Oregon State University College of Engineering

OSURC OSU Robotics Club

CS Capstone Requirements Document

April 26, 2018

OSU Robotics Club Mars Rover Ground Station

Revision 1.1
Changed requirements to match competition change and slow hardware development from Rover team

Signatures
[illegible] 4/26/18
[illegible] 4/26/18
[illegible] 4/26/18
[illegible] 4/26/18

Prepared for

OSU Robotics Club

Nick McComb

Prepared by

Group 30
Ground Station Software Team

Kenneth Steinfeldt
Christopher Pham
Corwin Perren

Abstract

This document covers the design requirements for the OSU Robotics Club Mars Rover Team's Ground Station Software. It begins with details about why the software is necessary, and a general overview of what it needs to accomplish. We then go into further detail about what the specific functional requirements will be, how they relate to each other in terms of dependencies, and then end with stretch goals if there is extra time to complete them.

CONTENTS

1 INTRODUCTION

1.1 Purpose

The purpose of this document is to formally define the characteristics of the software to be implemented. These requirements can then be used to determine if the software is complete once the project is done, while also serving as a reference during development.

1.2 Scope

We will be creating the ”Mars Rover Ground Station” software package. This software will allow Rover team members to interact with the Oregon State University Robotics Club’s Mars Rover for the purpose of a competition taking place in Drumheller, Alberta, Canada during August of 2018. The software will provide Rover systems monitoring as well as control of drive and arm systems. During competition, the software will be run on a remote base station primarily operated off the back of a truck, allowing users to tele-operate the Rover during manned controlled missions, or to monitor progress during autonomous missions. Properly functioning and fully featured ground station software will be a major factor in the success of the Mars Rover during this competition.

1.3 Definitions / Acronyms / Abbreviations

- ROS - Robot Operating System
- QT - A multi-platform GUI and application framework
- GPS - Global Positioning Satellite
- User - Person operating rover from ground control station
- GUI - Graphical User Interface
- RTT - Round-trip Time, the length of time for a signal to be sent plus the length of the time it takes for an acknowledgement of the signal to be received.
- Bogie - Groups of two individual wheels on the Rover, each with the ability to be driven independently.
- IMU - Inertial Measurement Unit
- CAD - Computer Aided Design

1.4 Overview

The remainder of this document consists of 4 sections and will describe the requirements that must be met in order for this project to be considered fully complete. The next section gives a description of the process. The fourth section describes the ground control software’s functions, user characteristics, constraints, and assumptions and dependencies. The fifth section describes the specific requirements that must be met for the software to be a success. Finally, stretch goals in the sixth section will describe hopeful, but not mandatory goals for the software.

2 DESCRIPTION

2.1 Product Perspective

This project interacts with and requires the Robotics Club’s Mars Rover robotics vehicle in order to be useful. If the Rover is not connected, the software will still be able to launch, but will not perform any useful functions until it has established a connection to the Rover. In order to accomplish interaction with the rover, the ground station software will use ROS for primary functionality and feedback from the rover.

3 Product Functions

The ground control software will primarily accomplish two tasks:

- Provide Rover control to the user. Direct rover control will be accomplished with joystick(s) and/or a SpaceNav mouse along with a keyboard and mouse.
- Enable and present rover feedback to the user. The rover will send health and status information to the ground control station.

3.1 User Characteristics

Users of the Rover Ground Station software will comprise of Rover team leads and members.

3.1.1 User Stories

- User should be able view ground station UI on two HD monitors so that valuable competition time will not be spent switching between screens on a single monitor
- User should be able to add way-points because way-points are needed to find useful points in the competition field and are necessary for autonomous navigation
- User should be able to remove way-points because way-points may become unneeded or need to be reset for a new competition event
- User should be able to edit way-points because there's a chance that the User may enter way-point details incorrectly and will need to change them
- User should be able to change the order of way-points because we might need to traverse back to a location, or fix improperly entered way-points
- User should be able to control the Rover by joystick(s) because that is the preferred way of controlling the Rover
- User should be able to select video streams because the Rover has more cameras than can be streamed at any one time, and being able to view any camera on demand will help the User drive the Rover and manipulate the arm
- User should be able to view the pitch, yaw, roll of the Rover because we want to know if the rover is on an incline, decline, or about to tip over
- User should be able to switch on and off autonomy because there are competition events that require autonomy and others that require manual control
- User should be able to tell that autonomy is enabled because that alerts the User that they can no longer manually control the Rover
- User should be able to view how the arm joints positions as that can show us if the arm is performing correctly
- User should be able to control the arm because competition events require the use of a Rover arm
- User should be able to set the Rover speed limit so that the User can more easily drive the Rover over difficult terrain
- User should be able to view the Rover CPU usage because the User can use this information to tell if there is a software problem on the Rover or if the Rover is trying to process too much information
- User should be able to view the Rover memory usage because the User does not want the Rover to crash or slow down to unusable levels due to running out of memory
- User should be able to view the Rover drive usage because the User does not want the Rover to crash if the drive is full

- User should be able to see the speed of the Rover because it can be used to determine if the Rover is driving at a speed that is reasonable for the current terrain
- User should be able to see how accurate the GPS positioning is because at the competition there is a requirement to be a certain distance from an object
- User should be able to see what heading the Rover is at because it allows for the User to know what direction to go
- User should be able to see the maximum throughput of the network because it will allow for video stream quality to go up or down depending on the connection quality
- User should be able to confirm connected devices because this will allow for debugging and checking of other connected devices

3.2 Constraints

Due to the nature of the Mars Rover project, there are multiple hardware and software constraints to ensure that the Ground Station software can properly communicate with the Rover. Additionally, the following constraints will help ensure future re-usability of the finalized software, which is a key goal of the Mars Rover team.

The software language allowed is the primary constraint for this project. Python 2.7 is required by the Mars Rover team as it maintains consistency with the software written on the Rover itself. The team also requires Python as it more easily allows future team members to reverse engineer code to be reused in future projects.

Use of the QT application framework is another constraint placed by the Rover team in order to facilitate rapid prototyping of the user interface for this project. It will also allow for faster and easier modification both by future members of the team, and during competition as that has previously been necessary.

The Rover internally uses ROS to handle routing and interpreting control and status information as well as video data. To be able to view this video data, status information, and to be able to send the Rover control information, this ground station software must also incorporate ROS.

At a hardware level, the Rover team will be providing an Intel NUC desktop computer running Ubuntu 16.04 that the Ground Station software will need to be able to run on. They will also provide two HD monitors, one or two USB joysticks, a SpaceNav mouse, as well as a keyboard and mouse that will be used to view and interact with the software.

As the Rover project is highly volatile where designs change frequently and hardware development often falls behind, any systems that cannot be explicitly implemented must at least have placeholders in both the GUI frontend and code backend to make eventual development easier when the hardware becomes available.

4 Specific Requirements

4.1 User Interfaces

Upon initial launch, the application should be in full-screen mode across both of the monitors. The left-hand screen will show navigation, Rover status, and miscellaneous Rover controls. The right hand monitor will show the three live video streams from the Rover. In the case that the software is started without a Rover to connect to, the software will display as-such and show placeholder information until such a connection is made.

4.2 Functional Requirements

4.2.1 Statuses / Informational Readouts

1) Rover Connection Status

- *Description:* This indicator will show a binary state of whether or not the Rover is currently connected.
- *Rationale:* The Rover must be connected in order for most of the software aspects to function. It is also useful to tell when full network dropouts happen.
- *Dependencies:* None

2) Joystick(s) Connection Status

- *Description:* Status indicators will visually show binary states as to whether joystick(s) are currently connected.
- *Rationale:* Joystick(s) must be connected to drive the Rover.
- *Dependencies:* Rover Connection Status

3) SpaceNav Mouse Connection Status

- *Description:* Status indicators will visually show binary states as to whether the SpaceNav mouse is connected.
- *Rationale:* SpaceNav mouse must be connected to control the Rover arm.
- *Dependencies:* Rover Connection Status

4) Rover Battery Voltage

- *Description:* This indicator will show the Rover's battery voltage.
- *Rationale:* The User should know what the battery voltage is to be able to determine remaining runtime and to diagnose other Rover errors.
- *Dependencies:* Rover Connection Status

5) Wheel Connection Statuses

- *Description:* These indicators will show whether each of the six wheels are connected to the Rover.
- *Rationale:* This will let drivers of the Rover know if a wheel has failed, which is a common occurrence.
- *Dependencies:* Rover Connection Status

6) Arm Connection Status

- *Description:* This indicator will be a binary state of weather or not the mechanical arm is connected or not.
- *Rationale:* This will show the User whether the arm is connected, and therefore whether they will be able to use the joysticks to control the arm.
- *Dependencies:* Rover Connection Status

7) Arm Joint Positions

- *Description:* These indicators will show Rover Arm joint positions in terms of degrees or via visual relationships.
- *Rationale:* These indications will help ensure the User knows where the Arm actually is, and whether it is being moved within its operating limits.

- *Dependencies:* Rover Connection Status, Arm Connection Status

8) Camera Presence Statuses
    - *Description:* This indicator will show which cameras on-board the Rover and connected and functional.
    - *Rationale:* The User can use such indicators to verify a failed camera for easy troubleshooting.
    - *Dependencies:* Rover Connection Status

9) IMU Status
    - *Description:* This will indicate the pitch, yaw, and roll of the Rover.
    - *Rationale:* This can be used by the User to help navigate uneven terrain and to avoid flipping the Rover.
    - *Dependencies:* Rover Connection Status

10) Map Visualizer
    - *Description:* This will display a map of the current Rover location and surrounding areas.
    - *Rationale:* This can be used to see terrain maps of the area and to place useful way-point and autonomous navigation markers.
    - *Dependencies:* Rover Connection Status

11) Rover Computer Statuses
    - *Description:* These indicators will show the current CPU, memory, and disk usage of the Rover's on-board computer.
    - *Rationale:* These can be used to ensure the Rover's processing computer is not overloaded. It can also show potential software failures while serving as a useful indication that the connection to the Rover is stable.
    - *Dependencies:* Rover Connection Status

12) Current Radio Signal Strength
    - *Description:* This indicator will display the strength of the connection between the Rover and ground station.
    - *Rationale:* The User can use this information to determine if the Rover is driving out of range to ensure a connection is not fully lost.
    - *Dependencies:* Rover Connection Status

13) Rover Network Potential Throughput
    - *Description:* This indicator will show the maximum theoretical throughput in Kbps between the Rover and this software as provided by the Ubiquity routers.
    - *Rationale:* The User can use the information to help tune video bit-rates and resolutions to ensure the connection is not overloaded.
    - *Dependencies:* Rover Connection Status, Current Radio Signal Strength, Rover RTT

14) Estimated Rover Distance
    - *Description:* This indicator will show the distance from the Ground Station radio to the Rover radio, as calculated by the Ubiquiti radio systems.

- *Rationale:* The User can use this information to help ensure they do not drive out of communications range.
- *Dependencies:* Rover Connection Status, Current Radio Signal Strength

15) Rover Speed Limit
    - *Description:* This display will show the Rover's speed limit in terms of 0 to 100 percent.
    - *Rationale:* During manual driving operations it is often useful to limit the max speed of the Rover. A visual indication of this limit will help avoid confusion if the User forgets that a limit has been set, or if the limit is accidentally changed.
    - *Dependencies:* Rover Connection Status, Rover Bogie Status
16) Display Data Collection Sensors
    - *Description:* Provide data displays for any scientific sensors attached to the Rover.
    - *Rationale:* During some competition phases, scientific sensors will be placed into the soil to gather scientific data. This data needs to be sent back to the Ground Station so it can be recorded.
    - *Dependencies:* Rover Connection Status

4.2.2 Rover Control

1) Rover Joystick Driving Control
    - *Description:* When both joysticks are connected and the Rover is also connected, the joysticks can be used to drive the Rover.
    - *Rationale:* The joystick control is needed to be able to drive the Rover.
    - *Dependencies:* Rover Connection Status, Joystick Connection Statuses, Bogie Connection Statuses
2) Rover SpaceNav Mouse Arm Control
    - *Description:* When the SpaceNav mouse is connected, the Rover is connected, and the Arm is connected, this mouse can be used to manipulate the Rover arm.
    - *Rationale:* During some competition events, the arm will need to be manipulated by the User to complete competition tasks.
    - *Dependencies:* Rover Connection Status, Joystick Connection Statuses, Arm Connection Status
3) Rover Speed Limit Control
    - *Description:* When both joysticks are connected and the Rover is also connected, buttons or levers on a joystick may be used to adjust the max speed limit of the Rover.
    - *Rationale:* This quick adjustment is often needed to allow for finer control of the Rover over difficult terrain or when manipulating the Rover arm.
    - *Dependencies:* Rover Connection Status, Joystick Connection Statuses, Bogie Connection Statuses

4.2.3 Navigation

1) GPS Status
    - *Description:* This will show whether the GPS on the Rover is connected and whether it has GPS lock.

- *Rationale:* This is needed to check to see if GPS is working and functioning properly.
- *Dependencies:* Rover Connection Status

2) Rover Speed

- *Description:* This indicator shows how fast the Rover is moving in meters per second via the GPS.
- *Rationale:* This will help the User ensure they are driving within the mechanical limits of the Rover.
- *Dependencies:* Rover Connection Status, GPS Status

3) Rover GPS Heading

- *Description:* This indicator will show what bearing the Rover is heading.
- *Rationale:* The User can use this information to help navigate the competition course.
- *Dependencies:* Rover Connection Status, GPS Status

4) GPS Satellites Connected

- *Description:* This indicator will show how many GPS satellites are currently active.
- *Rationale:* This can help tell the User if the Rover is in a location that is bad for GPS reception, allowing them to drive to a better location before the GPS lock is lost.
- *Dependencies:* Rover Connection Status, GPS Status

5) GPS Accuracy

- *Description:* This indicator will how accurate the current GPS location is in meters.
- *Rationale:* This will allow the User to judge how much the GPS location of the Rover can be trusted, as well as giving useful information about how well the Rover might perform during the autonomous challenge.
- *Dependencies:* Rover Connection Status, GPS Status, GPS Satellites Connected

6) Way-point Placement

- *Description:* Give the ability to place way-points on a map by GPS location, by current Rover location, or via manual GPS co-ordinate entry and add this way-point to a queue. These way-points will also need to show up visually on the map.
- *Rationale:* This will used to map out the course during the autonomous challenge as well as to place points of interest during manual driving events.
- *Dependencies:* Rover Connection Status, GPS Status, GPS Satellites Connected, Map Visualizer

7) Way-point Editing

- *Description:* Allow for editing any way-point in the queue by drag and drop of the way-point, or via manual editing of the GPS co-ordinates.
- *Rationale:* This will allow for easy adjustment of way-points if they were placed incorrectly.
- *Dependencies:* Rover Connection Status, GPS Status, GPS Satellites Connected, Way-point Placement, Map Visualizer

8) Way-point Re-Ordering

- *Description:* Allow for changing the order of way-points.

- *Rationale:* This will allow the User to skip unneeded way-points or fix incorrectly ordered way-points.
- *Dependencies:* Rover Connection Status, GPS Status, GPS Satellites Connected, Way-point Placement, Map Visualizer

9) Active Way-point Selection

- *Description:* Allow for the selection of a way-point from the queue as currently active. Setting the way-point active will update a ”Heading To” indicator to help a User drive to the desired way-point.
- *Rationale:* This is useful to make navigating to competition provided co-ordinates easier.
- *Dependencies:* Rover Connection Status, GPS Status, GPS Satellites Connected, Way-point Placement, Map Visualizer

4.2.4 Autonomy

1) Autonomy Switch

- *Description:* A toggle to enable and disable the Rover’s autonomous mode operation.
- *Rationale:* This is necessary to place the Rover into autonomous mode for that portion of the University Rover Challenge competition.
- *Dependencies:* Rover Connection Status

2) Autonomy Indicator

- *Description:* An alert or indicator for when the Rover is in autonomous mode.
- *Rationale:* This indicator will make it easy to tell whether the joystick(s) can presently be used to drive the Rover.
- *Dependencies:* Rover Connection Status, Autonomy Switch, Way-point Placement

4.2.5 Video / Cameras

1) Video Stream Displays

- *Description:* The software should show three video displays in terms of a primary, secondary, and tertiary display. These displays will show streamed video feeds from cameras on the Rover and should be able to be individually disabled.
- *Rationale:* The User will need video feeds from the Rover to navigate the competition course and to actuate the Rover arm.
- *Dependencies:* Rover Connection Status, Camera Presence Status

2) Camera Source Selection

- *Description:* For the primary, secondary, and tertiary video displays provide controls to switch which Rover camera is currently active.
- *Rationale:* During competition, the User will frequently be changing the active cameras to best navigate the competition course and interact with objects.
- *Dependencies:* Rover Connection Status, Camera Presence Status

3) Camera Quality Adjustments

- *Description:* For the primary, secondary, and tertiary video displays provide controls to adjust the camera resolution and/or bit-rate.
- *Rationale:* As the Rover gets farther away from the ground station, connection quality will diminish and require that the video stream qualities be lowered to ensure smooth Rover operation.
- *Dependencies:* Video Stream Displays

4.2.6 Miscellaneous

1) Offline Functionality

   - *Description:* The ground station software must function without a connection to the Internet during competition. Software elements that require the Internet such as maps must be able to be cached for use offline.
   - *Rationale:* The competition is in a remote location with no access to an Internet source.
   - *Dependencies:* None

2) UI Dark Theme

   - *Description:* The software must be visually "dark" themed.
   - *Rationale:* The dark theme will help ease User eye strain and was also a client request.
   - *Dependencies:* None

3) Getting Started Document

   - *Description:* A 1-2 page document describing general useful details for future Rover teams on where to begin for reusing the software.
   - *Rationale:* This was requested by the client.
   - *Dependencies:* All Prior Requirements

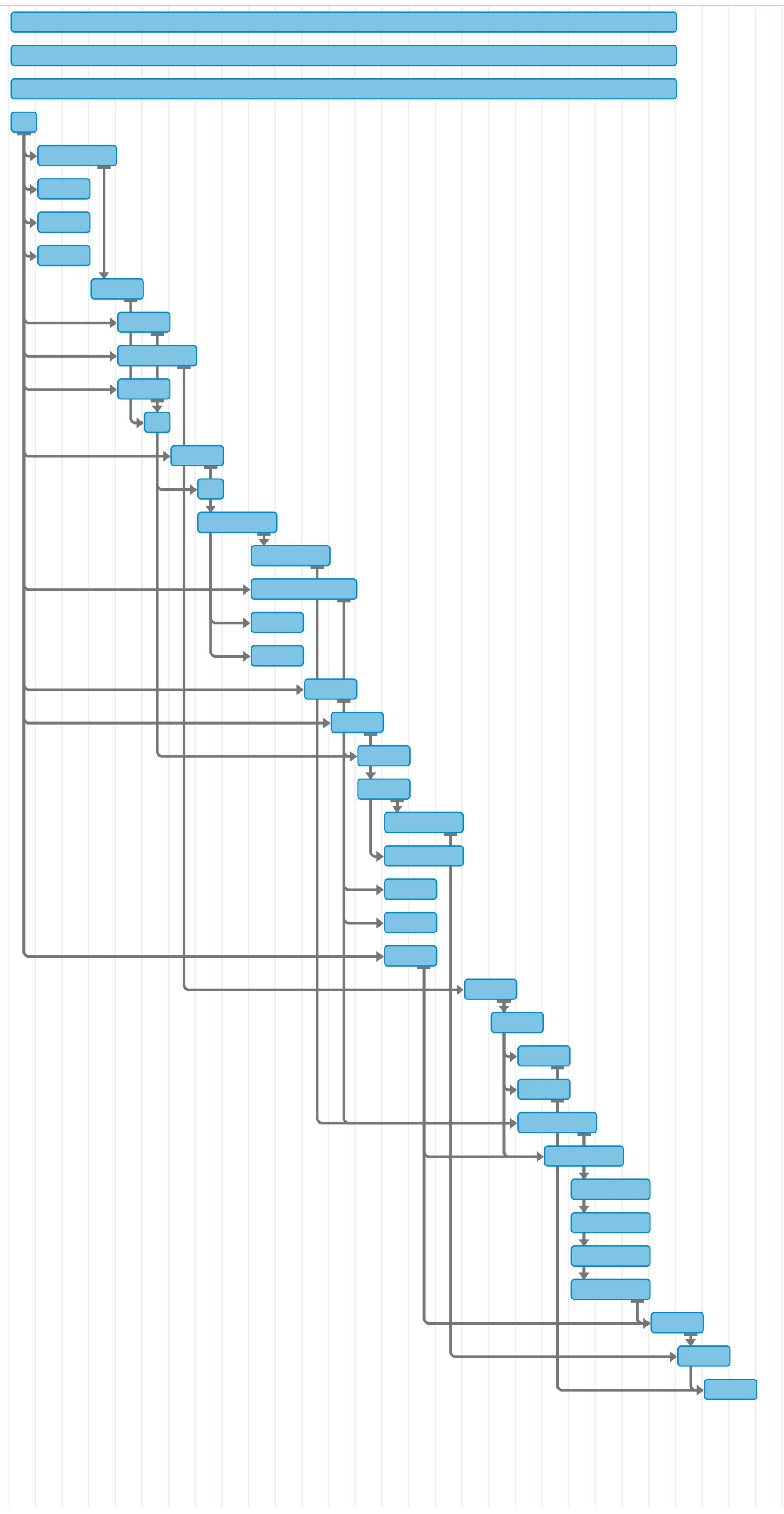
4) Quick Deployment

   - *Description:* This requirement is to package all items necessary for the Ground Station to function in a quickly deployable physical case that can be set-up and software started in less than two minutes during competition setup time.
   - *Rationale:* Fast setup times will allow the User to spend extra time testing Rover communications during the setup phases of competition.
   - *Dependencies:* All Prior Requirements

5 GANTT CHART

Mars Rover Ground Station

		Sub Tasks	Task Prog.
0	Full-Screen Mode		0%
1	Dark UI Theme		0%
2	Two Monitors Minimum		0%
3	Rover Connection		0%
4	Arm Connection		0%
5	IMU Visualizer		0%
6	Rover Computer		0%
7	Display Data Collection Sensors		0%
8	Arm Joint Positions		0%
9	Joystick Connection		0%
10	Camera Presence		0%
11	Rover Battery		0%
12	Rover Joystick Arm		0%
13	GPS Status		0%
14	Rover Voltage		0%
15	GPS Satellites Connected		0%
16	GPS Accuracy		0%
17	Map Visualizer		0%
18	Rover Speed		0%
19	Rover Heading		0%
20	Bogie Connection		0%
21	Current Radio Signal Strength		0%
22	Rover Joystick Driving		0%
23	Rover RTT		0%
24	Rover Network Potential Throughput		0%
25	Estimated Rover Distance		0%
26	Rover Speed Limit		0%
27	Rover Speed Limit Control		0%
28	Autonomy Switch		0%
29	Video Stream Displays		0%
30	Camera Source Selection		0%
31	Camera Quality Adjustments		0%
32	Video Stream FPS Counter		0%
33	Way-point Placement		0%
34	Autonomy Streams Overlay		0%
35	Way-point Editing		0%
36	Way-point Re-Ordering		0%
37	Way-point Navigation Path		0%
38	Active Way-point Selection		0%
39	Autonomy Indicator		0%
40	Log File Viewing		0%
41	Getting Started Document		0%
42	3D Rover Arm Visualization		0%
43	Quick Deploy Case		0%
44	Automatic Video Stream Quality Adjustment		0%



6 Stretch Goals

1) Autonomy Streams / Overlay

   - *Description:* This would display a specialty video stream either independently or via an overlay showing the course obstacles and/or features the Rover is detecting during autonomous mode navigation. This would depend on the computational overhead remaining on the Rover processing computer.
   - *Rationale:* This would help the team judge autonomous navigational performance of the Rover.
   - *Dependencies:* Rover Connection Status, Autonomy Switch

2) 3D Rover Arm Visualization

   - *Description:* Represent the arm joint positions using a 3D CAD model of the arm.
   - *Rationale:* A 3D representation of the arm and its positioning will make it easier for the user to precisely use the arm to manipulate objects.
   - *Dependencies:* Rover Connection Status, Arm Connection Status

3) Automatic Video Steam Quality Adjustment

   - *Description:* The software would automatically detect poor video quality through use of low FPS counts and/or high network latency and use this information to adjust video resolution and bit-rates to increase video stream FPS to reasonable levels.
   - *Rationale:* Automatic control of video stream quality would help the User avoid spending time adjusting values during valuable competition time.
   - *Dependencies:* Video Stream Displays, Camera Quality Adjustments, Video Streams FPS Counters