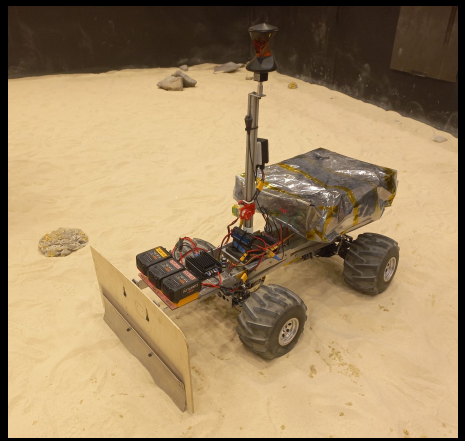
Lunar ROADSTER

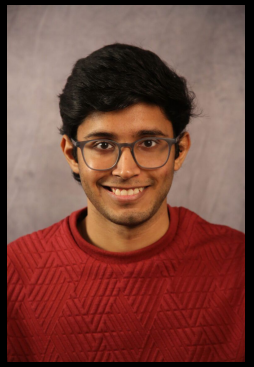
(Robotic Operator for Autonomous Development of Surface Trails and Exploration Routes)



"Starting with a foothold on the Moon, we pave the way to the cosmos"

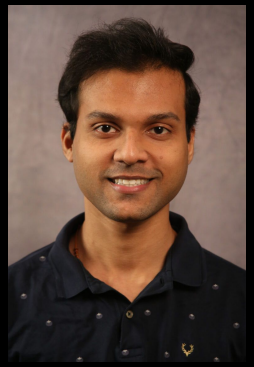
The Team

Ankit Aggarwal



Deepam Ameria

Bhaswanth Ayapilla

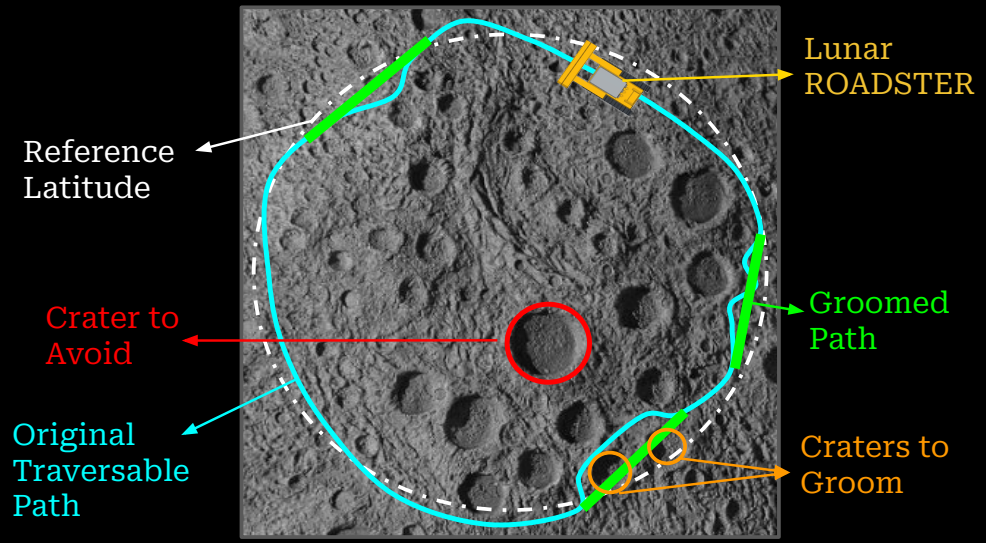


Simson D'Souza

Boxiang (William) Fu

Dr. William "Red" Whittaker

Project Overview: Lunar ROADSTER



An autonomous moon-working rover capable of finding ideal exploration routes and creating traversable surface trails

Technical: Subsystem Status

Subsystem	Completion %	Future Work
Sensors	75%	Integrate New IMU and Fisheye Camera
Computations	60%	
1. Jetson and Docker	90%	Set Up New Sensor Drivers
2. Localization Unit	80%	Implement Skycam Localization
3. Transport Planner Unit	60%	Implement New Transport and Tool Planner
4. Navigation Planner Unit	40%	Generate Map and Implement Global and Local Navigation Planner
5. FSM Planner Unit	70%	Update FSM to Support New Modules
6. Validation Unit	10%	Detect Craters and Implement Validation module
External Infrastructure	90%	Implement New Total Station Localization Method
Mechanical	90%	Fabricate and Install Actuator Arch
Actuation	85%	Linear Actuator Upgrade/Tuning
Electrical Power	100%	None

Technical: Key Challenges and Associated Plans

Subsystem	Challenges	Plan
Computations		
1. Localization Unit	External localization, while highly accurate, introduces latency that degrades navigation performance	Implement Skycam-based localization, if successful, adopt it permanently, otherwise revert to total station localization
2. Navigation Planner Unit	Nav2 package is computationally intensive, its path planning is not well-suited to our robot's mobility and negatively impacts overall system performance	Develop a custom navigation planner and controller optimized for our use case
3. Validation Unit	Limited progress has been made on the validation module, and implementing the validation pipeline will be difficult	Start development early and break the validation pipeline into smaller, manageable stages to ensure steady progress
External Infrastructure	Current total station method requires a full re-setup whenever the device is turned off or its batteries are replaced	Implement a new total station–based localization method to eliminate the need for repeated setups
Actuation	Tool (Blade) linear actuator frequently jitters during motion	Tune the PID controller or replace it with a higher-quality linear actuator

Project Management Methodology

Amalgamation of Traditional and Agile PM

- **High-level V-model** with clearly defined project scope and work units to complete
- **Low-level Agile method** with 2-week sprints
- Milestones set every sprint to coincide with PRs
- Regular **standup meetings**
- Predictive budgeting, scheduling and resource allocation, with adjustments based on real-world testing and feedback
- Task allocation based on a combination of skills and interests - optimizing **team member motivation and learning objectives**
- Team-level decision making is **decentralized**; all members get a say and the team converges to a solution

Project Management Methodology

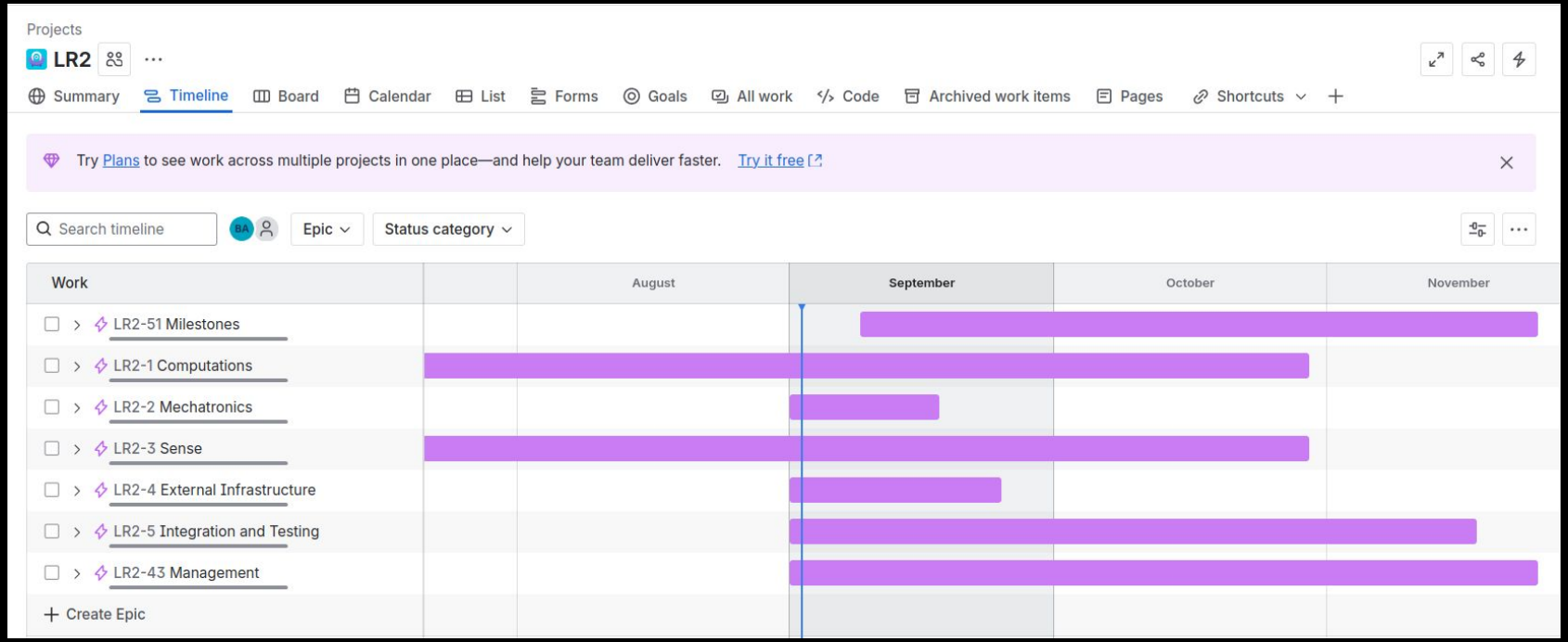
Standup Meetings Methodology

- Standups on MWF
- Weekly Meetings with Sponsor - Dr. William ‘Red’ Whittaker
- Everyone is and has been showing up :) (Sometimes members attend virtually)
- Questions being used:
  - What have you worked on since the last standup?
  - What are you going to work on?
  - Is there any help you need? Does your work affect anyone else’s ongoing work?
- Modifications from previous semester:
  - Switching from daily to every other day (lesser workload)

Project Management Tools

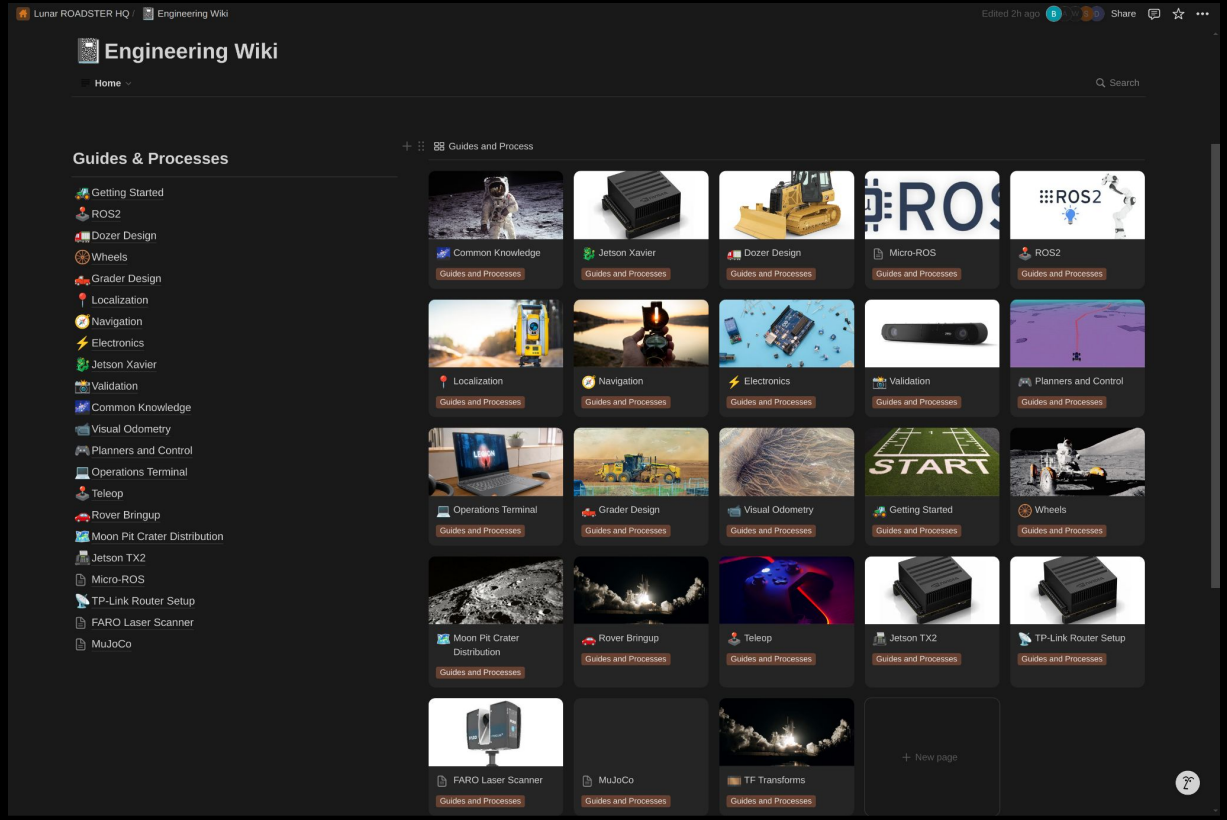
Jira - Managing Schedule

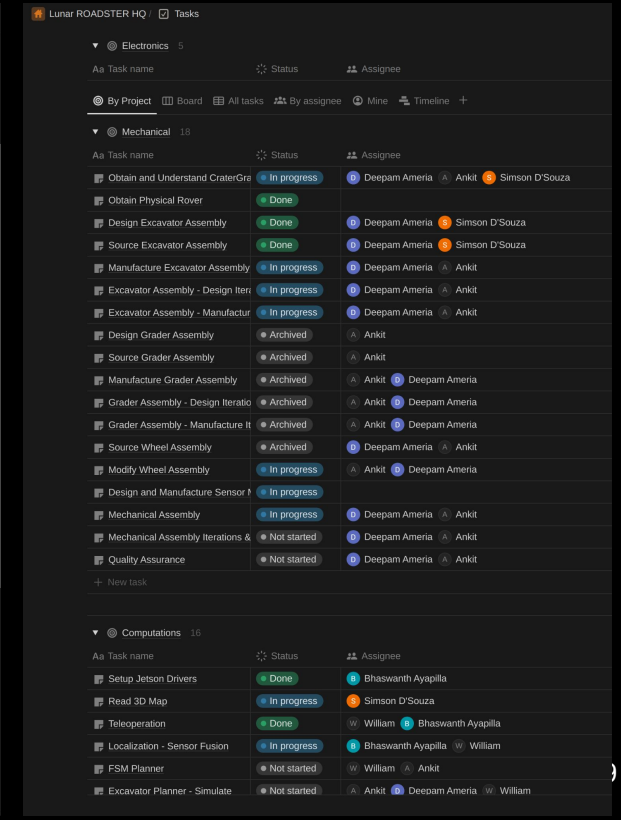
- Provides clear timeline view
- Gantt chart to track progress of each subsystem



Project Management Tools

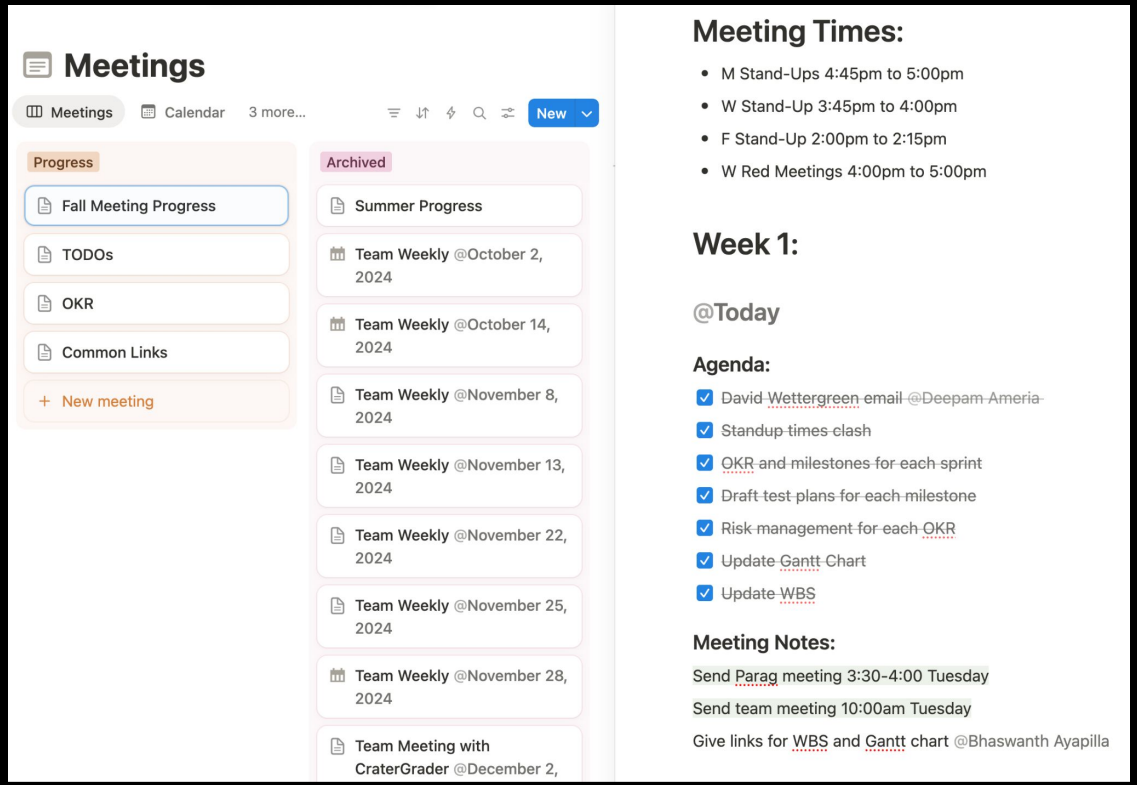
Notion - Documentation, Task Assignment & Meeting Notes





Project Management Tools

Notion - Documentation, Task Assignment & Meeting Notes



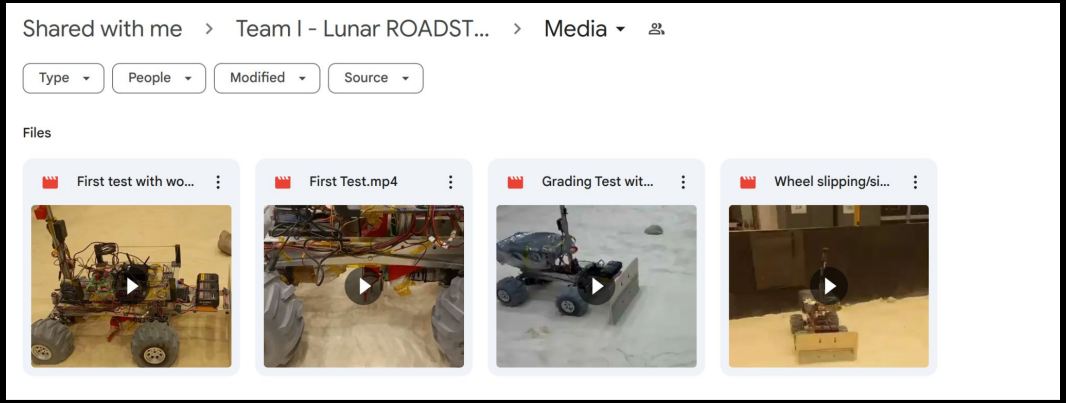
Project Management Tools

Discord

- Share important resources
- Hold virtual meetings
- Share test results and other relevant media

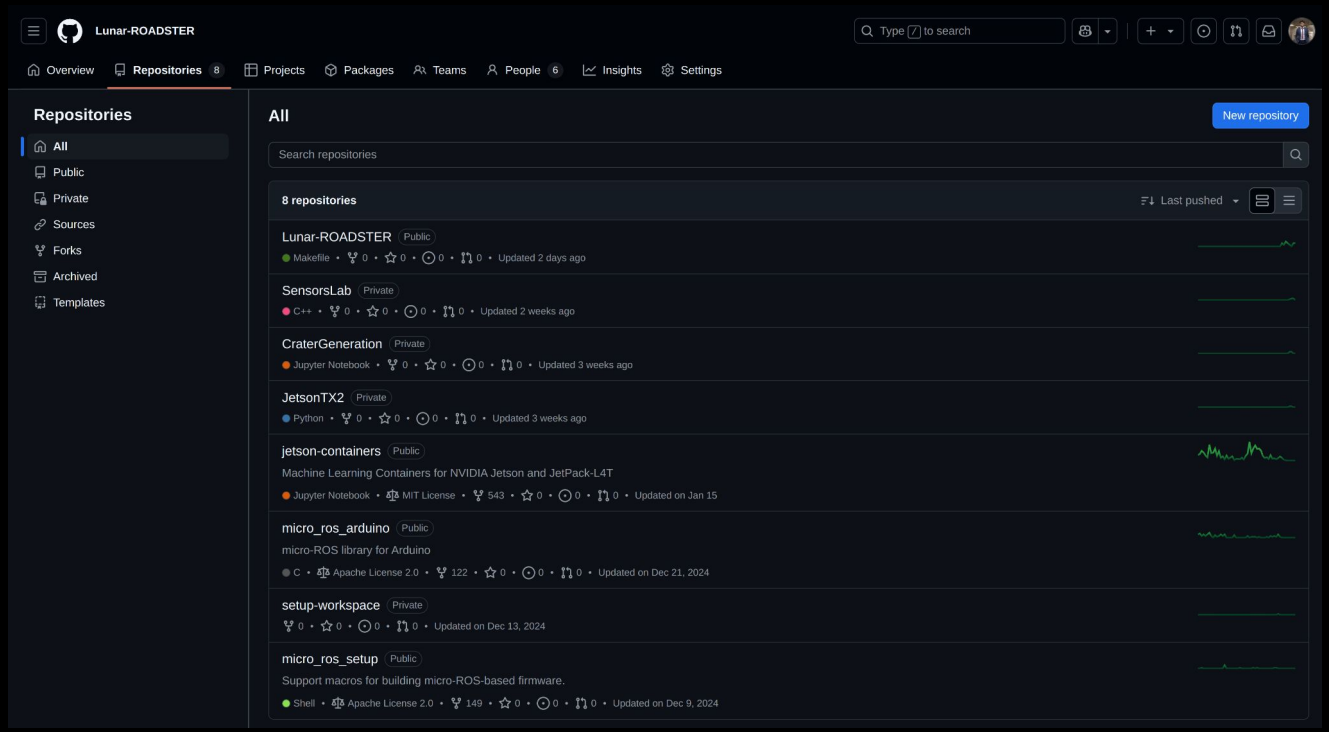
Google Drive

- Upload testing videos
- Upload project relevant documents



Project Management Tools

GitHub - Managing Code Collaboration



Lessons & Improvements

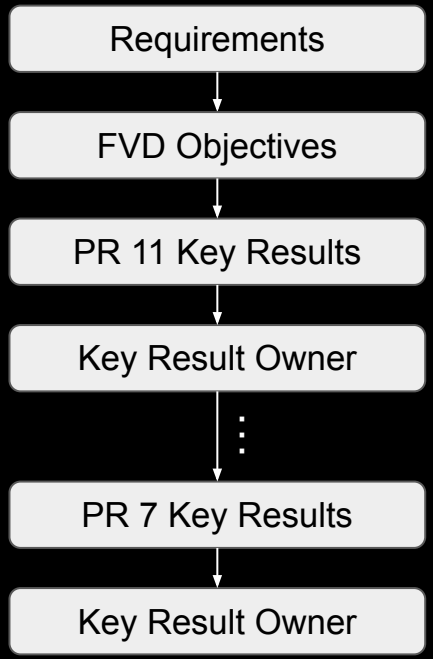
S.No	Lesson	Improvements
1	Deadlines and milestones missed leading to time crunch close to SVD	Introduced Objectives & Key Results (OKR) framework to track progress with development deadline in PR 10
2	Code sometimes breaks or fails to build due to newly added code	New GitHub version control to track new additions to codebase
3	Poor code documentation and difficult to read code	Standardized code convention and architecture
4	Testing schedule often missed as development is prioritized	Testing tracked as a part of OKR; testing schedule bundled with sprint milestones
5	Hardware needs to be robust, even small issues affect the entire timeline	Consistent testing of software on hardware in the MoonYard
6	Integration being left to the end causes unprecedented issues and delays	Following a Continuous Integration and Testing methodology

(New!) Objectives & Key Results (OKR)

Definition:

Framework used to define goals (Objectives) and track progress toward them (Key Results)

- Used mandatory performance requirements defined in Critical Design Review to come up with clearly defined Objectives for FVD (PR 12)
- Back-substituted to previous PRs to determine what Key Results needs to be achieved to achieve the Objectives for the next PR
- Assigned owners to each Key Result to benchmark contribution



(New!) Objectives & Key Results (OKR)

OKR Identifies:

- Risks
- Dependencies between work units
- Downscoping of requirements
- Stretch goals

OKR Guides:

- Work Breakdown Structure
- Gantt chart
- Testing schedule
- Risk mitigation plan

PR9 @October 8, 2025

@William: Validation stack tuning

@Deepam Ameria: Validation stack tuning

@Simson D'Souza: Local navigation controller, navigation tuning

@Ankit: Validation stack tuning

@Bhaswanth Ayapilla: Local navigation controller, navigation tuning

Test 4: Navigation planner has maximum deviation of 10%

Test 5: Avoid craters >= 0.5 meters

Test 6: Validation stack can accurately crater parameters

Test 7: Repeatability test of local nav controller

PR10 @October 29, 2025

@William: SkyCam

@Deepam Ameria: SkyCam

@Simson D'Souza: Planning stack

@Ankit: Tool planner, Planning stack, BEN stack

@Bhaswanth Ayapilla: Planning stack

Test 8: Autonomously fill craters of up to 0.5 meters in diameter and 0.1m in depth

Test 9: Groom the trail to have a maximum traversal slope of 5°

Test 10: SkyCam localization is reasonable (metric TBD)

Test 11: CPU/GPU usage of entire autonomous stack is below Orin compute limits

Risk 1: New Skycam method might not work (mitigating action is to revert back to total station)

Risk 2: Compute might be too high and over Orin compute limit

PR11 @November 12, 2025

@William: Skycam QA, Localization QA

@Deepam Ameria: Validation QA

@Simson D'Souza: Nav QA

@Ankit: Planning QA

@Bhaswanth Ayapilla: Hardware/wires QA, Arduino QA

Test 12: Maintenance, Reliability and Quality Assurance Test

Test 13: FVD preparation test

Risk 1: Unsatisfactory components for QA

FVD @November 17, 2025

O1: Plan a path with cumulative deviation of <= 25% (show in slides)

O2: Follow planned path to a maximum deviation of 10% (table with stats)

O3A: Climb gradients up to 15°

O3B: Contact pressure of less than 1.5 kPa (show in slides)

O4A: Avoid craters >= 0.5 meters (in demo)

O4B: Avoid slopes >= 15° (FVD Encore stretch goal)

O5: Fill craters of up to 0.5 meters in diameter and 0.1m in depth (in demo)

O6: Groom the trail to have a maximum traversal slope of 5° (in demo)

Test 14: Climb gradient of 15 degrees

Test 15: Run a bunch of tests prior or FVD and tabulate planned path deviation

Test 16: FVD Test

(New!) GitHub Code Version Control

Introduced documentation and coaching on version control and using pull requests and branching in GitHub

Version Control

Owner Tags

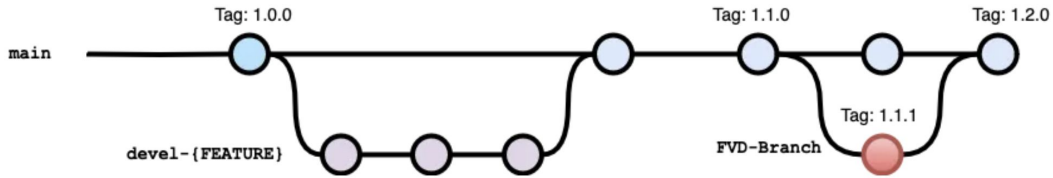
William Guides and Processes

Motivation:

Version control is an important aspect of software development. You will surely need to use it in your future workplace environment. It is used to keep track of code changes and manage the entire software development process. Ideally, we should not be developing in `main`, and instead should be developing in individual branches and merge into `main` after the developed code has been tested and staged. We will be using Git version control in our project, as it integrates nicely with our GitHub repository.

Workflow:

Ideally, the version control workflow should be something like the diagram below:



(New!) Code Quality & Architecture

Code Quality:

- Standardized code namespace, header information, and package.xml metadata

```
/**
 * @file
 * @brief DESCRIPTION OF CODE
 *
 * @author AUTHOR NAME
 * @version 1.0.0
 * @date CURRENT DATE
 *
 * Maintainer: MAINTAINER NAME
 * Team: Lunar ROADSTER
 * Team Members: Ankit Aggarwal, Deepam Ameria, Bhaswanth Ayapilla, Simson
 *
 * Subscribers:
 * - /SUB: [MSG TYPE] Description
 *
 * Publishers:
 * - /PUB: [MSG TYPE] Description
 *
 * Services:
 * - /SRV: [SRV TYPE] Description
 *
 * @credit NAME IN PACKAGE XML, Team CraterGrader (if code was originally v
 */
```

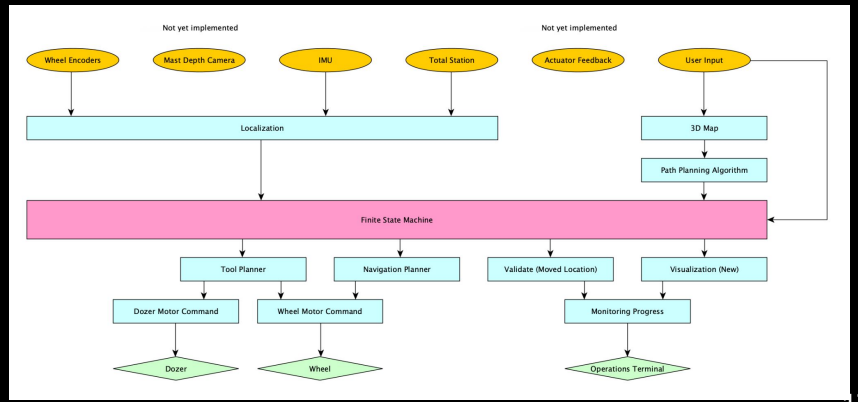
Code Architecture:

- Refactored ROS topics to conform to a Directed Acyclic Graph (DAG) architecture

Definition:

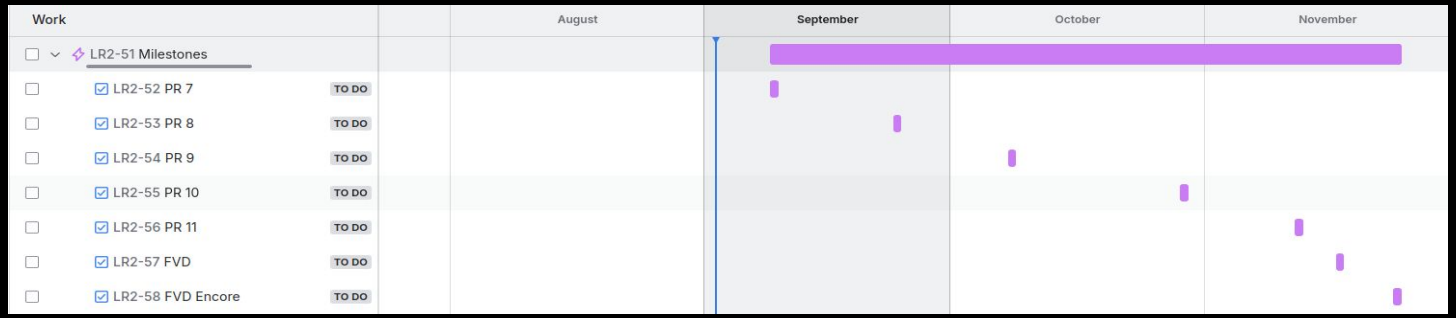
A finite directed graph with no directed cycles.

Arrows dictate the flow of information

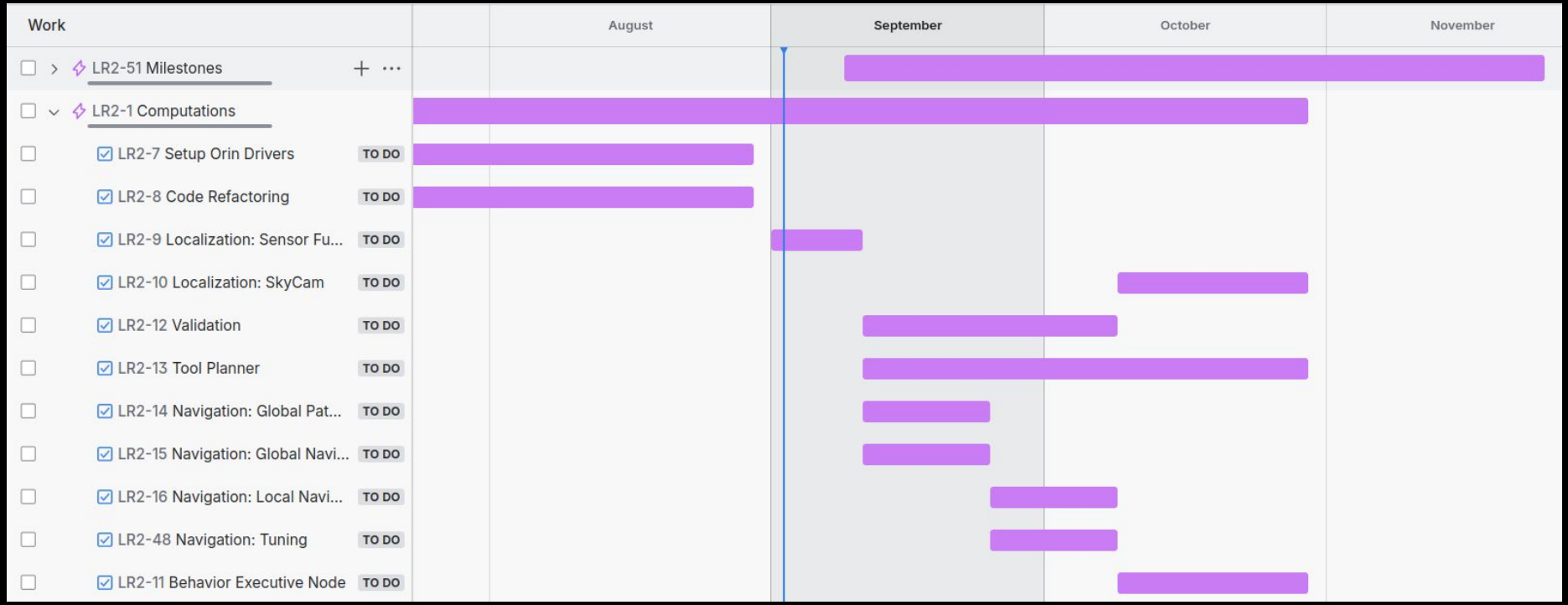


Schedule and Milestones

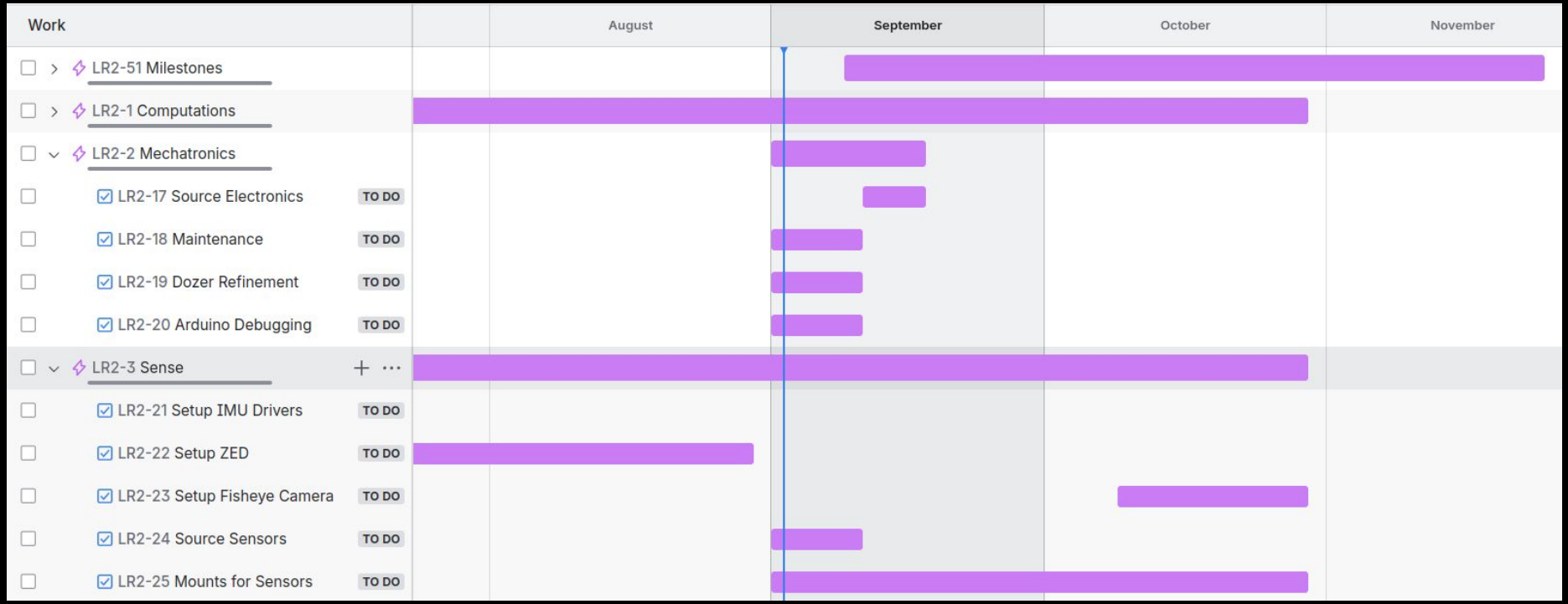
Milestones	Goals
PR 7 (Sept 10th)	Hardware and software refinement
PR 8 (Sept 24th)	Initial setup of validation stack and navigation planner
PR 9 (Oct 8th)	Tune validation stack and navigation stack
PR 10 (Oct 29th)	SkyCam-based localization for improved global positioning
PR 11 (Nov 12th)	Full system integration and quality assurance testing
FVD (Nov 17th)	Autonomous grading of multiple craters
FVD Encore (Nov 24th)	Autonomous grading of multiple craters using SkyCam



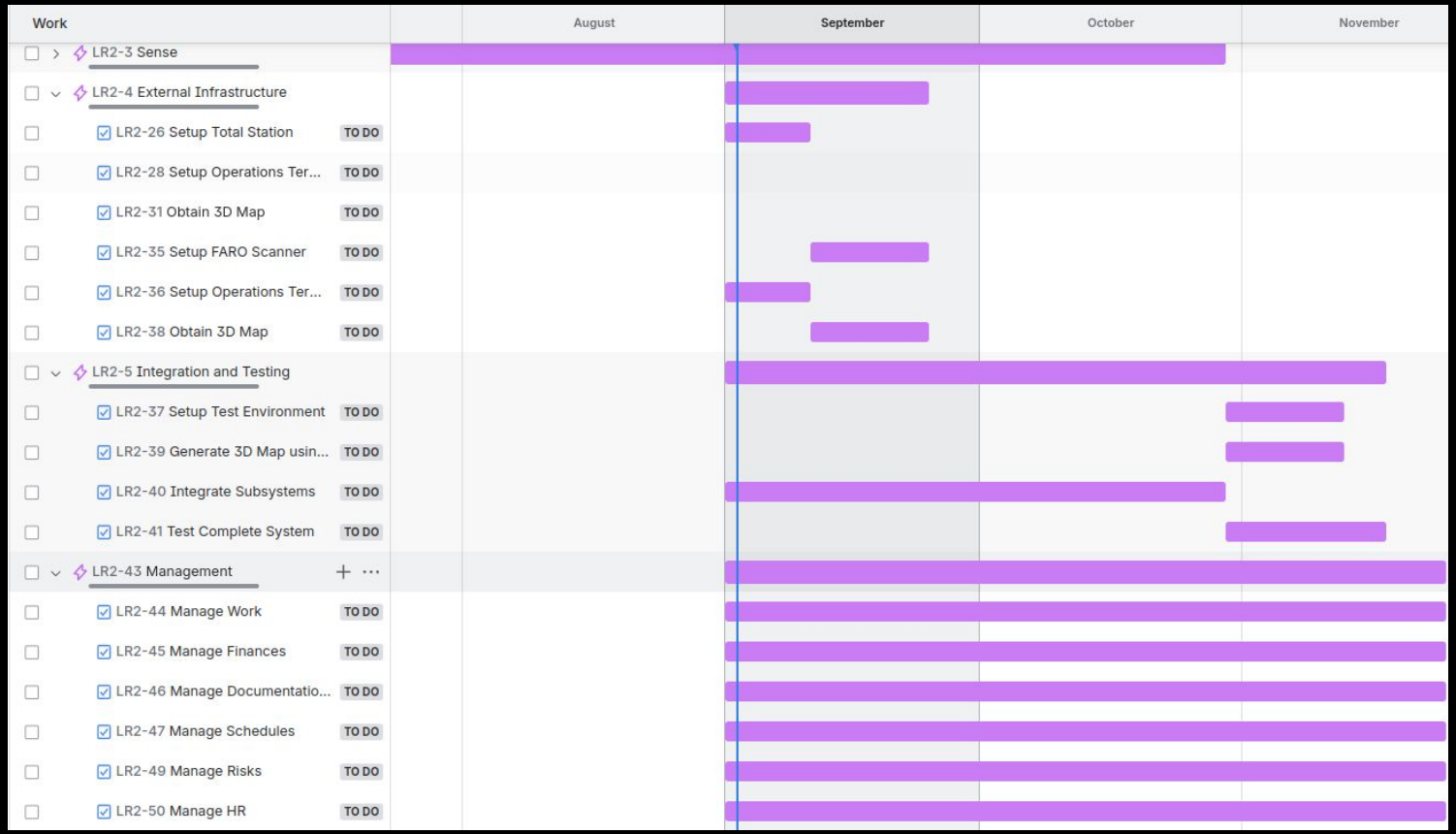
Schedule and Milestones



Schedule and Milestones

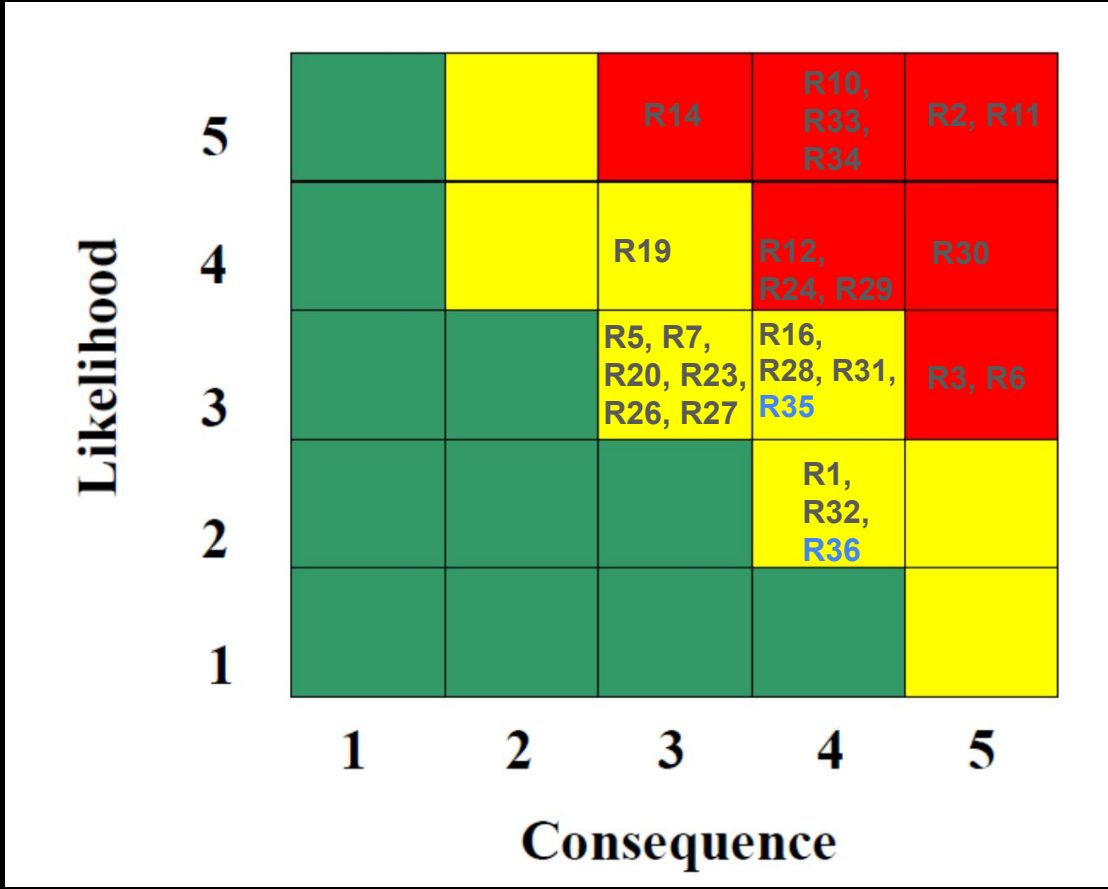


Schedule and Milestones



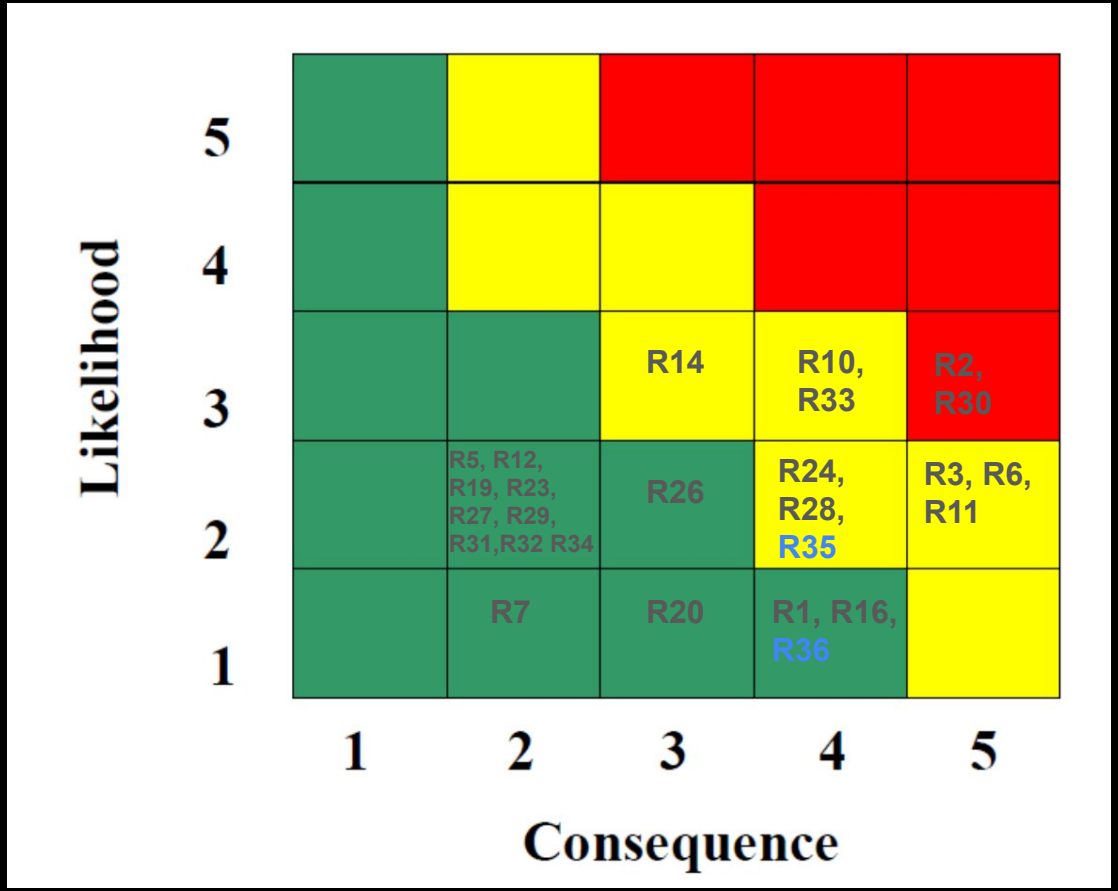
Risk Management (Updated)

Risk Summary



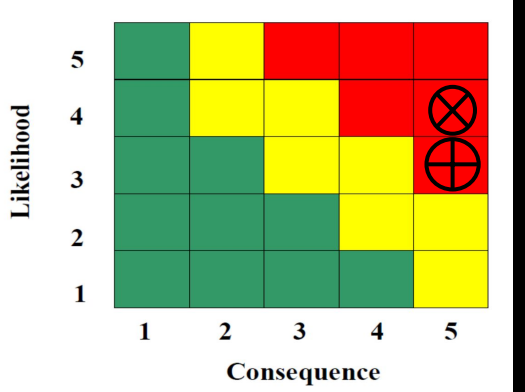
Risk Management (Updated)

Reduced Risk Summary



Top Risks

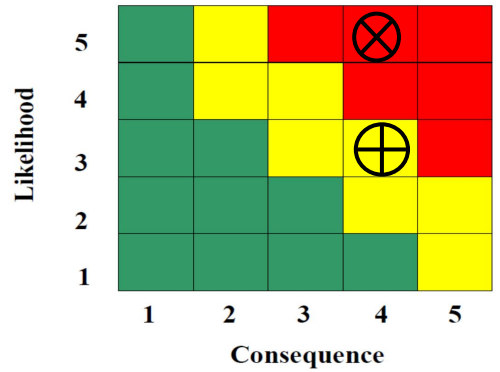
Risk ID	Risk Title	Risk Owner	Risk Type:	Logistics
R30	No spares available	Team		
Description		**Date Added**		
Discontinued model, spare parts unavailable		3/4/2025		
		Date Updated		
		8/30/2025		
Consequence				
The whole project falling through, or redo almost all subsystems on a different rover.				



Action/Milestone	Success Criteria	Date Planned	Date Implemented
Check out eBay and other similar platforms for spares	Successfully find exact spares on these platforms	3/6/2025	
Check out and stock similar parts if not same	Successfully find and stock similar parts	3/6/2025	
Find a twin rover that was used by a previous team on campus	Successfully find the twin rover and scavenge parts	3/6/2025	3/7/2025
Maintain all parts, especially mechanical parts	Successfully avoid future breakdowns and part failures	9/10/2025	

Top Risks

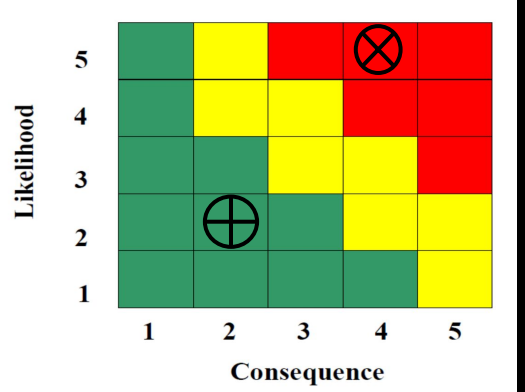
Risk ID	Risk Title	Risk Owner	Risk Type:	Technical
R33	Localization frame shift after total station battery swap	Bhaswanth, William		
Description		**Date Added**		
Battery replacement in the total station causes small frame offsets, leading to localization inaccuracies.		3/4/2025		
		Date Updated		
		9/1/2025		
Consequence				
Leads to poor navigation performance and risk of missing the crater during grading operations.				



Action/Milestone	Success Criteria	Date Planned	Date Implemented
Implement resection method using three known prism locations instead of orientate-to-line	Successfully fix the frame consistently after battery swaps	9/1/2025	
Explore and test alternative localization methods (using SkyCam)	Successfully maintain localization accuracy	10/9/2025	

Top Risks

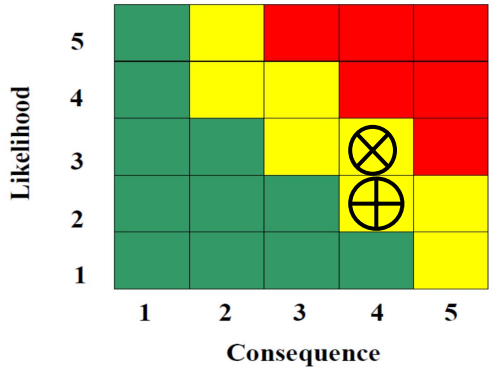
Risk ID	Risk Title	Risk Owner	Risk Type:	Technical
R34	Arduino requires reset before operation	Bhaswanth		
Description		**Date Added**		
Arduino needs to be manually reset each time before starting autonomy or switching between autonomy and teleoperation modes.		3/4/2025		
		Date Updated		
		4/10/2025		
Consequence				
Slows down setup time and impacts operational readiness, delaying mission start and mode transitions.				



Action/Milestone	Success Criteria	Date Planned	Date Implemented
Check USB port permissions and drivers issues on Jetson	Successfully establish consistent serial connection without reset	4/26/2025	
Verify that Arduino is connected via USB 3.0 instead of USB 2.0 port	Ensure stable high-speed communication	4/26/2025	
Check for ROS node frequency mismatches causing packet loss to Arduino	Match ROS publish/subscribe rates	4/26/2025	

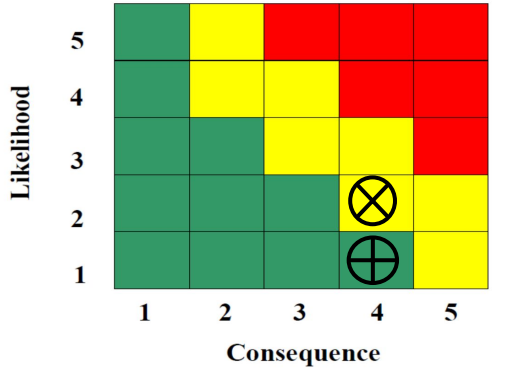
Top Risks

Risk ID	Risk Title	Risk Owner	Risk Type:	Technical
R35	Integration of ZED Camera with Lunar-ROADSTER stack	Deepam		
Description		**Date Added**		
Installing and setting up ZED SDK and ROS2 Wrappers in Docker might pose a problem due to insufficient support		4/24/2025		
		Date Updated		
		8/30/2025		
Consequence				
Delays in development and integration of Validation and Tool Planning sub-systems				
Action/Milestone		**Success Criteria**	**Date Planned**	**Implemented**
Start installation and setting up early (during Summer)		Successfully install the ZED SDK and ROS2 wrapper in Docker	4/26/2025	8/22/2025
Revert Back to Intel Realsense D435i		Successfully integrate Realsense D435i instead of ZED	4/26/2025	



Top Risks

Risk ID	Risk Title	Risk Owner	Risk Type:	Logistics
R36	PRL Moonyard Access	William		
Description		**Date Added**		
Securing Moonyard access for testing/demos will be restricted and challenging		8/29/2025		
		Date Updated		
		8/29/2025		
Consequence				
No testbed available for testing and/or SVD				



Action/Milestone	Success Criteria	Date Planned	Date Implemented
Devise and discuss a testing and demo plan with Prof. Red and Prof. David Wettergreen beforehand and reserve slots	Successfully meet and discuss the schedule of high priority projects	9/9/2025	
Complete Medical Evaluation to get unrestricted but controlled access	Successfully complete the Medical Evaluation and get unrestricted access to the Moonyard	9/5/2025	

Amalgamation of Traditional and Agile PM

Traditional PM:

- Well-defined requirements and a clear project scope.
- Structured schedule with milestones to meet deadlines.
- Tasks are assigned based on a planned workflow, with periodic check-ins to track progress and ensure alignment with project goals.
- Team members document their work thoroughly for transparency and tracking.
- Test plans and risk management studies are conducted to identify and mitigate potential issues.
- Budget and resources are carefully planned to ensure the project stays within budget.

Agile PM:

- Early failure, quick adaptation, and rapid improvements.
- Sponsor and stakeholder inputs drive subsystem refinements for the best MVP.
- Individual ownership of tasks; linear organization
- Daily stand-ups and Kanban boards for progress and scheduling.
- Test-Driven Development ensures robust hardware and software reliability.
- Team members collaborate across disciplines as needed.

Amalgamation of Traditional and Agile PM

Why we chose amalgamation?

- Structured Planning with Adaptive Execution.
  - Clear objectives and well-defined requirements, with quick adaptation to changes in subsystems without changing the scope
- Risk Mitigation and Continuous Improvement
  - Predictive risk analysis and mitigation plans, with iterative approach for quick failure, fast learning and continuous enhancement
- Efficient Team Workflow
  - Traditional task allocation and ownership, with cross-functional collaboration and decentralized decision making
- Predictability and Adaptability
  - Predictive budgeting, scheduling and resource allocation, with allowing adjustments based on real-world testing and feedback

Amalgamation of Traditional and Agile PM

Challenges of amalgamation

- Daily stand-ups improve collaboration, but if not kept concise and focused, they can become repetitive and time-consuming, reducing productivity.
- Agile allows frequent feedback, but balancing changes with a fixed timeline is challenging.
- Traditional PM requires detailed documentation, while Agile prioritizes speed. Finding the right balance is crucial.

What Might Be Done Differently?

Better Schedule Tracking -

- Identification of risks
- Update issue log in timely manner

Issues Log

10 entries per page

Search:

Issue ID	Date Initiated	Date Resolved	Participants	Description	Options	Resolution	Justification
I01	11/28/2024	12/04/2024	Team	Too many performance requirements for SVD.	Have revised performance requirements separately for SVD and FVD.	Revised performance requirements down to 6. Clearly defined SVD and FVD objective split.	Conducted meeting with Crater Grader team and discussed what is feasible and what is not in the given time.
I02	01/20/2025	01/27/2025	Boxiang Fu	Unable to login to TX2 chip.	Flash the chip and build docker container from scratch.	Found that chip was used by LunarX team. Got in contact and obtained login details.	No need to reinvent the wheel if not necessary.
I03	02/10/2025	02/14/2025	Ankit Aggarwal	Steering mechanism components failed due to wear-and-tear.	Replace broken parts.	Replaced all components of the assembly and fitted new screws and bolts.	Replaced old parts as a precaution for further failure due to wear-and-tear.

Showing 1 to 3 of 3 entries

1

Stand-Up Meetings

- Daily Standups on Weekdays (5:30 - 5:45 PM)
- Weekly Meetings with Sponsor - Dr. William ‘Red’ Whittaker (Friday)
- Everyone is showing up :) (Sometimes members attend virtually)
- Questions being used:
  - What have you worked on since the last Standup?
  - What are you going to work on?
  - Is there any help you need? Does your work affect anyone else’s ongoing work?
- Some key highlights where standup solved problems
  - Resource Reallocation to meet a deadline - Project Course Assignments, Internal Milestones
  - Re-assignment of member’s tasks based on help needed
  - Bandwidth Management
  - Insights into technical work: Helping wherever a member is ‘stuck’
  - Awareness of all ongoing work - makes us feel like we are ‘working in a team’, not in silos