Individual Lab Report - 10

Lunar ROADSTER

Team I

Author: **Bhaswanth Ayapilla**
Andrew ID: bayapill
E-mail: *bayapill@andrew.cmu.edu*

Teammate: **Deepam Ameria**
ID: dameria
E-mail: *dameria@andrew.cmu.edu*

Teammate: **Boxiang (William) Fu**
ID: boxiangf
E-mail: *boxiangf@andrew.cmu.edu*

Teammate: **Simson D'Souza**
ID: sjdsouza
E-mail: *sjdsouza@andrew.cmu.edu*

Teammate: **Ankit Aggarwal**
ID: ankitagg
E-mail: *ankitagg@andrew.cmu.edu*

Supervisor: **Dr. William "Red" Whittaker**
Department: Field Robotics Center
E-mail: *red@cmu.edu*

November 12, 2025

Contents

1 Individual Progress

1.1 BEN Stack

The Behavior Executive Node is our high-level Finite State Machine (FSM) which integrates all the software packages, which includes the navigation planner and controller, perception, planning pose extractor, and validation. We sat together as a team to decide the overall flow of the BEN Stack, all the various states and transitions, and then assigned sections of it to each team member.

I worked on the global navigation planner and the manipulation planner states, and also worked with Simson on the global navigation controller and manipulation controller states. I also collaborated with Ankit, Deepam and Simson in ensuring correct state switching from navigation to every other state, fine tuning the BEN code and additional feature development.

Figure 1 shows the overall BEN Stack, with the red boxes denoting ROS2 actions, and the green boxes denoting ROS2 services.

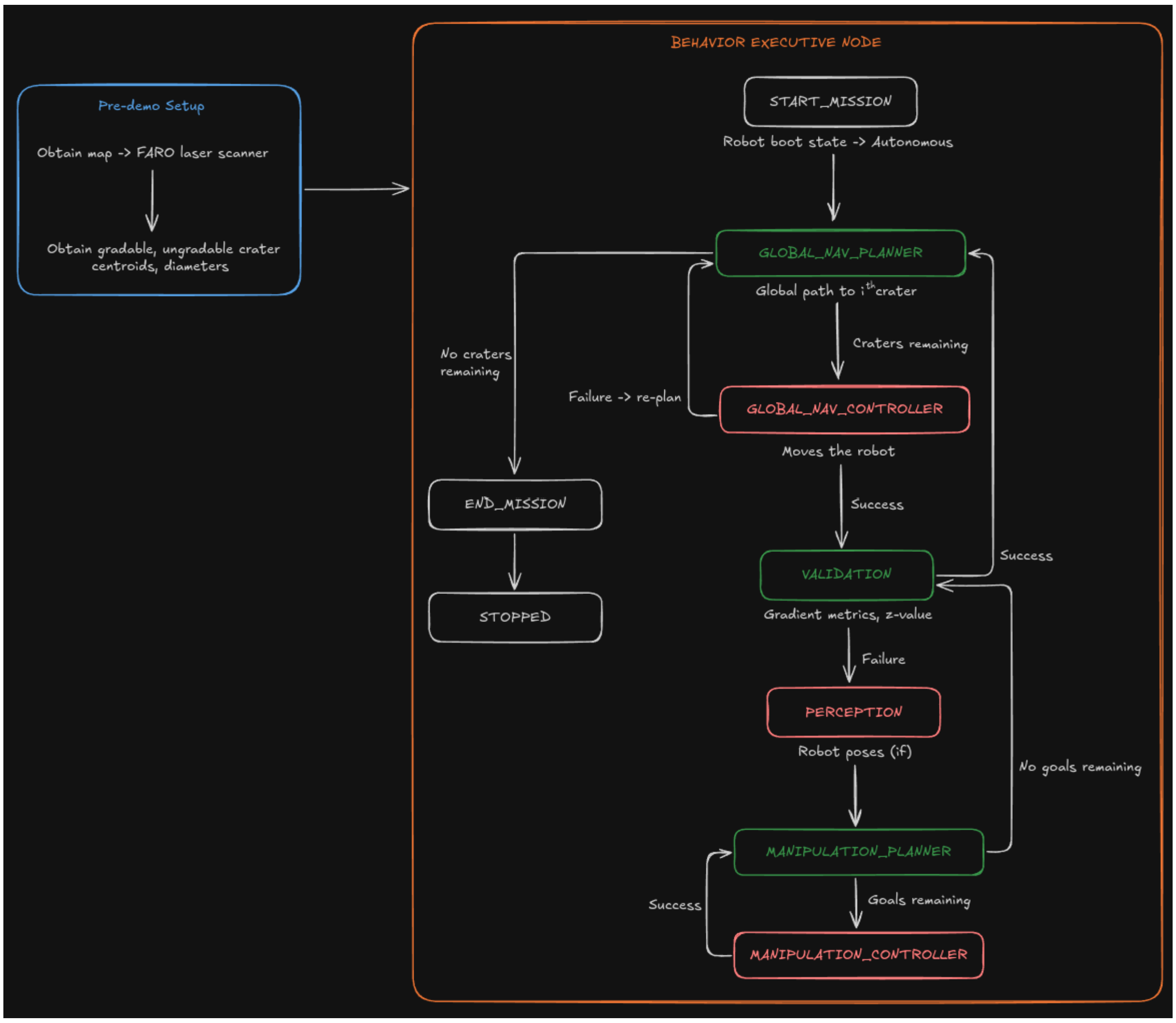


Figure 1: Behavior Executive Node

1.2 Navigation Tuning

The first step in this was tuning my global navigation planner. I tested this repeatedly on the rover, to ensure that the rover always plans a path to every crater regardless of its current position. This took some time as I had to play around with some costs and weights, and eventually everything worked out. The global navigation planner can now always plan a path to every gradable crater.

I worked with Simson in tuning the global navigation controller. Since the algorithm is based on Pure Pursuit, we played around with the lookahead distance and goal tolerance values, and eventually converged upon a value that works best for our rover. Based on the path planned by the global navigation planner, the global navigation controller always moves the rover towards the gradable craters, and always stops at the "source" pose, which is the rim of the crater.

Now once we reach the crater with the help of the global navigation controller, the validation, perception and pose planner states take over. At the end, they spit out four poses. These are the source, sink, source backblade and sink backblade poses, which will then be sent to the manipulation planner and manipulation controller states.

For the manipulation planner, I implemented a simple A* search algorithm. This plans a path from the rover's current position to every goal pose that I obtain from the perception and planning stacks, one after the other.

Now using this planned path from the manipulation planner, the manipulation controller, which is the same pure pursuit algorithm, moves the rover to each of the goal poses by also moving the tool. From source to sink, the tool is lowered. From sink to source backblade, the tool is put up again. And finally from source backblade to sink backblade, the tool is lowered again to perform backblading. I worked with Simson in fine tuning the lookahead distance and goal tolerance, and also implementing the correct tool position logic.

1.3 Planning Tuning

The planning stack was implemented by Ankit and comes under the perception package (implemented by Ankit and Deepam). This planning stack gives 4 goal poses - source, sink, source backblade and sink backblade. We earlier found these values to always be slightly offset to the left of the crater. I worked with Ankit and Deepam in fine-tuning this, by adding manual offsets and also making perception better. We now solved this and the goal poses that we obtain are in line with the crater's actual centroid.

Figure 2 shows the outputs obtained from the planning stack, which can be visualized by the magenta markers over the first crater.

1.4 Quality Assurance and Testing

We are in a very crucial Quality Assurance stage right now and every package needs to be individually tested so that it is robust and offers the best possible performance. These are validated by some tests as discussed below:

1.4.1 Localization QA

Localization plays a very important role in ensuring our rover accurately reaches the craters. We fuse information from the total station, wheel encoders and the IMU. I

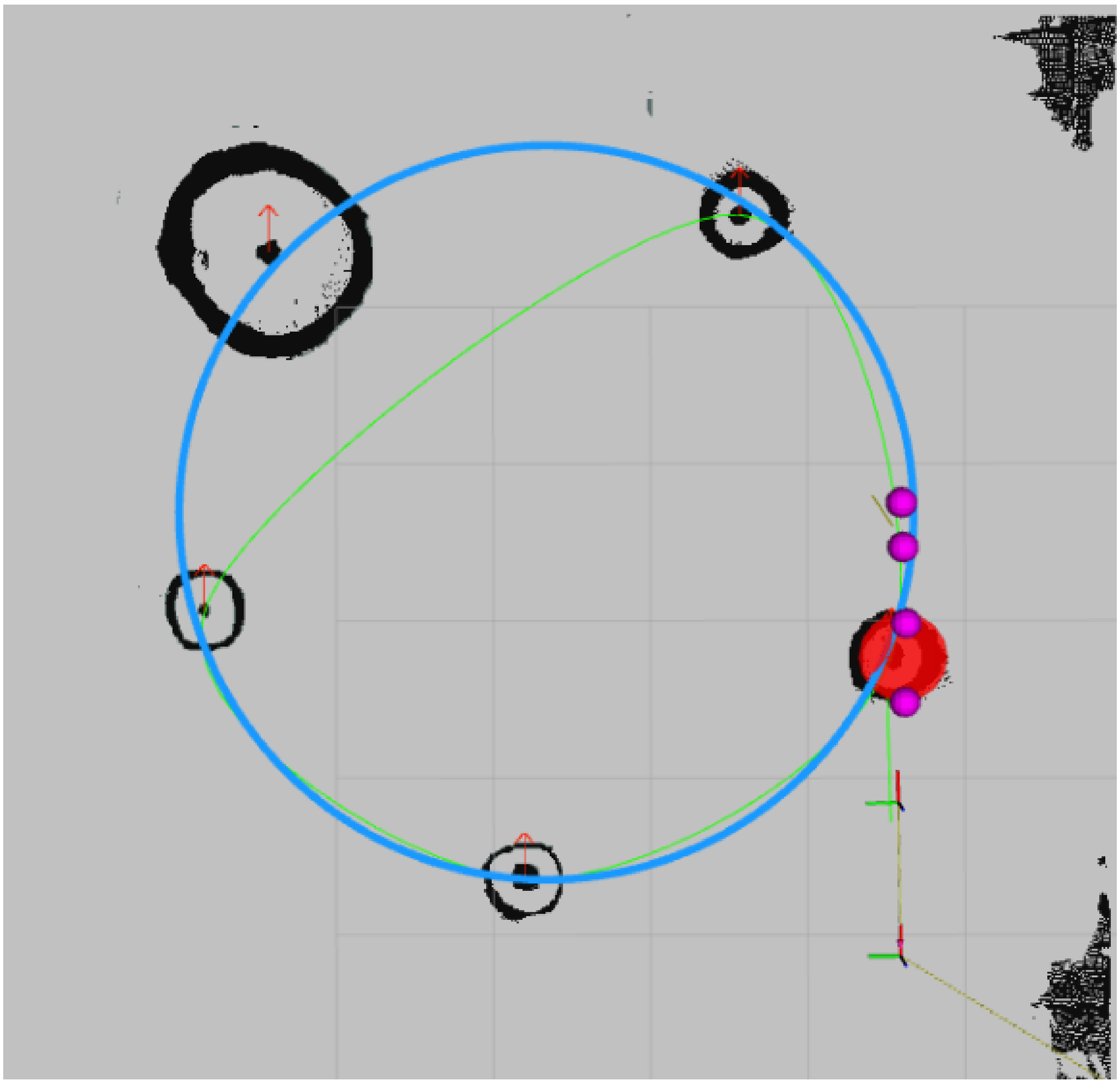


Figure 2: Planning Stack

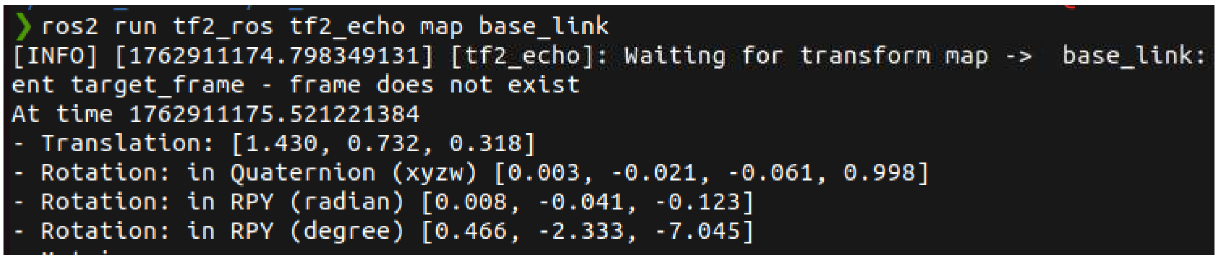
performed a test where I keep the rover stationary, print out the TF transform between the `map` and `base_link` frames, look at the translation values obtained and then measure its actual distance from the `map` frame origin using a measuring tape. The measured values were pretty much the same as the ones obtained from the TF tree, and this validates the localization quality assurance. The same is shown in Figures 3 and 4.

1.4.2 Navigation QA

T02: Global Planner Accuracy Test

In order to prove performance requirement "M.P.1: Will plan a path with cumulative deviation of <= 25% from chosen latitude's length", I am maintaining a google sheet where we log all the global navigation planner deviation statistics. This is shown in Figure 5. From these results, it is very clear that the percentage deviation for most runs in well within the 25% limit. We will keep logging in more of this information as we test, so that we can show more in FVD and FVD Encore.

T03: Filtering and Selection of Gradable Craters



```
❯ ros2 run tf2_ros tf2_echo map base_link
[INFO] [1762911174.798349131] [tf2_echo]: Waiting for transform map ->  base_link:
ent target_frame - frame does not exist
At time 1762911175.521221384
- Translation: [1.430, 0.732, 0.318]
- Rotation: in Quaternion (xyzw) [0.003, -0.021, -0.061, 0.998]
- Rotation: in RPY (radian) [0.008, -0.041, -0.123]
- Rotation: in RPY (degree) [0.466, -2.333, -7.045]
```

Figure 3: Localization TF tree output between map and base link frames

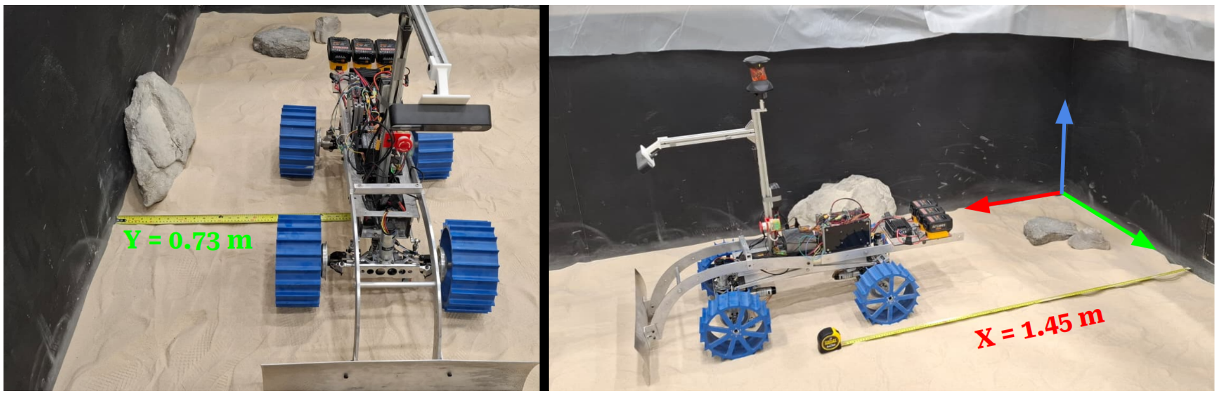


Figure 4: Localization measured values

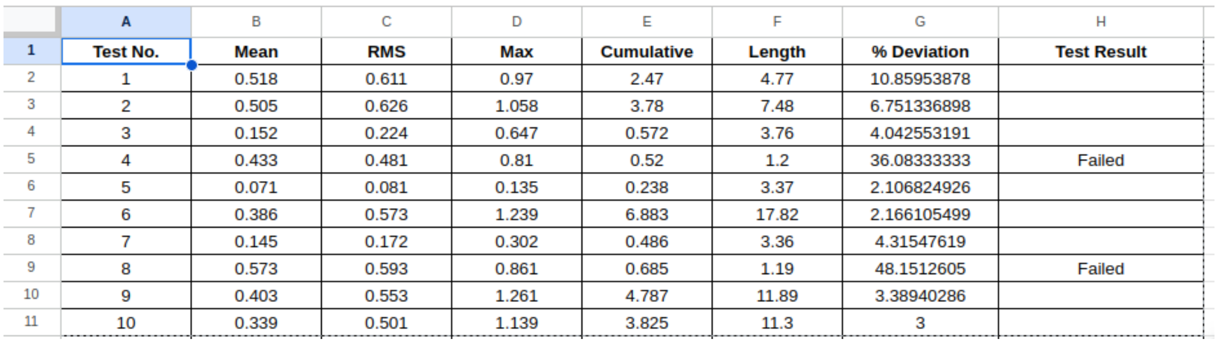
The global navigation planner can clearly differentiate between gradable and ungradable craters based on their diameters. Any crater whose diameter is greater than 0.5m will be classified as ungradable. This is clearly shown in Figure 6 where the green markers show gradable craters and the red marker shows the ungradable crater.

T04: Navigation Controller Maximum Deviation Test

In order to prove performance requirement "M.P.2: Will follow planned path to a maximum deviation of 10%", I am maintaining another google sheet where we log all the global navigation controller deviation statistics. This is shown in Figure 7. From these results, it is very clear that the percentage deviation for most runs in well within the 10% limit. We will keep logging in more of this information as we test, so that we can show more in FVD and FVD Encore.

T06: Repeatability Test of Navigation Controller

I also performed a repeatability test for the navigation controller, by giving every crater as a goal and moving it in a circular path. I repeated this for multiple runs and the rover always moved the every crater and followed roughly the same path. This verifies the T06 and we show a video of this in our PR 11.



	A	B	C	D	E	F	G	H
1	Test No.	Mean	RMS	Max	Cumulative	Length	% Deviation	Test Result
2	1	0.518	0.611	0.97	2.47	4.77	10.85953878	
3	2	0.505	0.626	1.058	3.78	7.48	6.751336898	
4	3	0.152	0.224	0.647	0.572	3.76	4.042553191	
5	4	0.433	0.481	0.81	0.52	1.2	36.08333333	Failed
6	5	0.071	0.081	0.135	0.238	3.37	2.106824926	
7	6	0.386	0.573	1.239	6.883	17.82	2.166105499	
8	7	0.145	0.172	0.302	0.486	3.36	4.31547619	
9	8	0.573	0.593	0.861	0.685	1.19	48.1512605	Failed
10	9	0.403	0.553	1.261	4.787	11.89	3.38940286	
11	10	0.339	0.501	1.139	3.825	11.3	3	

Figure 5: T02: Global Planner Accuracy Test

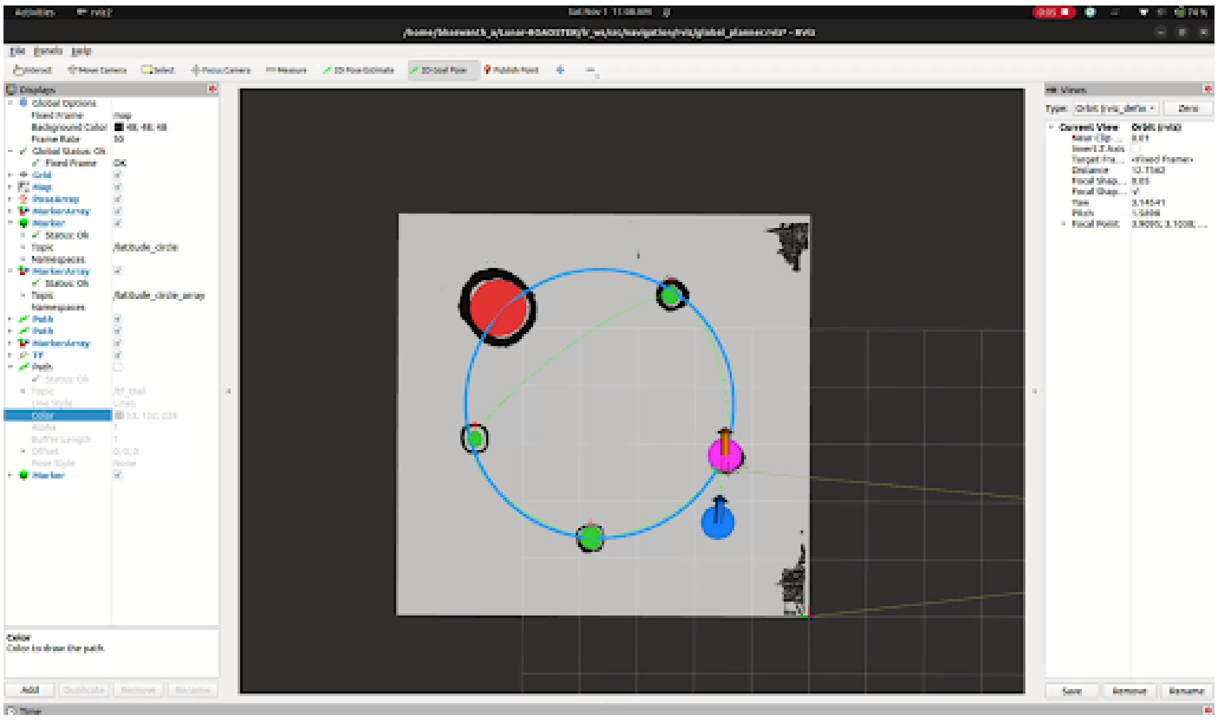
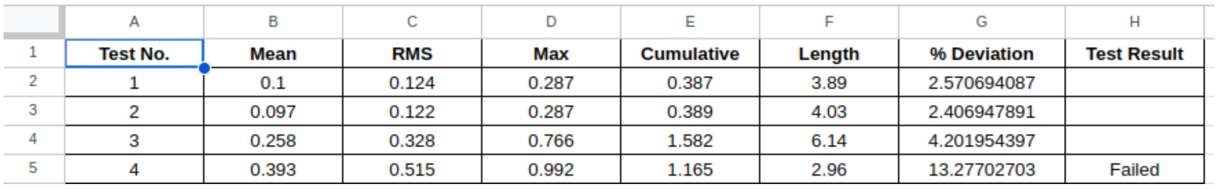


Figure 6: T03: Filtering and Selection of Gradable Craters



	A	B	C	D	E	F	G	H
1	Test No.	Mean	RMS	Max	Cumulative	Length	% Deviation	Test Result
2	1	0.1	0.124	0.287	0.387	3.89	2.570694087	
3	2	0.097	0.122	0.287	0.389	4.03	2.406947891	
4	3	0.258	0.328	0.766	1.582	6.14	4.201954397	
5	4	0.393	0.515	0.992	1.165	2.96	13.27702703	Failed

Figure 7: T04: Navigation Controller Maximum Deviation Test

2 Challenges

The team faced challenges in integration, since we don't expect everything to work on the very first go when we bring all the software packages together. There were race conditions and incorrect code logics, all of which were fixed during repeated testing. We also faced hardware issues as the front drive motor shaft started to wear off. Ankit and Deepam fixed this by replacing the motor and also the front assembly from a twin rover.

3 Team Work

- **Bhaswanth Ayapilla:** I worked on the full system integration and testing. I collaborated with Simson in fine tuning the global navigation controller and manipulation controller. I also worked with him in writing the BEN stack and ensuring correct logic switches within navigation. I worked with the entire team in deciding the BEN stack architecture, tuning it and feature development to make it robust. I worked with Ankit and Deepam in tuning the planning stack and ensuring correct poses relative to the actual crater centroid. I worked with William in debugging issues related to the validation stack. The entire team also collaborated together in deciding the flow of the FVD presentation. Moreover, I conducted localization and navigation quality assurance tests.

- **Ankit Aggarwal:** Ankit worked on the full system integration and testing. He collaborated with Deepam to define the perception stack's actions and services. He worked on writing the BEN stack for perception and collaborated with the whole team in deciding the BEN architecture. He worked with me for tuning offsets of robot pose extractor part of perception. He also collaborated with Deepam and Simson to replace the front drive actuation system. He also collaborated with William to integrate validation and perception together. The entire team also collaborated together in deciding the flow of the FVD presentation.

- **Deepam Ameria:** Deepam worked with the entire team on the full system integration and testing. He collaborated with Ankit to define the perception stack's actions and services. Together with William they developed a camera mount and gimbal system that would mechanically orient the skycam so that the roll and pitch of the rover is canceled out. He worked with me for tuning offsets of robot pose extractor part of perception. Deepam also collaborated with Ankit and Simson to replace the front drive system and modify the drive motors to optimally fit the assembly. Deepam also conducted the Perception QA for this PR

- **Simson D'Souza:** Simson worked on full system integration and testing, collaborating with me to fine-tune both the global navigation and manipulation controllers. He worked on tool planner logic and Foxglove GUI setup. Additionally, he worked with the entire team to define, tune, and develop features for the BEN stack architecture, enhancing its robustness. He collaborated with Deepam and Ankit to replace the front drive actuation system, and the whole team jointly planned the structure and flow of the FVD presentation.

- **Boxiang (William) Fu:** William's work since the last progress review focused on solving issues on the Skycam localization unit. The unit originally had a lot of drift when the roll and pitch was non-zero. Together with Deepam, they developed a camera mount and gimbal system that would mechanically orient the skycam so that the roll and pitch of the rover is canceled out. Next, William worked with the team on the Behavior Executive Node (BEN) and integration of all subsystems. William wrote the skeleton code for the BEN and set up the modules for each subsystem. He also wrote the debug trigger and manual override state for testing and debugging. Finally, he worked with the entire team to integrate all the subsystems into the BEN.

4 Plans

The only next goal is to get successful FVD and FVD Encore runs. The entire team is continuously testing to achieve robustness.