Individual Lab Report 10

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*

November 12, 2025

Carnegie Mellon University
Robotics Institute

1 Individual Progress

Since the last progress review, I mainly worked on resolving some bugs related to the Skycam localization. The roll and pitch of the rover caused the localization to drift significantly. I used a mechanical gimbal to solve this problem. Next, I worked on implementing the skeleton code of the Behavior Executive Node (BEN). This is the main integration unit for our entire rover. Finally, I worked with the entire team to integrate all subsystems and perform quality assurance.

1.1 Skycam Localization

As mentioned in the previous progress review, the new Skycam localization method's accuracy was susceptible to the roll and pitch of the rover. This is because the main optimizer used is a closed-form Levenberg–Marquardt. The algorithm assumes that the rover is operating on a flat surface. Therefore, whenever this assumption is invalidated, the localization would drift. We were able to solve this problem by mounting the fisheye camera on top of a mechanical gimbal. The motors in the gimbal would make sure to de-roll and de-pitch the rover's movements so that the skycam is always planar to the ceiling. The mechanical setup of the gimbal is depicted in Figure 1. My teammate Deepam helped me with the CAD design of the fisheye mount and Ankit helped me with 3-D printing the mount.

Figure 1: Gimbal setup for the fisheye camera

After the fisheye camera is mounted on the gimbal. Extensive testing of the new localization unit was performed. Testing videos of the skycam can be found on YouTube here. The test shows that the roll and pitch issues have been solved, and the localization accuracy is very close to the ground truth provided by the total station.

1.2 Behavior Executive Node

My next task was on implementing the BEN stack. This will be used as the high-level behavior logic for our entire system. The states and transitions were discussed and finalized by the entire team during a brainstorming session. We finalized on the architecture shown in Figure 2.

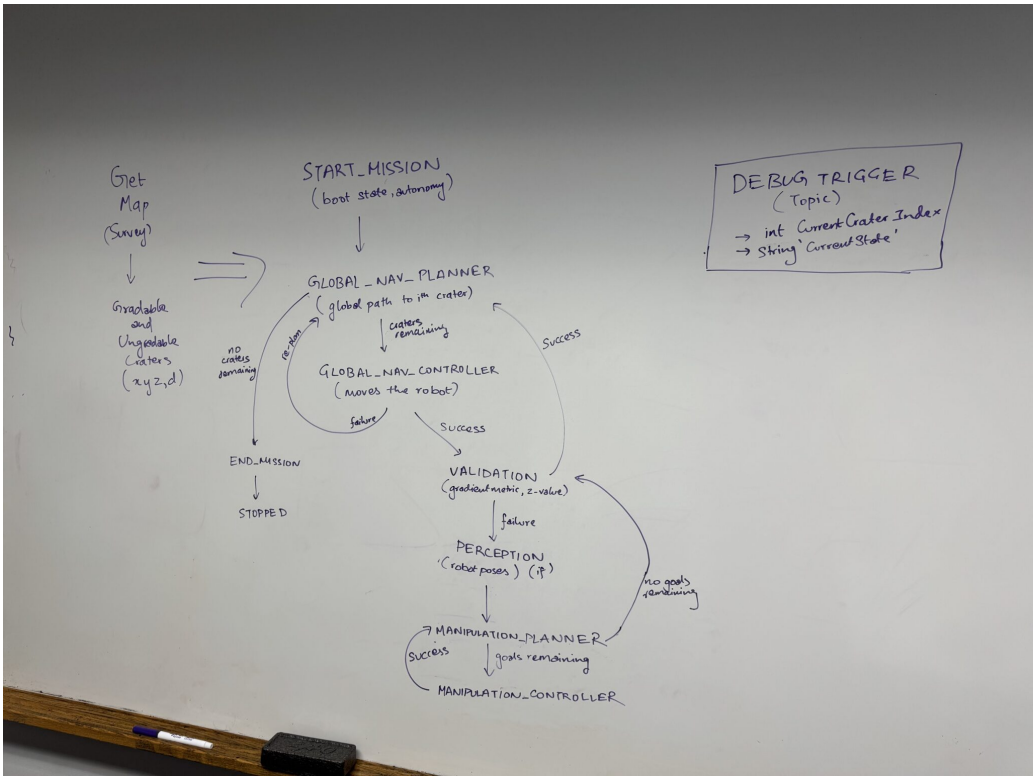

Figure 2: BEN architecture

After the architecture is finalized, I wrote the skeleton code for the architecture. This involved writing the FSM class (Figure 3) and FSM callbacks (Figure 4). For each callback, a `fsmRunState` function is called. Each team member is tasked with populating the topics, services, and actions called from their `fsmRunState` function to their ROS nodes. I was responsible for implementing the `fsmRunStartMission`, `fsmRunValidation`, `fsmRunEndMission`, `fsmRunStopped`, `fsmRunDebug`, and `fsmRunManualOverride` function callbacks.

```
public:
    enum class State {
        START_MISSION,
        GLOBAL_NAV_PLANNER,
        GLOBAL_NAV_CONTROLLER,
        VALIDATION,
        PERCEPTION,
        MANIPULATION_PLANNER,
        MANIPULATION_CONTROLLER,
        END_MISSION,
        STOPPED,
        DEBUG,
        MANUAL_OVERRIDE
    };

    // Constructor
    FSM(){};
    FSM(State start_state_);

    // Destructor
    ~FSM();
```

Figure 3: FSM state class

```cpp
void BenNode::fsmTimerCallback()
{
    RCLCPP_INFO(this->get_logger(), "--------------------------------------------------------------------------");
    RCLCPP_INFO(this->get_logger(), "[FSM] Current State: %s", fsm_.currStateToString().c_str());

    switch (fsm_.getCurrState())
    {

    case lr::ben::FSM::State::START_MISSION:
        fsmRunStartMission();
        break;

    case lr::ben::FSM::State::GLOBAL_NAV_PLANNER:
        fsmRunGlobalNavPlanner();
        break;

    case lr::ben::FSM::State::GLOBAL_NAV_CONTROLLER:
        fsmRunGlobalNavController();
        break;

    case lr::ben::FSM::State::VALIDATION:
        fsmRunValidation();
        break;

    case lr::ben::FSM::State::PERCEPTION:
        fsmRunPerception();
        break;

    case lr::ben::FSM::State::MANIPULATION_PLANNER:
        fsmRunManipulationPlanner();
        break;

    case lr::ben::FSM::State::MANIPULATION_CONTROLLER:
        fsmRunManipulationController();
        break;

    case lr::ben::FSM::State::END_MISSION:
        fsmRunEndMission();
        break;

    case lr::ben::FSM::State::STOPPED:
        fsmRunStopped();
        break;

    case lr::ben::FSM::State::DEBUG:
        fsmRunDebug();
        break;

    case lr::ben::FSM::State::MANUAL_OVERRIDE:
        fsmRunManualOverride();
        break;

    default:
        RCLCPP_WARN(this->get_logger(), "[FSM] State not recognized!");
        break;
    }
}
```

Figure 4: FSM callback

Additionally, I also set up the parameter YAML files for the BEN. This file is used to declare parameters so that tuning can be easier. I also implemented different callback groups and made the node run as a multi-threaded executor. This is so different units can interact with the BEN node simultaneously and not be blocked by another process. Finally, I also implemented robot localization callbacks and triggers for the debug state, verbosity, and manual override (teleop).

1.3 Quality Assurance

For the remainder of the time, I was involved in performing quality assurance (QA) on the packages I implemented. I first performed QA on the validation unit. I tested the unit on a flat surface, a flat surface with footsteps, and a crater with 15 degree elevation. The validation results returned are shown in Figure 5. The slope calculated and the grading success boolean returned by the unit is very accurate. Therefore, the validation unit passed quality assurance and is ready to be integrated to the rover system.

My next QA task was the compute limits of the Orin. I tracked the compute usage of the Orin while it performed our grading routine. The CPU compute rarely exceeded 80% and the average 15 minute load on the compute was 4.66 cores on a 8 core system. A snapshot compute usage is shown in Figure 6. Therefore, the compute unit passed quality assurance and is sufficient for our rover system.

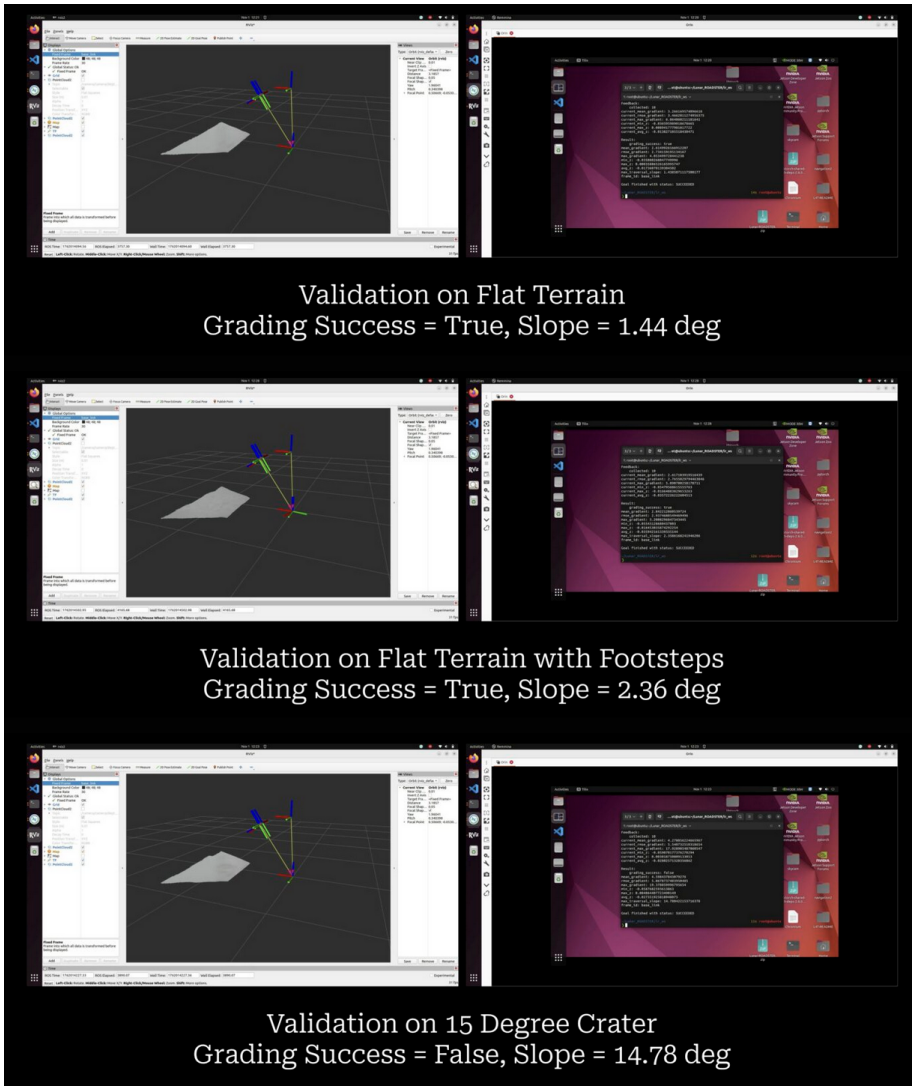

Figure 5: Validation QA

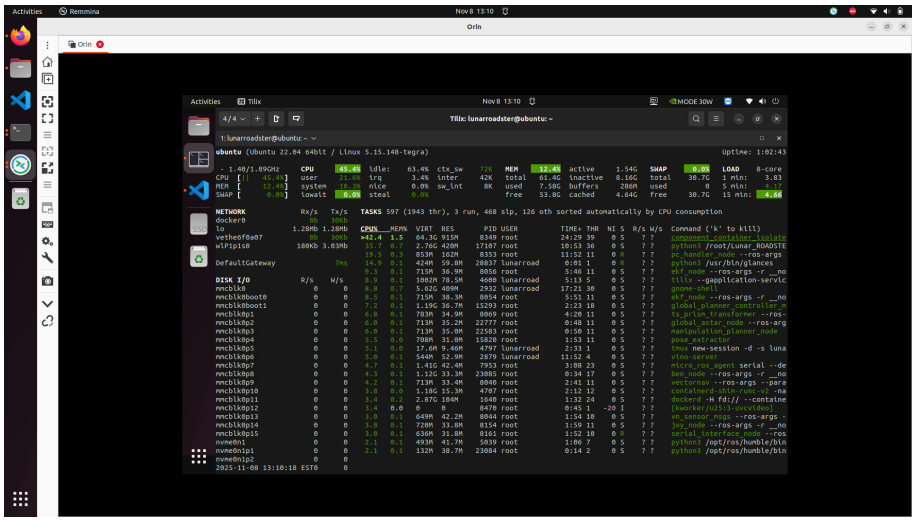

Figure 6: Orin compute QA

2 Challenges

The most significant challenge is coordinating the integration of the entire system. Each unit needs to be integrated with the BEN stack. However, each unit may also have dependencies from the other units. For example, the validation unit's output is used by

perception to determine the tool height. The manipulation planner state uses waypoints provided by the planning unit to guide the grading. Sorting out the dependencies between the units and debugging integration issues caused significant challenges to the entire team.

Furthermore, as the FVD date grows closer, the Orin is almost always in use and the compute resource needs to be shared between team members. Whenever other team members are using the Orin, it blocks my progress as I can't use the compute. Therefore, sorting out who uses the Orin at what time is also a challenge leading up to FVD.

3 Teamwork

For this progress review, I collaborated with Deepam and Ankit to design the skycam mount on the gimbal. Afterwards, I collaborated with the entire team on integrating the entire system and debugging integration issues.

- **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. Ankit worked with Bhaswanth for tuning offsets of robot pose extractor part of perception. He also collaborated with Deepam and Simson to replace the front drive actuation system. Ankit 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 Bhaswanth 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.
- **Bhaswanth Ayapilla**: Bhaswanth worked on the full system integration and testing. He collaborated with Simson in fine tuning the global navigation controller and manipulation controller. He also worked with Simson in writing the BEN stack and ensuring correct logic switches within navigation. Bhaswanth worked with the entire team in deciding the BEN stack architecture, tuning it and feature development to make it robust. He worked with Ankit and Deepam in tuning the planning stack and ensuring correct poses relative to the actual crater centroid. He 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, Bhaswanth conducted localization and navigation quality assurance tests.
- **Simson D'Souza**: Simson worked on full system integration and testing, collaborating with Bhaswanth 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. Simson 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.

4 Plans

From now until FVD, I will continue working on the integration of the various subsystems of the rover. In particular, I plan on mostly working on the integration of the validation subsystem and the skycam unit with the BEN unit. Since FVD is approaching on November 17th, I will be wrapping up on implementing new functionalities and instead focus my effort on integration and testing out everything works as intended for the demonstration.