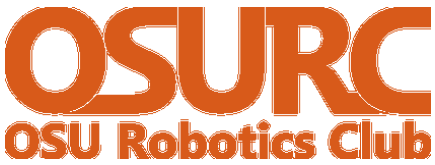

CS CAPSTONE PROBLEM STATEMENT

OCTOBER 10, 2017

OSU ROBOTICS CLUB MARS ROVER GROUND STATION

PREPARED FOR

OSU ROBOTICS CLUB

NICK MCCOMB

PREPARED BY

CHRISTOPHER PHAM

Abstract

The purpose of this document is to prove my understanding of the problem that will occur. By listing out the requirements that are requested of this project, this allows for I to define, explain, and bring into my own words how the problem is perceived and allows for misunderstandings to be brought up sooner rather than later.

CONTENTS

1 Overview

For this project, our group is looking to build a ground control station for the Mars Rover group which is a part of the Oregon State Robotics Club. The product is necessary for their club because this is going to view, control, and access the rover while it is on the ground in competition or even possibly in space. Our client, Nick McComb, is the Mars Rover team lead and we possibly might be in contact with other people of the Mars Rover team to fit their needs as this is an integral part of their combined systems. We had our first meeting October 8th to go over our requirements and expectations for our project.

2 Logistics

For our group, we are expected to use Asana which will be our team tracking software and will delegate tasks as decided by Nick or some other leads to accomplish goals in time. Asana will be used compared to other software because of the request of the Robotics team and there is already a subscription with them. Our version control will be the Robotics club's because it is what they requested because other code pertaining to the event will be located there too.

3 Software

For the software we are creating, there are requirements that are requested by our client. There might possibly be more requests by club members but those can be appended on as we progress.

3.1 Requirements

There is a requirement for us to use Python 3 with PEP-8 standard linting. We are required to use the QT framework under the wrapper PyQt5 to make any graphical user interfaces (GUI). All the code needs to be deployable by May 31, or the day of the competition. For the software we are creating, we will need to use Robot Operating System (ROS)framework to handle sending and receiving controls and statues, video streams over remote Ethernet link. All the code must be multi-threaded using QThreads and/or multi-process with ROS node system.

3.1.1 GUI

As per request of the client, we need to build a system that has a dark theme that is suitable for the location of the competition near Hanksville, Utah, such that the desert light does not cause a bunch of glare when trying to use the system. There needs to be a system that can adjust the bitrate of the video streams to allow for network interference at the competition and flexibility. The system should adjust the video bitrate automatically but a user could override the quality.

The project most likely will run on two 1080p (1920x1080) monitors in full-screen mode on both displays. The left monitor should contain an active video selection for the other display and should show a navigational map which contains the rover's current location, points of interests, and way-points. The map should have the ability to put pins for points of interests by the rover's location, by clicking on the map, or by explicitly putting in GPS coordinates. A way of editing the way points is required to show active way points, change order of way-points, or delete way-points. It should also contain a compass indicator, an Inertial measurement unit (IMU) visualization like pitch, roll, yaw like a flight attitude indicator. Autonomy settings are required like enabling it, video overlays showing processing/processed data if possible, and indicator for when the rover reaches its location and stopped. There is a need to show the statues

of the core systems: ”bogie” connection statuses, arm connection status, camera presences, joystick connection status, current connection latency, current network max throughput, radio calculated distance, current radio signal strength, GPS connected, Rover’s Next Unit of Computing (NUC) CPU usage, NUC Ram Usage, NUC file-system usage, GPS speed in m/s, GPS heading, GPS connected satellites, GPS accuracy. There should be a way to monitor voltages like minimums or approximate safe discharge time. Another request was finding a way to simulate the moving arm using the OpenGL wrapper in Python/PyQT to show the operator how the arm is moving. Another display would be showing the science probes like soil sample sensors or biological sensors attached to the rover. The right monitor will contain the primary video display or a secondary display in ”picture in picture” mode or ”picture by picture” mode and tools to save the video stream that is coming from the rover.

4 Hardware

The client requests that we run the software on an Intel NUC (2016) or a laptop that will be running Ubuntu 16.04 LTS. There will be two 24” 1080p monitors attached to the system. We will need to somehow to figure a way to interface the two universal serial bus (USB) joysticks to control the rover or via keyboard and mouse.

5 Suggested Solution

The project itself leans towards like a wrapper like system for ROS and end user to control using PyQT to see, inputs via USB to control, and Python to control and send back data to the rover. For this project, most of it is broken up into the front-end QT/OpenGL and back-end python code to run the front end. To solve how to place points of interests via clicking, it could be done by calculating where in the ”box” the click landed and then comparing that to the map texture. To simulate an IMU, you could use OpenGL and make a lens and sphere and calculate the changes needed. An overlay for autonomy mode finishing would just overlaying an object over the video stream which could be an OpenGL object, or covering it up with Qt. The rest of the GUI requests are just displaying the info to screen. The way of solving the way-point system would be harder than expected by using some database base system like SQLite and allowing for path finding.

6 Performance Metrics

This project needs to be done before May 31, 2018, the date of the competition but the metrics that are necessary and required are the requirements above. Any addendum to the project will be listed under requests and not mandatory for project completion.