Individual Lab Report 6

Lunar ROADSTER

Team I

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

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

Teammate: **Bhaswanth Ayapilla**
ID: bayapill
E-mail: *bayapill@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*

September 11, 2025

Carnegie Mellon University
Robotics Institute

1 Individual Progress

Since the last progress review in April, I worked on improving our code quality and readability. I also worked on refactoring some parts of the codebase to make sure our code architecture conforms to a particular architecture. Finally, it is my turn to be the project manager this semester. I took some time to work with the group and sort out our work breakdown and tasks that needs to be completed this semester.

1.1 Code Quality Improvements

Over the summer, we refactored our code so that we improved its readability and quality. Our previous codebase had three different namespaces. They were: 1. `cg` from CraterGrader, 2. `lx` from Lunar-X, and 3. no namespace. We standardized all the level one namespaces of our code to be under the same `lr` (Lunar ROADSTER) namespace. The level two namespace is standardized to be the node name. Personally, I was responsible for refactoring the sensing, motion control, launcher, and driver (which contained sub-directories) packages.

We also added header information to all our files containing ROS nodes. A template header is shown in Figure 1. It is written in Doxygen config format and includes important information such as its author, version number, date last edited, maintainer, and the publishers/subscribes/services the node interfaces with.

```
/**
 * @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 \
 */
```

Figure 1: Template header information for ROS node

We also standardized our codebase's `package.xml` metadata using the template shown in Figure 2. It includes important information such as the package name, its version number (synced with the version in the header), description of what the code does, its maintainer, and its license (we have decided to be more open source and adopted the Apache 2.0 license). Our intention for these code quality improvements is to make the code more readable and adaptable should future MRSD teams or projects decide to use our code.

```
<name>PACKAGE NAME</name>
<version>1.0.0</version>
<description>INTENDED USE for Lunar ROADSTER project</description>
<maintainer email="NAME@cs.cmu.edu">NAME</maintainer>
<license>Apache License 2.0</license>
```

Figure 2: Template for package metadata

1.2 Code Architecture Improvements

During my summer internship, I learned of a code architecture called the directed acyclic graph (DAG) architecture. A DAG is a finite directed graph with no directed cycles. In a DAG graph, each node represents an algorithm operation and the arrow between nodes represents the workflow direction (see Figure 3). There are no loops in the graph (although loops inside each node are permitted). They are useful for visualizing and understanding complex systems, reduce debugging problems, enable efficient scheduling of tasks, and ensuring that operations are performed in the correct sequence. Its greatest advantage is that information only flows in one direction, so we can monitor the connection edges (e.g. its Hz rate, its content, etc) for points of failure and resolve issues quickly.

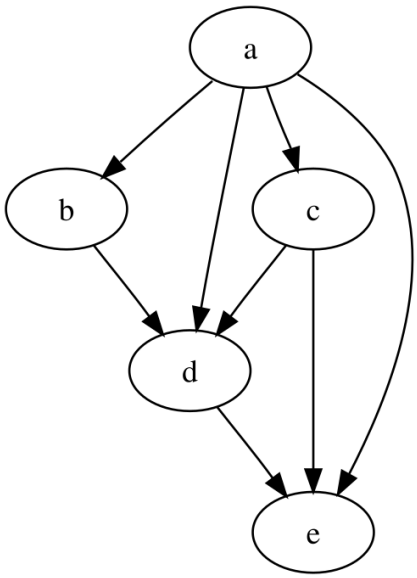

Figure 3: A directed acyclic graph

After realizing that our existing codebase only requires minimal refactoring efforts to conform to this architecture, we decided to refactor the non-conforming nodes so that our pipeline follows the DAG architecture. The main changes made were to the validation stack and visualization stack. After changing our software architecture and making adjustments, our new software architecture is visualized in Figure 4 and conforms to the DAG architecture.

1.3 Project Management Duties

I inherited the project management role from Ankit and spent some time planning out my duties. Such tasks are more managerial in nature and included items such as how to best track work progress and how to allocate tasks. A major contribution that I implemented this semester is the Objectives and Key Results (OKR) framework. It is used to define goals (Objectives) and track progress toward them (Key Results). We

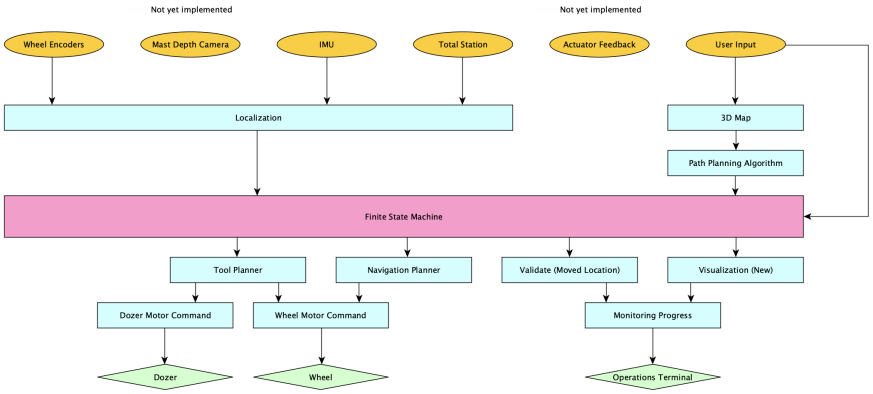

Figure 4: Lunar ROADSTER software architecture

start off from the mandatory performance requirements defined in the Critical Design Review to come up with clearly defined Objectives for FVD (PR 12). We then back-substitute to previous PRs to determine what Key Results needs to be achieved to obtain the Objectives for the next PR. Finally, owners are assigned to each Key Result to benchmark contribution.

I worked with the team and sorted out the objectives required for each PR/sprint. Ownerships were assigned depending on each team member's expertise and interests. By doing this at the start of the semester, we were able to sort out our Work Breakdown Structure (WBS) and Gantt Chart for the semester. For each Objective, we were also able to brainstorm and come up with possible risks that might impact completion and tests that could be included in our Fall Test Plan to confirm functionality. An example OKR for Progress Review 8 is shown in Figure 5.

PR8 @September 24, 2025

@William: Validation stack code, validation stack integrated

@Deepam Ameria: Validation stack code, validation stack integrated

@Simson D'Souza: Global navigation controller, selection of gradable craters and FARO map generation

@Ankit: Validation stack code, validation stack integrated

@Bhaswanth Ayapilla: Global path planner

Test 2: Global path planner paths with cumulative deviation of <= 25%

Test 3: Filtering and selection of gradable craters

Risk 1: No real experience in validation stack

Risk 2: Detecting point cloud is hard in uniform sand environment

Risk 3: Pure pursuit not enough space in Moon Yard for overshoot

Risk 4: Pure pursuit turning radius is large

Risk 5: Pure pursuit wiggle thing

Risk 6: Path planner how to add constraints

Risk 7: Accuracy of determining gradable craters

Figure 5: OKR for PR8

2 Challenges

The main challenge faced was that our Arduino was behaving irregularly ever since the Spring semester. During SVD Encore, a reset issue caused by the Arduino meant that we were not able to initiate the autonomy stack, and resulted in the rover becoming idle during the Encore demonstration. More recently on our test day on Monday Sept 8th, the Arduino's connection with the compute unit (both Jetson and Orin) was totally lost. We were unable to send commands (via the Arduino) to the motor encoders, and thus our rover was stationary and unable to move. This is a major blocker and we are hoping to resolve this issue as soon as possible.

Another challenge faced was that during the Summer semester, the school implemented new access policies for the Planetary Robotics Lab Moon Yard. This is because of the fine particulate regolith causing respiratory issues. The policies meant that we are restricted from using the Moon Yard until the appropriate safety trainings are completed. Even then, our access would be controlled and limited. We are attempting to resolve this challenge by planning further ahead and setting a fixed schedule for testing. We are also pushing to get our medicals evaluated and the safety training completed as soon as possible.

3 Teamwork

A breakdown of the contributions of each team member are tabulated below:

- **Ankit Aggarwal**: My work was carried out in collaboration with Simson and Deepam to resolved hardware issues. This involved complete maintenance of the wiring connections and re-routing them in a safe and efficient configuration. Additionally, I worked on the designing mounts for the Orin and the Zed 2i camera. I am also working with Bhaswanth and Deepam to debug the Arduino communication issues. Beyond this, the entire team collaborated on systems engineering discussions, brainstorming ideas for planning and improvements, and running maintenance tests on the rover in the Moon Yard.
- **Deepam Ameria**: My initial work was in collaboration with Bhaswanth over the summer, we worked on setting up the new Docker on the Orin. We worked on solving several dependency and version issues. I also worked on integrating the ZED Camera (SDK and ROS2 Wrapper) with the new Orin. I also worked with Simson and Ankit to carry out the hardware maintenance of the rover. We worked on replacing damaged wires, rerouting them, and soldering the connections to make them more robust. Additionally, I'm working with Bhaswanth and Ankit to solve Arduino communication issues. Moreover, I collaborated with the team on systems engineering discussions, brainstorming ideas and plans for upcoming semester and conducting maintenance tests on the rover in the Moon Yard.
- **Bhaswanth Ayapilla**: My initial work involved collaborating with Deepam over the summer to set up Docker on the Orin. Together, we resolved several Docker-related issues, configured Docker Compose, and added useful aliases, although most of the set up was completed by him. He also supported me in developing the Master Launcher. Currently, I am working closely with Ankit and Deepam to debug the Arduino communication issues. Beyond this, the entire team collaborated on systems engineering discussions, brainstorming ideas for planning and improvements, and running maintenance tests on the rover in the Moon Yard.

- **Simson D'Souza**: My work was carried out in collaboration with Deepam and Ankit to identify and resolve hardware issues. This primarily involved replacing damaged wires, rerouting them for a cleaner setup, and soldering connections to ensure reliability during transportation. In addition, I worked on the setup of the graphical user interface (GUI); an initial version of the dashboard has been created and will be updated as new modules are integrated. I also contributed to the collaborative effort of refactoring the codebase and conducting rover maintenance tests.

4 Plans

From now until Progress Review 8, I will be mainly focusing on resolving the Arduino connection and reset issue discussed above in the Challenges section. This is a major blocker and has my highest priority. After resolving this issue, I will be focusing on finalizing the methodology for the validation stack. This includes the code architecture and gateways to other nodes. If time permits, I plan to also begin implementing portions of the validation stack.