**Boston University**

**Electrical & Computer Engineering**

**EC464 Senior Design Project**

Final Report

Mars Rover: Autonomous Navigation

Submitted to

Osama Alshaykh

8 St.Mary’s Street

Room 424

858-361-9043

[osama@bu.edu](mailto:osama@bu.edu)



by

Team #28

Autonomous Navigation

Team Members

Daniel Crawley crawley@bu.edu

Brian He brianhe@bu.edu

Tommy Lam tlam11@bu.edu

SeungYeun Lee sylee538@bu.edu

Linden Vo lindenvo@bu.edu

#### Table of Contents

[**Table of Contents**](#_heading=h.gaspa1z7x9w1) **2**

[**Executive Summary**](#_heading=h.bk9hi692o011) **3**

[**Introduction**](#_heading=h.1fob9te) **4**

[**System Technical Description**](#_heading=h.pok8alpwqii7) **5**

[**Second Semester Progress**](#_heading=h.3dy6vkm) **7**

[**Technical Plan**](#_heading=h.1t3h5sf) **8**

[**Cost Breakdown**](#_heading=h.4d34og8) **9**

[**Requirements**](#_heading=h.2s8eyo1) **10**

[**Appendix 1: Gantt Chart**](#_heading=h.3rdcrjn) **13**

[**Appendix 2: Hardware Diagram**](#_heading=h.y4jp1ngo7lxz) **14**

[**Appendix 3: ROS Node Graph**](#_heading=h.isu465ljkzz6) **15**

[**Appendix 4: Test Result**](#_heading=h.26in1rg) **16**

[**Appendix 5 : Team Information**](#_heading=h.i7nlym5cv5vw) **18**

# 

# Executive Summary

Mars Rover Autonomous Navigation

Team 28

Our goal for this project is to design, build, integrate, and test a rover capable of autonomous movement to a specified AprilTag with the use of GPS, visual and inertial navigation. We have been following guidelines outlined by the University Mars Rover Challenge. The system is a prototype meant for use on Earth, but can be designed to be used on Mars. While adhering to size, weight, power, budget, and environmental constraints, the design of the rover used an Intel RealSense T265 tracking camera and D415 depth camera to generate a map of its surroundings, detect obstacles and simultaneously navigate around them to acquire the AprilTag target. The entire software stack used ROS nodes with different topics subscribing and publishing to each other. While much of the final prototype testing was in progress, the team accomplished many of its goals in the final semester and left the project in a state where another group would be able to come in, replicate the results, and complete the task of autonomous navigation.

# Introduction

Autonomous navigation is no easy task, but when done right, could tremendously benefit many industries. The Autonomous Navigation Market is estimated at $2.2B in 2018 and projected to reach $13.5B by 2030. Societal impacts of this technology include the opportunity to service and explore unknown terrain, safely conduct search and rescue operations, autonomous agricultural farms and much more.

Even though our project was initially given by Mars Rover University Challenge, our focus has shifted from Mars Rover to Autonomous Navigation since autonomous navigation can benefit many industries. Our autonomous rover is not only useful to NASA but also industries such as traditional farming and food delivery services. For farms, autonomous robots could capture data on yield and health of crops at the level of individual plants and on a massive scale. For food delivery, instead of a person delivering the food item, it will be carried to the customers doorstep on four wheels. With this technology, both industries can see increased operational efficiency and reduced operating costs.

# System Technical Description

The project entails delivering a functional vehicle with a chassis and suspension and fully integrated sensors such as a tracking and depth camera system, a powerful computer and image processing unit, an inertial measurement unit (IMU), and a second microcontroller capable of driving motors and controlling the rover’s movement. A significant portion of the deliverables will be autonomous path planning and control algorithms and smart decision making in regard to obstacle detection and collision avoidance.

Overall, the team provides a fully integrated vehicle as a proof of concept that can autonomously traverse terrain and detect an AprilTag while using inertial and visual navigation to avoid obstacles.

**Hardware Overview (Appendix 2)**

The rover is a Traxxas 1/8th 4WD Electric Power R/C Rock Crawler with front and rear stepper motors, an electronic speed controller (ESC), a differential steering servo for the front axis, and a 7.2V, 2000mAh LiPo battery for powering the rover.

The main computer that performs image processing and makes smart decisions for the Mars rover is the Nvidia Jetson TX2. An Arduino Uno R3 acts as the microcontroller which drives the ESC and motors on the rover and interfaces with all other sensors for movement.

An Intel Realsense D415 Depth Camera and Intel RealSense T265 Tracking Camera are used in tandem to detect obstacles, perform simultaneous localization and mapping, and navigate towards an AprilTag target.

Our components all rely on battery power. The PowerAdd Pilot Pro2 23Ah/4.5A/5-20V Power Bank is a portable battery that powers the Jetson TX2 which requires 19V input as well as a 3.0 USB hub which powers the Tracking Camera. The TX2 powers the Depth Camera and the Arduino Uno R3 (via a PCIe to USB host card). The rover’s built-in 7.2V, 2000mAh LiPo battery powers the servo and ESC.

**Software Overview (Appendix 3)**

Our software stack consists of 4 layers. The first layer controls the motor which allows the vehicle to move and turn. The second layer enables and receives input from the cameras. The third layer implements simultaneous localization and mapping (SLAM) with the visual data from the cameras and generates an occupancy map of the vehicle’s surroundings. Lastly, the fourth layer implements the vehicle’s path pathing and navigation.

This is all accomplished through the Robot Operating System (ROS). ROS allows our devices to communicate with each other by using nodes which take in data from buffers (topics), processing the data, and then sending it to other topics. We have 4 nodes: the occupancy node, the AprilTag Node, the navigation node, and the serial node.

The occupancy node receives camera data as the input and generates an occupancy map. This map is then generated in Rviz which creates a 3D visualization. The AprilTag node detects whether an AprilTag is in sight of the cameras. It sends this data to the navigation node which makes movement decisions for the vehicles. Finally, the serial node converts the navigation commands into PWM signals for the ESC and servo motor.

# Second Semester Progress

Our project goals were advanced by leaps and bounds in the first few months of 2020. Starting off the year, some of the individual components of the rover were at the beginning stages of functionality. This was developed into a functional autonomous rover in time for the prototype demonstration. Such progress can be defined in three categories: making the rover mobile, redesigning the software stack, and integrating all components.

In order to make our rover mobile, we first had to tweak our chassis. Over winter break, we amended our issue with turning fully by adjusting the servo mount, and adding an L298N driver so that the servo could be powered by the drivetrain battery rather than the Arduino. We also built two chassis platforms on which we later mounted all of our electrical components. After winter break, we acquired a battery to power the Jetson TX2, and 3D printed mounts for our tracking and depth cameras. We completed our goal of going mobile by setting up an ssh server on the Jetson so we could start up the software remotely and mounting all the components.

For the second category, redesigning the software stack, we aimed to make our rover run purely on one ROS application. This was divided into reading camera data into ROS, producing an occupancy map for object avoidance, identifying and tracking april tags, using PID to make navigation decisions, and producing servo and motor commands from these decisions. Within our window we put together this entire software stack (cf Appendix 4). Much of the headaches were caused by adapting the ROS libraries for the Realsense cameras to work with the Jetson; and we thank Phillip Schmidt from Intel and Jim Benson from JetsonHacks.com for their guidance in this regard. Due to this time spent tweaking, we were able to build and view the occupancy map but were not able to use it as part of the navigation decision-making within our time window, thus leaving object-avoidance to be done by our successors.

Lastly, the topic of integrating components. In addition to merging software components, we also found that the Jetson’s default to using as little power as possible caused us to rethink some of the hardware design. We acquired a powered USB hub to connect our cameras to in order to avoid the mouse and keyboard shutting off when the cameras were powered on. We also circumvented the issue of limited USB bandwidth by purchasing a PCIe to USB host card and using it as the middleman between the Arduino and Jetson. These advancements were essential to achieving full hardware and software integration.

# Technical Plan

The performance period of this plan is not decided due to COVID-19. However, these tasks were planned to be finished by April 24th, 2020. Gantt Chart (aka. timeline) for each task is shown in Appendix 1.

Task 1. Optimize Navigation

*Currently, the rover is able to navigate autonomously and manually in a straight line towards an AprilTag once it is detected. When the AprilTag is slightly off the vision of the rover, it tries to navigate towards it; however, sometimes it fails to stop or goes in the wrong direction. In order to fix these problems, we have to make the rover navigate to the AprilTag more accurately by tuning the PID controller. In addition, in order to avoid a situation where the rover cannot detect where the AprilTag is, we have to build modules that control the rover movement to look for the AprilTag.*

Task 2. Finalize Obstacle Avoidance

*In order to navigate autonomously without running or bumping into obstacles, we need to finalize obstacle avoidance. This specific task has been causing a lot of problems running with Jetson TX2. Professor Tron has advised us to use the slam\_gmapping node package from ROS, which provides SLAM (Simultaneous Localization and Mapping) and is compatible with our ROS melodic system. In addition, having the move\_base package as part of the ROS navigation stack would try to navigate to the goal (i.e. the AprilTag) while avoiding obstacles.*

Task 3. GPS Integration

*One of the requirements is GPS integration. We have to have our rover move to a specific point by providing GPS coordinates or clicking on a map generated in RViz. In terms of hardware, this would require us to connect the Adafruit FONA 808 shield to our Arduino, and acquire the proper GPS antenna and battery for its operation (refer to the Cost Breakdown for specific hardware information). In terms of software, we would have to either find a ROS package which could interpret GPS data or otherwise modify the software stack such that the Arduino published to a GPS data topic and then have the navigation node subscribe to it.*

Task 4. Test Launch Outside and Debug

*In order to ensure that our rover can sustain rough terrain like Mars, we need to test our rover outside. We have been testing inside, and have been adjusting our rover to the inside environment to begin with. However, since our goal is to have the rover autonomously navigate in rough terrain, we should be testing outside and adjusting our rover to the outside environment. We anticipate that weather-proofing the rover would require building some housing for the electronics and modifying the settings on the cameras to reduce glare.*

# Cost Breakdown

|  |  |  |
| --- | --- | --- |
| **Item** | **Description** | **Cost** |
| 1 | Battery (for drivetrain) | $22.95 |
| 2 | Intel RealSense D415 Depth Camera\*\* | $149 |
| 3 | Arduino Uno R3 set\*\* | $35 |
| 4 | Adafruit FONA 808 Shield - Mini Cellular GSM + GPS for Arduino\*\* | $49.95 |
| 5 | Passive GPS Antenna uFL - 15mm x 15mm 1 dBi gain | $3.95 |
| 6 | Lithium Ion Polymer Battery - 3.7V 500mAh | $7.95 |
| 7 | NVIDIA Jetson TX2 Developer Kit (Education Discount) | $330.09 |
| 8 | Servo | $21.47 |
| 9 | Intel RealSense T265 Tracking Camera | $199 |
| 10 | USB to PCIe Host Card | $19.11 |
| 11 | USB 3.0 Powered Hub | $23.36 |
| 12 | PowerAdd Pilot Pro2 23Ah/4.5A/5-20V Power Bank | $82.99 |
| 13 | 5 yards of Strenco 2” Velcro Tape | $12.89 |
|  | Total Cost (before tax) | $957.71 |

\*\*Provided via Donation

A green background indicates that the item was not acquired

Items 9 to 13 were acquired this semester. Of these items, the ones that stand out the most are the T265 Tracking Camera, and the PowerAdd Power Bank. The Tracking camera, with its fisheye lense and built-in IMU, was used in conjunction with the Depth camera to produce our occupancy map and odometry data. The Power Bank was connected directly to the Jetson TX2, and provided a stable power source that fit within the 19V/4A requirement while also powering the USB hub.

Items 5 and 6 were designated as part of our GPS development plan . They consisted of the GPS antenna and small cellphone-like battery that would be connected to our Mini-Cellular shield. This plan was not implemented within our shortened time frame and so the items were not acquired.

# Requirements

The requirements of our project were initially defined by the [University Mars Rover Challenge](https://7aec5dcb-a-3f6a8980-s-sites.googlegroups.com/a/marssociety.org/urc/files/University%20Rover%20Challenge%20Rules%202020.pdf?attachauth=ANoY7cpLqkXuYQCD0uW64Q3HTPOaNsRQ05ab-AumhtX_F_jY3DKxqUJM-x4Eevf3YLeGNm7ehF8cQKKQOMAAetMp6pzvBHyMh7tyg5-fsObt9LNl5b00d6K5aWYfG4I5T_6Thz2nkoM3BvNwU9fTXOYXqDuqfkUqFQ2ezP3MInWR-0SnoFmCFno-WXadJWFOj2gOniuh-mOHi7fwkgxKzHJtl96PwQHzfYQHPCELA0y0VJM53lxDdy7BKkZoUYM0guwFLp7G7OOM&attredirects=0), sponsored by NASA. However, for the purposes of senior design, in order to reach a more attainable goal, the group worked with the client to define these requirements, which if needed, could be upscaled for the competition.

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Value, range, tolerance, units** | **Requirement met?** |
| Dimensions | 1.2m x 1.2m | Yes, the measured dimensions of our rover fit well within 1 meter in terms of length, 0.5 meter in width, and 0.4 meters in height. (Appendix 3: Figure1) |
| Weight | 50 kg | Yes, the hardware and chassis of the rover was well within our constraints, weighing no more than 10 kg. The rover was lightweight and portable enough to be lifted by one human hand. |
| Navigation towards AprilTag upon detection | Must move to within 1 meter of target | When the rover was directly in front of the AprilTag, the tracking camera would be able to detect it in the camera’s field of view to a high degree of accuracy, up to 10 meters away.(Appendix 3: Figure 3) As distance to the left or right of the AprilTag increased, the rover would fail to move within target range. The team attributed this error due to the PID gains being unstable enough to guide the rover during its navigation course. With more fine tuning of the PID, the team is confident that navigation towards an AprilTag will succeed 100% of the time. One of our main goals in the final stages of testing before the project was disrupted was to make our rover extremely robust. At all times during operation, as long as the AprilTag was in camera view, the system was able to obtain the coordinates of the AprilTag relative to the rover’s coordinates with minimal error. |
| Mapping | Generate a map of the environment | The team had used custom edited Intel Realsense occupancy mapping ROS modules to generate a map in RViz to some success (Appendix 3: Figure 2), however failed to integrate the map with the Arduino navigation node for obstacle avoidance. Before CDR, the decision was made to swap to slam-gmapping algorithm, which has a prebuilt ROS package where the node automatically publishes to the ROS navigation node called move-base. In tandem the team is able to use slam-gmapping and move-base to map the environment and avoid obstacles. |
| Navigation using Generated Map | Be able to click at any point on generated map and navigate rover towards target, avoiding obstacles simultaneously | Part of the task of generating a map in Rviz was to be able to click on the map and navigate rover to specified point, while also detecting obstacles with the camera and telling the navigation node to avoid obstacles by steering left or right. The team was in the process of integrating slam-gmapping with move-base to accomplish this task. The test plan detailed using hallways and open street areas with rocks and small obstacles placed in the rover’s field of view, potentially blocking the AprilTag target, and having the rover move forward and around obstacles in search of the AprilTag. Because AprilTags are very angle, shear, translation and motion invariant, then the rover would still be able to acquire the AprilTag and navigate towards it after avoiding obstacles. The location of the AprilTag target is known prior to testing. If needed, support for an unknown AprilTag location could be implemented in the form of sweeping cameras on a servo motor swiveling to search for and acquire the target. |
| Obstacle Detection and Avoidance | Must be able to move from point A to B, moving around rocks and small obstacles in the way | The team had still struggled with implementing obstacle avoidance algorithms using the ROS navigation node. The idea was to integrate the ROS Topics outputted by slam-gmapping algorithm and subscribe to the published messages. Testing would have been to place small, simple objects in front of the rover and cameras and ensure the proper PID outputs were enabled for the rover to write to the navigation node, moving forward and left or right of the obstacle. This key requirement was in the process of testing and fine tuning and required extensive testing through the month of March before prototype delivery, however the project was cut short. The team believes with the time to develop and resources and riding the wave of progress we had achieved all semester that the project would have been completed. |
| Manual Control | Using keyboard teleoperation commands to manually navigate to within range of AprilTag, and switch to autonomous mode | Yes, the team was easily able to integrate manual navigation using ROS Serial and Navigation nodes on the Jetson TX2. Simple testing included W, A, S, D keyboard controls over wireless SSH, driving down hallways with the rover in range of the host. |

A green background indicates that the requirements were successfully met.

An orange background indicates that the requirements remain to be tested.

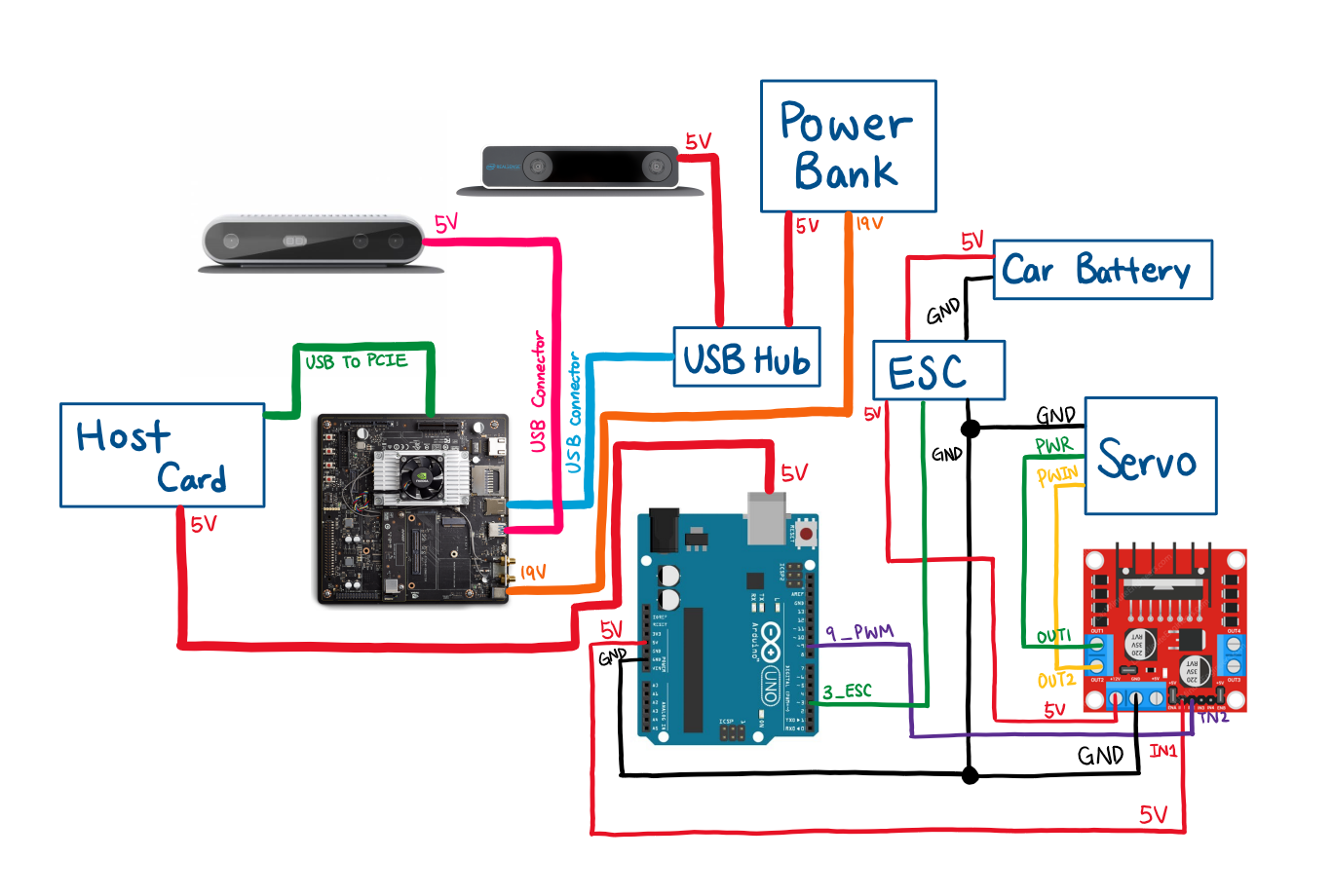
# Appendix 1: Gantt Chart

# 

*Figure1. This is a Gantt Chart for all the tasks for the 2nd semester.*

Note: This gantt chart was our original timeline of tasks that we need to do in order to finish the project; however, we were advised to stop working on our project due to COVID-19

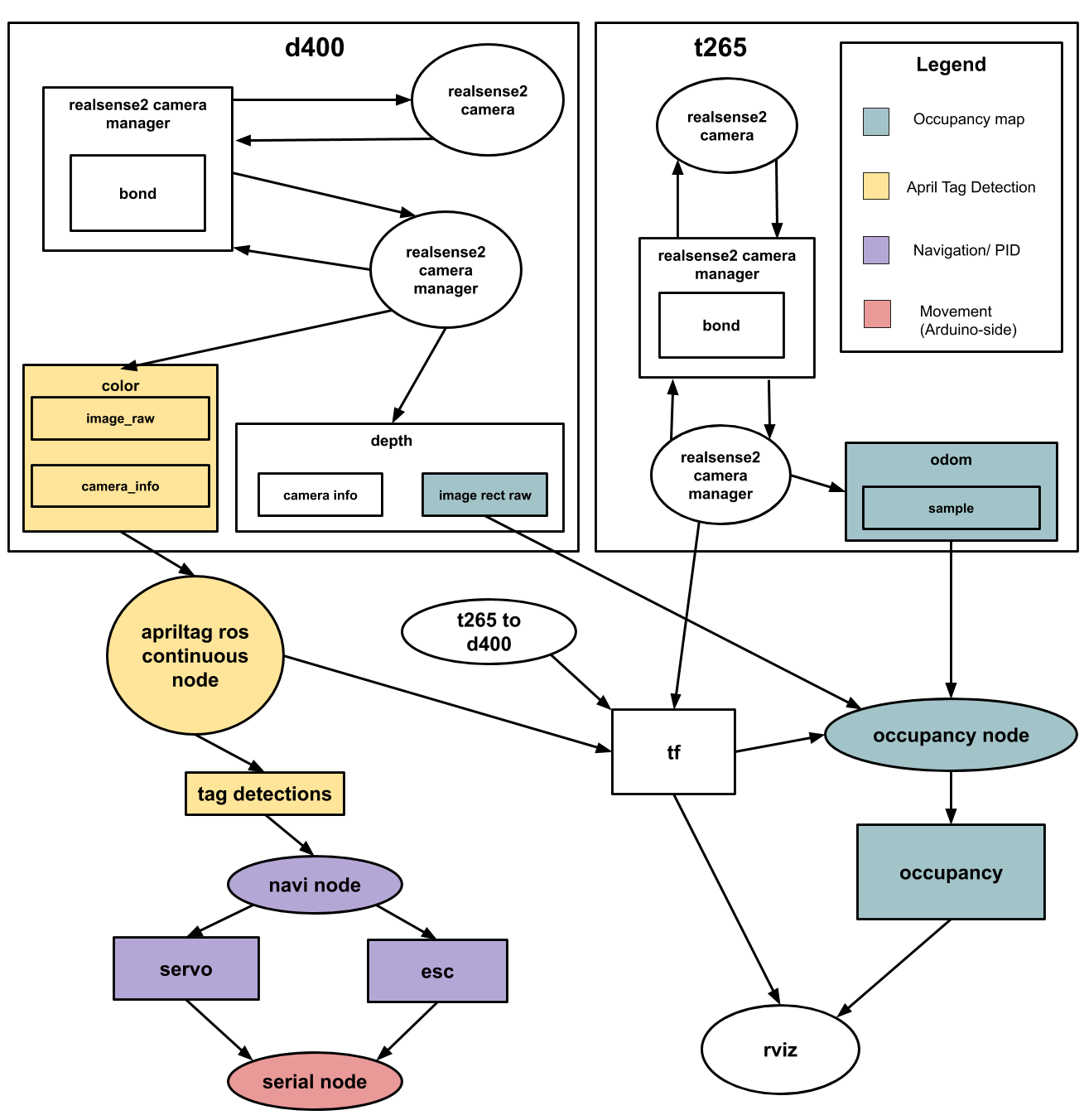
# Appendix 2: Hardware Diagram



***Figure 1.*** *This is a hardware schematic of the rover. It shows all the hardware components that we mounted on the rover*

# Appendix 3: ROS Node Graph

This is a simplified node graph based off of one generated by the ROS tool RQT. ROS operates on the simple principle of having nodes which take data from (aka subscribe to) buffers (aka topics), process the data, and send data (aka publish) to other topics. Here, topics are represented as rectangles and nodes as ellipses. The four distinct features of our software stack are color-coded for the viewers’ convenience.

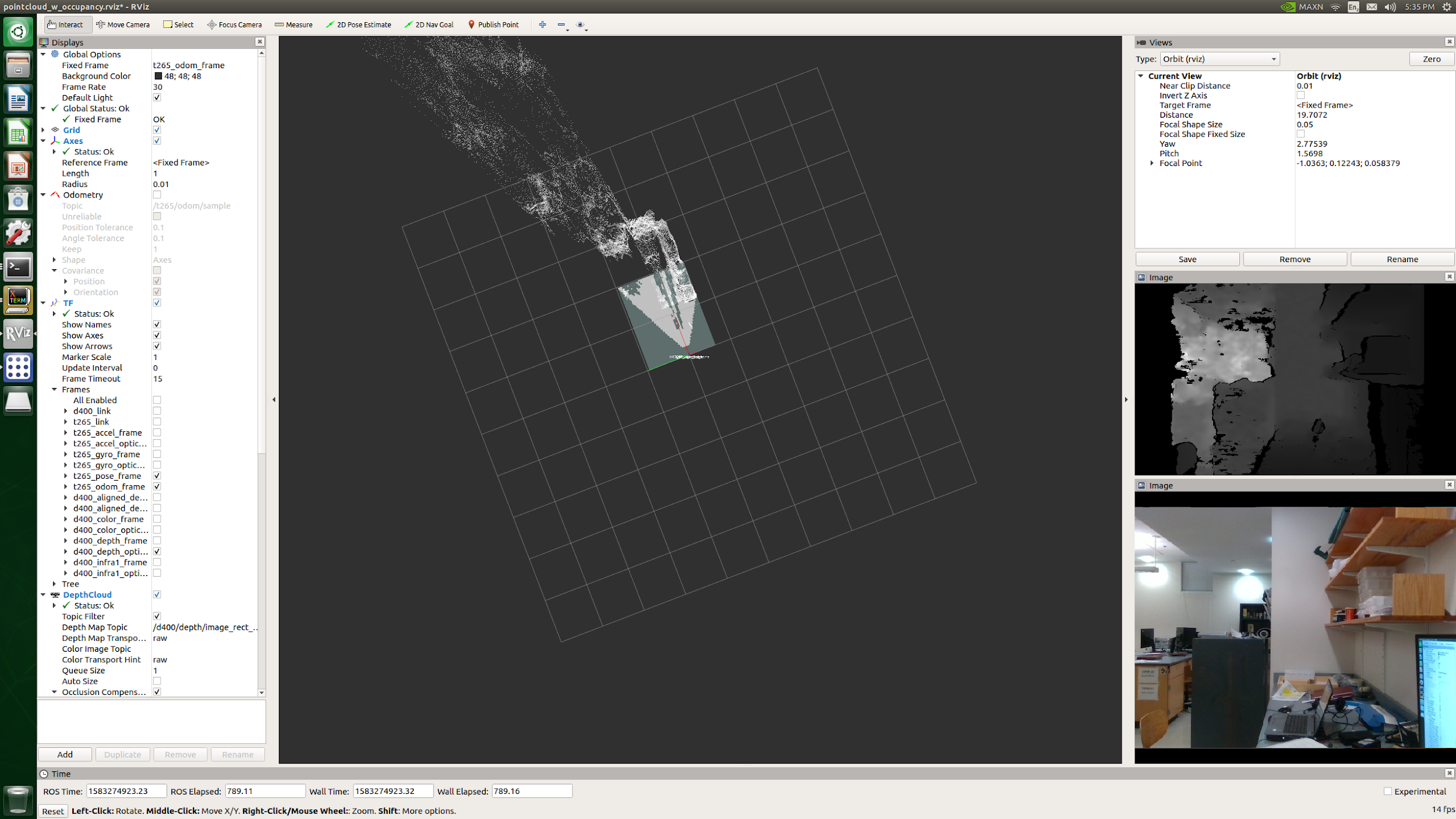
****

# 

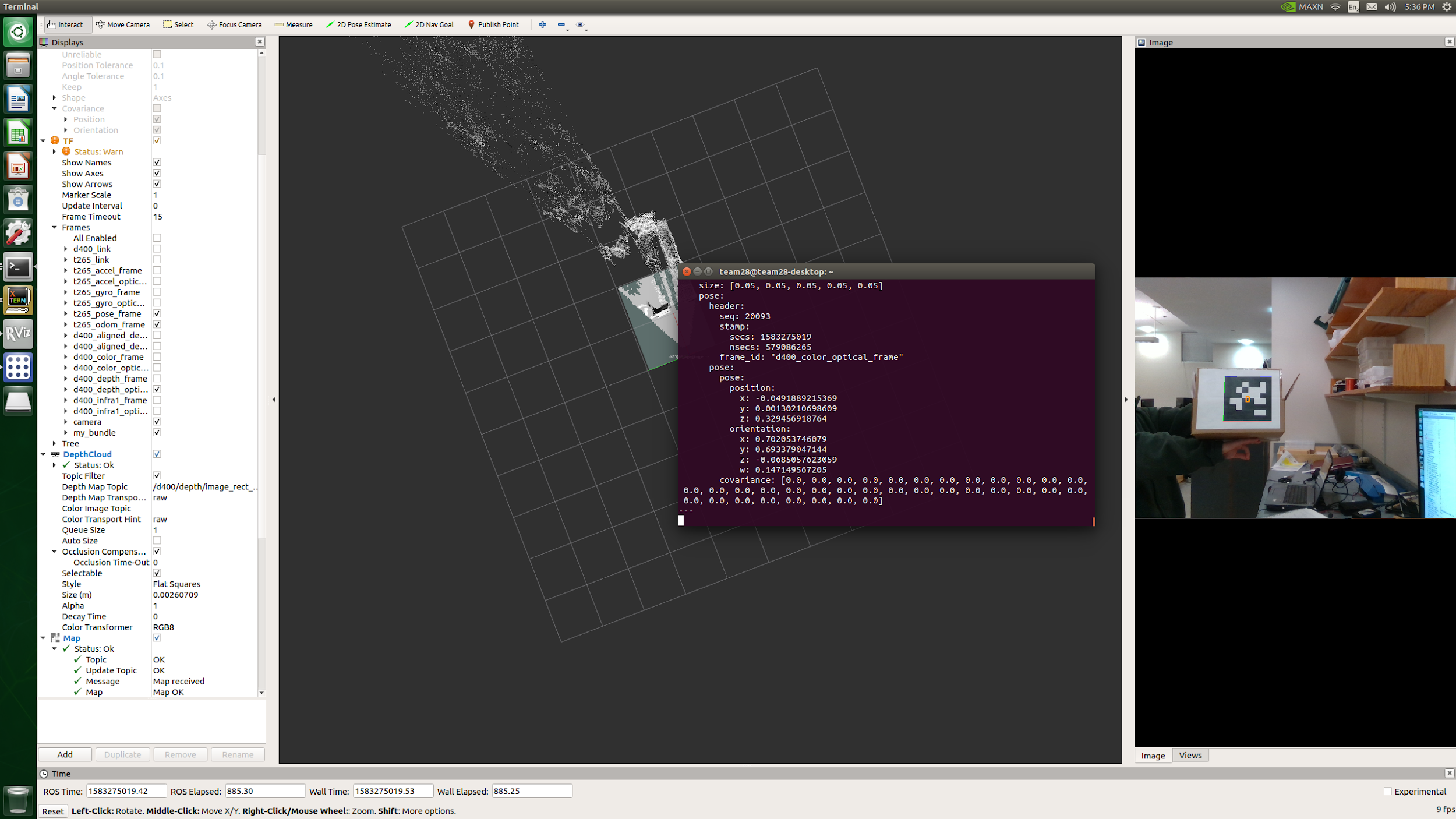
# Appendix 4: Test Result



***Figure 1.*** *Overall Design of the Rover with Hardware Components*



***Figure 2.*** *Occupancy map by using Intel RealSense D415 Depth Camera on RViz*



***Figure 3.*** *AprilTag Detection and Tracking using Intel RealSense T265 Tracking Camera*

# Appendix 5 : Team Information

**History of Team and Company**

Team 28 assembled over the summer of 2019. We anticipated proposing a new and brilliant design concept to work on for the year. After much deliberation we decided on making a water-traversing, bird-hazing hovercraft. Unfortunately, this idea did not come to fruition and we were instead offered a chance to work on the Mars Rover project.

The original concept for the Mars Rover project was for us to build part of the system to participate in the annual Mars Rover Society University competition. Our design would be fundamental to tackling the Autonomous Traversal Challenge set out by the Mars Rover Society. As of now, we are unclear as to whether we will enter the competition this year, as the club is in its first year and there are many dependencies on other teams and club members.

**Team Member Information**

**Daniel Crawley**

[crawley@bu.edu](mailto:crawley@bu.edu)

Daniel was born in Boston and has since not ventured too far, attending high school a stone’s throw away from Boston University. He is currently pursuing a BS in Computer Engineering with an Electrical Engineering minor. His current academic interests include High Performance Computing and Embedded Systems. One of the few interesting facts about him is that he is a citizen of three different countries, and may give you a penny if you can guess all three.

**Brian He**

[brianhe@bu.edu](mailto:brianhe@bu.edu)

Brian was born in New York. One interesting fact about him is that in highschool, he was on his school's Table Tennis and Swimming team. He does not exercise anymore because he is too busy studying Computer Engineering at Boston University. His favorite food is sushi and he loves traveling to different places.

**Tommy Lam**

[tlam11@bu.edu](mailto:tlam11@bu.edu)

Tommy was born in Minnesota and moved to Boston when he was 8. Some time has since passed and he is now pursuing a BS in Computer Engineering with a minor in Computer Science at Boston University. He is looking for a full time employment opportunity in software engineering. A fun fact about him is that he once gained 30 pounds over one summer.

**SeungYeun(Kelly) Lee**

[sylee538@bu.edu](mailto:sylee538@bu.edu)

Kelly is from South Korea, and she came to the United States for school. She’s been in the States for 10 years, and now she is currently pursuing a BS in Computer Engineering at Boston University. She is currently looking for a software engineering/computer engineering job in Boston. One fun fact about Kelly is that she changed her major from Mechanical Engineering to Computer Engineering thinking that it would be easier but it really isn’t.

**Linden Vo**

[lindenvo@bu.edu](mailto:lindenvo@bu.edu)

Linden, hailing from Randolph, Massachusetts, is pursuing a BS in Electrical Engineering. He is seeking full time positions in digital hardware and software design. He enjoys working with FPGAs and processors and looks to further hone his skills in Verilog, VHDL, and System Verilog for system design, test, and verification. Linden loves playing games such as League of Legends and was captain of his high school varsity volleyball team. He is also the president of BU’s chapter of the Society of Asian Scientists & Engineers and actively involved with BU’s Vietnamese Student Association.