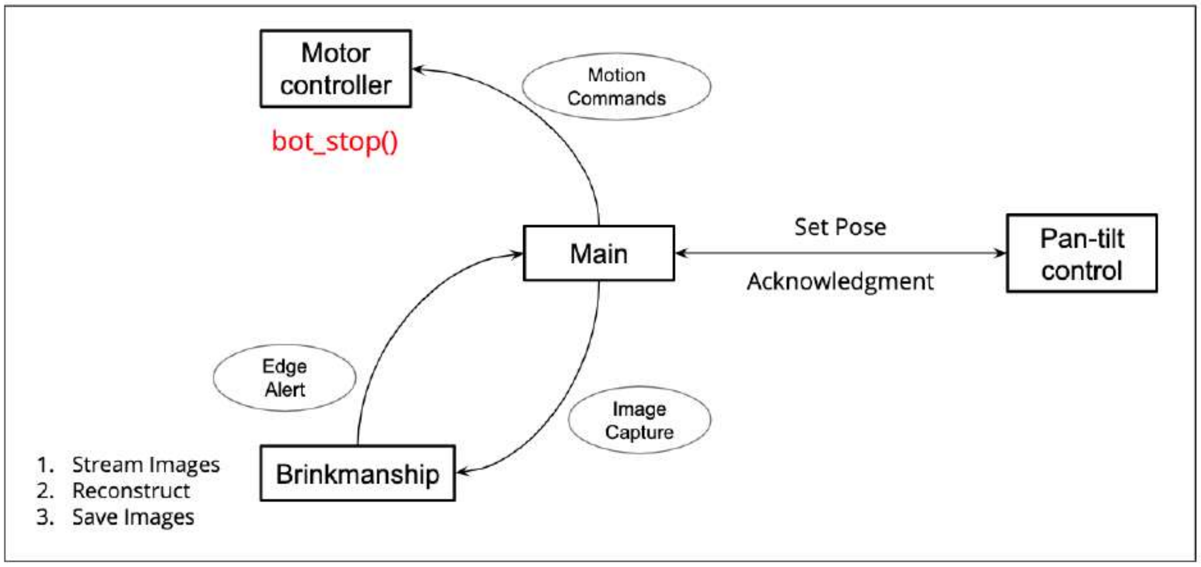


Figure 8: Control Flow Between System Functions

6.2.3 Planning Subsystem

The planning subsystem is critical to the navigation of the Pit Navigator project. This subsystem is responsible for generating the motor commands and navigating around the environment in a manner that will not cause the robot harm. This system takes in a known global map and a local map of its immediate environment and navigates through the local maps, avoiding known obstacles in the local map. The planning subsystem does this by generating motor commands, known as twist messages, that will safely move the rover to its goal location.

The algorithm that does this known as the TEB local planner. This algorithm was chosen over the base ROS planner because it produces optimal plans, which reduces energy use and increases speed made good by only traversing the shortest possible path. While it can take longer to plan an optimal path, we have found that the trade-off of time spent navigating and time spent planning favors an optimal path in our case when there are few obstacles to avoid.

In our current state, the global map also serves as our local map. The local map is an expanded sliding window of the global map and one pixel is turned into a 50 by 50 square. Normally this local map is added to with /obstacles that represent untraversable terrain, but there are no publishers to that topic at this time. That means that this local map stays empty where the global map is empty and the robot will ignore obstacles that it encounters in the environment due to not perceiving and logging them. The obstacle avoidance algorithm that is not implemented will add the perceived obstacles to this /obstacles topic and the local planner will avoid them. We do not expect errors from the planning subsystem when implementing the obstacle avoidance algorithm once the two are integrated.

These twist messages that the planner outputs are accepted by our simulated robot controller and will move the robot in simulation. The motor controller on Blue2 also accepts twist messages,

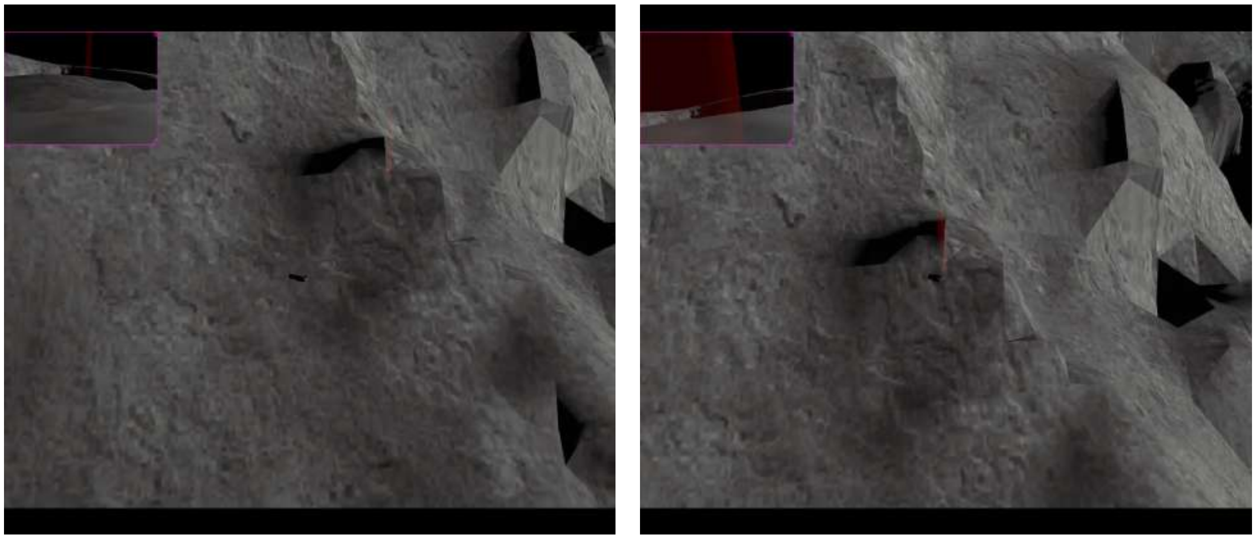


Figure 9: Waypoint Traversal In Simulation

but porting the planner onto the rover has not been attempted. This generalization to twist messages was made specifically to keep the code mostly rover independent, as it is the ROS standard, and will work on any ROS rover.

The planner was demonstrated in the spring validation test 3. The specific validation criteria for the test that pertained to the ability of the navigation software was navigating to each waypoint to within 0.45 meters or a robot length away. The planner passed this criterion and also currently satisfies the performance requirement of planning an optimal path between waypoints in less than 20 seconds. While the subsystem needs integration, the subsystem itself is nearly complete.

6.3 Modelling, Analysis, & Testing

As the Pit Navigator project got underway, some prior work had been done to design a brinkmanship system that used the RealSense depth camera to detect edges in the rover's vicinity. Due to the lack of space-rated depth cameras or LIDAR systems, it was determined that this method would need to be replaced with one that derive similar depth information using only the stereo RGB images produced by the camera. These stereo images would need to be converted into a point cloud which could then be used by the brinkmanship subsystem. This feature was initially designed and tested by capturing images of the MRSD lab to fine-tune the stereo-to-depth pipeline, before being incorporated into the brinkmanship routine later in the semester.

Another experiment that was performed using the RealSense camera was a simple program to control the exposure of the images produced. This was performed because the lunar rover would need to be able to handle a wide range of lighting conditions to produce the largest quantity of useful data. In addition, this provided an opportunity for the team to develop familiarity with the RealSense hardware and API. The lessons learned from this experiment enabled effective use of the RealSense camera in later testing.

The pan/tilt turret chosen to support the RealSense camera on the surrogate rover was the PhantomX from Trossen Robotics. This turret used two AX-12 servos, controlled by an Arbotix-M Robocontroller board. The Arbotix-M could be programmed in the Arduino language, but a premade program that allowed the board to handle commands sent from an external computer also existed. Several programs that interfaced with the Arbotix-M board were also available. One of these, a terminal-based program that provided a limited command set to control the speed, position, and IDs of the servo motors, was used to perform initial validation testing of the assembled turret. Once the motor function had been verified, the terminal program was used as a guide for implementing position control of the turret into the rover code base.

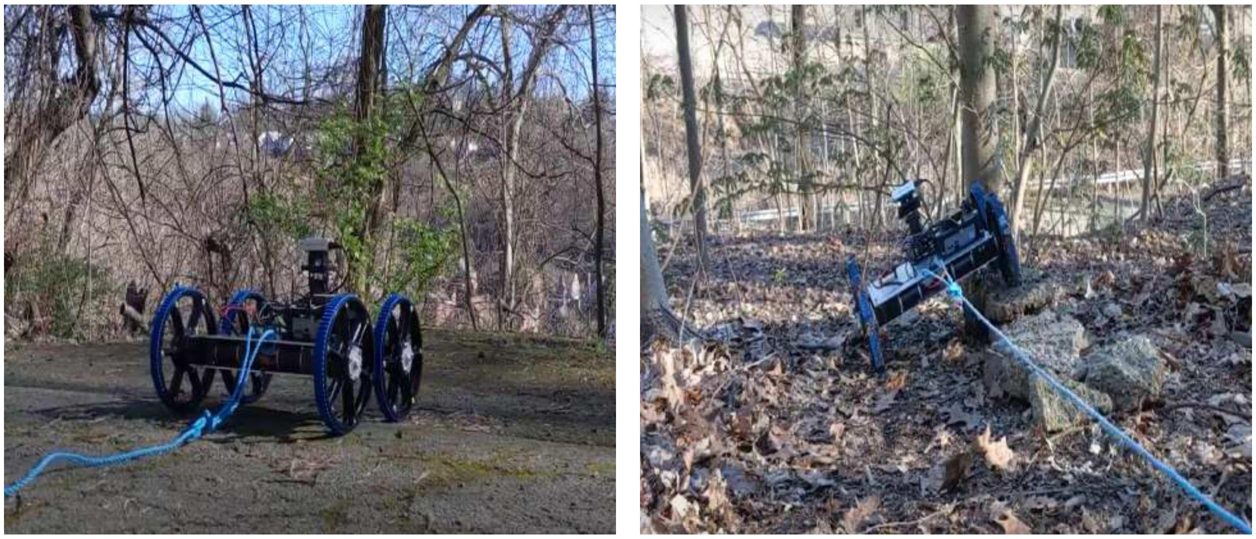


Figure 10: Rover Operation (Schenley Park Field Test)

On April 3rd, the first field test of the surrogate rover was performed. Blue was taken to a location in Schenley Park which had been identified as having ledges and slopes that would be useful for gathering data. No brinkmanship code had been implemented on Blue at this point in the project. Instead, Blue was simply teleoperated to drive forward towards each ledge or slope, while the computer saved data from the motor encoders and the RealSense camera into a rosbag file. A rope was attached to Blue's chassis to prevent it from fully falling over the brink as it approached. Tests were performed with the camera at tilt angles of 0 and 15 degrees. The stereo image data was then reconstructed into a point cloud and used to develop the brinkmanship edge detection criteria used in later tests.

Parallel to the testing using the rover surrogate, the simulation environment was also being developed. A previous MRSD team had done work navigating in a lunar pit environment using the WeBots simulator, so it made sense to build off their progress. The first order of business was to improve the pit model in the simulated environment. By experimentation, it was discovered that the pit could be handled in the same way as the rock objects which already existed in the simulation, and this enabled an existing 3D model of the lunar pit Lacus Mortis to be imported into the simulation. With this environment available for testing, the team was able to use the simulation to test the planning subsystem. These tests consisted of implementing pre-existing planning algorithms that charted a path for the rover between human-selected waypoints, and then having the simulated rover execute the path. In the future, this environment will continue to be used as the planning subsystem develops further.

6.4 SVD Performance Evaluation

6.4.1 Image Capture Test

The first of the Spring Validation Demonstrations that we proposed, referred to as the Image Capture test, was designed to assess the functionality of the camera operation subsystem. The proposal went through several iterations as the project developed and world events restructured our priorities and resources. However, the core concept of the demonstration always remained intact. A program running on the rover computer would identify a feature within the images returned by the camera, then actuate the pan/tilt turret to center that feature within the camera frame. Once the feature was centered, an image file would be saved on the computer.

This test met most of the proposed success criteria. The first requirement was that the turret center the marker within the camera image within 3 seconds of detection. In practice, this process took less than a second. For the marker to be considered "centered", it had to be within 50 pixels of the true image center. The test code set a more stringent requirement that the marker be within 25 pixels of the image center in both the X and Y directions, and the rover accomplished this 100% of the time. Only five centered images were required to be saved to meet the second test requirement, but upwards of 50 images were generated over the course of the live SVD and SVD encore. The last requirement stated that the pan/tilt angles at which the images were taken must be saved. This feature was not added to the code for the Image Capture test, and so this requirement was not met. However, pan and tilt angles corresponding to captured images were saved in the Pit Identification test, and it would be a simple task to add that functionality to the Image Capture test code if necessary.

Success Criteria	Evaluation
Pan/tilt turret moves so that tag is within 50 pixels of image center within 3 seconds; image is captured when tag is centered	1. Tag within 50 pixels of center 2. Centering within 3 seconds
Save >= 5 images captured at different angles in local directory on rover computer	1. >5 images captured 2. Images taken at different angles
Save the data of the pan/tilt angles corresponding to the images	1. Correct pan/tilt angle calculated 2. Angles not saved (saved in Test 2)

Table 2: Image Capture Test Performance Evaluation

6.4.2 Pit Identification Test

The pit identification test was formulated to evaluate the performance of the brinkmanship subsystem. The main objective of this test was to evaluate whether our system reliably detects the presence of an edge in its surrounding. We performed the test with different settings like camera position and thresholds used in our heuristic for determining an edge. Later we fixed the parameters used in our heuristic and recorded the results for different camera positions. The test had a couple of success criteria for each trial and the test was assumed to be successful if these criteria were met for 5 successive trials. The recorded results of 8 successive trials are shown below.

Test	Tilt Angle (degrees)	Response Time (microseconds)	Stopping Distance (cm)
Run 1	25	191	6.5
Run 2	25	135	6.5
Run 3	25	236	5
Run 4	25	159	3
Run 5	25	120	5
Run 6	15	122	26.5
Run 7	15	252	26
Run 8	15	121	23

Table 3: Recorded Data From 8 Trials

We conclude that the rover achieves a stopping distance of less than 0.25m when the camera is tilted at an angle of 25 degrees. However, the rover stops just behind the required distance when the camera is tilted at an angle of 15 degrees or less. The stopping time required is calculated as the time between the reception of the alert signal and the execution of the stop command sent to the motors. In all the cases, a stopping time of fewer than 300 microseconds was achieved whereas the success criteria was at 0.5 sec. The SVD performance evaluation is shown in Table 4.

6.4.3 Simulation Planning Test

The planner was demonstrated in the spring validation test 3. There were 3 validation criteria and only two that were expected to pass. The three validation criteria are listed below.

1. Speed made good of 0.21m/s over all of the waypoints
2. Locally navigate until 1 meter from the edge of the pit
3. End navigation .45 meters from the center of the waypoint

By making the speed made good 0.21m/s we are saying that we will average 30% of our max speed. This criterion is based on the speed made good of 50% of the max speed of the Moon-Ranger. Pit-Navigator expects to be going slower because of the additional tasks like imaging the pit and carefully approaching the pit. While our target speed made good is 0.21m/s we were able to achieve 0.38m/s, passing 50% of the max speed. This is due to not doing some of those extra actions like taking pictures of the pit. While not a complete evaluation of the integrated speed made good, for now, the planning subsystem is successful.

The second criterion, locally navigating to 1 meter from the edge of the pit, was the criterion we did not expect to pass during this test. We did not expect to pass because of the limitations of correctly setting the global map to the real-world terrain. The exact location of the global map and the rover is difficult to make out from orbit. We expect there is some distortion in the scale, shape, or distance between the two. Any mistake made by humans is on the scale of meters from the exact location of the edge of the pit and failing this criterion after trying our best only exemplifies the need to implement our brinksmanship algorithm. Our target was 1 meter from the edge of the pit. We were able to achieve 5.32 meters from the edge of the pit after a ground truth comparison and a second attempt, an act that would not be available during the mission. System integration with the brinksmanship algorithm should allow the criterion to be passed.

The final criterion specifically tests the planning algorithm and how well it can execute in the simulation space. Ending navigation 0.45meters from the center of the waypoint is ending a robot length away from a waypoint after traveling 700 meters or more is a feat and one that the algorithm passed splendidly. The target for ending navigation was 0.45 meters, and the planning algorithm ended navigation an average of 0.406 meters from the waypoint center. In summary, the planning algorithm passed 2/3 of the success criterion. This is good news and with system integration should be able to pass 3/3 of the criteria.

Success Criteria	**Evaluation**
Rover stops within 0.5 seconds of edge identification	1. Average stopping time for 8 tests = 200 microseconds
Rover stops within 0.25 meters of cliff edge	1. Meets criteria for camera tilt = 25 Degress 2. Exceeds threshold for camera tilt = 15 Degrees
Rover meets above criteria on 5 successive tests	1. Met criteria for 5 successive tests at camera tilt of 25 deg.

Table 4: Pit Identification Test Performance Evaluation

Success Criteria	**Evaluation**
Speed made good of 0.21m/s over all waypoints	1. Speed made good of 0.38m/s
Locally navigate until 1 meter from the edge of the pit	1. Average Distance: 5.32 meters
End navigation .45 meters from the center of the waypoint	1. Average Distance: .406 meters

Table 5: Simulated Planning Test Performance Evaluation

6.5 Strong/Weak Points

The hardware design aspects of the project are the most complete at this point. This is partially because our intention with the surrogate rover and camera was merely to give us access to a platform with which to test our software, not to design a mission-ready hardware system. Therefore, the requirements to consider the hardware aspects of the project "complete" are generally less stringent. Furthermore, the need to have these hardware subsystems functional in order to perform our various tests meant that this portion of the work needed to be front-loaded in the project schedule, so that software development would not be held up by testing difficulties later on.

The remaining hardware-related work falls into two categories. The first is improving component performance. During brinkmanship testing, it was observed that the rover would veer slightly to the right when commanded to move straight, an issue which can most likely be resolved by improving the motor tuning. The second category of improvement is replacing existing hardware components with versions that are closer to mission-analog. The highest priority in this regard is to build and run our software on the Jetson TX2 computer, instead of the Gigabyte NUC that we are currently using. We know that the TX2 is the intended flight computer for the eventual moon mission, so it is important that we demonstrate our software's compatibility with that system.

The issue of development on other aspects of the wider Pit Ranger project still being nebulous or incomplete is why we have rated the "Interfacing Operation" aspect of our project as a weak point. It is difficult to establish the interfaces between the Pit Navigator system and the other subsystems that will ultimately run on the lunar rover when those systems are still being conceptualized. We will provide input on the development of those other systems in the areas where our knowledge can be valuable, and in the meantime we will focus our efforts on those aspects of our project that can be developed without knowledge of the wider system.

The remaining portions of the project are those on which notable progress has been made, but which still require significant development before they can be called complete. The first of these is the brinkmanship subsystem. This system is functional enough to achieve the goals set for our Spring Validation Demonstration (full test results available in Section [INSERT SECTION HERE]. The brinkmanship code can parse a point cloud generated from stereo images and compare the number of points in the cloud to a threshold to determine whether the rover is close to an edge.

Future work on this subsystem would involve developing more sophisticated methods of detecting edges, and more complex response behaviors.

The second subsystem in this category is the planning subsystem. Work on this subsystem has occurred primarily in simulation up to this point. Within the simulated environment, a planner has been implemented that can generate a path between a series of manually-selected waypoints. This planner produces results at speeds consistent with the requirements of the project and allows the simulated rover to reach the waypoints with an acceptable degree of accuracy. However, the expectations of this system will eventually require it to adapt in response to brinkmanship results, lighting conditions, and accumulated risk over time. Thus far, none of those features have been implemented.

7 Project Management

7.1 Work Breakdown Structure

The Work Breakdown Structure seen in Figure 11 shows the division of the Pit Navigator project into its various subsystems. The boxes with dots in the corner represent areas where work is ongoing, and boxes that have been fully colored represent areas where the work is complete.

1.1 Hardware is the only top-level section that has been completed so far. As discussed in the previous section, this is due in significant part to the need for a functional test platform with which to test and verify the software components of the project. This front-loading of the hardware related work extends to the software components of the project that are most closely related to the operation of the rover and camera. 1.2.1 Camera Operation and 1.2.5 Movement Control are the specific sections which encompass this development. It was important and necessary to implement these low-level software functions quickly to provide a foundation for the more complex tasks of perceiving and navigating through the environment.

These higher-level software tasks are represented within the WBS by sections 1.2.2 Brinkmanship and 1.2.3 Planning. These sections have seen significant development in all the subsections they cover, but there is still plenty of work to be done. The Spring Validation Demonstrations were largely centered around showcasing basic brinkmanship and planning functionality. The task for the fall semester is to improve the robustness of these features and integrate them to enable more complex behaviors for the rover.Section 1.2.4 Interfacing Operation is the only section under the heading of 1.2 Software which has not yet begun to be developed. This is due to the relative lack of work that has been performed on other aspects of the Pit Ranger mission beyond the scope of this project.

The final high-level heading, 1.3 Management, covers all the project management aspects of this project. As will be shown in the following sections, project management is a task that has been actively maintained throughout the duration of the project and will be ongoing until the project completes. Purchases are tracked with a regularly updated budget, progress is measured against a clearly defined schedule, and risks are tracked and mitigated wherever possible.

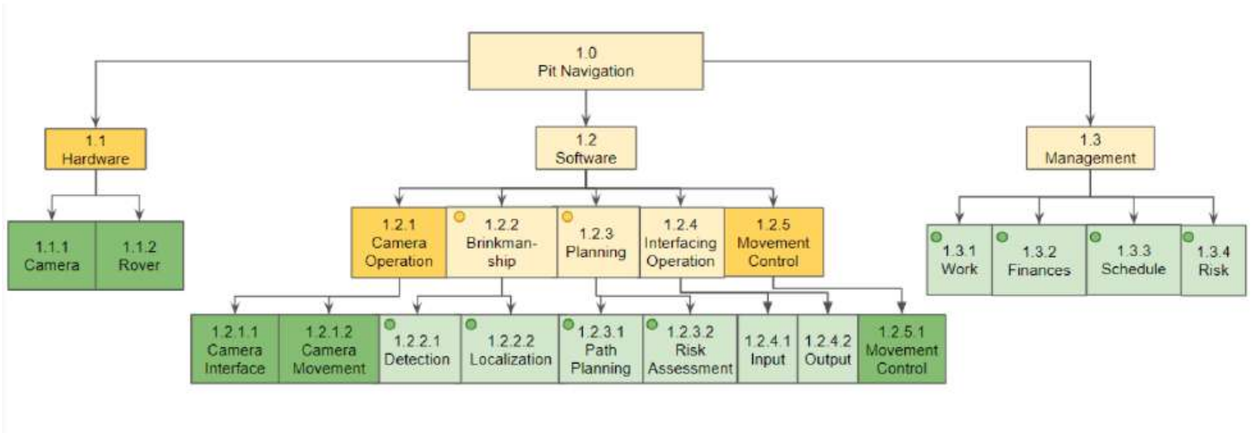


Figure 11: WBS Tree

7.2 Schedule Status

The only milestone for Spring 2020 that was not met was the incorporation of brinkmanship into the simulator. The chief obstacle to implementing this feature is the question of how to translate the simulator environment into a point cloud that mimics what could be generated from the stereo cameras on the rover. While it is possible that some work can be done to achieve this milestone over the summer, given that one team member will be working on simulating the Pit Ranger mission as part of his summer internship, the schedule for Fall 2020 makes the assumption that room will need to be made for that work during the first months of the semester.

Much of the other work scheduled for the first half of the fall semester will also take place in simulation, both because this is a logical progression for the project and because of the restrictions imposed by the possibility of continued campus lockdown. The first priorities for Fall 2020 will involve designing more complex robot behaviors in response to brinkmanship edge detection and incorporating a risk assessment component into the planning algorithm. Work will also be done to increase the sophistication of the pit edge detection heuristic, relying largely on data collected during previous tests in the spring.

In the latter half of the semester, work will continue within the simulator, but some of the advancements made in the earlier months will also be ported back to a physical rover to be demonstrated in a field setting. The primary goal for the simulator will be to execute an entire simulated mission, in which the rover makes multiple trips from a lander location to the pit, avoiding obstacles in the simulated environment along the way. The rover will drive to pre-established waypoints around the pit edge, adjusting the exact destination locations according to readings from the brinkmanship system, and generate images of the pit interior from each location. A subset of these skills will also be demonstrated on the physical rover; the proposed demonstration involves placing the rover near a cliff edge or similar brink, and having it autonomously navigate to a location along the brink and capture images from that vantage point.

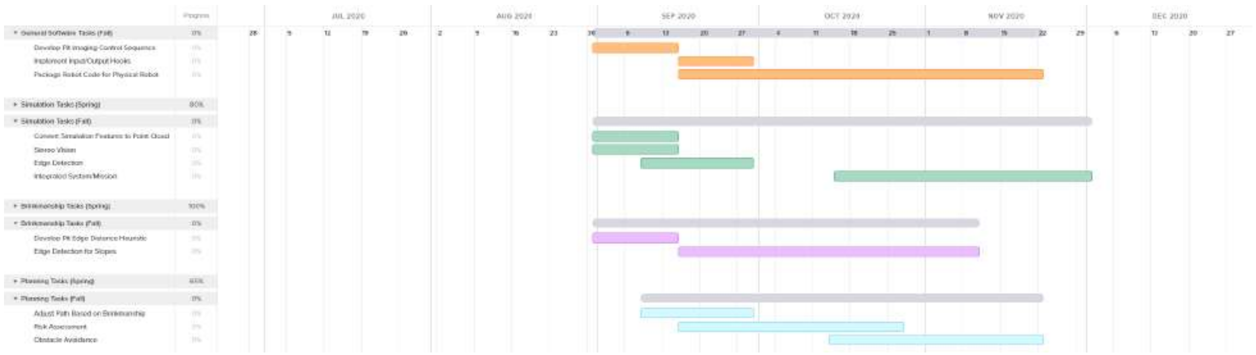


Figure 12: Fall Schedule

7.3 Test Plan

For next semester we have developed a set of deliverables that lead our team to complete the project for the Fall Validation Demonstration (FVD). At each of these deliverables, we will show some added functionality that allows our rover to complete its mission better, or will allow us to verify our functionality. These deliverables all have an associated test that confirms the functionality of the deliverable. The progress reviews happen every two weeks starting in mid-September, and finishing with the FVD happening in early December. Our proposed deliverables are in table 6, shown below.

Progress Review	Deliverable	Test Plan
PR7	Develop pit edge distance heuristic Add stereo vision to sim rover at the proper height	Get distance from camera recordings Show vision in simulation
PR8	Add edge detection state to sim Develop pit edge detection for slope type (1/4) Create pit capturing control sequence	Approach edge in the sim with brinkmanship Detect edge in previous rosbags Physical robot demonstration camera mount
PR9	Develop pit edge detection for slope type (2/4) create input and output hooks	Detect edge in previous rosbags Test in skeleton MoonRanger state machine
PR10	Develop pit edge detection for slope type (3/4) Add risk assessment to planning	Detect edge in previous rosbags Simulator demonstration returning to the lander
PR11	Develop pit edge detection for slope type (4/4) Separate sim and robot code	Detect edge in previous rosbags Present separated code
PR12	Add obstacle avoidance Properly package robot code onto the physical robot	Simulation demonstration avoiding rocks
FVD	Integrated System in Simulation	Pit Edge Data Capture Test Simulated Mission Execution Test

Table 6: Planned Deliverables for Fall 2020

We have two Fall Validation Demonstrations to showcase the functionality of our Pit-Navigator robot. The first test, generally speaking, is to drive our earth-based surrogate rover to the brink of a natural cliff, and verify that the robot will stop itself before driving over the cliff. The second test is a simulation of a part of the expected lunar mission. Each of these tests require our functionality to be fully implemented and operational to pass our success criteria.

During the first test, the natural high brink test, our objective is to identify distance to the vantage point(s) near the brink and capture images from the point(s). This will allow us to verify requirements M.P.3, M.P.4, M.P.6, and help verify that M.P.7 is true outside of simulation.

While ledges are numerous in this manufactured world, natural brinks that simulate the dropoffs on the moon are uncommon. We have chosen to use an 18-meter cliff on the Gascola testing site in Penn Hills Pennsylvania. While normally used to test self-driving cars with NREC, we have acquired permission to use this cliff as a testing location after getting a safety rundown for the site.

We chose this location due to its relative lack of vegetation, the dirt terrain, natural wear and erosion, its accessibility to CMU campus, and its height. While not a perfect simulation of the brinks on the moon, the variety of slopes and shapes of the brink edge is the closest that we could access. This cliff is shown in figure 13, from the bottom. We are planning on testing without snow, but the snow makes the top of the cliff edge very clear.

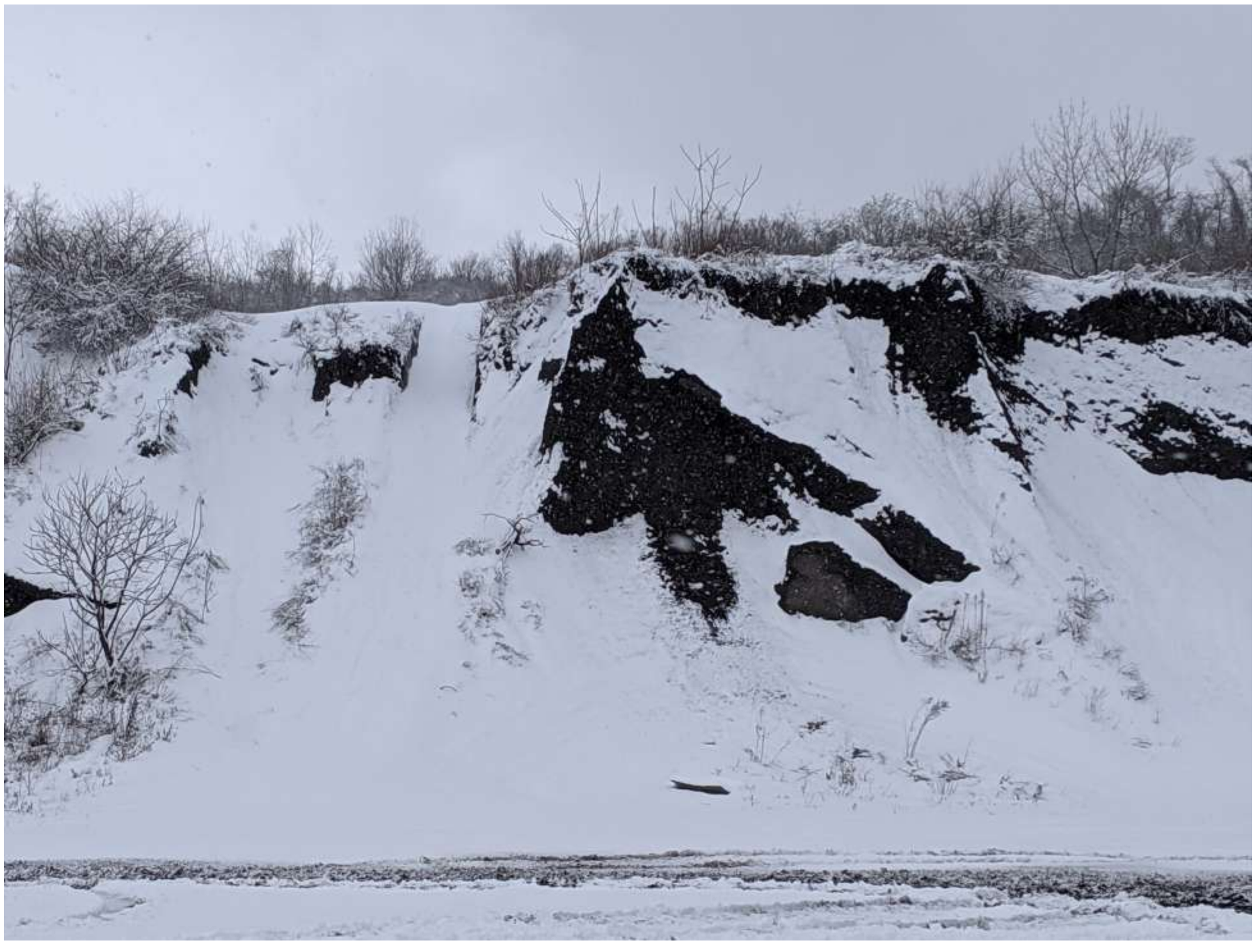


Figure 13: Gascola Testing Site

To set up for this test, there are a few extra steps to make the testing location more moonlike. First is to clear vegetation, greater than 0.25 inches in diameter, 2m from the anticipated vantage point, and 3 meters back. Measure the slope along the 3-meter trek (if it can be done safely) to determine where the rover should stop. The rover should stop when the average slope is greater than 30 degrees from horizontal. If a stopping point is found before the obvious cliff edge, leave a marker. Set the rover loaded with the Pit-Navigator code at the end of the cleared vegetation. Attach a ruler extending out from the rover, that lies outside the original stereo image matching line of sight but in line of sight for pit image capturing. Then begin the brinksmanship code.

To achieve this set up some tools are required. First and foremost you need a skid steer rover with a stereo camera pair looking forward, loaded with Pit-Navigator code. Next, you will need vegetation clearing equipment like saws, hedge trimmers, extended trimmers and weed whackers.

Lastly, you will need a rope for the rover and PPE for anyone testing, hardhats, high vis vests, working gloves, and possibly a harness and climbing rope if the vegetation clearing calls for it.

Once the setup is complete at one location, to test the rover one should follow these steps:

1. Activate rover
2. Rover detects brink
3. Rover identifies distance to a vantage point near the brink
4. Rover drives to a vantage point(s) and captures images off the brink
5. Rover drives back to the starting point
6. Repeat for completeness and in many locations
7. Review footage afterward to confirm distance to the vantage points using the ruler

To verify the success of the captured data, compare the results to the criteria listed below.

1. Average distance to pit when the image capturing: $< .5$m
2. Amount of usable data captured per location > 15 MB
3. Produces tagged data and the stitched panorama

During the second test, the simulation test, our objective is to demonstrate all of the rover's capabilities in one simulation. This will allow us to verify each of our performance requirements inside a simulation. While the simulation could run on any computer, a top-notch set of hardware will allow the sim to run at half of real-time, dramatically increasing the speed of the simulation. The best computer that we have access to is the MRSD simulation lab computer that allows for this speed and is the primary candidate to run our final simulation on.

To get the computer set up to run the simulation we have developed a list of commands that will download and create the simulation. This will add ROS and Webots to the computer and the code requires 17 GB of space to download. The code supports an Ubuntu 16 OS and ROS Kinetic. It will use the native python 2, 3 combination to run its code, and source its catkin workspace. As previously stated the equipment required to run this simulation is a computer with a top-notch graphics card and processor.

In order to run this test, one needs to create a world environment, create a traversable map of the world, create the desired waypoints, and join the rover to the simulation code. If one is downloading from our GitHub repository, then this has been done already. Any new model will need to follow the above steps. Once that is done, the running of the test involves opening a terminal window, sourcing the setup.bash of the catkin workspace, then launching the launch file.

The simulation will open Webots, begin the ROS nodes, and begin moving towards its first target once the Webots sim is running. The rover will follow the route through the waypoints as defined by the global plan. The rover will adjust the plan based on time available in the mission to get to the remaining waypoints, returning to the lander to 'save' some of the captured data. The rover will then return to the pit as many times as necessary to visit and capture data at every waypoint, or until time runs out.

To verify the success of the captured data, compare the results to the criteria listed below.

1. Average distance to pit when image capturing: $<$1m
2. Amount of usable data captured 500 MB
3. Speed made good $>$ 0.21m/s
4. Risk never goes over threshold, 5:1
5. Mission Completion

The final two success criteria require more explanation. Risk is calculated as mission success : mission failure. There are many places that the mission could fail, and this criterion captures all of them into a single criterion. When testing we will notice trouble spots, mark them, and improve that portion until the total risk is at criteria. Mission completion is defined as capturing data at every near pit waypoint and dropping off the data at the lander with time leftover. Figure 14 is an example of the robot capturing data at a vantage point around the pit. What the robot sees is shown in the top left of the image.

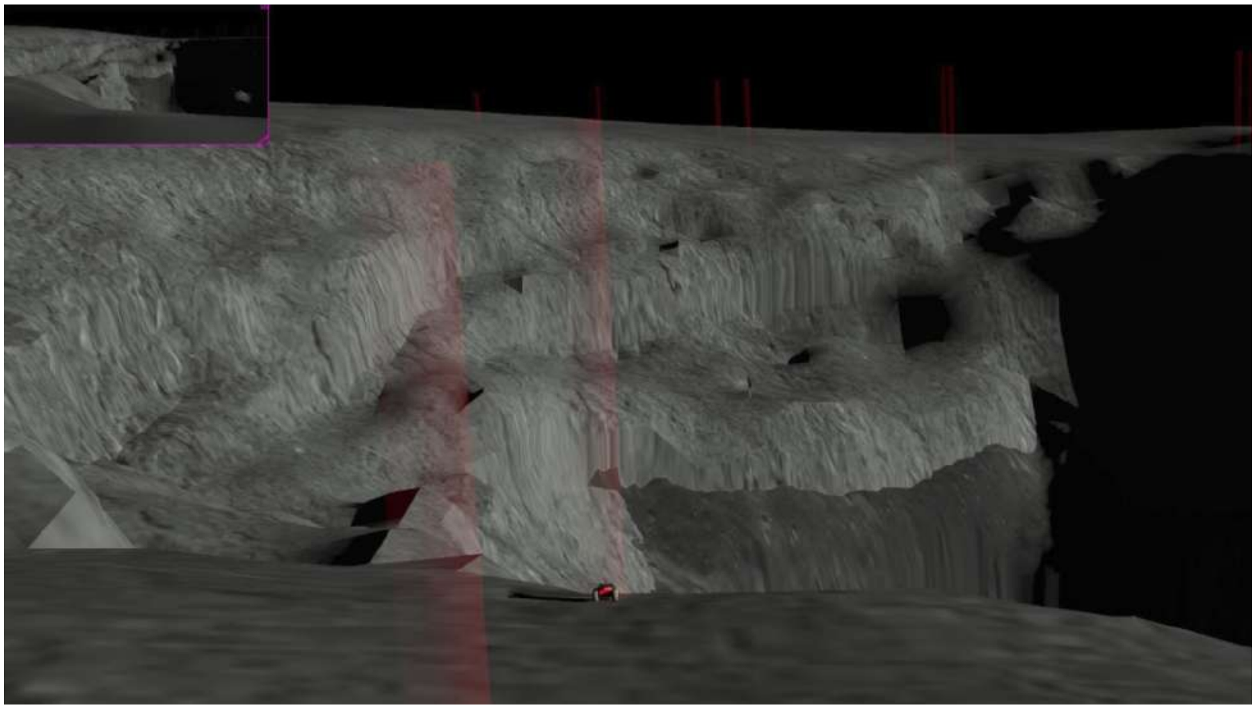


Figure 14: Rover Looking Into Pit

7.4 Budget Status

Our team has two significant advantages in the budgeting of our project. The first is that our project is primarily software, which allows us to avoid the expense of purchasing physical components. Second, our project is contained within the umbrella of a larger development (the PitRanger project) which has significant financial resources that we can leverage. Because we expect our costs to be minimal, and to be able to outsource many expenses to the wider organization, our expected budget is fairly low.

However, our total spending until the end of the spring semester has gone upto $1700. Most of the expenditure has been with regards to building a new robot. The complete list of items has been shown below.

Item	**Quantity**	**Cost/ea.**	**Total**
12v 6A Regulator	1	$23	$23
Glass Standoff 1/2"x2"	1	$17	$17
Handle	1	$10	$10
IMU	1	$276	$276
Wheels	4	$23	$92
Hubs	8	$10	$80
Motors	4	$175	$700
Roboclaw 2x15A	1	$90	$90
Roboclaw 2x30A	1	$125	$125
Encoder Cable 4 pack	1	$12	$12
Encoder Cable Breakout Board	4	$2	$8
1/8" Aluminum Sheet	2	$0	$0
Pan-Tilt Mount	1	$100	$100
Wire 1	1	$35	$35
Wire 2	1	$15	$15
Heat Shrink Tubes	1	$9	$9
Left angle USB	1	$6.50	$6.50
Right angle USB	1	$7.50	$7.50
Arbotix M board	2	$40	$80
Total			**$1686**

For the fall semester, we expect our expenditure to be minimal as we plan to work mainly in simulation.

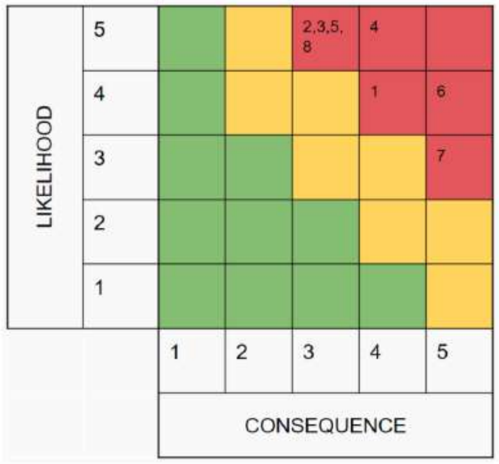
7.5 Risk Management

The Pit-Navigator has 8 major risks to the project that are likely to happen and have major consequences if they do.

The number of risks has not changed but over the semester many of our risks have become situations. For example, Risk 3, we were waiting on the rover Blue to become available for regular use for our team, but it never did. We had to build Blue2 as a result. Risk 8 was included before it became a situation but was identified with very little time to spare, as there was not enough time to properly mitigate the risk. These situations were dealt with and handled better because we saw them coming.

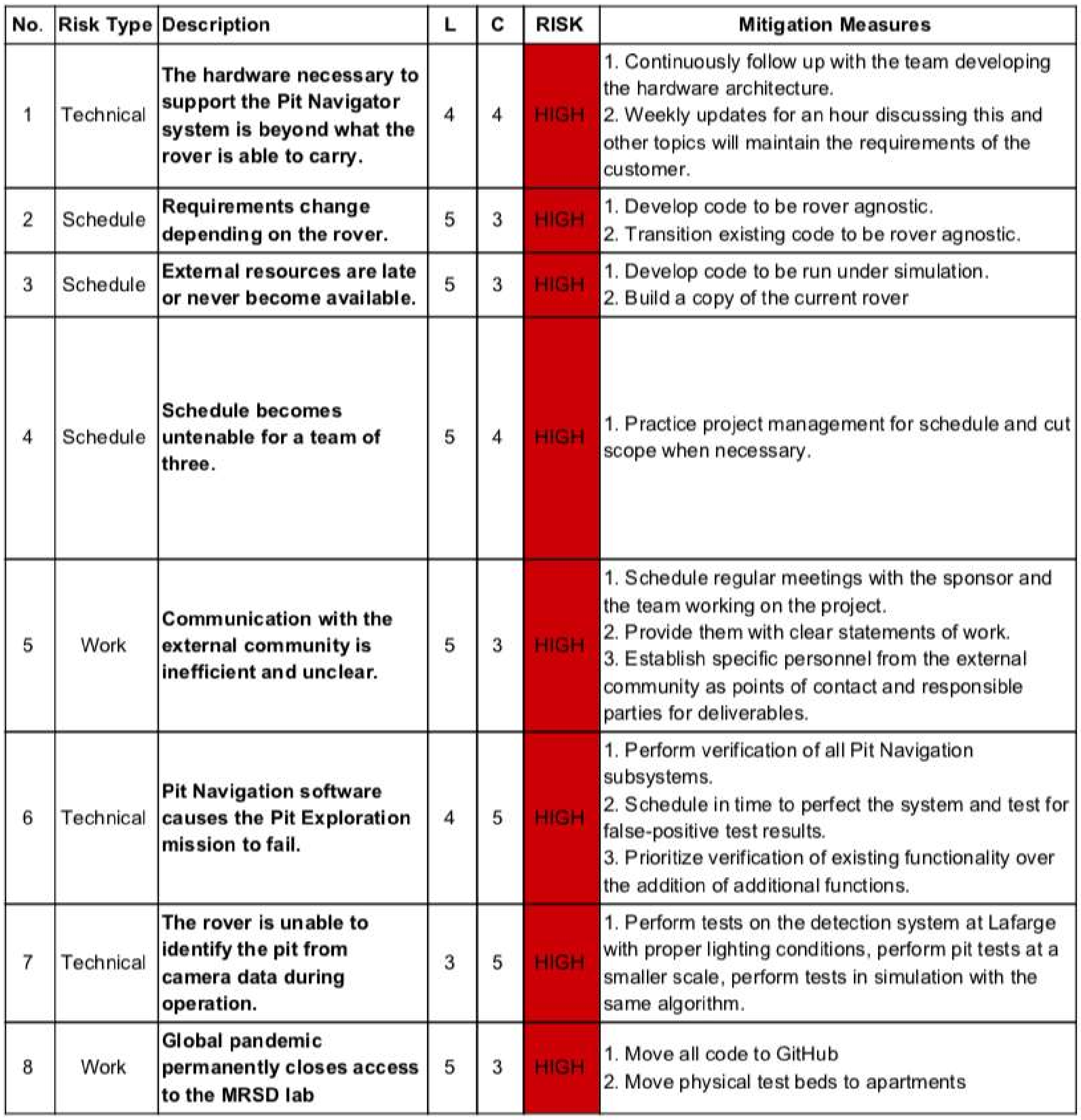
There are also ongoing changes that are currently budding situations, like Risk 2. The engineering model of the rover has not been complete and has pressures to change the form of the rover and the mission if the physical abilities are able to traverse slopes. This could significantly change the flight code of our mission. We are currently adding perception and maneuvering techniques that would allow traversing onto slopes to account for a wide variety of strategies that will come out of changing the rover's shape and vantage points.

After review from the CDR presentation, there seem to be risks that are close to retiring. Risks 3, 5, and 7 may be close to retirement from the highest danger level as the likelihood or cost may drop for them in the coming semester. For Risk 3, the Pit-Navigator project is become more and more self-sufficient, needing less critical components from external resources and is likely to reduce in cost. Risk 5 will likely see a drop in likelihood as the teams have gotten more efficient at communicating what is needed. And as the rover has gotten more developed, and run more tests, the likelihood of Risk 7 will go down with continued testing of the system. Our risks are shown below, and their risk matrix is shown in figure 15.



LIKELIHOOD	5	🟩	🟨	🟥 2,3,5, 8	🟥 4	🟥
	4	🟩	🟨	🟨	🟥 1	🟥 6
	3	🟩	🟩	🟨	🟨	🟥 7
	2	🟩	🟩	🟩	🟨	🟨
	1	🟩	🟩	🟩	🟩	🟨
		1	2	3	4	5
		CONSEQUENCE				

Figure 15: Likelihood-Consequence Matrix



No.	Risk Type	Description	L	C	RISK	Mitigation Measures
1	Technical	**The hardware necessary to support the Pit Navigator system is beyond what the rover is able to carry.**	4	4	HIGH	1. Continuously follow up with the team developing the hardware architecture. 2. Weekly updates for an hour discussing this and other topics will maintain the requirements of the customer.
2	Schedule	**Requirements change depending on the rover.**	5	3	HIGH	1. Develop code to be rover agnostic. 2. Transition existing code to be rover agnostic.
3	Schedule	**External resources are late or never become available.**	5	3	HIGH	1. Develop code to be run under simulation. 2. Build a copy of the current rover
4	Schedule	**Schedule becomes untenable for a team of three.**	5	4	HIGH	1. Practice project management for schedule and cut scope when necessary.
5	Work	**Communication with the external community is inefficient and unclear.**	5	3	HIGH	1. Schedule regular meetings with the sponsor and the team working on the project. 2. Provide them with clear statements of work. 3. Establish specific personnel from the external community as points of contact and responsible parties for deliverables.
6	Technical	**Pit Navigation software causes the Pit Exploration mission to fail.**	4	5	HIGH	1. Perform verification of all Pit Navigation subsystems. 2. Schedule in time to perfect the system and test for false-positive test results. 3. Prioritize verification of existing functionality over the addition of additional functions.
7	Technical	**The rover is unable to identify the pit from camera data during operation.**	3	5	HIGH	1. Perform tests on the detection system at Lafarge with proper lighting conditions, perform pit tests at a smaller scale, perform tests in simulation with the same algorithm.
8	Work	**Global pandemic permanently closes access to the MRSD lab**	5	3	HIGH	1. Move all code to GitHub 2. Move physical test beds to apartments

Figure 16: High-level Risks for Pit-Navigator Project

8 Conclusions

8.1 Lessons Learned

The spring semester provided many opportunities to learn and apply new skills. No member of the team had extensive or formal project management experience prior to this project, so we divided those responsibilities between us to give everyone a chance to practice. Awadhut tracked the project budget, Alex managed the risks and their mitigations, and Justin maintained a schedule for the semester.

Another new skill was the development of a comprehensive test plan. With the limited time available to us, it was crucial that we make the most of every opportunity to test our hardware and software. In particular, the field tests required significant effort to set up, so to make that effort worthwhile we needed to perform as many tests as possible. To accomplish this, we clearly established the deliverables for the test, and used those to derive what sort of data was necessary to collect. We identified variations of the test process that would give us the most useful distribution of data in the most efficient manner.

Assembling a complex system such as the surrogate rover required several trade studies for different components and processes. Because our work will eventually be a part of a system that operates on the moon, the conclusions we reached from these trade studies was even more important. The choices we made needed to align with the development of the larger system and needed to be robust enough to stand up to the challenging environment. We learned to assess the most relevant criteria for deciding between similar options, and how to construct our studies so that the information was clear and readable.

This project provided many opportunities to develop technical skills, as well. We learned quickly that there is a difference between breadboard circuitry of the kind we had practiced in labs before and designing a functional electrical system for a robot. We learned to interpret data sheets for controllers, motors, and other mechanical components to ensure that each component was powered safely and effectively. Proper battery safety and storage was also an important skill for us to engage with, especially after moving to working from our homes.

The software side of the project also presented a great deal of complexity, and with it many opportunities for us to learn. While we had some prior experience with ROS, integrating the various components of our system into a coherent ROS network was a new level of challenge. Working with different packages required us to troubleshoot version compatibility issues and read documentation to understand the underlying mechanisms. Through practice and hard work, our familiarity with the systems and packages increased, allowing us to work more effectively and address problems when they arose.

The significant vision-related component of our project allowed us to practice the theory we had picked up from Computer Vision, a class which two of the team members were taking during the spring semester. We had to apply our knowledge to process the images collected by the RealSense camera and generate useful 3D data about the rover's environment so that brinkmanship functionality could be achieved.

8.2 Key Fall Activities

The fall semester will bring many chances to apply the lessons we have learned, as well as new challenges that will require us to learn additional lessons. Our major goals for the fall are to improve our planning and brinkmanship subsystems to enable the complex behaviors that the rover will need to navigate the pit environment.

We will develop more sophisticated heuristics for identifying brinks within the point clouds collected from the stereo images. This will improve the robustness of the brinkmanship subsystem, so that it can allow the rover to recognize and safely approach a variety of brinks and slopes. The brinkmanship functions will also be implemented in the simulator, to allow a wider range of scenarios to be tested. The rover will be given additional behaviors to enact when it detects a brink, beyond just halting its current movement and taking images at hard-coded angles. The image capturing cycle will be refined and parameterized, and once the requisite images have been captured the rover will navigate away from the pit to continue traveling around the rim.

Much of the work scheduled for the upcoming semester involves the planning subsystem. Risk assessment is a significant aspect of our project that has only begun to be explored. We intend to design a system that will allow the rover to adjust its intended path as it travels, based on data accumulated during the expedition, in order to gather data about the pit and return it to Earth efficiently and safely. We recognize that if the rover was to experience a mission-ending accident while exploring the pit, any data contained within its memory would be lost, and the rover must be able to account for this by returning to the lander at regular intervals to store and transmit its collected data. The planning subsystem must also be able to account for the direction of the sunlight because the rover is solar powered and cannot operate when no sunlight is available. This will affect what portions of the pit it will be able to access throughout the mission. Lastly, the rover must be able to avoid obstacles in its immediate vicinity while traversing the path between waypoints. Although we intend for this functionality to be derived from the MoonRanger mission being developed in parallel with our project, we still need a working implementation in simulation to test the planning code that we develop.

All these tasks contribute to our ultimate goals for the Fall Validation Demonstration. The first of our proposed demonstrations is an execution of the entire mission in the simulated environment, which will apply every aspect of our project to allow the rover to safely navigate a lunar pit and make multiple trips back to the lander location to deposit data. Our second demonstration will be performed by Blue 2 and will involve the rover navigating autonomously to a location along a brink and capturing images once it has reached that location. With the lessons learned over the past semester, and the tasks we plan to accomplish in the fall, we are confident of our success in both these demonstrations.

9 References

[1] Figure 4: Phatom-X Pan-Tilt Mount, https://www.trossenrobotics.com/phantomx-pan-tilt

[2] Figure 5: Arbotix-M Controller, https://www.trossenrobotics.com/p/arbotix-robot-controller.aspx