Oregon State University
College of Engineering

CS Capstone Design Document

May 11, 2020

An underwater ROV for exploring the uncharted ice-ocean boundary

Prepared for

Oregon State University

Jonathan Nash
Signature Date

Prepared by

Group48
The Iceberg Explorer

Coulby Nguyen
Signature Date

Michael Huang
Signature Date

Isaac Peters
Signature Date

Abstract

The border between glacier calving and ocean is too dangerous for human exploration. The goal of this project is to break this barrier, so that we can measure the physics of glacial melt directly against the ice surface, underwater. We will use a Remotely operated vehicle to achieve this goal. This paper describes our hardware/software design choices to achieve network communication, ROV control, Videofeed, and topology acquisition.

Contents

1 Project Overview

1.1 Introduction

This project will focuses on expanding an already existing proposal by our client. Their central goal is to create new methods of measuring active glacial melt in the arctic. The final product this project should provide is an ROV capable of traveling 30m underwater in the Arctic ocean and be able to scan a timelapse of the changing topography of a melting glacier. To deliver this, chose the following process/hardware that are expected to function

- BlueROV2 - An Open-Source ROV capable of traveling underwater
- Autonomous Kayak - An already-built boat connected to our ROV, it is capable of automatic movement
- User Control Station (UCS) - A ship holding the users, it has a computer and a wireless receiver
- Wireless Communication - All data from the ROV will be sent to the UCS over a wireless network
- Active Videofeed - A camera livestream of the ROV's front-facing perspective
- Open Source Software - Controls the ROV remotely through controller commands
- Structural Analysis Sensor - Either a LIDAR or photogrammetric system to determine the structure of the glacier wall and provide detailed structural changes over time
- Depth Sensors - Used for giving the user more information on distance from destination, as well as being used in part with the mapping of the topography

1.2 Intended Audience

This document is meant primarily for the software developers, collaborating engineering teams, course instructors, client, and anyone wishing to continue the project beyond its original goals. It is to be treated as the front-facing document from the software development team and what is expected to be completed by the project's deadline. Users who read this document will find the high-level description of the project itself with detailed explanations of each design choice and how it functions within the project's entirety.

1.3 Definitions

ROV: Remotely Operated Vehicle

Calving: (of an iceberg or glacier) split and shed (a smaller mass of ice)

Photogammetry: the science of making reliable measurements by the use of photographs

UCS - Abbreviation for User Control Station, the computer setup for the users controlling the ROV

Light-Sheet: A grid used to create a topography of a 3d object.

Application Programming Interface (API): A set of functions designed to interface with the program's design features

1.4 UML/Flowchart

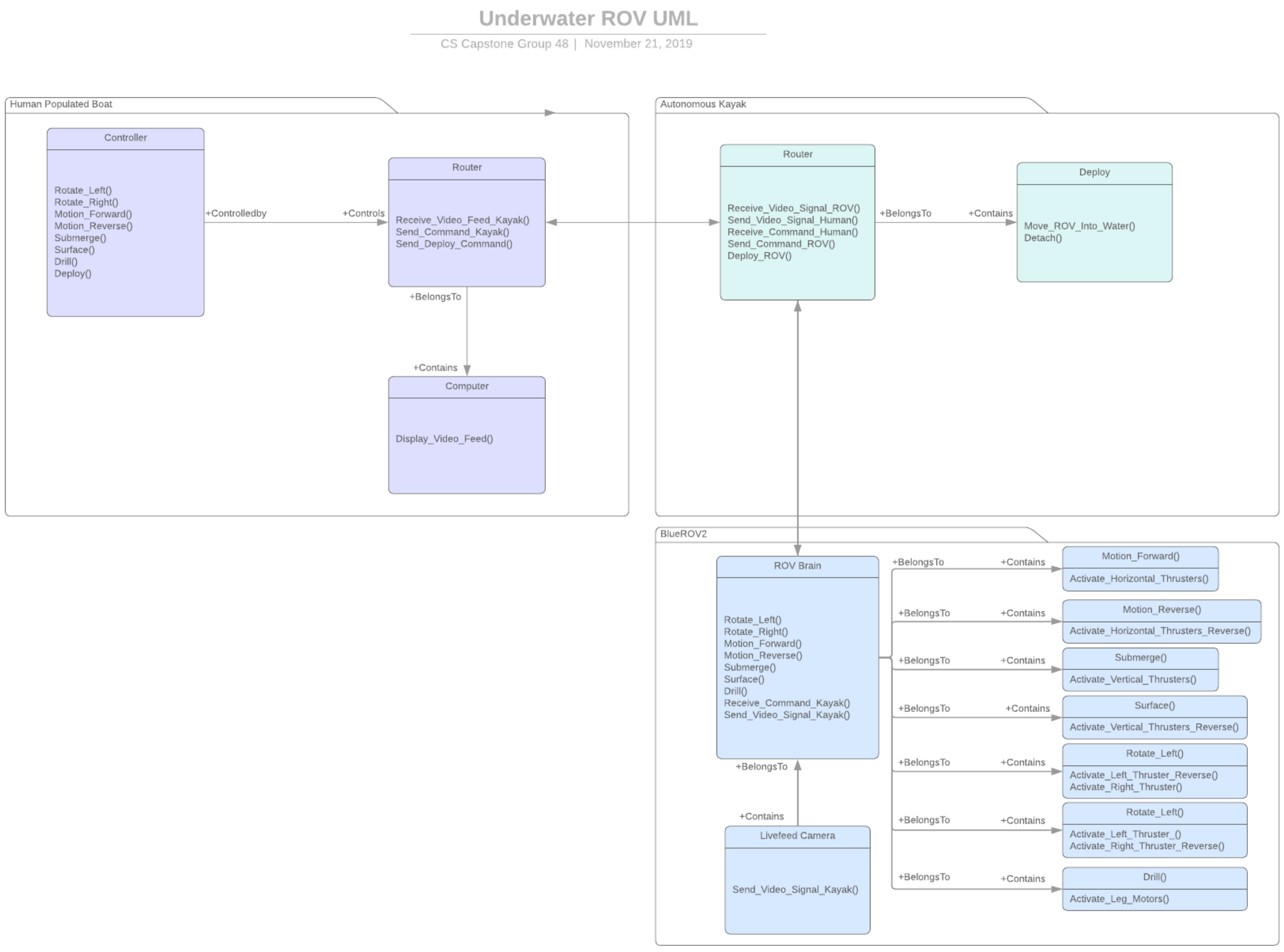

Fig. 1. The relationship between controller, autonomous kayak, and the ROV

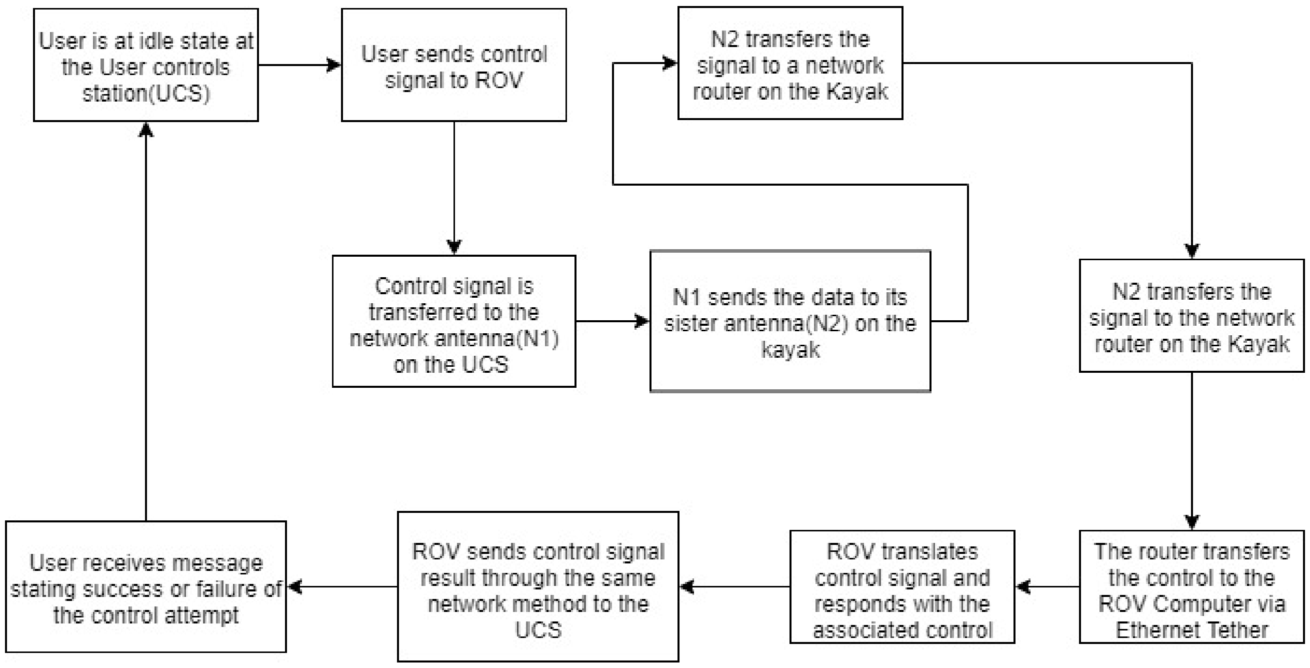
User is at idle state at the User controls station(UCS)
User sends control signal to ROV
Control signal is transferred to the network antenna(N1) on the UCS
N1 sends the data to its sister antenna(N2) on the kayak
N2 transfers the signal to a network router on the Kayak
N2 transfers the signal to the network router on the Kayak
The router transfers the control to the ROV Computer via Ethernet Tether
ROV translates control signal and responds with the associated control
ROV sends control signal result through the same network method to the UCS
User receives message stating success or failure of the control attempt

Fig. 2. Event flowchart of the system

1.5 Document Changes

This document has been revised to the current status of the project, these are the current changes.

- System Data has been updated to reflect the relevant data, LIDAR/Depth Sensor data has been removed, Camera information has been updated.

2 System Data

This is the data expected to exist during the testing and active use of the whole system, all data is to be passed using ethernet and/or a wireless network.

- Camera - H264 Codec: H264 stored and sent directly via BlueROV2's native camera
- Control Signal - IP Packet: Digital command sent to the ROV for controlling one motion on the system

3 Software Control of the ROV

The control of the ROV should be relatively simple as complicated controls can result in a lost product or unintended actions that can cause adverse effects such as draining battery life. Since the hardware uses a Raspberry Pi 3, which has1 gigabyte of RAM, the more straight forward the logic, the better. With less complex logic the robot can perform autonomous macro instructions from deployment until it reaches the face of the iceberg or if sent off course.

3.1 Implementation

The implementation of this piece will be modifying the official BlueROV2 software. This software has pre-defined and coded controller options that can be compatible with remote controllers such as the Xbox One wireless remote controller. This implementation already takes care of basic movement in underwater environments and navigation. However, modifications need to be made in order to instruct the legs to successfully mount and latch onto the face of the iceberg.

The modification of this piece should be straight forward as there already exists two sets of BlueROV software. The first package is a Windows/Linux/Mac executable installer that provides a graphic user interface (GUI) as well as a better user experience. However looking into this option, it seems that it is much harder to edit the source code. The second option is that the BlueROV have published their source code for a Robot Operating System (ROS) framework on github. This would mean that it would be very easy to modify and add the changes required to control our hardware modifications, but would require the user to be more skilled at controlling the software without a GUI. Initially the project will take the GUI option, and if there is no success, it can be quickly changed to the open source alternative.

3.2 Blockers and Obstacles

For this piece of the project there are no blockers or obstacles that would delay initial implementation. However, for this piece to be fully completed, the hardware modifications must be completed entirely as the hardware will need to be controlled through this software.

3.3 Testing

Testing for software will have 3 stages

Stage 1 Testing Build

- Install BlueROV2 software and packages on testing computer
- Import additional libraries that are specific to this project
- Test compilation

Stage 2 Testing Motion Underwater

- Test submerge
- Test surfacing
- Test rotation 90 degrees left
- Test rotation 90 degrees right
- Test motion forward 1 meter
- Test motion reverse 1 meter

Stage 3 Testing Mount

- Move ROV towards imitation block of ice
- Activate mount motors
- Test to check penetration results

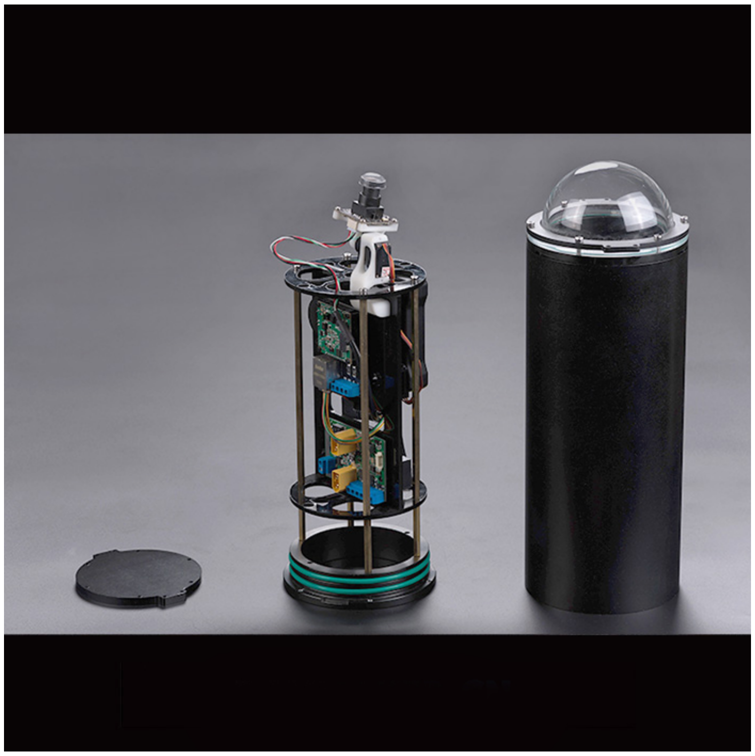

Fig. 3. Disassembled Water Tight BlueROV Capsule complete with Raspberry Pi **[RovControlSystem]**

4 Active Videofeed / Cameras

Active camerafeed from the ROV is required for the user's ability to control the ROV. This feed must be passed from the ROV to our UCS.

4.1 Functional Requirements

All devices, regardless of choice, must be able to fulfill the noted requirements

- Provide a clear videofeed in an underwater environment
- The cameras must be able to survive pressures of below 50m and be waterproof
- The videofeed must be able to be passed wirelessly
- The feed must be archived and stored in .mp4 format
- Feed must be viewable in low-light conditions
- Full setup should run on low power to reduce energy cost

4.2 Implementation

We have chosen to use a serial camera for our videofeed. Serial is better equipped to handle the electrolyte pollution noted in saltwater. This serial camera is to be connected in front of our ROV. If it is not innately waterproof, a waterproof casing will be devised to hold the device and its wires.

The serial output is to then be put through and IP encoder, this will allow the analog data to be connected to as if it were an IP camera. Note, serial is **not** capable of transmitting through Ethernet. This encoder must be placed on the ROV and be within its own underwater casing. Once converted, the data can be wireless transmitted to our UCS

The UCS will receive raw H264 data from the IP encoder, we will have software capable of turning that H264 into a live format and implement it into our API. This will be written as a Python script using the SHRLEX library.

5 LIDAR / PHOTOGAMMETRY

In order to study the physical effects that are causing melting on the glacial surface, we will create a high detail topographical map of the glacier over time as it is melting, so that we can connect the actual melt of the glacier with any other data that we get.

5.1 Functional Requirements

The provided solution must fulfill the following requirements:

- Provide a clear enough map of the glacier surface that we can track melting over time; sub-centimeter accuracy would be preferred
- Provide enough time resolution that we can see a clear progression over time; taking a scan every minute would provide this
- Must be small enough to physically attach to our ROV while maintaining full mobility
- Data generated must be transmittable back to the user of the ROV

5.2 Implementation

The current choice is to use ”Agisoft”. This program is capable of using a set of images and creating a 3D model. We plan on using OpenCV alongside three cameras to record three videos. From there, we’ll take a set of images from each video and combine them to create a working model. We will then compare the models provided to us by Agisoft to display the melting surface.

Another option that we have considered is buying an industry solution that would be easy to add, and would provide higher detail than a photogrammetric approach would provide. Our team is working with a company that creates a wide range laser scanner (LIDAR) that is equipped to work underwater; this solution would save us time developing an photogrammetric algorithm to generate topography, but is expensive and most likely outside of our budget. If we can create a working prototype with a photogrammetric approach, we may switch to a LIDAR solution if we can find funding for it.

5.3 Testing

Stage 1

- Begin live videofeed, take feeds of varying lengths both in literal distance and in video length. Test for latency

Stage 2

- Repeat stage 1 except transmit the data wirelessly

Stage 3

- Repeat stage 1 and 2, now connect the devices to the ROV and test in underwater conditions

6 Network Communication

The ROV must be able to communicate with the user control station, this includes sending controls to the ROV and receiving live videofeed from the ROV itself. The distance we are expected to be from the ROV is about 150 meters, the wireless connection must be strong enough to cover at least 20% above that minimum expectation and provide a high-quality signal for data transfer. We've chosen to pass our data through an Ethernet tether that connects an autonomous kayak to our system, this means all data must be translated to an Ethernet viable form.

6.1 Functional Requirements

Note that while we've chosen to go with a specific method, if we were to switch to another device, it would still fulfill the requirements noted below.

- The wireless network must be capable of providing a strong, low latency ¡10ms, connection for a minimum of 150 meters.
- All data transferred to and from the ROV must be in a usable or otherwise convertible format, this includes assuring that all devices connected to the ROV have correctly formatted data.
- The transceiver and receiver must be able to connect regardless of direction
- All devices purchased must remain below the total $15,000 cost of the ROV.

6.2 Implementation

We've chosen to use two Ubiquiti M2s, these are two network devices capable of interfacing with each other over a 1km distance while outputting a network we can connect to. The transceiver on the kayak is to be connected to a separate router on the kayak, the router will then receive the Ethernet connection from the ROV. This data will be sent wirelessly from the transceiver to the receiver mounted at the user station. The user station will have a computer connected to this network and will be actively translating this data into a usable format, this includes active videofeed, sensor data, and other modules to be included.

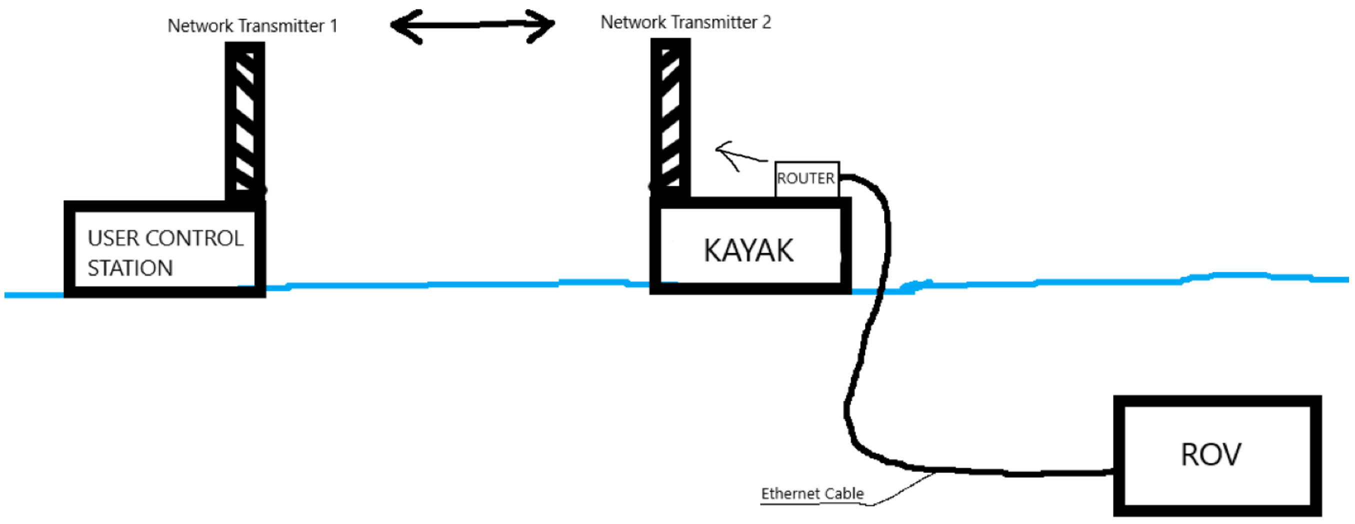

Fig. 4. Gantt Chart

6.3 Testing

We will test the devices in multiple stages.

Stage 1

- Connect the receiver and transceiver in total with the minimum distance of 150m
- Connect the router to the transceiver and the Ethernet tether to the ROV
- Move the tether at max range on a dry surface, we should expect consistent connection between the ROV and router
- Place the tether and ROV into saltwater, record any differences in connection

Stage 2

- Disconnect and remove the ROV from the testing environment
- Setup the UCS and the kayak, now replace the Ethernet tether with a Ethernet cable and connect it to a separate computer
- Pass data between the UCS and the computer through the two network antennas. Verify that the passed data is correct and make note of the latency.

Stage 3

- Repeat stage 2 setup
- Using software that has converted our data into Ethernet-viable form, send that to the other device (i.e. camera feed)
- Verify data structure, check for latency

Stage 4

- Repeat stage 3 with the full setup, now including the ROV.
- Now, repeat stage 3 with the full setup, but now submerge the ROV in water.

7 System Outlook

7.1 Timeline - Gantt Chart

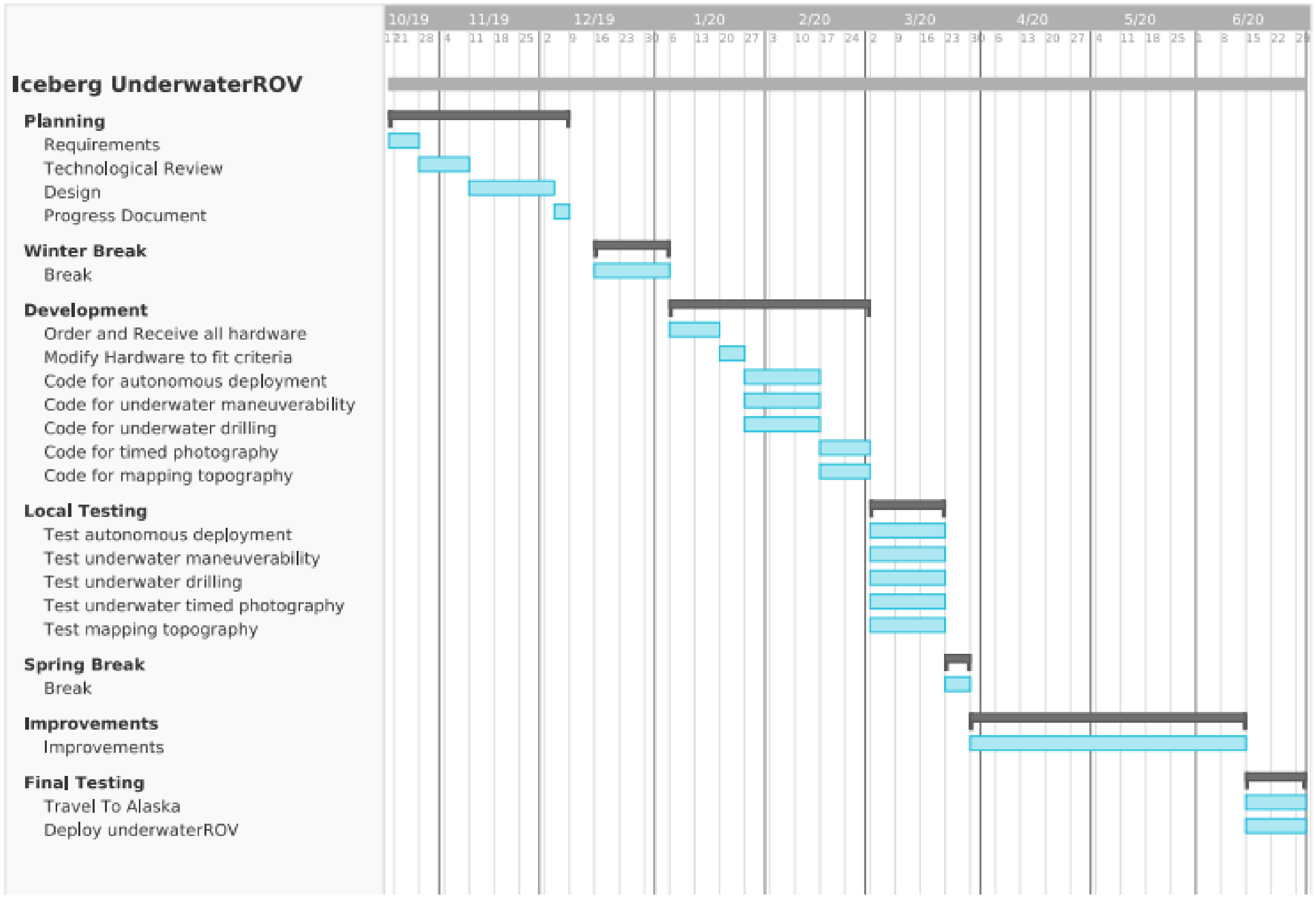

Fig. 5. Gantt Chart

7.2 System Testing

This testing is dedicated to testing the whole system, there will be redundancy with testing the individual pieces detailed in the earlier sections of the report.

Stage 1 - Above Ground

- Prepare kayak and ROV by connecting all necessary peripherals (antennas, router, ethernet tether etc)
- Test all motion controls, keeping track of latency
- Test Active Videofeed, keep track of latency, fps, and resolution of the camera
- Test topography method by scanning a flat wall, curved wall, tree bark, and melting ice
- Continuously increase the distance between the UCS, kayak, and ROV to test if the distance causes anomalies.

Stage 2 - Submerged

- Submerge the ROV, and repeat the stage 1 test cases

Acceptable Output: 0-40ms Latency, Active Videofeed with a 24fps minimum, 95% Network uptime, no control failures

8 Major Milestones

The major milestones of the project are as follows, each have varying degrees of dependency

- Fully accessible 200 meter+ omnidirectional wireless network akin to a home network
- 720p, 24fps minimum live camera feed from the ROV with 0 - 40ms latency
- Complete, easy to understand controls for all functions of the ROV
- Tool(s) capable of recording and modeling the topology of a changing surface

9 Conclusion

Once completed, the Underwater ROV provides reliable, highly accessible, and when properly coded, a salable method of measuring any underwater topographical structure, be it ice, rock, or many other surfaces. Through the use of depth sensors, large scale networks, and active camera feeds, the system allows researchers and users to do their work without endangerment.

References

[1] ROV Openrov Diy Kit Control System Electronic Cabin Ardusub Sealed Cabin Aluminum Full Set for BlueRov Remote Operated Vehicle, `https://www.aliexpress.com/i/32914530243.html`

[2] Direct observations of submarine melt and subsurface geometry at a tidewater glacier Sutherland, D., Jackson, R., Kienholz, C., Amundson, J., Dryer, W., Duncan, D., Eidam, E., Motyka, R. and Nash, J. (2019). *Science, 365(6451), pp.369-374.*

[3] An overview of using directional antennas in wireless networks. International Journal of Communication Dai, Hong-Ning & Ng, Kam-Wing & Li, Minglu & Wu, Min-You. (2013). Systems. 26. 413-448. 10.1002/dac.13481.