University of Arkansas – CSCE Department
Capstone II – Final Report – Spring 2021

NASA Robotic Mining Competition

William Burroughs, Calvin Franz, Z. Gunner Lawless, Jett McCullough, Carson Molder

Abstract

The goal of this project was to design, implement, and evaluate improvements to the software of an autonomous lunar mining rover developed by the Razorbotz team at the University of Arkansas. We implemented new autonomy features to the existing code and established a new foundation of documentation and refactoring to facilitate continued robot development. Ultimately, the goal of this project is to win the NASA Robotics Mining Competition (RMC) in May 2022 using the updated robot and control systems. In the future, lunar mining robots could be used to extract minerals on the moon, reducing the costs of such materials on Earth while eliminating the environmental impacts from their extraction.

1.0 Problem

NASA's Artemis program has goals including putting the first woman and next man on the moon by 2024, exploring more of the lunar surface than ever before, and later sending astronauts to Mars. This mission will require an incredible quantity of resources and infrastructure to be present in the lunar environment to house astronauts and provide a base for lunar operations. A 2019 estimate of the commercial cost of transporting payloads to the moon is about $1.2 Million per kilogram [1]. Given the many resource needs of humans and the extreme cost of sending resources into space, improving the technologies that allow us to harvest resources for the production of fuel, water, and/or oxygen from local materials while on space missions will be an important step towards enabling sustainable lunar surface operations while decreasing supply needs from Earth. Without the ability to gather resources while in space, extended space exploration will be infeasible due to the incredible costs of supplying resources from Earth.

Rovers will be a valuable tool to future astronauts, especially if they can autonomously gather useful local materials. However, autonomy presents its problem. Rovers are complex machines that need to operate in extreme environments continuously and reliably. The Artemis Student Challenge (this project) challenges college-level teams to develop a lunar excavator prototype capable of excavating lunar regolith to extract local resources so that those resources need not be transported from Earth. The challenge requires our rover to be capable of teleoperation as well as various levels of autonomy, and it strongly rewards the development of lighter rovers. The rover will be faced with specific challenges from the abrasive nature of the regolith and icy-regolith simulants it must mine, weight and size limitations for the rover itself, and the ability to navigate and operate remotely and autonomously.

Our Capstone team was part of the Computer Systems sub-team of the U of A’s “Razorbotz” team dedicated to this challenge. Our specific tasks were to implement all the software systems required by the rover, including controls, operator interfaces, sensor reading, computer vision, and autonomy.

2.0 Objective

The objective of this project was to design a strong, sturdy, fully modular robot that can maneuver on the arena terrain, collect the maximum amount of icy regolith within the time limit, and make two collection trips autonomously using ROS2 (Robot Operating System) and several types of sensors. Each sub-team has its separate objective which combined, gave the overall objective of the project. Our team was part of the Computer Systems sub-team and was responsible for executing the objectives of autonomy and control via software development. Other objectives included gaining experience with efficient hardware design and software-architecture planning, learning the processes involved in working with a multidisciplinary team, and obtaining valuable engineering skills for future careers. If our project is competitive enough by the 2022 competition, it may win the competition and benefit NASA by providing innovative robotic and excavations concepts that may inspire future ideas and solutions.

3.0 Background

3.1 Key Concepts

We used a wide variety of technologies and key concepts to build the computer systems for the robot. This year's Computer Systems sub-team’s main goals were to refactor old code and add further autonomy to the robot’s functionality with the help of computer vision.

The robot uses computer vision, specifically an object recognition neural network, to seek out objective locations, identify key targets, and perform path generation within the competition arena. Python, ZED SDK, and Darknet were technologies used to develop the computer vision model needed for the robot [2]. We chose YOLO to be our object detection neural network framework. The computer vision model is run on the Nvidia Jetson Nano board, a small microcomputer that is designed for artificial intelligence models in low-power environments. The Nvidia Jetson Nano serves as the onboard computer to control the robot.

We mounted a ZED camera on the robot which allows for visual sensor input to be passed to the computer vision model via the ZED SDK. This input is passed to the computer vision model built using Darknet. The model is application-specific but was built using transfer learning since the model was originally based on a pre-trained network trained on a large, diverse image dataset. We built a new dataset using the ZED image sensor data in the robot testing environment to retrain the initial model into the model that will be used by the robot. The model decides which functions the robot will need to perform automatically.

All robot functionalities were built using ROS2 (Robot Operating System). ROS2 supports development in C++ and Python. This year’s Computer Systems sub-team had experience with a wide variety of programming languages, including Python, C, C++, and Java, so the paradigms from these languages were helpful for programming ROS2 modules.

Additionally, background knowledge of embedded systems programming was crucial for interfacing with the robot’s hardware and sensors. Specifically, the ROS2 software controlling the robot can be interpreted as a real-time operating system (RTOS) that must accomplish desired goals within time limits. While there was existing code for manually controlling the robot’s functionalities, this year’s Computer Systems sub-team rebuilt the interface for operating the robot manually.

The former code for interfacing with the robot was built primarily by mechanical engineering students, so the usability, functionality, and readability of their code were limited due to their programming knowledge. This year’s sub-team contained more students in computer-related fields, leading to the adoption of proper computer interface development practices and techniques. These techniques were introduced for the manual interface redesign to produce more intuitive controls. This includes using Git for version control. While many new functionalities and standards were implemented by this year’s Computer Systems sub-team, the team focused on learning and using computer vision, the ROS2 RTOS, and modern software development techniques.

3.2 Related Work

Teams from previous years have built computer systems focusing on navigation, internal systems, controls, and manual operation with feedback [4, 5, 6]. This year’s computer systems team started the restructuring of a large portion of the existing codebase and implemented good coding practices. We developed well-documented code, unit testing, and used Git as a version control manager. Additionally, our code improved readability so future teams could continue to use the codebase we designed. Another key goal was to implement autonomy of the robot’s functions with the help of computer vision. In prior years, the competition did not have as much of an emphasis on the robot’s autonomy with its tasks [7, 8]. For this year’s competition, several points were allocated to the autonomous functionality of the robot [1], so autonomy was a primary focus when developing the new computer system’s codebase. Since this year’s computer systems team had an abundance of students from computer-related fields, the robot’s computer systems saw several improvements compared to previous years concerning programming standards, software stability, and robot autonomy.

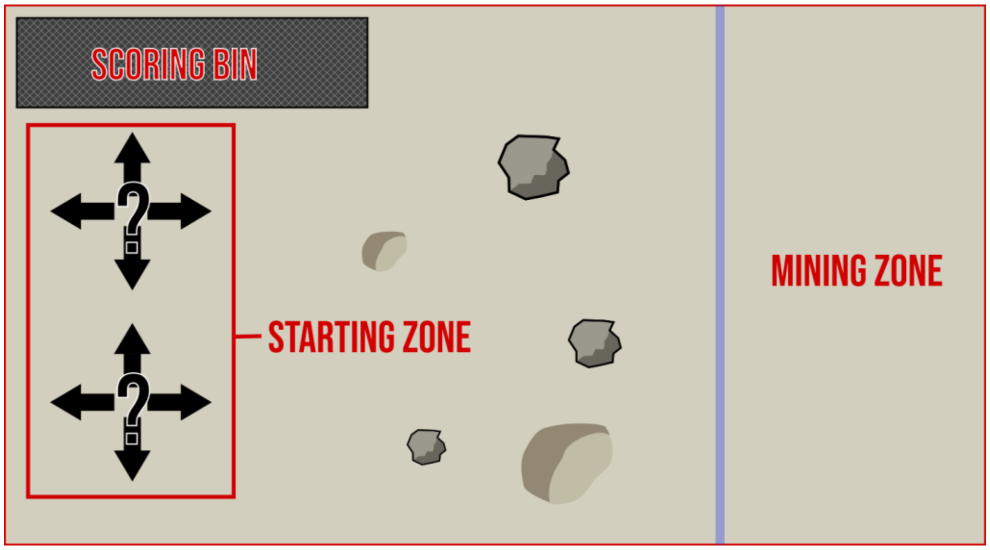
4.0 Design

4.1 Requirements and/or Use Cases and/or Design Goals

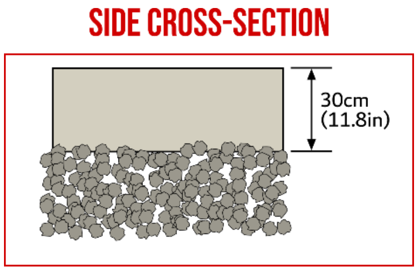
- Design must satisfy volume specifications set by the NASA rubric
- Design must satisfy volume specifications of .5m x .5m x 1m, as set by the NASA rubric
- Prove team can communicate wirelessly with the robot by March
- Autonomous navigation of the arena
- Autonomous excavation of the regolith

4.2 High-Level Architecture

The design of the robot for this year was heavily modified from the previous years. In years past, the robot relied on a spinning drum to remove the soil simulant called BP-1 from the gravel beneath and used a shovel to scoop the gravel into a collection bin. Both the soil composition and the arena layout are shown in the figures below. This year, due to changes in the size requirements, the robot uses an auger to drill into the soil and collect the gravel. The significant changes in the design have brought equally significant changes in the hardware and software needed to run the robot. The robot uses four motors to run the drivetrain and four ODrive motors to run the excavation assembly.



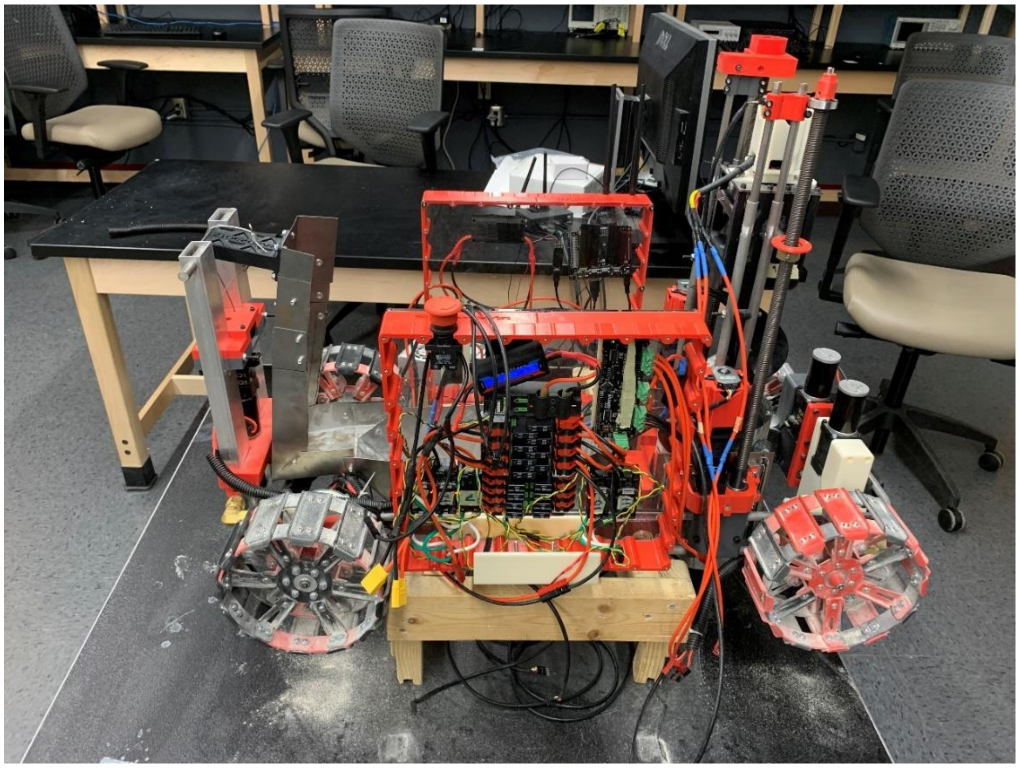
NASA Competition Arena



Side View of Arena Regolith Composition

The software needed to run the robot is broken into four main systems: Navigation, Movement, Excavation, and Autonomy. The navigation system is responsible for using the camera inputs to plot a path around obstacles and keep track of the position of the robot. The movement module is made of the drivetrain motors and the various sensors attached to keep track of the movement of the wheels and the robot. The excavation system consists of the ODrive motors, the attached encoders, and the other sensors attached to ensure the operation of the excavation assembly is working smoothly.

The final and most important system is the autonomy module. This module is responsible for all aspects of autonomous operation, including using the other three systems to read in the information necessary and to take the actions required to achieve the goals laid out by the system. It uses a combination of sensor inputs, graphics processing, and logic to run the course fully autonomously. The robot is programmed with the Robotic Operating System 2 (ROS2). This operating system allowed the the team to write individual programs, known as nodes in ROS2, in many different languages and have them all interact without errors if the message passing is compatible. ROS2 was used to in replace its predecessor, ROS1, because of Windows support.



The rover as of 04/27/2021

4.3 Risks

Risk	**Mitigation**
Autonomy system fails during competition	Trained on a diverse set of rock images, attempted to mimic the competition rock pit and rock samples as accurately as possible, used a neural network model that is robust to varied lighting and environmental conditions
Glitches from porting ROS1 code to ROS2	Documented all changes, kept code compartmentalized to make debugging easier
M.E. students may not understand our changes because of limited computer science knowledge	Wrote comprehensive documentation that explains concepts and functions well enough that someone who has only taken Programming Foundations I (or equivalent) could understand them
Setting the M.E. team back by losing old code	Backed up the old code before making changes, used GitHub for effective version control
Loss of key functionality from porting ROS1 to ROS2	Kept a checklist of functionalities to implement in the refactored code, checked items off as they are ported and updated, and only removed functionalities that were deemed no longer applicable

4.4 Tasks

* = tasks left to future RMC members

Goal	Steps taken
Refactor and upgrade old code	1. Upgraded ROS1 code to ROS2 • Exclusively used ROS2 going forward 2. Trimmed current modules • Studied code to understand functionality • Removed code that was not applicable to the current rover • Discussed these changes with design teams 3. Created a Docker container for off-robot development • Contains ROS2, CUDA, and other packages
Create and update documentation	1. Created a GitHub wiki detailing each of the ROS2 nodes implemented 2. Created auxiliary documents that discuss how to use development tools • Setting up Docker container • Building ROS2 modules
Improve manual control and user interface	1. Updated UI to be compatible with current robot 2. *Create a more user-friendly UI design
Implement excavation autonomy (identifying and digging rocks from the surface)	1. Set up YOLO [3] object detection neural network • Implemented YOLO on the rover computer using Darknet and ROS2 • Evaluated object detection performance on labels from the COCO [10] object dataset 2. Built a dataset of rock images from the test pit • Annotated bounding boxes around rocks • *Train YOLO to detect lunar rocks 3. Set up communication between YOLO and the ZED camera using a custom communication node 4. Automatic control of the excavation arm
Implement dump autonomy (depositing payload at base)	1. Extend YOLO to detect the mining base 2. *Create a pathfinding algorithm to estimate the shortest/fastest path to the base
Implement travel autonomy (movement around the test environment)	1. *Use YOLO to detect lunar rocks and hazards 2. *Create a pathfinding algorithm to estimate the optimal path (fastest, most efficient) for mining a series of rocks
Implement failure management	1. *Detect component failures 2. *Implement backup features for detectable losses
Win NASA RMC Competition	1. *Travel to Florida and win

4.5 Schedule

Tasks	Date
GUI/CLI Design Started	9/1/2020
GUI Design Review	9/26/2020
GUI Programming/Code Rewrite Started	10/1/2020
Start Manual Controls	10/1/2020
Preliminary Design Review	10/10/2020
Finish GUI/CLI Design Overhaul	12/18/2020
Revisit Scope of Project	1/16/2020
Start Excavation Macro	1/23/2021
Start Camera Implementation	1/23/2021
Start Sensor Implementation	1/23/2021
Finish Manual Controls Overhaul	2/6/2021
Begin Full Scale Manual Control Testing	3/17/2021
Start Training AI and Dataset Creation	2/20/2021
Complete Target Localization with ZED camera	3/6/2021
Begin Testing Camera and Sensors While Driving	4/30/2021*
Finish Excavation Macro	5/8/2021*
~~Begin Full Scale Autonomous Testing~~	~~4/10/2021~~
~~Finish Final Testing of Autonomy~~	~~4/24/2021~~
Graduate	5/7/2021
~~Compete at NASA~~	~~5/18-23/2021~~

* future/in-progress work

4.6 Deliverables

- Design Document: We submitted a report of the software systems to the Project Manager. The design document included a diagram of how the ROS2 nodes are connected, a description of the computer vision system used for navigation autonomy, and additional details about the software that controls the robot.
- ROS2 Nodes: ROS2 nodes for autonomy, excavation, navigation, and movement were designed and implemented. These nodes are made up of many subfiles and comprises most of the codebase of the project.
- Documentation: The previous code had almost no documentation explaining functions, subroutines, and variables. We refactored and updated old code with good programming

practices by adding comments in the code and creating a GitHub wiki explaining the utility of each function and class. Documentation was crafted for all new code and functionality added to the rover.

- Robot Testing Data: We built a dataset of images identifying rocks and obstacles. The neural network to analyze these images is completed for when the robot is in a finalized state and ready to be tested in the off-campus testing pit. Results of the robot's performance will be documented in the future as a deliverable.
- Final Report: This final report explain the steps we took to complete the project, challenges and setbacks faced, tasks that were altered from our Project Proposal, and the results. This report is supplemented by all the code written for the project including both the ROS2 nodes and computer vision system.
- Project Website: The project website is updated to provide all information on the project. It hosts the results of the project, a link to the code repository, the final report, and additional relevant material. This website is hosted on the capstone.csce.uark.edu website.

In addition, we intended to seek the 2021 RMC Competition prize. However, the competition was canceled due to COVID-19 considerations. Information on the future of our competition engagement is presented in Section 5.0.

5.0 Future Work

The goal of full assembly and operability of the Lunar Rover by May 2021 was delayed due to design and construction delays that pushed the timeline for the implementation of key planned features beyond the end of the Spring 2021 semester. Because of the lack of a testing platform to ensure the autonomous features of the robot were properly working, many of the original functions proposed were unfortunately removed from the scope of the project for this year. However, frameworks and plans are in place to greatly increase the autonomous functionality of the robot for the competition next year. A future Capstone group would be in an excellent position to implement these features because of our foundational development.

The computer systems team this year set up key features that should allow future teams to finish implementing the automation of the excavation assembly by training the neural network to recognize the rocks and holes. The neural network information will then also be used to run pathfinding through the arena. Finally, when the automation portion is completed, we will modify the Graphic User Interface (GUI) to dynamically display information from the robot. These deliverables would be finished, but the excavation assembly was completed nearly eight months late, which prevented the computer systems sub-team from performing physical testing during this school year. However, the neural network is prepared and will be trained with the rock dataset we constructed. This neural network will run on the Jetson to allow the robot to recognize the obstacles to avoid.

The current implementation of the robot struggles to use the neural network due to power supply issues and the overcurrent throttling of the ZED camera. This makes accurate data difficult to obtain while the robot is running. These issues will be addressed by redesigning the power supply and possibly upgrading to a more powerful processor to allow for more accurate real-time image analysis.

Additionally, the GUI to control the robot manually is currently hardcoded to refer to the current implementation of the robot. The team is working to design a message protocol that would allow the robot to send an initial message to the controller that would cause the client-side GUI to dynamically create the information panels that are needed to display the information from the motors on the robot. Development of the robot is currently on-going. Assuming there are no other setbacks preventing the other sub-teams from building the robot, the robot, and its computer systems, should be ready for competition in May 2022.

6.0 Key Personnel

William Burroughs – Burroughs was a senior Computer Science major in the Computer Science and Computer Engineering Department at the University of Arkansas. He has been on the team for 1 ½ years and was the Controls sub-team lead in 2020. While a member of the Capstone group, he was the Computer Science sub-team lead for the RMC project. For the Capstone group, he oversaw the creation and testing of the excavation node, GUI overhaul, and project deliverables.

Calvin Franz – Franz was a senior Computer Science major in the Computer Science and Computer Engineering Department at the University of Arkansas. He has experience in writing software for vehicle systems in the F-16 fighter plane and has worked on projects featuring Robotics, AI, and Software Development. This was his first year on the team, and he used his software engineering experience in vehicle systems to help with the implementation of ROS2 nodes and write descriptive multi-level documentation for the project.

Zachary "Gunner" Lawless – Lawless was a senior Computer Science and Computer Engineering major in the Computer Science and Computer Engineering Department at the University of Arkansas. This was his first year on the team; however, his previous research experiences attacking vulnerabilities in deep neural networks and working with embedded systems should help contribute to building the robot's computer systems. Lawless was responsible for contributions in developing the dataset that will be used to train the computer vision model, documentation, and incorporating Docker into the tech stack.

Jett McCullough – McCullough was a senior Computer Science major in the Computer Science and Computer Engineering Department at the University of Arkansas. This was his first year on the team. He has experience with working on diverse project teams and foundational knowledge in AI, Big Data, and Software Development. He was responsible for interpreting ROS nodes and writing documentation.

Carson Molder – Molder was a senior Computer Engineering major in the Computer Science and Computer Engineering Department at the University of Arkansas. He had experience in artificial intelligence, machine learning, and computer systems. He performed computer vision research with Professor Justin Zhan. For this project, Molder developed the robot's computer vision system.

Alex Westbrook – Westbrook was a senior Mechanical Engineering major in the Mechanical Engineering Department at the University of Arkansas. He was a fourth-year team member and was the project manager this year. He oversaw all sub-teams and oversaw the overall schedule for the project.

Dr. Uche Wejinya– Dr. Wejinya is an Associate Professor in the Department of Mechanical Engineering at the University of Arkansas. He performs research in robotics and mechatronics and is the faculty advisor for Arkansas Razorbotz.

7.0 Facilities and Equipment

7.1 Facilities

- Mechanical Engineering Robotics Lab: Laboratory in the Mechanical Engineering building where the robot is assembled, and the software is tested
- Campus Test Site: Shipping container with a simulated lunar environment to test designs

7.2 Equipment

- Talon SRX Motor Controller: Uses CAN protocols to relay position information of motors to the Jetson
- Victor SPX Motor Controller: Similar functionality to Talon
- NVIDIA Jetson Nano Developer Kit: Run ROS nodes, includes neural network hardware
- Vex BAG Motors: Motors used to power each wheel
- ODrive Boards: Motor controller boards used to interface between Jetson and ODrive motors
- ODrive Motors: Used to control excavation assembly
- ODrive Encoders: Tracks positions of excavation motors, allowing for more precise control
- Zed Stereo Camera: Used for depth perception and navigation; images are passed through to the neural network to predict where rocks and obstacles are
- Power Distribution Panel: Used to distribute and track power consumption of various components
- Logitech Extreme 3D Pro Joystick: Used to manually control the robot
- Arduino Modules
- NVIDIA RTX 2080 Ti: Used to train and simulate computer vision models for the rover off-device

8.0 References

[1] “NASA Robotic Mining Competition (RMC) Lunabotics 2021, Registration, Rules and Rubrics,” NASA, 2020, url: https://www.nasa.gov/sites/default/files/atoms/files/000_rmc_lunabotics_rules_rubrics_2021.pdf

[2] "PyTorch: An Imperative Style, High-Performance Deep Learning Library," Adam Paszke, Sam Gross, Franciso Massa, et.al., 2019, url: https://papers.nips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf

[3] "You Only Look Once: Unified, Real-Time Object Detection," Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi, arXiv, 2016, url: https://arxiv.org/abs/1506.02640

[4] "Razorbotz CS Intro," William Burroughs, 2020, url: https://www.youtube.com/watch?v=BSq90vnPwJ4&feature=youtu.be

[5] "2020 Systems Engineering Report," Razorbots Team 2019-2020, University of Arkansas, 2020.

[6] "Razorbotz/RMC-Code-19-20," Razorbotz Team 2019-2020. 2019-20, url: https://github.com/Razorbotz/RMC-Code-19-20

[7] "RMC 2019 Registration, Rules and Rubrics," NASA, 2018, url: https://www.nasa.gov/sites/default/files/atoms/files/rmc2019_registrationrulesrubrics_09062018.pdf

[8] "2020 Systems Engineering Report," Razorbotz Team 2019-2020, University of Arkansas, 2020.

[9] "Jetson Nano: Deep Learning Inference Benchmarks," Nvidia Developer, n.d, url: https://developer.nvidia.com/embedded/jetson-nano-dl-inference-benchmarks

[10] "Microsoft COCO: Common Objects in Context," Lin, T., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollar, P., and Zitnick, C.L. European Conference on Computer Vision. 2014, pp. 740-755.