# SOFTWARE REQUIREMENTS SPECIFICATION

# for

# Finger Lakes Explorer ROV

**Version 1.0 approved**

**Prepared by Trevor Sherrard**

**P20250 Tiger Sharks**

**December 4, 2019**

# Contents

# Revision History

| Trevor Sherrard December 2019 | | | |
|---|---|---|---|
| 21 | 22 | 23 | 24 |
| 31 | 32 | 33 | 34 |

# 1 Introduction

## 1.1 Purpose

The requirements outlined in this document pertain to design and implementation of software for the Finger Lakes Explorer ROV for Team P20250 of Rochester Institute of Technology's Multidisciplinary Senior Design (MSD) Program.

## 1.2 Intended Audience and Reading Suggestions

This document is intended for the developer(s) of the overall systems software and all invested stakeholders in the project overall. These stakeholders include (but are not limited to) the project customer Dr. Jason Kolodziej, the project guide Gerald Geravuso, and the head of the MSD program Dr. Elizabeth DeBartolo.

## 1.3 Project Scope

The Finger Lakes Explorer ROV is designed to be operable by a single user for recreational underwater exploration in the Finger Lakes of Western New York State. The ROV will be able to navigate within a 20 foot radius of the launch point while operating at a depth of 100 feet. The ROV will capture a video feed (and thus still images) along with analog sensor data from pressure and temperature sensors. This data will be transmitted to the surface via a twisted pair data tether to a base station that will store this mission data in a historical context. The base station will allow the user to control the ROV via commands entered via an X-Box controller.

## 1.4 References

None yet.

# 2 Overall Description

## 2.1 Product Functions

This systems has two major domains of design. Those two domains being the ROV domain and Base station domain. Each of these domains serves different functions needed for the overall systems. These two systems are linked via the tether domain. The functions for each domain can be seen in this section.

### 2.1.1 ROV Domain

This domain contains all the software functionality required to maintain a bi-directional connection with the base station over the tether, collect mission data from on-board sensors and camera, and actuate motors and lights via hardware controllers due to user input.

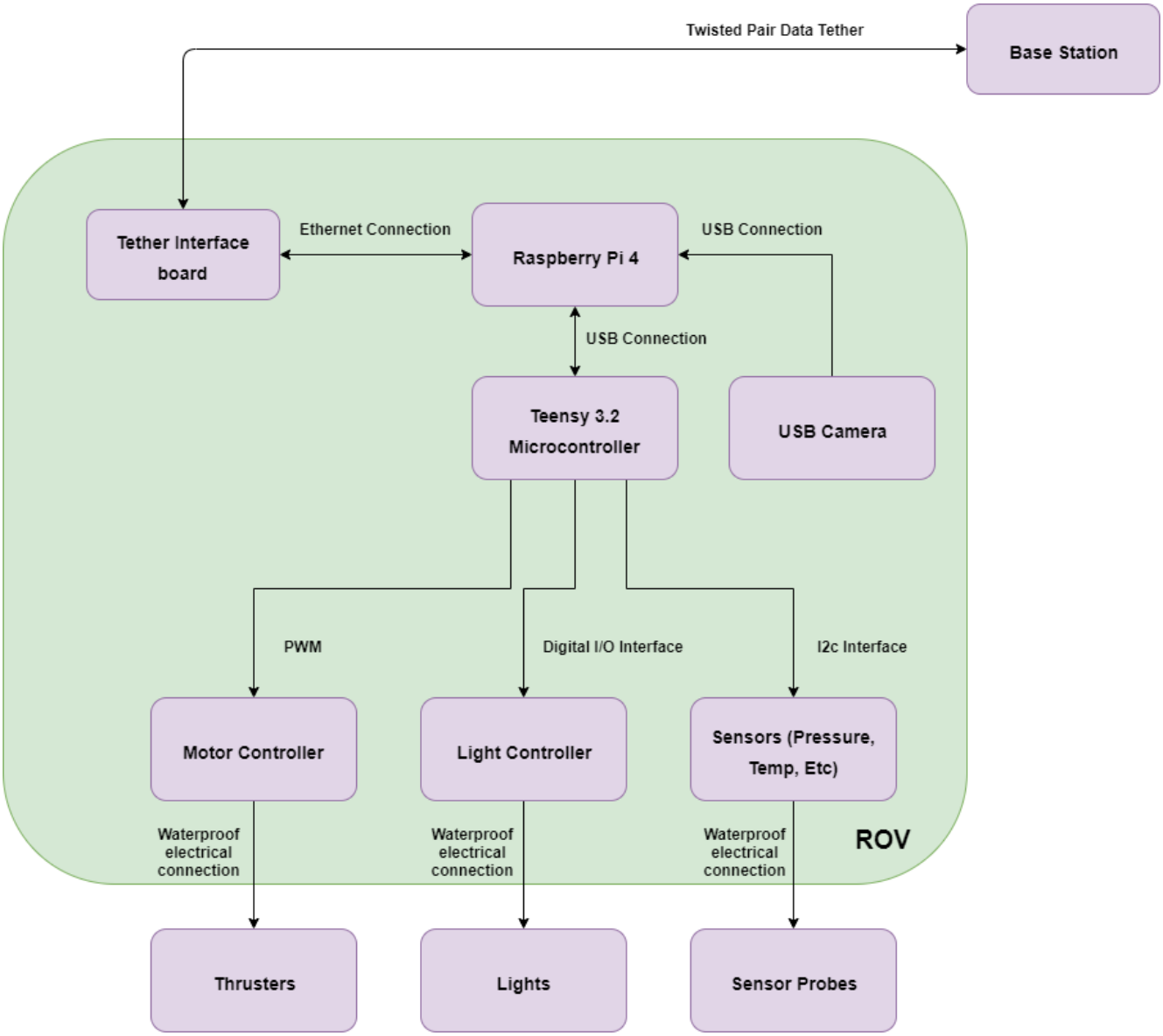


Figure 2.1: ROV Software Domain

Figure 2.1 in this section depicts the overview of the ROV software domain. From the diagram above, the following software functions fall out.

- ROV-Side Tether Communications Manager
- ROV-Camera Control Manager
- Micro-Controller and Raspberry Pi Serial Bridge

- Micro-Controller Software For Motor Control via PWM interface
- Micro-Controller Software For Light Control via digital I/O interface
- Micro-Controller Software For Sensor Data Collection via I2C interface

Later section in these documents will outline these functions in much more detail.

### 2.1.2 Base Station Domain

This domain contains all software functionality to maintain a bi-directional connection with the ROV over the tether interface, store mission data received from the ROV in the data archival system, start and end dive procedures, and update the HUD screen as required.

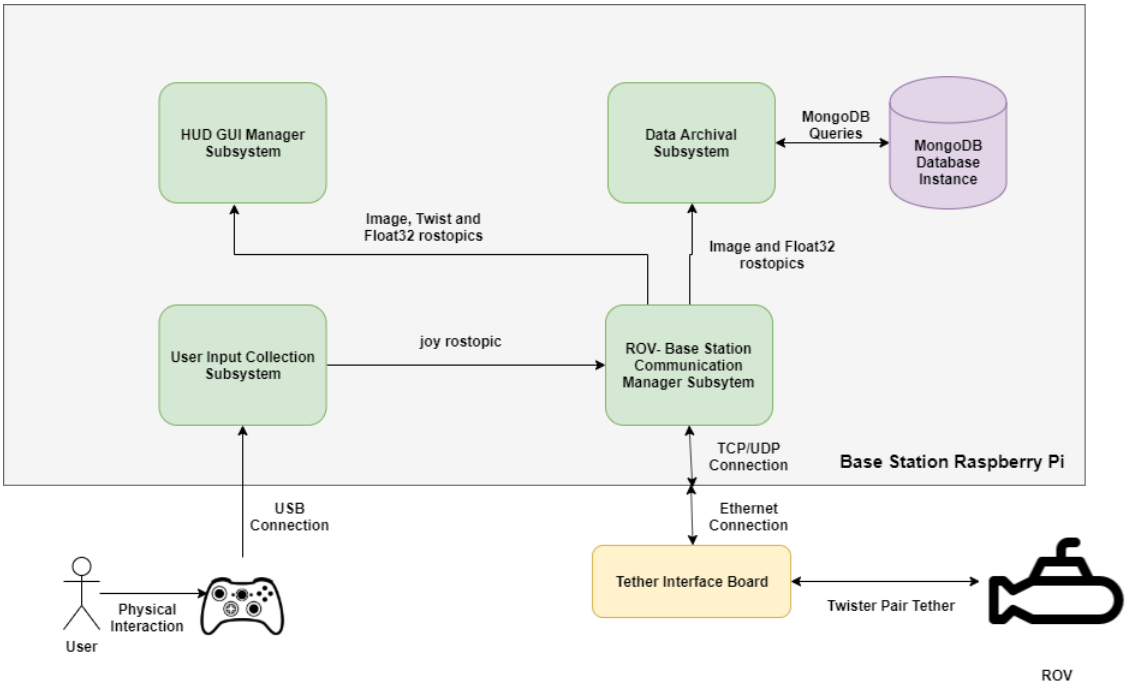


Figure 2.2: Base Station Software Domain

Figure 2.2 in this section outlines the overview of the Base Station software domain. From the diagram above, the following software functions fall out.

- Base Station Side Tether Communications Manager
- User Input Collection Manager
- HUD Front End Interface
- Data Archival Insertion Software

- Data Archival CSV Export Software

Later section in these documents will outline these functions in much more detail.

## 2.2 Operating Environment

The system's computational resources consist of two Raspberry Pi 4 single board computers and one Teensy 3.2 Micro-Controller. Both Raspberry Pi 4 devices have 4GB of available memory, and a Broadcom BCM2711B0 quad-core A72 (ARMv8-A) 64-bit processor operating at 1.5GHz. The ROV Raspberry Pi will have 16GB of hard disk space, and the base station will have 64GB of hard disk space. All Software shall be written to handle potential faults that occur from running computational resources in an real-time embedded setting.

## 2.3 Design and Implementation Constraints

Both Raspberry Pi devices only have 4GB of RAM and software on both base station and ROV, and thus software on both computational resources should be developed accordingly. The data archival system should be designed to interface with a MongoDB database instance. The maximum effective bandwidth over the twisted pair tether is 80 MBit/sec. Data packets being sent over the tether should be constructed and scheduled to account for this limitation. The hard disk size for the ROV Raspberry Pi is 16GB and the hard disk size for the base station hard disk size is 64GB and software on either raspberry pi must be designed within these limits.

## 2.4 User Documentation

The project will be delivering all generate source code and system launch scripts upon completion of the project. Documentation of system configuration and source code as implemented will also be delivered upon project completion. A user manual for operation of ROV will be constructed and delivered with the final system prototype.

## 2.5 Assumptions and Dependencies

The overall system will use ROS for message passing between the different nodes on both the ROV and base station domains. Data acquisition will be implemented through the use of a third party ROS node known as mongodb_store. This also requires the data storage database to be a MongoDB instance. The user collection software subsystem relies on a third party device driver for communicating with the Xbox controller via the base station computer. All third party source being considered for use is licensed under a free and open source software licence and thus can be used for this project's purposes.

# 3 External Interface Requirements

## 3.1 User Interfaces

The Heads Up Display (HUD) Software System will be the main interface with the ROV operator. This interface will be developed as a multi-page react web application with interfaces to a backend containing collected mission data and live frames collected from the ROV. This interface will display live video feeds from the ROV camera. It will also display an instantaneous reading of each mission data parameter, as well as an estimated pose of the ROV in three dimensional space. The main page of the HUD interface will also have a list of recent commands run via the Xbox controller interface as well as a box to display any messages sent from the ROV to the base station. Figure 3.1 in this section outlines a graphical mockup of this webpage.

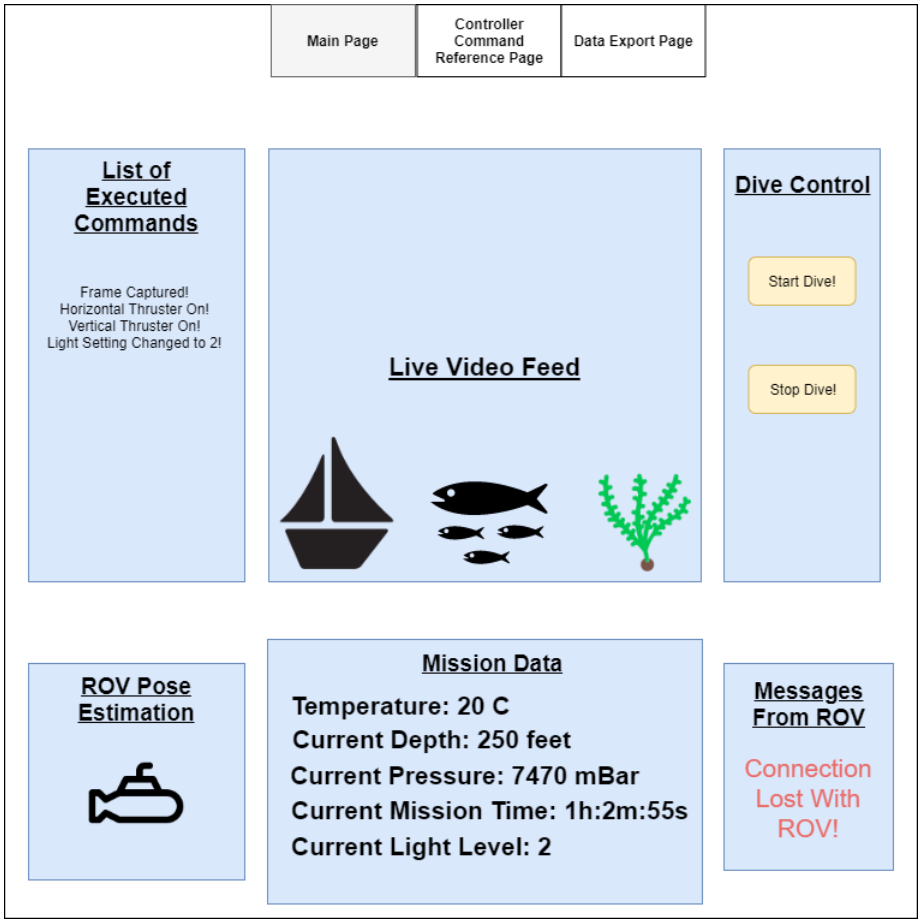


Figure 3.1: Main HUD Page Mockup View

The second page of the HUD interface will contain a reference for accessible commands able to be executed from the Xbox controller interface. This page will provide a graphical depiction of the Xbox controller to reference against the available list of commands. Figure 3.2 in this section outlines the graphical depiction of the controller reference view.

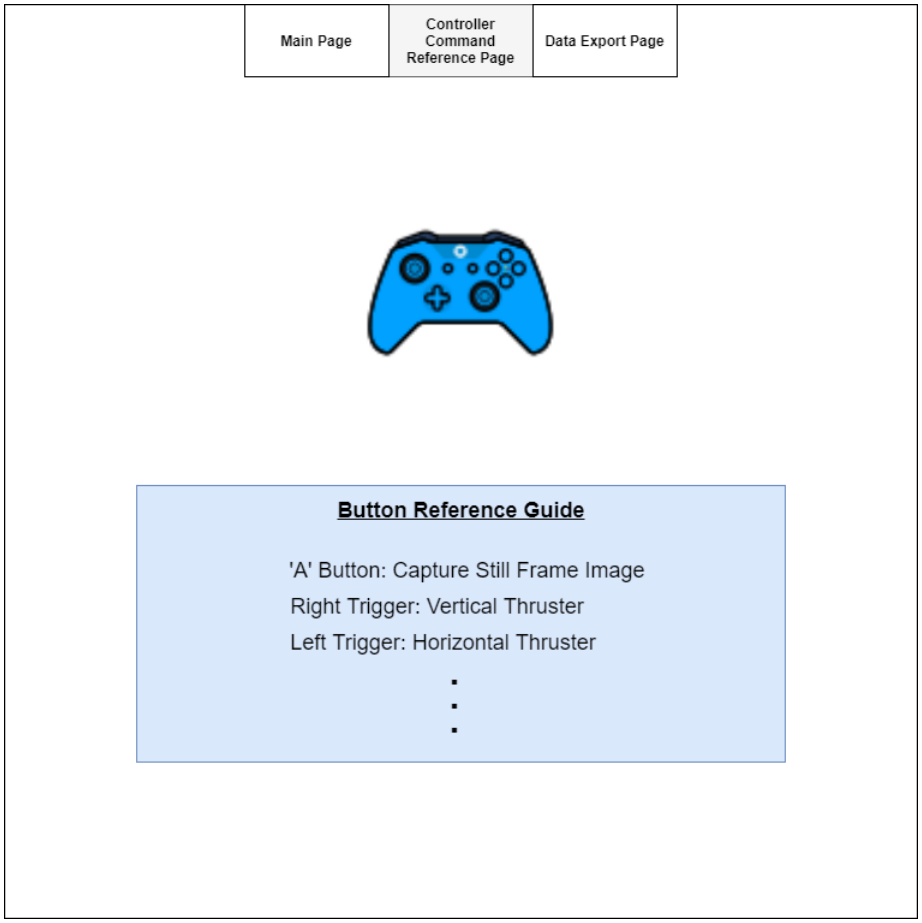


Figure 3.2: Controller Reference Page Mockup View

The third page of the HUD interface will contain an interface for the exporting of mission data to a CSV file based on a datetime range. The user will select a 'to' datetime and a 'from' datetime range and all data within the MongoDB database instance that fits that range will be compiled into a CSV file on the backend server and will be downloaded through the web interface. The user can then insert a flash-drive or other portable storage media to extract this CSV file from the system. Figure 3.3 in this section outlines the graphical depiction of the data export page view.


Figure 3.3: Data Export Page Mockup View

## 3.2 Hardware Interfaces

The main hardware interface that interfaces with users is the Xbox controller. This will communicate with the base station computer via a third party device driver from Microsoft. All sensors on-board the ROV will interface with the Teensy micro-controller via an I2C interface. The Teensy will also communicate with the motor controllers and lighting controller via PWM and digital I/O interfaces, respectively. The ROV and Base station software domains are coupled via the tether hardware interface for bi-directional data communication. Figure 2.2 depicts the hardware interfaces needed for the ROV software domain. The only major hardware interface on the base station side is the tether interface board to couple digital communication over the tether to the base-station Raspberry Pi to its Ethernet interface.

## 3.3 Software Interfaces

The software interfaces and their detailed design requirements are outlined within this section for both the ROV and base station software domains, respectively. The software design in this section of this document are to be implemented on the overall system, as this is the final software design. As previously stated, any changes between the software outlined in this document and the software implemented in the final system will be provided in the final project documentation.

### 3.3.1 ROV Domain Software Interfaces

The ROV software domain will have software interfaces to allow for communication between the Teensy microcontroller and the raspberry pi single board computer. It will also provided software interfaces for the communication between the base station and the ROV via the tether interface board. Lastly, the system will also allow for collection of images for live streaming and single frame capture. Figure 3.4 in this section depicts the overall view of the software interfaces present on the single board computer of the ROV.

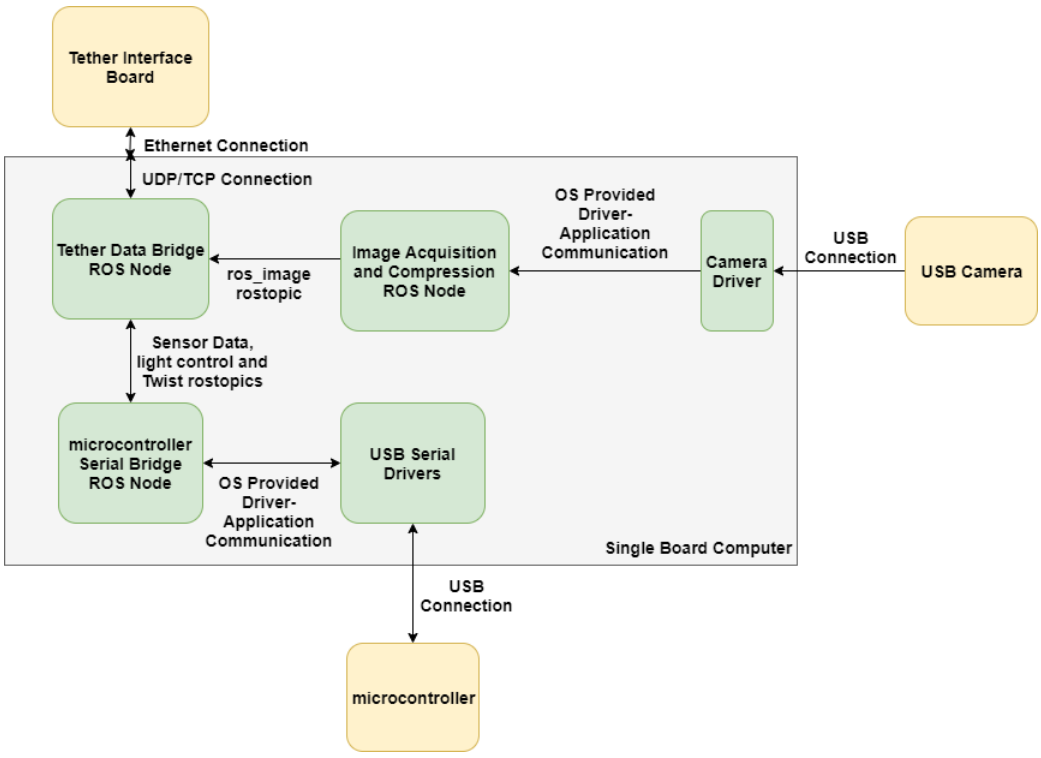


Figure 3.4: Single Board Computer Software Interface Diagram

Within figure 3.4, The interfaces between each software component can be seen. All data retrieved from the microcontroller and USB camera will interface with their respec-

tive hardware drivers before sending each given data stream to ROS nodes the subscribe to the ROS topic in question.

### 3.3.2 Base Station Domain Software Interfaces

The Base Station domain will have software interfaces to allow for data acquisition from the Xbox Controller, and software interfaces to allow for communication between the base station raspberry pi and the base station tether interface board. There will also be an API provided by a flash back end application to allow for the front end web application to communicate with the back end. The back end application will have software interfaces for communicating the the MongoDB database instance. All of these interfaces allow the fulfilment of the required functionality needed to control and monitor the ROV during the duration of the dive. Figure 2.2 depicts the high level software interfaces for the base station software domain.

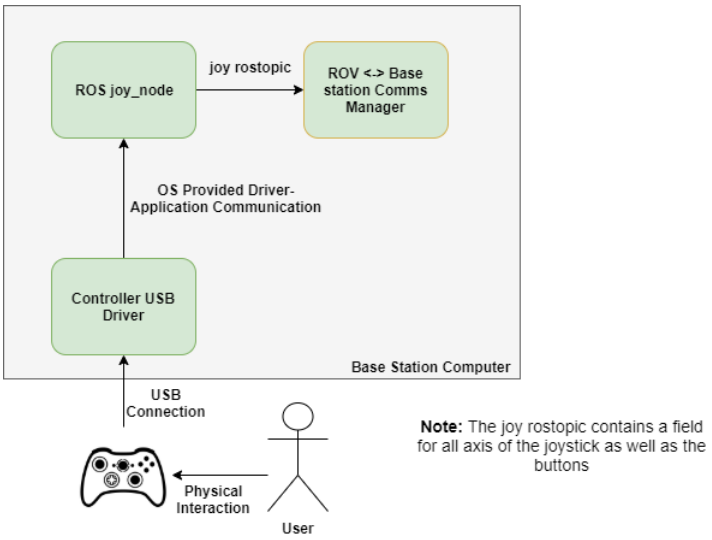


Figure 3.5: User Input Collection Software Interface Diagram

Figure 3.5 in this section depicts the software interfaces between the software components involved in user input collection via the Xbox controller. Data obtained from the Xbox controller will be published to any subscribing nodes via the ROS joy topic. It will likely be the case that the only node requiring this information in this phase will be the ROV¡-¿Base station communication manager.

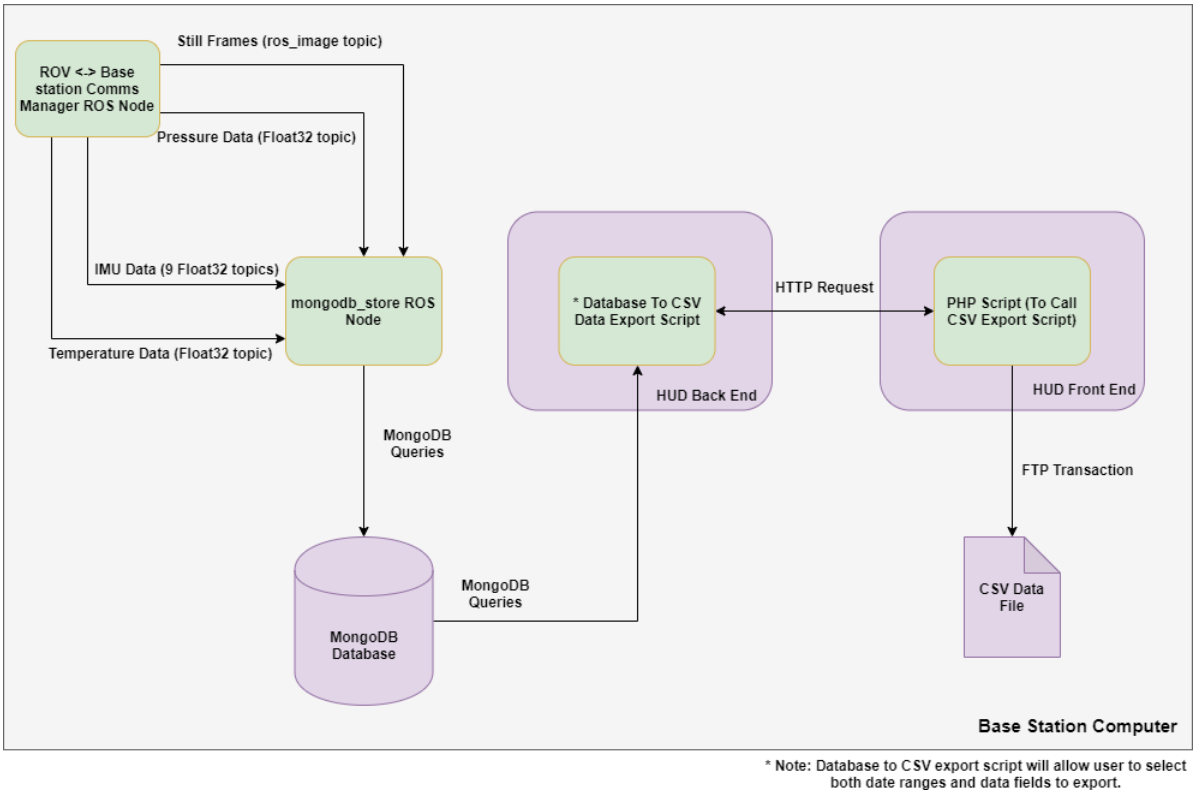


Figure 3.6: Data Archival Software Interface Diagram

Figure 3.6 in this section depicts the software interfaces between the various software components involved in the archiving of mission data into a MongoDB database instance, and the subsequent software required to retrieve that data at a later time. The MongoDB_store ROS node allows for data streams via ROS topics to be aggregated into a database in a time-series manner without an issue.

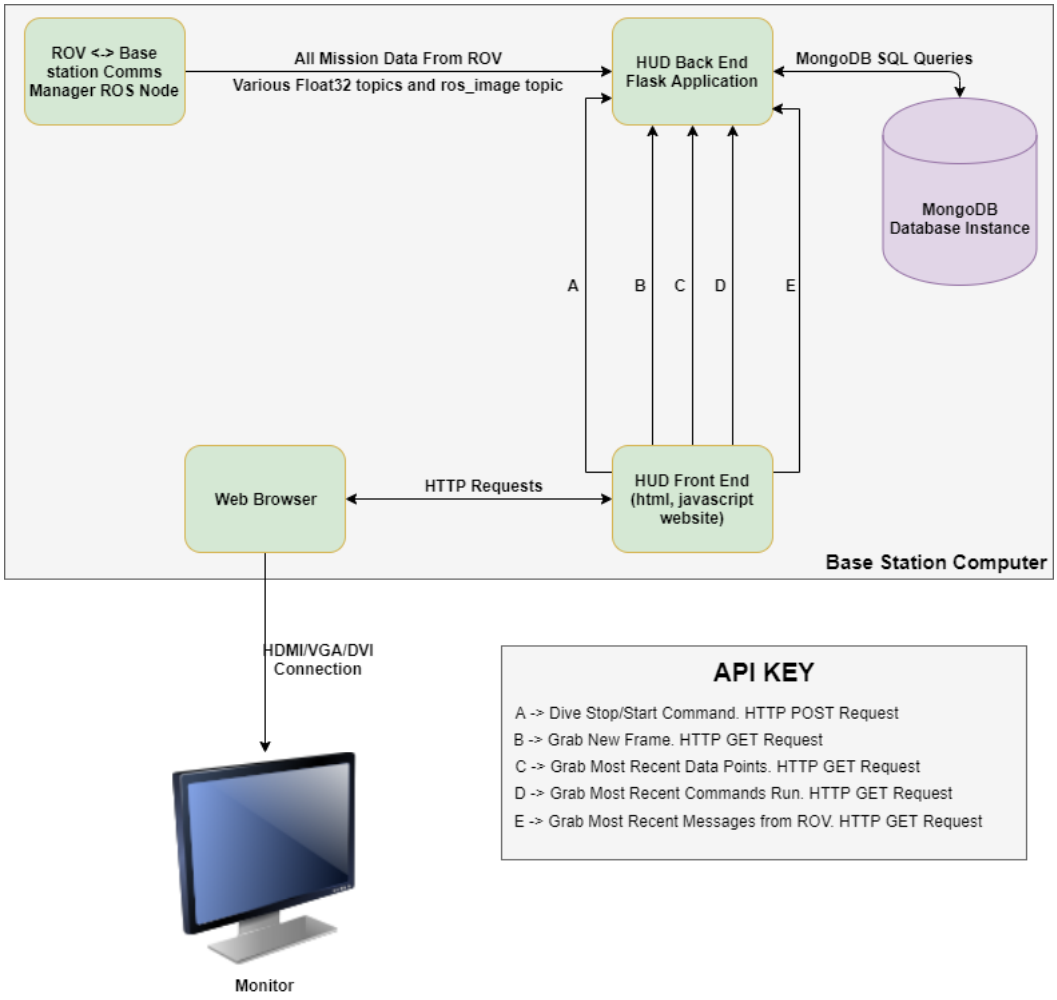


Figure 3.7: Software Interface for Main GUI Page Management

Figure 3.7 in this section depicts the software interface required to support the main page of the HUD interface seen in 3.1. An API exposed within the back end flask application allows for the front end to request new frames, new mission data points, messages from the ROV, and a list of most recently run commands.

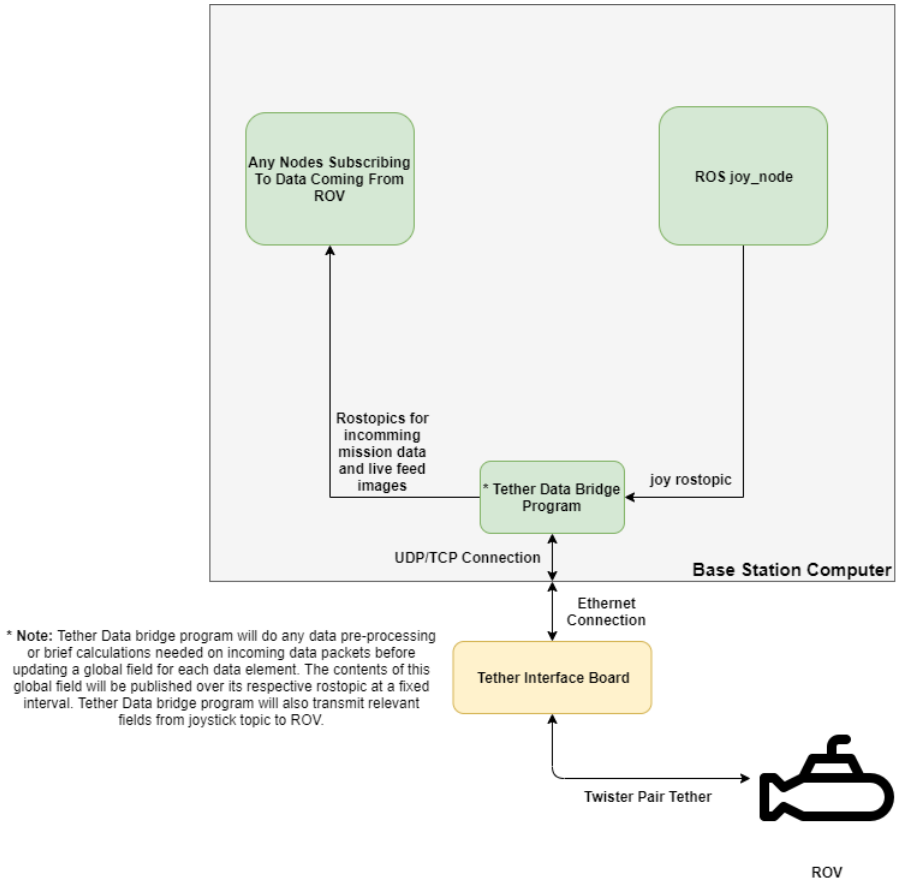


Figure 3.8: Software Interfaces For The Base Station Communications Manager

Figure 3.8 in this section depicts the base station communications manager software interface diagram. All data coming up the tether from the ROV is distributed to the rest of the system via rostopics through the tether data bridge program node. Data from the Xbox controller will be sent to the ROV via the joy rostopic through the tether data bridge program. Dive start and stop commands collected in the front end HUD interface seen in figure 3.1 will be sent to the ROV through the back end via the tether data bridge program node.

### 3.3.3 Stimulus/Response Sequences

This section outlines the Teensy micro-controller stimulus state diagram. This diagram outlines the looping structure that the micro-controller will perform at a fixed rate to ensure all data collected will be able to be retrieved by the rest of the system, and that they system can actuate the lighting controller and motor controllers in a responsive manner. Figure 3.9 in this section outlines this state diagram.

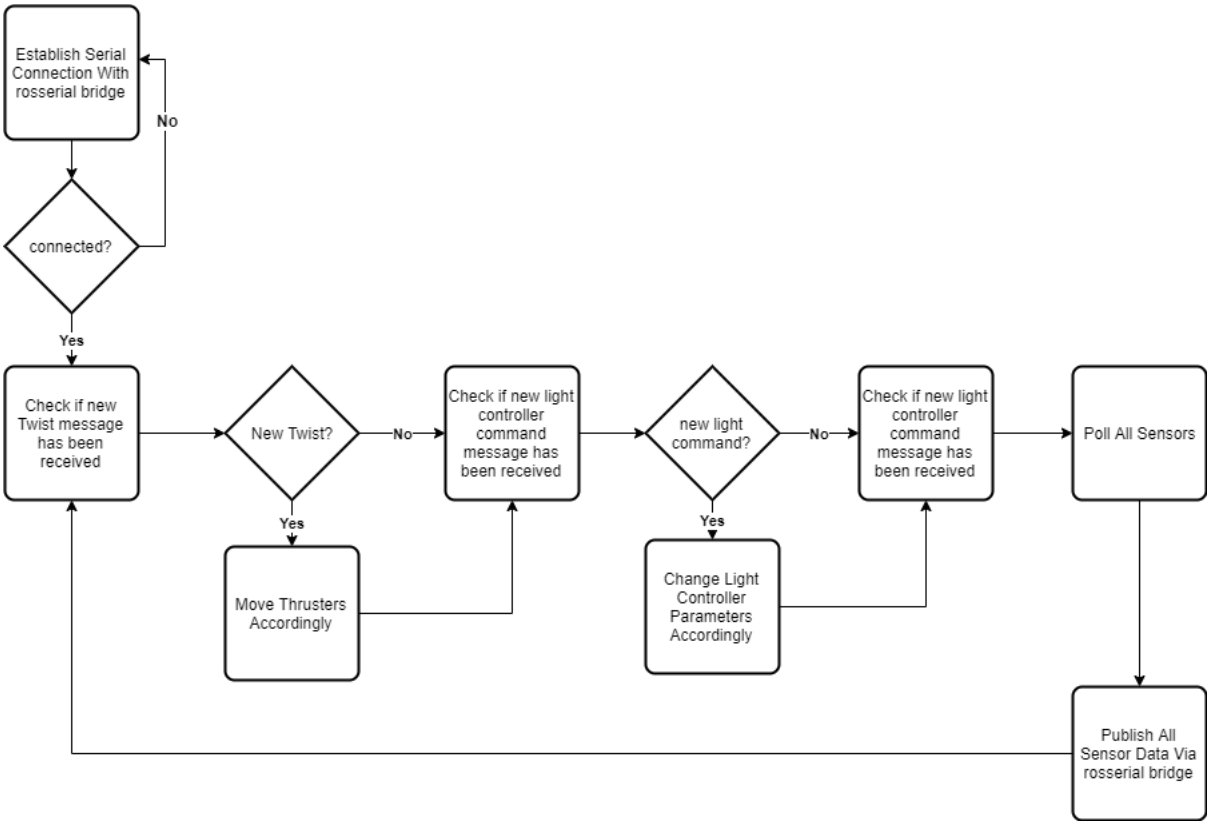


Figure 3.9: Teensy Micro-Controller State Diagram