Miniature Mars Rover

DESIGN DOCUMENT

Team Number: sdmay19-09

Client: Dr. Zambreno
Adviser: Craig Rupp

Team Members/Roles:

Calvin McBride:	Hardware/Software Developer, Communication, Status Reporter, Meeting Note Taker
Sam Westerlund:	Software Developer, Communication
Mitchell Freshour:	Electrical Lead
Douglas Kihlken:	Hardware Developer

Team Email: sdmay19-09@iastate.edu
Team Website: sdmay19-09.sd.ece.iastate.edu

Revised: 2018-12-2 Version 3

Table of Contents

1 Introduction

1.1 Acknowledgement

This project would not be possible without the support of the Iowa State University ECpE Department, our advisor Craig Rupp, and our client Dr. Joseph Zambreno. The funding, advice, and motivation they provide us has made it possible for us to build and extend the JPL Open-Source Rover, a demanding yet rewarding project. It is with their feedback and support that we hope future students and teams may benefit from our work on the miniature Mars Rover.

1.2 Problem and Project Statement

In today's evermore complex and demanding job market, potential students and interest groups must make hard decisions on what to study and where to get an education that will help them fulfill their dreams. For students looking to delve into engineering and related disciples, it can be especially difficult to see all of the potential benefits and freedom that an engineering degree can provide them.

To help spur interest in Iowa State University's engineering programs and catch the interest of prospective students, ISU's ECpE department has devised several display pieces that demonstrate the engineering prowess their senior students wield. One of these projects is the Mars Rover project, a miniature 6-wheeled mars rover model that requires knowledge of software, mechanical, and electrical engineering to construct.

Our team's primary goal is to get the rover (based on JPL's Open-Source Rover project) up and running, with secondary but preferred goals being the use of computer-visual and machine-learning to navigate, classify its environment, and possibly handle objects. It is our hope that these features, along with the ability to remote-control the rover, will help catch the interest of visiting prospective students, as well as give future senior design teams a well-built platform to extend for their own projects.

1.3 Operational Environment

The expected operational environment that the miniature rover will operate in is inside the buildings of the Iowa State University campus. These buildings have hard, smooth floors, clean (non-dusty) air, and a reasonable amount of room to maneuver. While we expect the rover to mostly operate indoors, the JPL Open-Source Rover is capable of operating outside in nice weather. Because of this, the the rover we designed will be able to operate outside, as long as the weather is acceptable with no rain or snow, and the operator keeps the rover on relatively flat terrain.

The Mars Rover will most likely have a small crowd around it, and because of so safety is key. We want to demonstrate to the public a cool piece of machinery and software in a exciting but safe

manner. We have developed several safety methods and failure points to ensure that the Mars Rover operates safely within its environment.

1.4 Intended Users and Uses

Our end-product is expected to have two different groups of users: student/faculty drivers and senior design teams.

The first group consists of potential students (mostly young tech savvy adults), interest groups, and faculty who will demonstrate the rover's features. This group will utilize the rover to see first-hand what can be built and what has been built by Iowa State University engineering students. Code will not be shown to this group, only the product of the code. They will also have access to view the image classification and manual control of the rover. The control and data view is a simple intuitive web interface, with instruction provided to not confuse or hamper the experience of the user. Overall, we want to impress this group with a well rounded finished product.

The second group are senior engineering students taking Senior Design who are tasked with extending the rover's functionality for their own senior design project. This second group is expected to have technical knowledge, contrary to the first group who are expected to have minimal technical experience. This group will essentially start where we left off, and to ensure a fast transition, well written documentation will be provided.

With these groups in mind, the rover will implement a range of both manual and autonomous navigation in order to "wow" members of the first group, while keeping extensibility and flexibility in mind so that members of the second group may easily build upon the rover.

1.5 Assumptions and Limitations

Assumptions:

- The rover will primarily be operated indoors, in a well lit area.
- The rover will be remote-controllable, with users able to view what the rover sees through its camera.
- The rover will utilize computer-vision and machine-learning to navigate when not manually controlled.
- The rover will have six independently-powered wheels.
- The rover will be based off the JPL Open-Source Rover.
- The rover will be charged by the operator between operations.
- The rover will have the processing power to run the neural network and its subprocesses.

Limitations:

- The JPL Open-Source Rover base-build has a battery-life of approximately 4 hours, without additional modifications. Expanding the power source is currently an unknown due to the amount of free space inside the rover body.
- The rover uses a Raspberry Pi 3, which has a limited number of GPIO pins that are necessary for extending further hardware add-ons.

- The project has a budget of $2,600.
- The JPL Open-Source Rover documentation estimates a minimum of 200 man-hours by experienced teams to build the base rover.
- Training a new neural network may not be possible, and also data-collection and training from the mars rover's camera is too time consuming of a task.
- The Intel Compute Stick has limited processing power, a smaller neural network must be used, sacrificing performance.
- Connectivity between the rover and the operator's web-browser may suffer connectivity issues, video streaming quality may be an issue. A certain amount of users can be connected at once.

1.6 Expected End Product and Deliverables

The final end-product will be a working mars rover based on the JPL Open-Source Rover project. This rover will utilize computer-vision and machine-learning to move around in its environment, as well as remote-control capabilities so that users may operate it manually. The end-user will be given equipment to properly operate the rover and maintain it (eg. charging). The rover is expected to work "right out of the box" with the end-user simply connecting to the control panel interface over WiFi on their phone or laptop.

The rover will be well-documented and have many GPIO pins so that future teams can work with and extend the rover design. Additional documentation on how to operate the rover, design choices, and a technical description will be provided. The rover, its documentation, and its source code will be delivered to the client at the end of the Spring 2019 semester, around April/May.

2. Specifications and Analysis

2.1 Proposed Design

We decided to use the JPL Open-Source Rover as the starting point for our miniature rover. Other robot chassis were considered, but JPL's rover was well-documented, open-source, and thoroughly tested, making it ideal for our team. Our rover will most-likely use the Intel Movidius Compute Stick as a small but functional VPU, upon which many computer-vision and machine-learning tasks will be executed. Our team is currently looking into camera systems that will satisfy our computer-vision needs, as well as other sensors such as LIDAR being considered.

The rover hardware/parts as specified in the JPL Open-Source Rove parts-list have been ordered, and its construction will begin once they are received.

Functional Requirements:

- Raspberry Pi takes input from an external web interface, controls the hardware, and processes hardware data.
- The rover will stream camera visuals as well as current labels of objects in view to the user.
- Working camera for computer-vision to self-navigate.

- Navigation algorithm to move around detected items.
- Working sensors (such as LIDAR, Sonar, Infrared, Camera) to detect and map it's surroundings, as well as navigate .
- Implementation of a neural network to detect common indoor objects and people.
- The Mars Rover must be autonomously capable of going from its current position to a destination selected by the end user. It must reach the destination in a reasonable amount of time compared to manually driving to the destination.
- The end user must be able to manually control the Mars Rover. To manually control the rover the user will connect to the rover in their browser at the same page of the camera input. There will be simple buttons to click (forwards, backwards, left, right) that will move the rover in that direction with latency of less than 1 second.
- The end user will select a destination position on the same website as the other controls. The position will be displayed to the user in the form of a map and they can simply click a waypoint.

Non-Functional Requirements:

- Battery life must have a minimum of 2 hours.
- Rover should take no more than 1 second to respond to manual user input.
- Rover should take an average of 5 seconds to scan, process, and create a plan for navigating its environment, with a worst-case time of 10 seconds.
- Remote-control signals should reach several meters, ideally throughout most of the a given building.
- Safety features must be in place (eg. stopping auto-navigation when operator disconnects) to prevent the rover running away or colliding with people or objects.
- Python will be used as the programming language to control the hardware. React will be used as the web interface.
- The rover will be able to handle inclines at 15 degrees, and uneven terrain where the disparity from the left to right wheels is less than 1 inches.

2.2 Design Analysis

The JPL Open-Source Rover project has been tested and verified by a much more technically knowledgeable and experienced team of engineers, and we feel an in-depth analysis of the base rover design is not necessary beyond the testing laid out by the JPL in their testing and calibration documents included in their documentation. This includes testing for the printed circuit boards, calibration of the motor controller units, as well as testing for connecting and operating the rover using the provided software.

For the components we will personally be developing and extending the rover with (computer-vision, machine-learning, etc.), in-depth testing and analysis will have to wait until hardware is received and much of the base rover is constructed. However, we have compiled what we believe are accurate strengths and weaknesses in our proposed design:

Strengths:

- Well-tested and thoroughly vetted rover base/starting point.

- Relatively cheap and easy to understand.
- Several sensors for accurate and thorough environmental analysis by rover.
- Raspberry Pi is well-documented, making tasks like computer-vision and machine-learning easy to start and develop with.

Weaknesses:

- Many electrical components are not vacuum-sealed, making them susceptible to bad weather and water.
- Rover construction involves many construction and fabrication techniques that our group is unfamiliar with.
- Computer-vision is a relatively young field in computer science, making it a somewhat difficult task to implement for our junior team.
- Computer-vision and machine-learning is a relatively high cost in terms of processing power, making the rover's 3-4-hour battery life even shorter.
- Using the Intel Compute Stick sacrifices accuracy, and we are not training the neural network model with our own data.

3 Testing and Implementation

3.1 Interface Specifications

The Raspberry Pi will communicate with several hardware entities, such as the camera, LIDAR sensor, wheel motors, and onboard GPU stick. In addition to this, the Raspberry Pi will have to connect to a web interface controller and stream video to the user wirelessly.

Testing these interfaces will involve observing that the hardware performs actions as they are instructed (cameras turn when instructed, wheels rotate and turn at specific speeds, etc.) and manually confirming that data being received from sensors is correct (ensuring the camera is working and streaming video, manually outputting data points from sensors, verifying the rover moves when remote-controller is used, etc.). Other means of verifying the accuracy of sensors and commands will be devised once hardware is received and the base rover construction has begun.

3.2 Hardware and software

Testing our design will utilize several hardware and software testing suites. Hardware testing will most likely be done with equipment located in Coover Hall's Electrical Engineering labs, and software testing will be done on university workstations or personal computers using software provided by the university or that is free/open-source. Hardware tools will include multimeters and oscilloscopes, while software tools will include IDEs and debugging software.

Testing the accuracy of the neural network model will be done via real world testing. Since we are not training the model, it will not be given it validation data to test against, only a test set or test scenario will be used. Accuracy will be measured based on the rovers actions in the real world.

Testing of electrical hardware will be done in accordance with the provided documents by the JPL. One document covers testing for the printed circuit boards required for the rover's operation, to ensure that voltage is properly . The other document covers calibration and testing of the motor controller units to ensure they are providing the motors with proper voltage in order to move, as well as transmitting data about the drive and corner motor's positions to be used by the software. Both will be used to ensure the functionality of the motors, and the proper delivery of power and data between the motors, the boards, and the Raspberry Pi.

3.3 Process

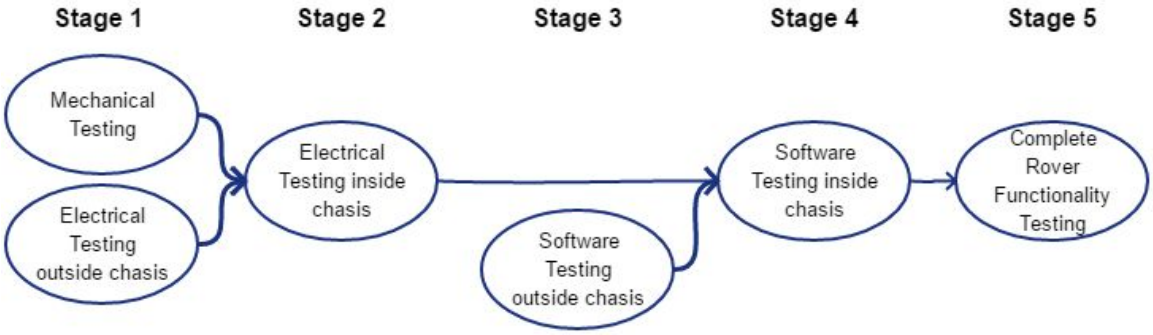
The first phase of testing we will do will be simple mechanical testing on the rover body. This will be done by setting various weights on the body to ensure it does not break, moving body parts to ensure they do not break off and have the expected range of motion, and pushing the body around to ensure the wheels and joints are operating optimally.

The second phase of testing will involve the electrical components after integrating it into the body. This will mostly involve insuring the various electrical components are getting power and turn on. Other testing (such as ensuring correct data transmission power levels) will wait until after software has been integrated.

The third stage of testing will involve static testing and analysis of software using an IDE on a computer, independent of physical hardware. This is the most basic form of software testing and is functionally agnostic (i.e. ensures that all software components "fit" together properly, not necessarily that they function correctly).

The fourth stage of testing will involve integrating the software into the rover. In contrast to statically analyzing and testing the software as in stage three, this step involves ensuring the behavior of the software correctly models how we want the hardware to run. Tests in this stage will involve reading hardware data, sending hardware commands from the Raspberry Pi, data processing, and other tests to ensure the hardware and software are properly interfacing.

Final testing will involve miscellaneous tests that weed out extreme or unlikely behavior, as well as ensuring that extending the rover can be done in a relatively simple manner.



4 Closing Material

4.1 Conclusion

Our project goal is to have a working miniature rover based on the NASA JPL Open-Source Rover that demonstrates the skills and technical knowledge wielded by Iowa State University senior engineering students in order to impress and draw in prospective students for the ECpE department. With the JPL Open-Source Rover as a base, we plan to extend it with computer-vision and machine-learning to enable both manual and autonomous navigation, as well as ensure that future senior design teams can work on and extend our rover. Our solution demonstrates the utilization of two fields of computer science that are becoming increasingly popular, as well as gives future engineering teams plenty of room and flexibility when building upon our design.

4.2 References

[1] Github (2018, Sept.). NASA JPL Open-Source Rover [Online]. Available: https://github.com/nasa-jpl/open-source-rover [Accessed Oct. 13, 2018]