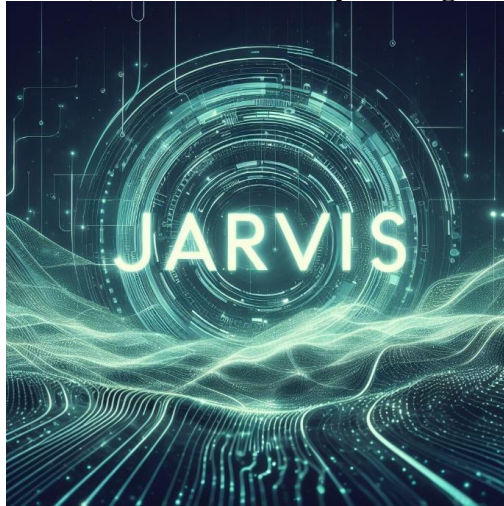


J.A.R.V.I.S. (Just a Rather Very Intelligent System)



Team 24 – Social Navigating Robot

Final Design Report

November 29th, 2023

Team Members:

Ben Pudlowski – bmpu223@uky.edu

Bryce Grant – bryce.grant@uky.edu

Erick Calvillo – erick.calvillo@uky.edu

Hasib Hasan – hasib.hasan@uky.edu

Instructor Team:

Dr. Regina Hannemann – regina.hannemann@uky.edu

Dr. Samson Cheung – sen-ching.cheung@uky.edu

Dr. Robert Adams – robert.adams@uky.edu

Daniel Muchow – daniel.muchow@uky.edu

Prakash Dhungana --

Santiago Posso Murillo – spo230@uky.edu

Project Sponsor:

Dr. Samson Cheung

Table of Contents

1. Abstract	7
2. Problem Statement	7
2.1 Need	7
2.2 Objective	7
2.3 Background	8
3. Design Requirements	9
3.1 Marketing Requirements	9
3.1.1 Changes to Marketing Requirements	9
3.2 Objective Tree	9
3.3 Engineering Requirements	10
3.3.1 Changes to Engineering Requirements	13
3.4 Design Impact	14
3.4.1 Standards	14
3.4.2 Economic	14
3.4.3 Social	15
3.4.4 Ethical	15
3.4.5 Health and Safety	15
4. Design Summary	16
4.1. Design Overview	16
4.2. Software Overview	19
4.3 User Experience Overview	22
5. Functional Decomposition	24
5.1. Level 0	24
5.2. Level 1	25
5.2.1. JARVIS	25
5.2.2. Home Base	27
5.3. Level 2	28
5.3.1. Robot	28
5.3.2. Remote Interface	31
5.4. Level 3	33
5.4.1. Robot Communication Input	33
5.4.2. Robot Communication Output	35
5.4.3. Motors and Encoders	36
5.4.4. Navigation Systems	37
5.4.5. Power Delivery	39

5.4.6. Computers/Microcontrollers.....	41
5.4.7. User Controls	43
6. Subsystem Integration.....	44
6.1. Subsystem Assignments	44
6.2. Navigation Systems	44
6.2.2. Testing	47
6.3. Hardware Communications.....	48
6.3.1. Summary.....	48
6.3.2. Testing	50
6.4. Remote User Interface.....	52
6.4.1 Summary.....	52
6.5. Home Base and Power Delivery	58
6.5.1. Summary Home Base	58
6.5.2 Summary Power Delivery	62
Testing 6.5.3.	63
7. Presentation of Final Design	64
7.1. Robot Structure	64
7.2. Home Base Structure.....	68
7.3. Software	70
8. Testing and Validation	71
8.1. Marketing Requirements	71
8.2. Engineering Requirements	72
8.3.Requirements Summary	74
9. Final Project Plan	75
9.1. Work Breakdown Structure.....	75
9.2. Gantt Chart	78
9.3. Final Costs.....	78
9.4. Final Team Member Contributions	80
10. Final Thoughts	81
10.1. Future Recommendations.....	81
10.2. Reflection	82
10.3. Conclusion.....	82
11. References.....	82

Appendix A - Analytical Hierarchy Process for Objective Tree Weights	83
Appendix B – Previously Completed Engineering Requirements	85

List of Figures

Figure 1 : Social Navigating Robot Objective Tree.....	10
Figure 2 - Hardware Design Overview	17
Figure 3 : Final Block Diagram	18
Figure 4 : Main Software Thread Diagram	19
Figure 5 : User Input Thread Diagram.....	20
Figure 6 : Lidar Software Thread Diagram.....	21
Figure 7 : Zoom Meeting Landing Page	23
Figure 8: J.A.R.V.I.S. Online UI	23
Figure 9 : Level 0 Functional Decomposition	24
Figure 10 : Level 1 Functional Decomposition	25
Figure 11: Home Base Level 1 Functional Decomposition	27
Figure 12 : Level 2 Functional Decomposition - Robot	28
Figure 13 : Level 2 Functional Decomposition - Remote Interface	31
Figure 14 : Level 3 Robot Communication Input Functional Decomposition	33
Figure 15 : Level 3 Robot Communication Output Functional Decomposition.....	35
Figure 16 : Level 3 Motors and Encoders Functional Decomposition	36
Figure 17 : Level 3 Navigation Systems Functional Decomposition	37
Figure 18 : Power Delivery Functional Decomposition	39
Figure 19 : Level 3 Computers/Microcontrollers Functional Decomposition	41
Figure 20: Level 3 User Controls Module	43
Figure 21: Lidar Mapping Code	45
Figure 22: Motor Commands Code	45
Figure 23: Lidar 2D Point Cloud	45
Figure 24: 3D Lidar Design	47
Figure 25: Voltage Divider Circuit for Battery Display	49
Figure 26 : Streaming LED Indicator Code	50
Figure 27 - Remote Landing Page	52
Figure 28 - Navigation Commands within cdn.html	53
Figure 29 - WebSocket Initialization	53
Figure 30 - Battery Percentage Code	54
Figure 31 - isOpen() and wsSend() Functions	55
Figure 32 - Navigation and Camera Commands.....	55
Figure 33 - Changing Robot's Speed Commands	56
Figure 34 - Autonomous and Manual Navigation Functions.....	57
Figure 35: Front Side of Charging Station.....	58
Figure 36:Front and Back View of Connection Point	59
Figure 37: Flange of 3D Print (left)and Limit Switch (right)	59
Figure 38: Home Base Metal Bar Connection Point	59

Figure 39: Micro Limit Switch	60
Figure 40: Back Plate of Home Base Connection Point	60
Figure 41: Robot Base Connection Part.....	61
Figure 42: 3D Printed Plastic Part Robot Side.....	61
Figure 43: Top and Front View of Robot Connection Point	61
Figure 44: Front Side Charging Station	62
Figure 45: Wire Connection Home Base	64
Figure 46 - Final Robot Structure	65
Figure 47 - USB Hub	66
Figure 48 - Robot Charging Leads.....	67
Figure 49 - Camera Mount Structure	68
Figure 50: Home Base Setup Back View	69
Figure 51 - Home Base Charging Leads	70
Figure 52 - Launching the Server	71
Figure 53 : Gantt Chart	78

List of Tables

Table 1 : Engineering Requirements for J.A.R.V.I.S	10
Table 2 : J.A.R.V.I.S. Functionality	24
Table 3 : Robot Functionality	26
Table 4 : Remote Interface Functionality	26
Table 5 : Power Delivery Functionality.....	28
Table 6 : Motors/Encoders Functionality	29
Table 7 : Audio/Visual Sensors Functionality	29
Table 8 : Navigation Systems Functionality	29
Table 9 : Remote User A/V Output Functionality	30
Table 10 : Computer/Microcontroller Functionality.....	30
Table 11 : Web Server Robot Side Functionality	31
Table 12 : User Controls Functionality.....	32
Table 14 : Robot A/V Output Functionality	32
Table 15 : Robot Information Functionality	32
Table 16 : Web Server Remote User Side Functionality	33
Table 17 : 360-Degree Camera Functionality.....	34
Table 18 : Microphone On The Robot.....	34
Table 19 : Computers/Microcontrollers Functionality	34
Table 20 : Computer/Microcontrollers Functionality	35
Table 21 : Speakers on The Robot Functionality.....	35
Table 22 : Display on The Robot.....	36
Table 23 : Computer/Microcontrollers Functionality	36
Table 24 : Motors/Encoders Functionality	37
Table 25 : Wheel Motors Functionality	37
Table 26 : Wheel Motors Functionality	37
Table 27: Lidar Sensors Functionality	38
Table 28: Computer/Microcontrollers Functionality	38
Table 31 : AC to DC Converter Functionality.....	40
Table 32 : Battery Charge Functionality	40

Table 33 : DC to DC Converter Functionality	40
Table 34: USB Hub Functionality	40
Table 35 : Video/Audio Processing Unit Functionality.....	42
Table 36 : Web Server Functionality	42
Table 37 : Navigation Controller Functionality	42
Table 38 : Secondary Microcontroller (Object Avoidance) Functionality	43
Table 39 : Battery Charge Reader Test Results	51
Table 40 : DROK 48V VS MEAN WELL Power Supply.....	62
Table 41 : Work Breakdown Structure	75
Table 42 : Original Bill of Materials/Inherited Parts	79
Table 43 : Purchased Parts	80
Table 44 : Decision Matrix for General Categories	83
Table 45 : Decision Matrix for Human Interaction	84
Table 46 : Decision Matrix for Interface/Connectivity.....	84
Table 47 : Decision Matrix for Safety/Privacy	84
Table 48 : Decision Matrix for Product	85
Table 49 : Previously Completed Engineering Requirements	86
Table 50 : Decision Matrix for Power Supply	90
Table 51 : Safety Matrix for Power Supply	91
Table 52 : Reliability Matrix for Power Supply	91
Table 53 : Affordability for Power Supply	91
Table 54 : AHP for 360-Degree Camera	92
Table 55 : AHP for Camera Cost.....	92
Table 56 : AHP for Camera Quality	92
Table 57 : AHP for Additional Camera Features.....	92
Table 58 : AHP for Camera Size	92
Table 59 : Final Decision Matrix for Camera.....	93
Table 60: Object Detection Matrix	93
Table 61: Sensor Type Final Priority Matrix	93
Table 62: Lidar Sensor Final Priority	93
Table 63: IMU Final Priority Matrix	94
Table 64: BLE Beacon Parameter Final Priority	94
Table 65: BLE Beacon Final Priority	94
Table 66 : Software AHP Matrix (Un-Normalized)	94
Table 67 : Software AHP (Normalized)	95
Table 68 : Final Software AHP Matrix.....	95

1. Abstract

J.A.R.V.I.S. (Just a Rather Very Intelligent System) has been tasked with redesigning an existing social navigating robot to provide an immersive remote experience for users unable to attend events in-person. This report is an overview of the system design as we prepare the project for Senior Design Day at the end of the semester. This report builds upon the previously completed Requirements Report and Preliminary Design Report, to both address the past comments instructors made on those reports, as well as updating the report to include our progress on design, implementation, and integration. The purpose of this report is to inform the reader on where we currently stand with the project, as well as what future work is planned and needs to be completed to have a successful Senior Design Presentation and Report at the end of the semester. This report will give an extensive overview of the entire project, from the need and objective that was initially established in coordination with our sponsor, an update on design changes that had to be made since our last report, our progress on each of the subsystems and integration, as well as the general information of the project like an updated Work Breakdown Structure and Bill of Materials.

2. Problem Statement

In this section, we will be providing some information on what problem our project will solve. We will start with a short paragraph describing the need for our project. The need was discussed in collaboration with our sponsor. In section 2.2, we will discuss the goals of this project and what our team intends on accomplishing. The end of this section will be introducing some basic background information on our project. This will contain information on robots with similar designs, as well as a short description of the first version of our project.

2.1 Need

Although we are no longer in the middle of the COVID-19 pandemic, there are still lasting effects on our society because of it. One of these effects is the increased number of Americans working from home. According to the U.S. Census Bureau, the number of Americans who primarily work from home more than tripled from 5.7% in 2019 to 17.9% in 2021 [1]. The use of a telepresence robot can promote social interaction between those who choose to, or are forced to work remotely, and those who they would regularly communicate with had they been able to come into the office physically. The sponsor of our project, Dr. Samson Cheung, is one of those individuals who now works remotely. He wishes for us to improve upon a telepresence robot which a previous group worked on. He would like us to create an affordable alternative that he can use while he is away from the University of Kentucky. Being able to use our robot would allow Dr. Cheung and other individuals like him the opportunity to communicate with students and coworkers in ways that he would not be able to do otherwise. Currently, individuals who work remotely can communicate with people via email or Zoom meetings but are unable to “walk through the halls” and have casual conversations with those around them. An affordable telepresence robot will allow these individuals to increase their relationships with the people they would normally interact with.

2.2 Objective

Our team's objective is to improve the social navigating robot built by the previous team. We are focusing on two main general improvements from the first version. First, the robot shall

have a more immersive viewing of environments through real time video streaming by using a 360-degree camera. Our second focus is improving the autonomy of the robot. The robot should be able to navigate back to a home base when not in use or it needs to recharge. Although these are our two focus points, we also want to improve some of the robot's other abilities. We plan to implement a better navigation system by introducing object avoidance. We also plan to improve the queue system, and the remote user interface. Meeting all these goals would allow the remote user to have a more improved experience, which we plan to display with our final prototype.

2.3 Background

J.A.R.V.I.S. or "Just a Rather Very Intelligent System," is a social navigating robot which aims to provide an immersive experience for remote users by standing in for them at meetings or events. J.A.R.V.I.S. is the second iteration of the social navigating robot R.O.B.B., built by students at the University of Kentucky in 2022. J.A.R.V.I.S. provides further innovation to the previous project with a myriad of process improvements and automation features which are further discussed in the engineering requirements in section 3.3, Table 1.

There are a litany of telepresence robots spanning multiple industries. PadBot, a Chinese company specializing in producing high quality telepresence robots, provides insight into a telepresence robot that can be used in many situations with an emphasis on customer facing roles and environments [2]. The PadBot 3, a reception robot, has a human-like appearance with a two-wheel shockproof base. PadBot 3's microphone array, inbuilt woofers, HD camera, and 4G LTE compatibility allow for high quality video communication with the option for voice command input. Additional features PadBot 3 has that we plan on implementing within J.A.R.V.I.S. include automatic obstacle detection and a self-charging system. The robot weighs under 40 pounds, has a continuous running time of 10 hours with a standby time of 60 hours. PadBot 3 offers a plethora of options for its users but is extremely expensive at \$5,875. Many of the features included in PadBot 3, such as dancing, singing and voice commands, are redundant for our situation and its egregious price makes the product unfeasible for the University needs. Additionally, with PadBot 3 measuring in at 3.25 feet, its 30° vertical camera range would require most users to have to look down when communicating with the robot.

VGo Robotic Telepresence [3], is an American company developing automated robotic solutions for healthcare, business, development, education and manufacturing applications [4]. VGo has a 6" LCD to display a person's face, a video streaming indicator, embedded Wi-Fi, auxiliary lights for use in dark places as well as a microphone array allowing for seamless capturing and presentation of audio and video. Additionally, the robot has a 12-hour battery life, lightweight build and self-charging capability, but these do not justify its \$4,875 price tag. This coupled with its poorly designed user interface and its lack of self-navigation and object detection makes this product unfeasible for the needs of the university.

We will keep the design of these similar robots in the back of our minds, but the most important robot to consider while we are designing and developing this robot is clearly the first version prototype that was given to us. This robot had basic functionality with lots of room for improvement. As we received it, the robot had the ability for the user to connect to a remote interface which was connected to the main computing unit on the robot. This allowed the user to use the robot's basic capabilities like the navigation of the robot. The robot also was able to provide two-way audio and visual communication between the remote user's display and microphone and those of the robot. This was essentially the extent of the robot's abilities. Although the movement works, none of the sensors on the robot were working, along with some

other problems. Based on the limited ability of the robot, we will implement the changes that will be detailed in this report to improve the experience of the user.

3. Design Requirements

In this section, we are going to start by detailing the marketing requirements which our sponsor expects the robot to satisfy. Then, we are going to show our objective tree, which shows the importance which we think each marketing requirement has in relation to the others. After that, we will discuss the engineering requirements which will need to be satisfied to satisfy the marketing requirements, and we will end the section with a brief discussion on the various impacts J.A.R.V.I.S. will have, as well as the impact that several outside factors will have on our final design.

3.1 Marketing Requirements

After discussing our project with our sponsor, Dr. Cheung, we were able to come up with the list of marketing requirements below. These requirements outline the basic functions of the robot, as well as introduce some improvements from the previous version of the robot. The changes to our original Marketing Requirements and the justifications for the changes can be found in the next section, 3.1.1. These were then used as a backbone for our engineering requirements, which will be discussed later in this report.

1. Robot shall be able to clearly communicate with the individuals surrounding it.
2. Robot shall display a real-time 3D viewing of its surroundings to the remote user.
3. Robot shall be able to automatically avoid any obstructions or hazards that are in its path.
4. Remote user shall be able to control the movement of the robot through an online user interface.
5. Robot shall pose no safety risks to its surroundings.
6. Robot shall have a 3-hour long battery charge.
7. Robot shall clearly indicate when video streaming is in progress.
8. Robot should have a user-friendly interface for setup and user operation.
9. Robot shall implement a queuing system for users to wait for remote access.
10. Robot should have a practical design and appearance.
11. Robot should have manufacturing costs that are competitive with similar products.
12. Robot shall have designated charging location that automatically charges robot.

3.1.1 Changes to Marketing Requirements

Due to obstacles encountered since we turned in our Preliminary Design Report, we have had to make changes to our list of Marketing Requirements. One Marketing Requirement we had to remove since our last report was “Robot shall be able to automatically return to a designated charging location.” It was determined we would not have sufficient time to implement complete autonomy in this aspect. Lastly, we changed the requirement “Robot should be able to automatically recharge from a designated charging location” into what is now our 12th Marketing Requirement. The updated version is clearer on our intentions with the home base and its functionality.

3.2 Objective Tree

The objective tree is shown in Figure 1. This was developed using our marketing requirements, decomposing them into four sections: Human Interaction, Interface/Connectivity,

Safety/Privacy and Product. The categories were then put into our Analytical Hierarchy Process (AHP) matrix where we performed calculations to determine the priority of each category and their respective subcategories, which can be seen in Appendix A. The number corresponds to the priority of each of the categories in accordance with their significance as determined by our team and our sponsor.

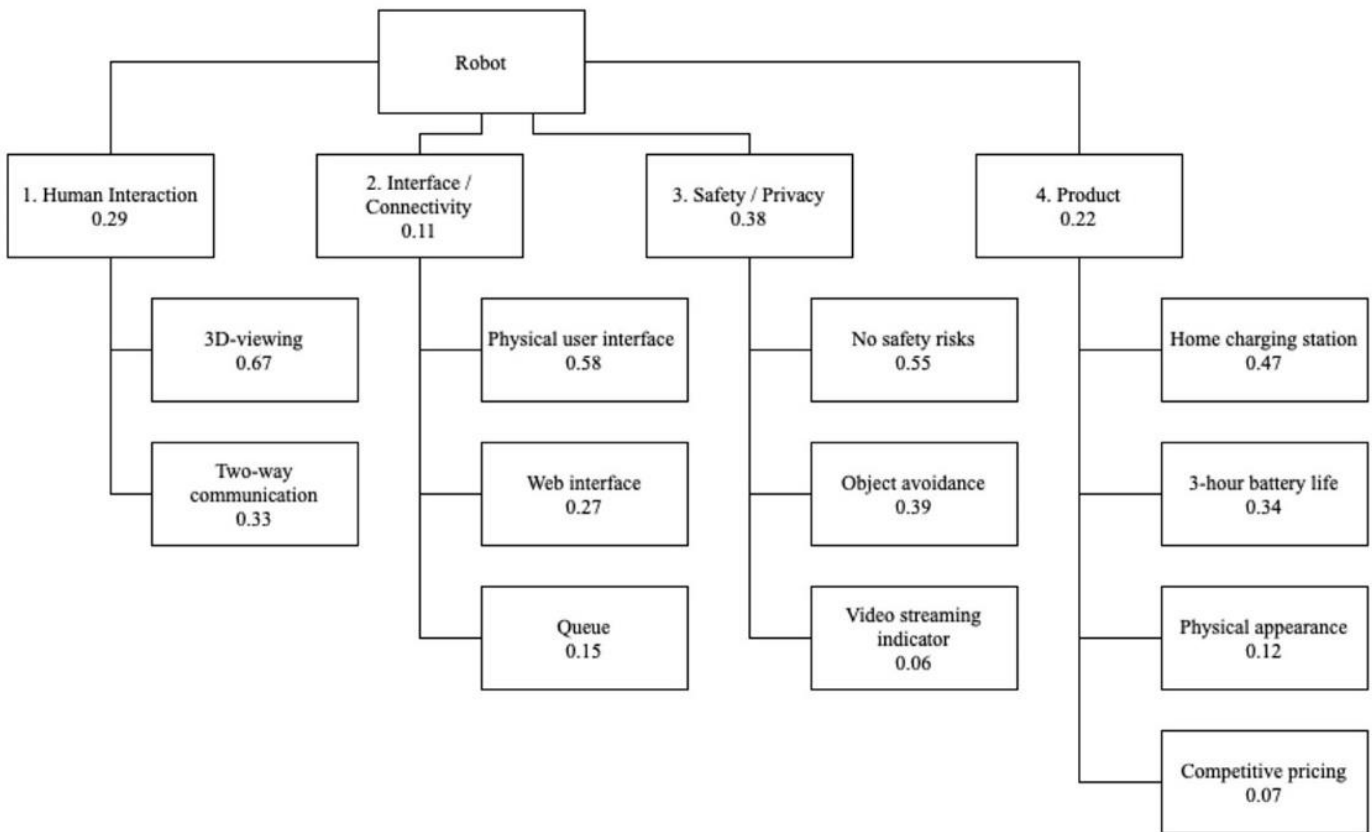


Figure 1 : Social Navigating Robot Objective Tree

3.3 Engineering Requirements

We have listed our engineering requirements in Table 1. The purpose of these requirements is to satisfy the marketing requirements listed in section 3.1. This table offers a numbered list of engineering requirements, the corresponding number of marketing requirements that were satisfied in accordance with that number, a brief explanation of why the requirement is necessary, and a means to verify whether the need was ultimately satisfied.

An important thing to note is that the engineering requirements in Table 1 were either created by the previous group and not satisfied or created by us to better fit the vision of J.A.R.V.I.S. that we have. Engineering requirements that were created by the previous group and have been satisfied are listed in Table 49 in Appendix B.

Table 1 : Engineering Requirements for J.A.R.V.I.S

ER #	Engineering Requirement	MR #	Rationale / Justification	Testability / Verification
------	-------------------------	------	---------------------------	----------------------------

1	Robot shall redirect or stop at 3 feet from the detected objects	3, 5	One arm length – or 36 in. – is required to move freely	Drive the robot towards an object and verify robot stops at 3 ft. away
2	Camera viewing angle shall be 360 degrees	1, 2	A panoramic view of the surroundings will make navigation and interaction easier and more immersive	Connect to robot as a remote user and verify panoramic camera is working properly
3	Upon object detection, the robot shall disable forward moving navigation control for the user	3, 5	Users should not be able to move forward if an obstacle is detected	Drive the robot towards an object and verify that once object is detected, forward navigation is disabled
4	User should see the remaining battery power available	6	User should know how much charge is available to use.	Log onto the web interface and verify that battery power is seen
5	Robot should keep track of battery level within 3% accuracy	6	An accurate battery measurement allows the user to know approximately how much time they have left	While robot is in use, take battery level measurement and compare accuracy to value displayed on remote interface
6	Robot should indicate video streaming is in progress via LED that is visible from at least 6 feet away	7	Individuals surrounding the robot need to be made aware the robot is streaming	While robot is in use, verify that the LED can be seen from at least 6 feet away, and that LED is turned off when not streaming.
7	Robot should contain an operation manual	8	The robot will need its own document independent of our report that tells the user how to use and troubleshoot the robot	Give robot to someone who has never utilized it and see if they are able to properly operate it using only the manual
8	Robot shall have a user-friendly interface for controlling the robot's movements and functions	4, 8	The UI (user interface) should be intuitive so that the user can easily navigate the system.	Allow someone with no knowledge of the project to connect to user interface and verify its ease-of-use
9	Web interface should provide a	9	If multiple people simultaneously want	Have more than 10 users attempt to connect to the

	queue for up to 10 users to wait for connection access		to connect to the robot, it needs a way to keep users in line for access	robot. Verify only 10 users are in the queue and that they were able to access the robot when it was their turn in line.
10	Robot should have no sharp edges	5, 10	Sharp edges can be a safety hazard for the people surrounding the physical robot	Confirm that all edges on the finished robot are not sharp enough to cause any damage.
11	Robot should weigh no more than 50 pounds	10	The robot should be light enough that it can be manually carried around by an adult when non-operational	Place robot on a scale and verify that it weighs no more than 50 pounds
12	Robot should be able to carry an object up to 5 pounds and up to 6x6x6 inches for remote user with proper stability	10	Depending on the environment, the robot's ability to carry around small objects can be helpful.	Put items of varying weights in the basket to verify the maximum load that can be carried is greater than 5 pounds and that volume of basket allows for 6x6x6 object to fit.
13	Robot shall be able to store sensor data for further training and development	3, 5	If the robot can train itself based on past sensor data, it will be able to improve itself and its movement capabilities.	Record and export point cloud data and use open3D to successfully visualize the data
14	Object Detection feedback shall be provided to the user within 1 second under normal conditions if object detected	3, 4, 5	Giving the user feedback about the surroundings prevents frustration and gives the user better overall control	Place object in front of robot and verify it sends feedback to user within 1 second
15	Robot should cost no more than \$3000	11	Similar robots are usually in the price range of \$3000-\$5000. We want to be at the low end of this range, as we are focusing on individuals or small businesses.	Review costs at the end of production and verify that all parts, including those inherited, and production cost is no more than \$3000
16	Real-time video streaming to user shall have a	1, 2	Low latency allows the remote user to respond	Join Zoom meeting as remote user and checking

	maximum latency of 150ms		to changes in the environment quickly	streaming latency through Zoom
17	Robot shall be controllable through web interface using a keyboard and mouse	4, 8	Keyboard and Mouse provide familiar tools to help with any necessary troubleshooting or start up	Use the interface and verify keyboard and mouse can be used to fully control the robot
18	Camera shall be controllable through web interface using a keyboard and mouse	2, 4	Keyboard and mouse provide the user with familiar tools to move the camera	Use the remote interface mouse and keyboard controls to verify the camera can be fully controlled
19	Robot shall be rechargeable using home base	6, 12	When the robot returns to the home base, the robot should be able to charge itself without requiring a user to plug it in to maintain a level of autonomy	Navigate the robot to the home base, and verify battery begins charging when leads are in contact with home base
20	Wiring should be completely enclosed in a casing	10	Loose wires are a safety hazard to both the robot and people and objects surrounding the robot and should be eliminated	Confirm there are no wires exposed on the completed robot

3.3.1 Changes to Engineering Requirements

Due to our removal and changes to our Marketing Requirements since the last report, some accompanying Engineering Requirements also needed to be removed or changed. Since we removed the Marketing Requirement “Robot should be able to stay within a predefined indoor area of operation,” we also needed to remove the Engineering Requirements associated with this Marketing Requirement. The two Engineering Requirements we removed because of this were “Robot’s boundaries shall be able to be changed or edited” and “Robot shall not travel more than 2 foot outside the navigation boundary.” Due to our changes to the Marketing Requirement regarding the home base, we also decided to remove the Engineering Requirement “Robot’s home location shall be able to be changed or edited.” Lastly, we removed three Engineering Requirements based on feedback received in our PDR, because they were either redundant or unnecessary. These Engineering Requirements include “Robot should have at least two connection options,” “Robot should be sturdy enough to withstand 40 Newtons of force,” and lastly “Robot should be able to notify the remote user in case of any technical issues or errors.”

3.4 Design Impact

In the design of a telepresence robot, there are several potential areas of impact which our team has considered. In this section, we will discuss categories which our robot may be affected by or that our robot could affect in society.

3.4.1 Standards

Our project, although this specific version may not be marketed toward the public consumer, should still abide by the rules and regulations that govern products similar to our own. We would like to make our iteration of this project one that would be easy to transition into a consumer market, which means using parts that are compliant with regulations, as well as following any protocols or guidelines, among other things. Additionally, this robot should operate in a safe and expected manner during all testing and demonstrations.

Although there are several guidelines and standards for robots and artificial intelligence in general, there are not many guidelines that cater to our specific style of robot. Many of the regulations that are currently in place are more focused on robots that are used in an industrial or manufacturing environment, as opposed to a telepresence robot that is used in more of a social environment. The Occupational Safety and Health Administration, or OSHA, has no listed guidelines for telepresence robots. Standards of other organizations such as ISO or IEEE have some guidelines that are slightly more applicable to our project.

The International Organization for Standardization, or ISO, has some high-level standards that can apply to our situation [6]. Standard IEC 61508, which deals with functional safety, and ISO 12100, which discusses risk assessment, can both be considered when designing our project. Referring to these will ensure that the robot is built in a way that makes it safe to use and operate.

Another thing that needs to be considered while we design our project is compliance with the FCC, or Federal Communications Commission. Since our device will be connecting to the internet and potentially interfering with other electronic devices, we need to keep in mind internet connection protocols. The FCC only regulates devices that are marketed to consumers, but the regulations are still important to consider for the future of this project. The previous version of the robot that we inherited is constructed entirely of parts that are FCC compliant. Another internet connection protocol that needs to be considered is IEEE standard 802.11 [7]. This is a very general standard that applies to all devices connecting to a wireless network, which our device does. IEEE 802.11 discusses the specifications and architecture of wireless local area networks, or WLANs. Our robot will need to follow these standards, so it is able to connect to Wi-Fi networks.

3.4.2 Economic

Telepresence robot robots have future market growth potential. In the United States, the telepresence market is currently valued at 262.44 million dollars in 2022 and is projected to grow at a rate of 20.93% from 2023 to 2030 [4]. There are many telepresence robots developed by various companies available on the market and it is our aim to build J.A.R.V.I.S. with similar features at more affordable prices so that J.A.R.V.I.S becomes a popular and accessible medium for remote interaction. We are designing our robot to function in a classroom environment where

multiple robots can be helpful to students and professors. This would allow students to use this style of robot in the case that they are unable to make it to school, but do not want to miss out on what was taught that day. Providing the telepresence robot at reduced cost compared to what is now available on the market, it is our hope that J.A.R.V.I.S. becomes a more accessible means of remote learning.

3.4.3 Social

The robot's main purpose is to allow users who cannot be physically present at a location or event to remotely socialize with people who are. The original intention was for the robot to take the place of anyone who is sick with COVID-19, but it's not just limited to that. Since attending work, school, and events remotely for the sake of convenience is more popular now than ever, this product has potential to have a notable social impact. Telepresence robots being readily available and adapted into both schools and workplaces allows people to get their responsibilities done without sacrificing both one's socialization and health. Being socially isolated for extended periods of time can cause loneliness and depression, but telepresence robots such as J.A.R.V.I.S. can be used to mitigate this. A study done in 2016 by UC Irvine found that most students who used a telepresence robot to attend school thought that being able to socialize with their classmates was their favorite part of the experience [5]. Although the students are primarily in school to learn, being able to properly socialize with other students helps keep the student engaged. Telepresence robots can be used to boost morale and increase productiveness for both students and employees, which is beneficial for all parties involved. J.A.R.V.I.S. strives to allow people to experience a normal everyday life in situations in which this was previously impossible.

3.4.4 Ethical

With this being a telepresence robot, many of the ethical challenges that come with the creation of a robot, such as AI bias or human unemployment, are not applicable in our robot's case. The main area of ethical concern in our eyes is privacy and surveillance. As this robot can record and stream its surroundings, the privacy of those around the robot must be considered. It is important that those around the robot know that the robot is recording its environment and streaming it to another user. We believe people have the right to know the robot is recording so they can stay quiet or leave the room if they are uncomfortable with the recording of their conversation. We made one of our requirements that the robot indicate when it is streaming for this purpose. It will allow for the robot and connected user to operate ethically, knowing that those around the robot can make their own choice about whether they want to remain around it.

3.4.5 Health and Safety

The impacts that J.A.R.V.I.S. could have on our health and safety could be beneficial and must be examined. The possibility of using this robot in the health industry could provide patients with a more convenient, and safe way to interact with other people and attend to their duties. The robot also has potential for use in other environments including conferences, museums, and showcases. In recent years, we have observed public events allowing a virtual or remote option for attendance. Although this option is convenient, it limits the interaction and connection between an environment or host, and their respective attendee. J.A.R.V.I.S. would

provide an experience that feels real, allowing someone to live life vicariously through its interface.

J.A.R.V.I.S could have a large impact from a public health lens as it would allow anyone with a contagious virus to still interact with the rest of the world in a meaningful way without having to risk their health or anyone else's. This robot could allow nurses and other health practitioners a way to interact with patients without risking the health of others and saving space in hospitals. The option to use J.A.R.V.I.S. would be desirable to those who may be immunocompromised or have other diseases that affect their quality of life in public.

It is also imperative to examine the impacts J.A.R.V.I.S. could have on our safety as a machine that is automated. Errors in the programming could result in J.A.R.V.I.S. moving in an unexpected or unsafe way, potentially causing damage to people or the environment surrounding it. A user with ill intent could operate the robot to cause harm to the environment around it, thus making the automatic override crucial to protect the safety of the world around it.

4. Design Summary

In the design summary section, we will be discussing the overall project design, both at a high-level and an in-depth level. The project will be broken down into different modules which are each responsible for different tasks and functions. These modules will also be discussed in detail.

Due to this project being inherited, there are some design decisions that were made by the previous team that we are choosing to keep for the sake of ease. One example of this is their choice of computing unit. The previous team decided to purchase and implement an Intel NUC as the primary computer for the robot. Since the entire previous project was designed, coded, and implemented with this design choice in mind, we find it will be a better transition between versions if we continue to use this computing unit, and other features on the robot. The transition between versions will be more seamless if we continue to use the previous team's computer, microcontrollers, and overall robot structure, including the microphone, speakers, and display. Some design changes that we decided would be worth the changes will be discussed later in this section.

4.1. Design Overview

Figure 2 shows an overview of the robot itself and the components included. It gives a short description of the functionality of each piece and what benefit it provides to the user.

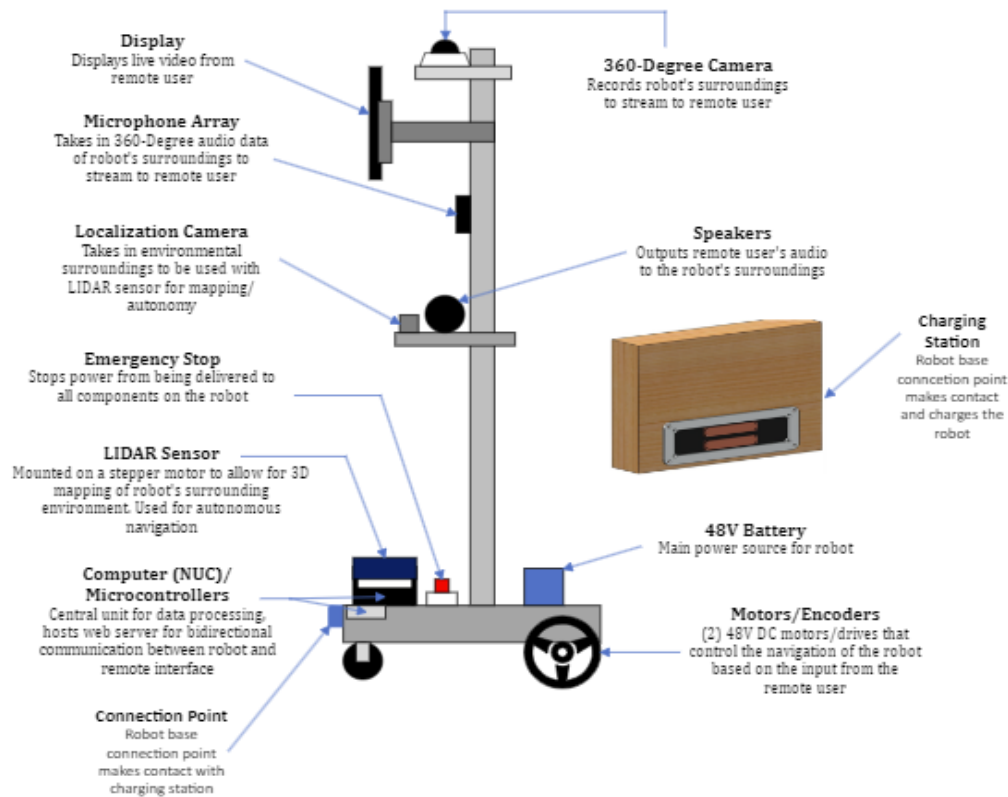


Figure 2 - Hardware Design Overview

Figure 2 shows the final diagram giving an overview of the project's functionality. This diagram is divided into three sections: the robot, home base, and the remote user interface connected by a web server hosted by the NUC. These blocks are color coded, with the power in red, the robot inputs in green, the computers and microcontrollers in orange, remote user inputs in blue, and all outputs in grey. The charging station or the home base is a square shaped box made of wood. The connection point is attached Infront of the home base, back side of the home base is open and inside the home base is the power supply unit which is wired through a micro limit switch to the two-metal connector. The other connector point is mounted on the base of the robot as shown on the figure and when the two connectors make contact, and the micro switch is triggered then the DC power flows through the connectors and charge the battery. From the battery power is sent to the NUC, which takes in environmental data through the lidar sensor and encoders and AV data through the 360-degree camera and microphone. The AV data is sent to the web server and output on the user's end so they can see and hear the robot's surroundings. AV data from the user's end as well as their control inputs are sent to the web server and back to NUC. The user's AV data is sent directly to the speakers and monitor, but the environmental data is used to adjust the input controls in cases such as an object directly in front of the robot being detected. This data is what is sent to the drive which ultimately controls how the motors move. The user camera controls don't need to be adjusted, so it is sent directly from the web server to the NUC and microcontroller. In addition, the servo motor for the lidar sensor and battery display both have their own programs that run continuously on the Arduino. As can be observed

from the diagram, all communication and processes are done through the NUC, making it the heart of our robot.

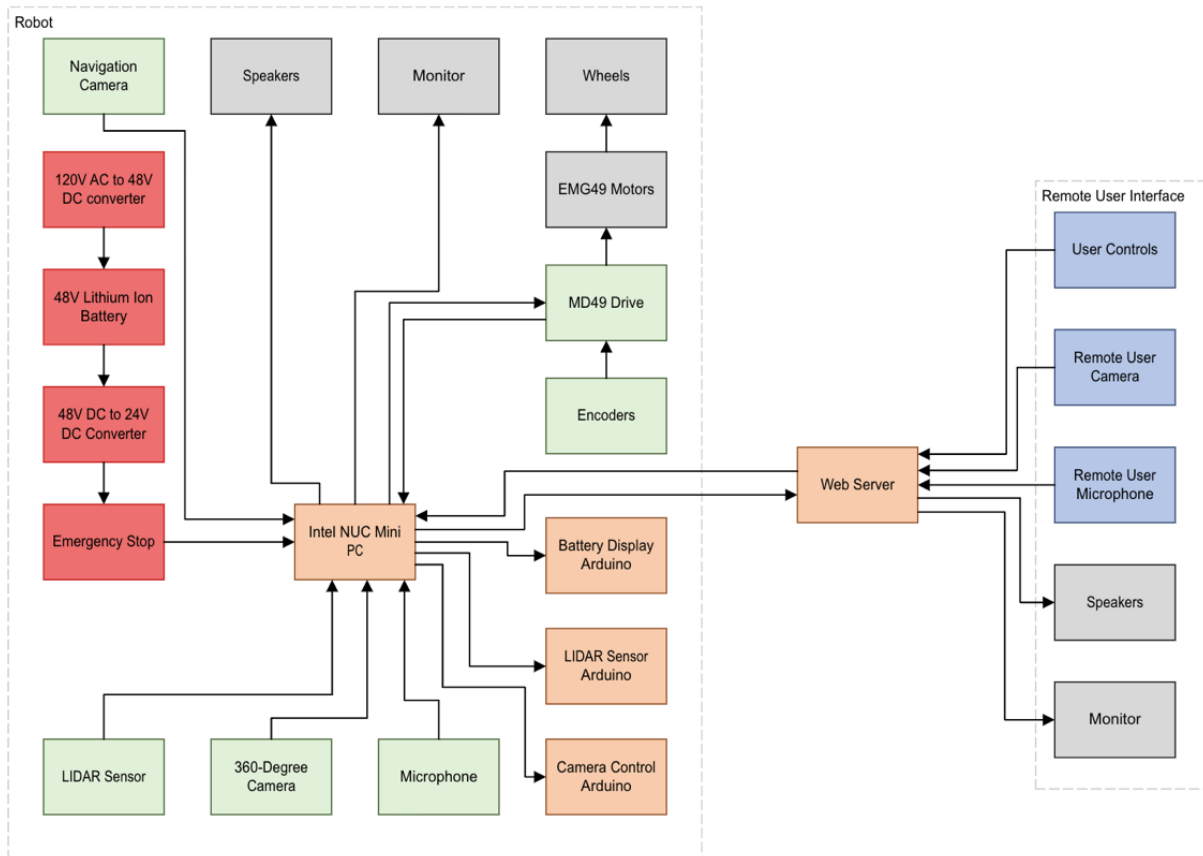


Figure 3 : Final Block Diagram

4.2. Software Overview

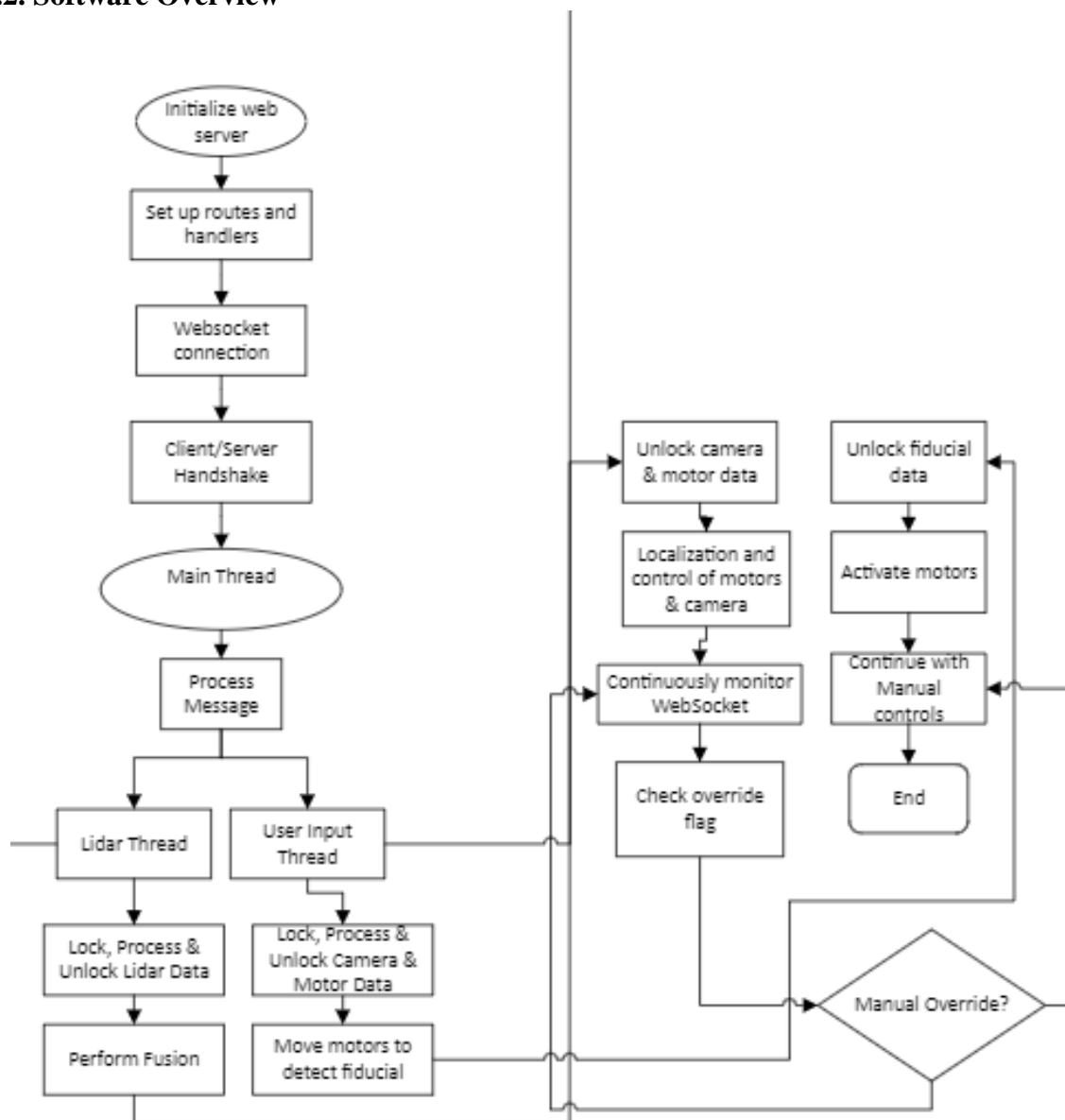


Figure 4 : Main Software Thread Diagram

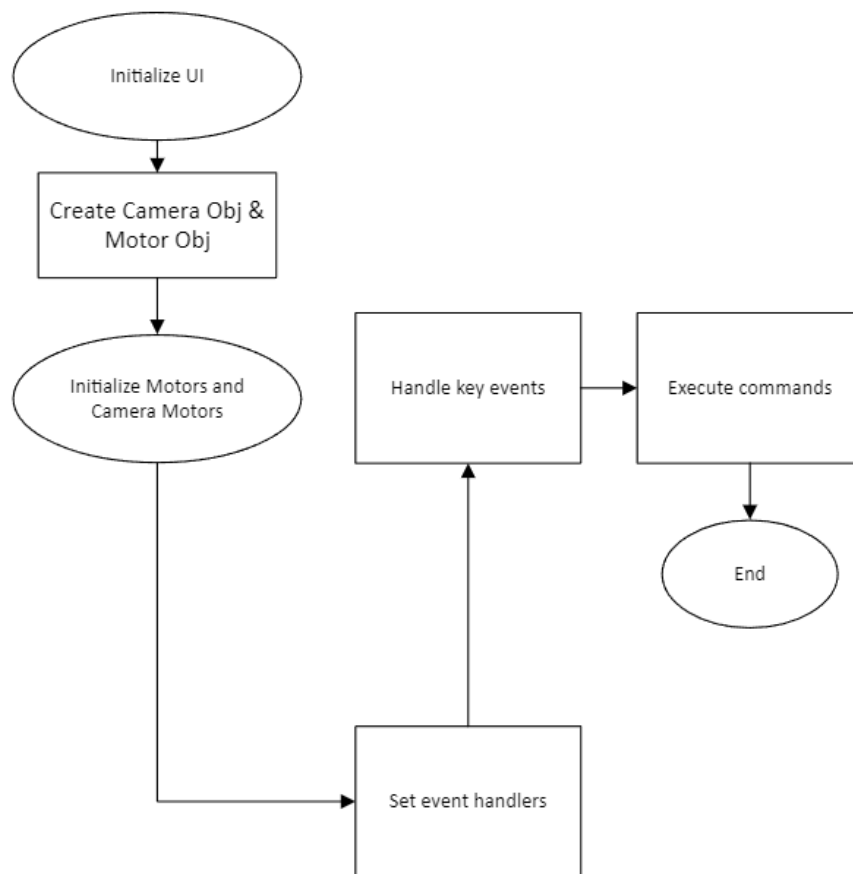


Figure 5 : User Input Thread Diagram

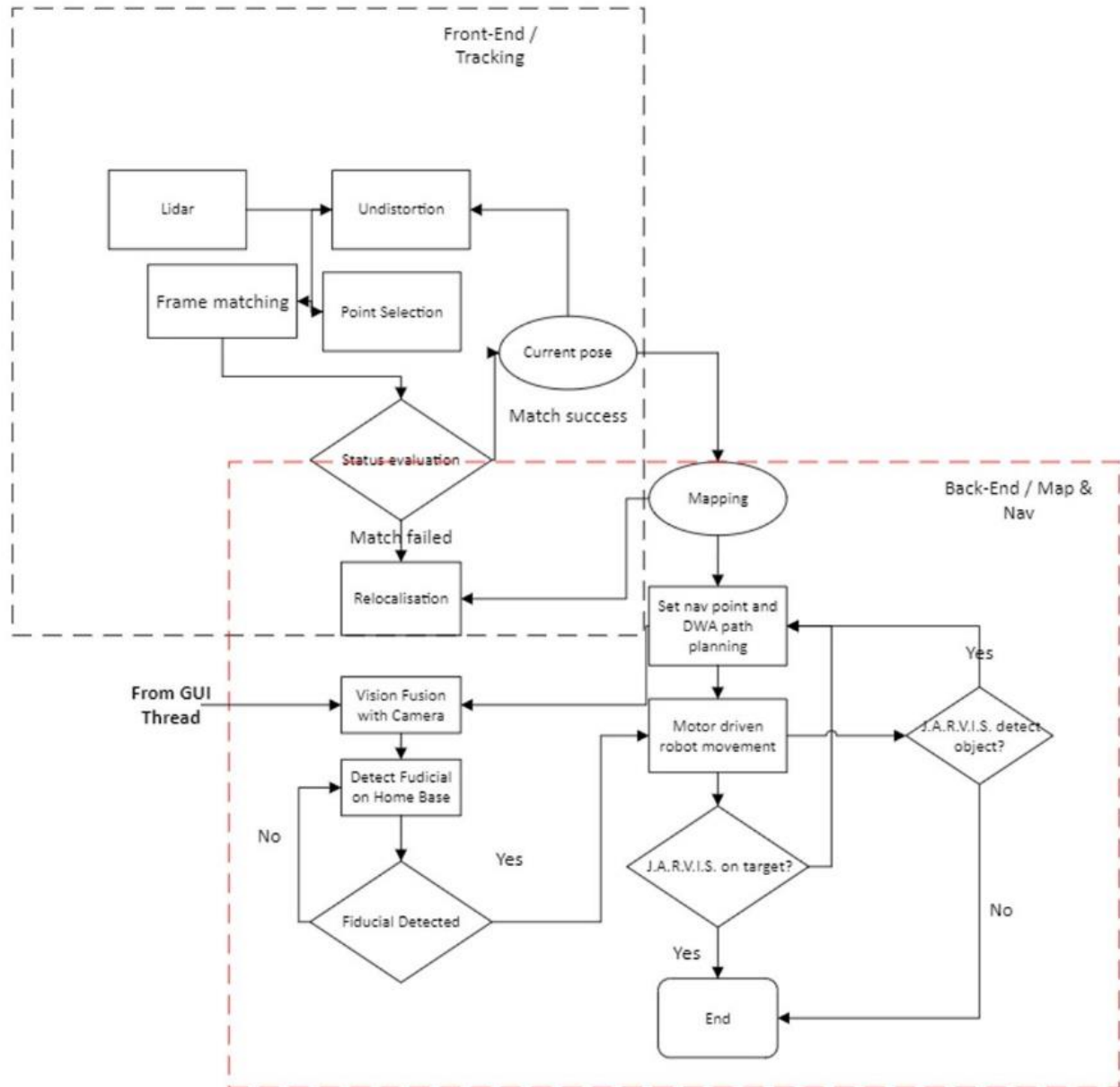


Figure 6 : Lidar Software Thread Diagram

The J.A.R.V.I.S. software architecture is designed for seamless integration of user interactions, sensor data processing, and autonomous navigation capabilities, all managed by three key components: the main thread, the lidar thread, and the user input thread. In contrast to the previous software architecture, our approach for J.A.R.V.I.S. diverges from utilizing the Robot Operating System (ROS). Instead, we've adopted a modular and adaptable framework, allowing us to leverage specific SDKs while maintaining our flexibility to integrate custom drivers and libraries seamlessly. Our decision is further bolstered by the extensive documentation we've provided, ensuring ease of understanding and future maintenance.

The heart of J.A.R.V.I.S. is a robust threading mechanism allowing for concurrent execution of multiple processes. The primary thread, an overview of which is illustrated in Figure 4, serves to initialize the web server, and set up routes and handlers. Key to our approach is the utilization of WebSockets, which provide a real-time communication channel for

concurrent processes. Once a user joins a zoom room and initiates the user interface, the main thread processes incoming messages and makes decisions regarding the movement of the robot.

Simultaneously, two auxiliary threads work in tandem with the main thread which includes the lidar and user input thread. By default, the user input thread, detailed in Figure 5, takes precedence in processing user commands. When a user interacts with the online interface, the user input thread translates these JavaScript commands into motor control instructions, allowing the user to manually navigate the robot.

In Figure 6, the processes involved in the LiDAR thread are illustrated. Beginning with frame matching, the sensor scans are aligned to form a representation of the surroundings before correcting inaccuracies through undistortion. Undistortion is a process which maps between the distorted input camera image and an undistorted output image. Point selection focuses on areas with features to form a point cloud map. Status evaluation monitors the quality of the data and if the frame matching succeeded, the thread will proceed to localization, determining the robot's exact position and orientation which ensures accurate navigation. If this fails, relocalization is performed which leverages the previously mapped data to reorient J.A.R.V.I.S.

The back end of the lidar thread, displayed in the bottom half of Figure 6, performs mapping and navigation. The environment is updated based on the current pose and through setting a navigation point, a dynamic window approach (DWA) algorithm is employed to chart a safe and efficient route while continuing to map the environment around it. The vision system fuses the lidar and mid-level camera to heighten the perception of the environment. The camera can compensate for the lidar sensor's inability to detect reflective surfaces by matching visual features to the map for enhanced localization which is a crucial step of the home base semi-autonomous return. Additional safety measures are implemented which allow the lidar thread to gain access to motor data from the user input thread and override, redirecting the robot away from potential collisions as illustrated in Figures 4 and 6.

This dual functionality ensures a balanced and safe operation of the robot. Users have the flexibility to manually control the robot, but in critical situations, the lidar thread can take control to prevent accidents. This combination of user interaction and autonomous safety measures enhances the overall usability and safety of the robot.

4.3 User Experience Overview

Before the remote user is able to connect to the webpage, the server needs to be launched. In order to do this, the operator of the robot needs to navigate to the JARVIS/Web_Server/Components folder. Once this path is established as the current directory using terminal commands, the operator needs to run the "npm start" command. This will launch the server and provide the URL that the remote user needs to navigate to. For the remote user to connect to the robot, they will pull up the specified URL in the browser of their choice. The URL is the IP address of the NUC, followed by "/nav.html" This URL is a webpage hosted by the NUC that allows for the remote user to communicate with the robot. The initial landing page that the user sees is shown in Figure 7.

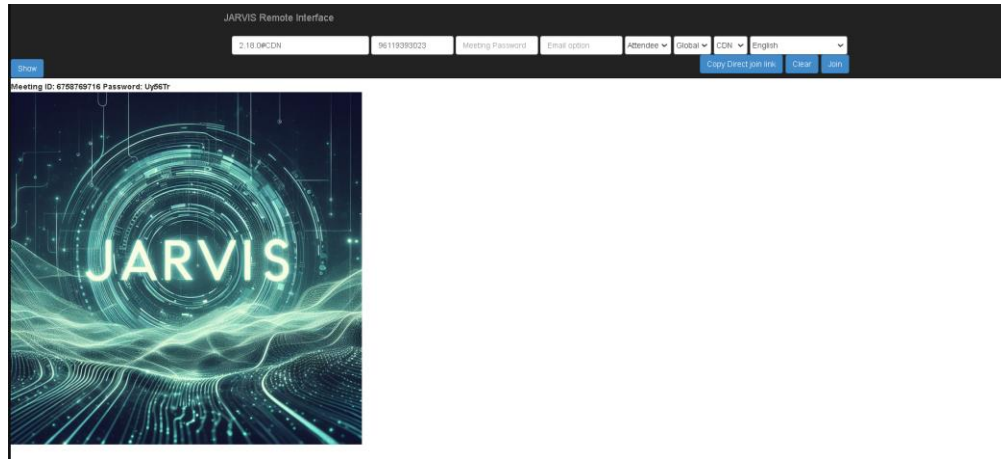


Figure 7 : Zoom Meeting Landing Page

The box on the furthest left of the screen is where the remote user will enter what name they want to appear under in the meeting. The next box to the right is where the User will enter the Meeting ID of the Zoom Meeting hosted by the robot, which is also shown under the blue “Show” button on the left side of the screen. If there is a password on the meeting, the user will be able to enter in that password at this menu as well. After entering in all necessary details, the user will click the blue “Join” button shown in the bottom right.

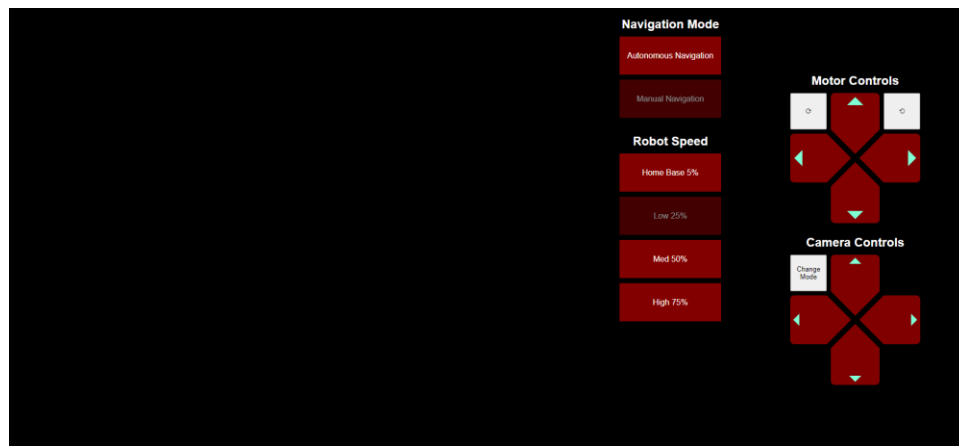


Figure 8: J.A.R.V.I.S. Online UI

After clicking “Join,” the user will be redirected to the main remote interface, which will show the AV stream from the robot, as well as the controls for the remote user to control the movement of the camera and the robot itself. The main remote interface is shown in Figure 8. The Zoom pane is supposed to be resizable on the left pane, however we were unable to get it working correctly. The camera servo can be moved up, down, left, or right. The navigation controls can move the robot forward, backward, left, or right, controlled by the regular D-Pad controls in red. To the left of these D-Pad controls are some extra navigation functions. The buttons titled “Autonomous Navigation” and “Manual Navigation” do exactly that: change the robot from either manual control via D-Pad or Autonomous Navigation, which utilizes the LiDAR sensor to avoid objects. These buttons launch the threads discussed in the previous section, with the Autonomous Navigation button launching the LiDAR thread, and the Manual Navigation button launching the User Input thread. Below these buttons are different speed

options. The default is set at 25% speed, as this provides the best balance between speed and stability of the robot. The user can choose to go faster or slower if they like. Driving up to the home base requires turning the speed to the lowest setting, in order to most effectively connect the charging leads on the robot to those on the home base.

5. Functional Decomposition

The following section is the functional decomposition of our project. This will be a breakdown with increasing detail of how our project works. This will start at level 0, which deals with basic inputs and outputs, and will finish at level 3, which will show a detailed breakdown of each subsystem within our project. These will also be color coded by individual subsystem. We decided to split our project into 4 sub-projects for our 4 team members: Power Delivery, Hardware Communication, Navigation Systems, and Remote User Interface. The colors associated with each sub-project will be discussed in the level 0 section. We also referenced the previous team's functional decomposition to complete this section. Since level 0 and level 1 are extremely general views of the robot, not much changed from the previous version to our version. Levels 2 and 3 are where one can start to see more clearly the changes that are being made from version 1 to version 2 of this robot. We have also denoted any system that the previous team completed that we will not be improving upon by greying out the block. This will show that any block in color is an addition or a development from the previous team's decomposition.

5.1. Level 0

Figure 9 shows the overall inputs and outputs of the system. J.A.R.V.I.S. will be charged by a charging station that is connected to 120 V outlet, controlled by the remote user's inputs, and will record both audio and video of the environment. These inputs allow J.A.R.V.I.S. to move, interact with the people around it, and show the user both the environmental data around the robot as well as any other information the user should know.

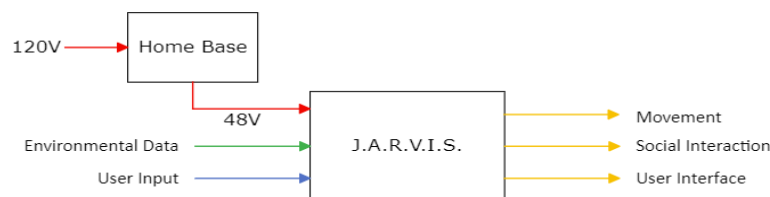


Figure 9 : Level 0 Functional Decomposition

Details of the overall inputs and outputs are listed in table 2 along with a summary of the functionality the system provides.

Table 2 : J.A.R.V.I.S. Functionality

J.A.R.V.I.S.

Inputs	120 V used to charge the robot
	Audio and video of the environment is recorded by the robot, as well as any sensor data to detect objects
	Remote user uses web interface to control robot and send audio and video of themselves
Outputs	Robot moves according to user inputs as well as sensor data
	Robot outputs audio and video of remote user to allow people around it to socialize with remote user
	Audio and video of the environment the robot is in is sent to user interface, as well as any additional information the user should know
Functionality	The combination of remote video chatting and control over robot allows user to simulate in-person social interaction.

5.2. Level 1

5.2.1. JARVIS

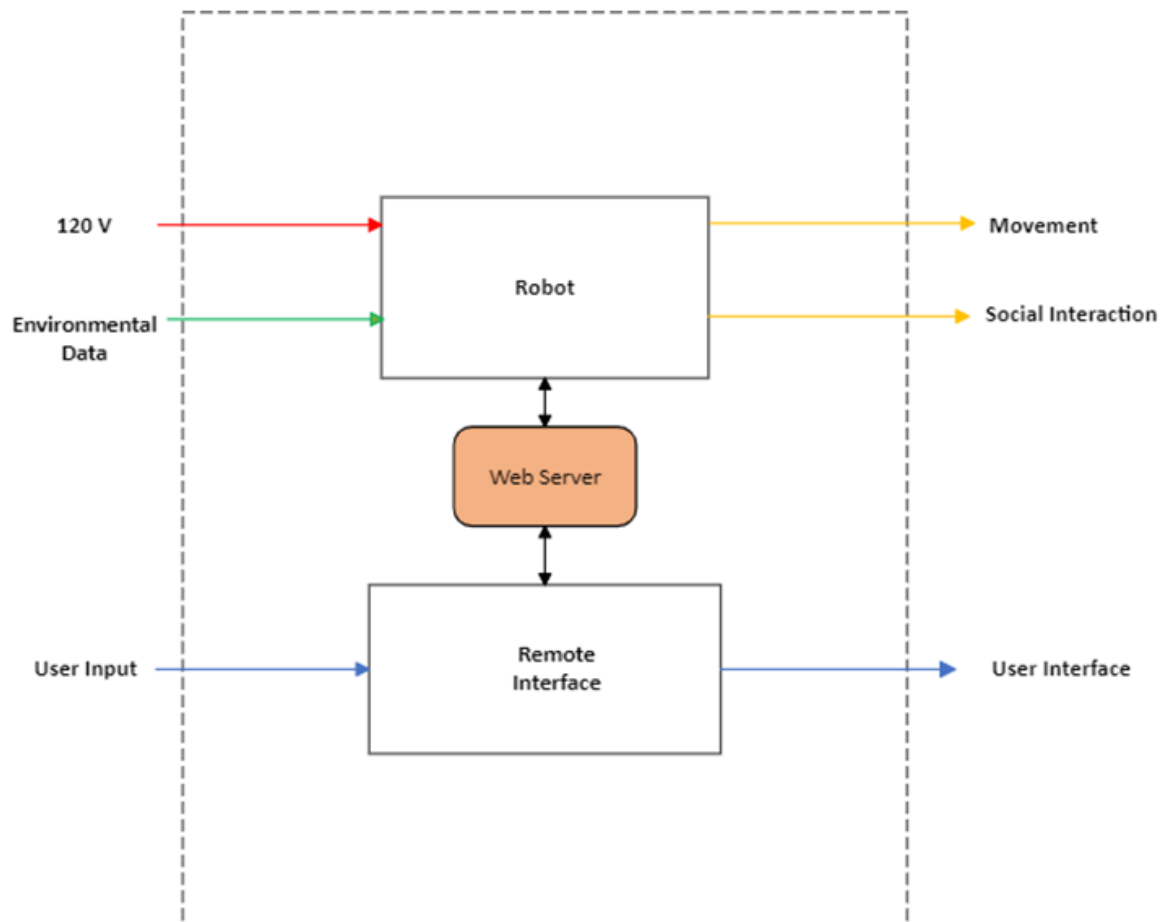


Figure 10 : Level 1 Functional Decomposition

Our level 1 decomposition divides our project into two different subsystems: the physical robot and the remote user interface that is shown in Figure 10. The robot subsystem deals with

anything on the physical robot. In general, this is most often dealing with sensor data, motor control, or power delivery. The remote interface subsystem is any inputs or outputs coming from the online user. This could also deal with motor control or audio/visual inputs and outputs on the remote side. The level 1 decomposition takes all the inputs and outputs from our level 0 decomposition and splits them between the two subsystems. These two subsystems will interact with each other through a wireless network connection via the microcontroller on the physical robot and the internet connection of the remote user.

Table 3 : Robot Functionality

Robot	
Inputs	48V DC from the Home Base
	Environmental data from the sensors
Outputs	Physical robot movement around the room
	Social interaction for the remote user
Functionality	The physical robot subsystem is responsible for hosting the web server for the remote user to connect to. The main computing unit within the robot will take the controls given by the user on the remote interface and then transform these controls into motion and communication for the remote user

Table 4 : Remote Interface Functionality

Remote Interface	
Inputs	User input and controls for the navigation system and camera
Outputs	An audio and video stream from the robot, visible to the remote user
	A set of controls for the upwards-downwards-left-right movement of the robot and camera
Functionality	The interface will act as a connection point between the remote user and the microcontrollers/computer on the robot, transform user inputs into outputs for the microcontrollers on the robot, and transform communication output and relevant information from the computing unit on the robot the robot

Table 5 : Web Server Functionality

Web Server	
Inputs	Commands entered from the UI, A/V data from robot and UI, information about the robot like errors or battery percentage
Outputs	An audio and video stream from the robot, visible to the remote user, and vice versa
	A set of commands to send to different elements on the robot corresponding to the user input
Functionality	The server is what facilitates the communication between remote user and robot. The server hosts the webpage that the user connects to. Once the user is connected, it also sends data between the robot and the UI and vice versa for the robot to behave how the user intends

5.2.2. Home Base

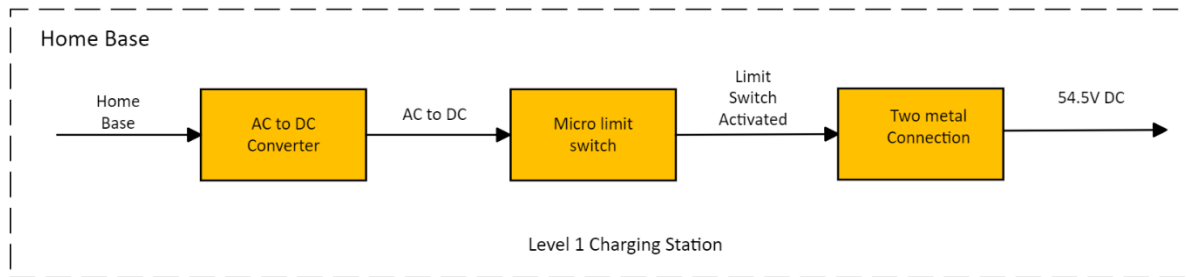


Figure 11: Home Base Level 1 Functional Decomposition

The home base or the charging station is a 15" x 15" x 15" cube shape box made of wood. On the front side there is a connection point attached to the box and the back side of the charging station will be open. Inside the box the power supply will be stationed. The power supply converts 120V AC to 54.5V DC power. The connection point which is two aluminum metal bars is connected to the power supply unit through micro limit switch with wire. When the connection point on the base of the robot contacts the charging station connection point, then the micro limit switch will be triggered and complete the circuit. As a result, power will flow and charge the battery. When the micro limit switch is not activated the circuit remains open, so no power runs on the metal bars that are connected to the charging station. The decomposition level 1 of home base or the charging station is shown in Figure 11. The home base and connection functionality shown in Tables 6 and 7.

Table : Home Base Functionality

Home Base	
Inputs	120V AC power from wall outlet
Outputs	Home base
Functionality	Provide AC power source to the home base

Table: Connection Functionality

Connection Lid	
Inputs	54.5V DC power
Outputs	54.5V DC Power
Functionality	Connects to the robot's charging point

5.3. Level 2

5.3.1. Robot

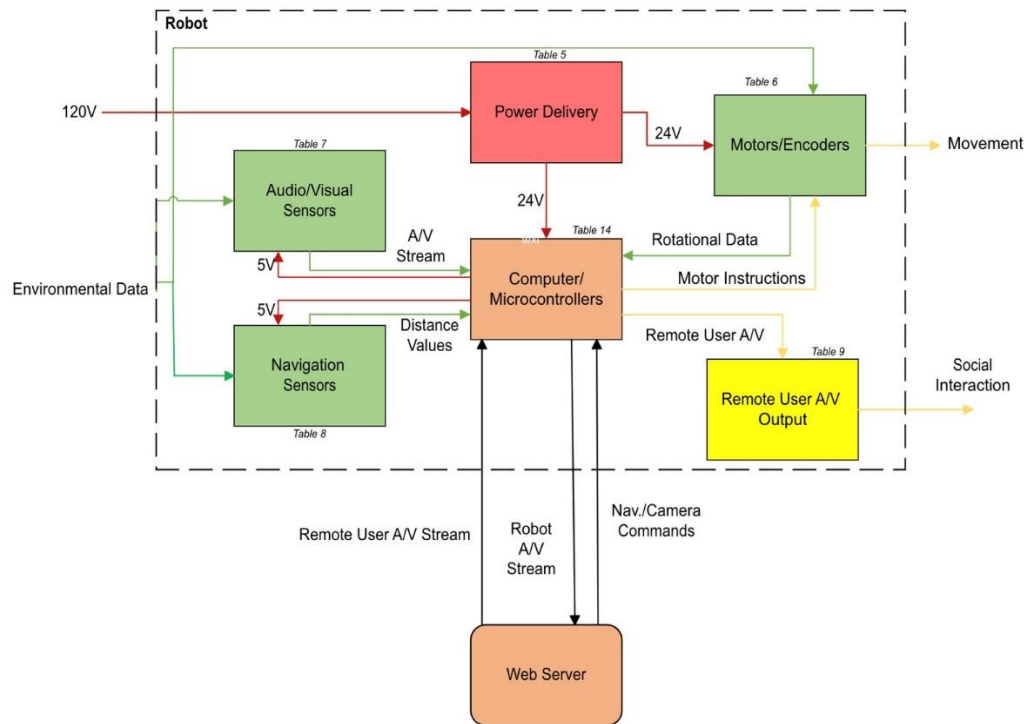


Figure 12 : Level 2 Functional Decomposition - Robot

Our level 2 functional decomposition breaks down the project into even more detail. This level is once again divided into the two previous subsystems, the physical robot, and the remote interface. This level shows how the different modules integrate in the final system and how each block communicates with the microcontroller(s) on the robot. Figure 12 focuses on the physical robot and its inputs and outputs, as well as giving more detail on what the robot does with each of those inputs and outputs. Level 2 also displays what information is being sent between the robot and the web server, which facilitates the communication between the robot and the remote interface.

Table 5 : Power Delivery Functionality

Power Delivery	
Inputs	120 V AC
Outputs	24V DC supplied to the motors/encoders
	24V DC supplied to the computer/microcontrollers
Functionality	To take the AC voltage from the wall outlet, convert it to DC, and apply this voltage to the desired outputs in order to power the robot and its components

Table X shows the functionality of the power delivery module. This module is shown in Figure 12 in the red box entitled “Power Delivery.”

Table 6 : Motors/Encoders Functionality

Motors/Encoders	
Inputs	24 V DC from the power supply
	Environmental data from the object detection sensors
	Motor instructions from the main computing unit, given by the remote user
Outputs	Rotational data is sent to the computer after receiving environmental data
	Physical movement of the robot around the room
Functionality	To take in the power and environmental data, convert this into rotational data to send to the computer, as well as to receive the motor instructions from the computer, initially given by the remote user, and translate these directions into movement

Table X above shows the functionality of the Motors and Encoders module. This module is shown in Figure 12 in the green box in the top right entitled “Motors/Encoders.”

Table 7 : Audio/Visual Sensors Functionality

Audio/Visual Sensors	
Inputs	Audio input signal supplied by the microphone on the robot
	Video input signal from the 360° camera on the robot
Outputs	Audio/Visual stream, taken in from the devices on the robot, and sent to the computer on the robot, to be displayed to the remote user
Functionality	Take in audio/visual data to display to the remote user and send captured data to computing unit

Table X above shows the functionality of the Audio/Visual Sensors module. This module is shown in Figure 12 in the green box in the top left corner entitled “Audio/Visual Sensors.”

Table 8 : Navigation Systems Functionality

Navigation Sensors	
Inputs	Data from the object-detection sensors
Outputs	Distance values which are sent to the computer and microcontrollers
Functionality	To record object-detection and range data, convert this information to quantitative distance values, and send these distance values to the computer to be used in motor instruction

Table X above shows the functionality of the Navigation Sensors module. This module is shown in Figure 12 in the green box in the bottom left entitled “Navigation Sensors.”

Table 9 : Remote User A/V Output Functionality

Remote User's Audio/Visual Output	
Inputs	Remote user's audio signal captured from microphone
	Remote user's video signal captured from webcam
Outputs	Remote user's audio signal captured from microphone sent to the speakers on the robot
	Remote user's video signal captured from webcam sent to the display screen on the robot
	Social interaction for the remote user
Functionality	Take the remote user's audio and video data from the microcontroller and output it to the robot so the remote user can be seen and heard

Table X above shows the functionality of the Remote User A/V Output module. This module is shown in the yellow box in Figure 12 entitled “Remote User A/V Output.”

Table 10 : Computer/Microcontroller Functionality

Computer/Microcontroller	
Inputs	A/V stream from robot comms input
	Distance values from navigation sensors
	A/V stream from web server
	Navigation commands from web server
	Rotational data from motors/encoders
	24V power from power delivery
Outputs	Motor instructions to motor/encoders
	Remote user A/V to robot comms output
	A/V stream to the web server
	Remote user A/V to robot comms output
Functionality	Receives power from the power system. Take the Audio/Visual data and distance values from navigation sensors and send motor instructions to motors.

Table X above shows the functionality of the Computer and Microcontrollers on the robot. This module is shown in Figure 12 in the orange box in the center of the robot module, entitled “Computer/Microcontrollers.”

Table 11 : Web Server Robot Side Functionality

Web server	
Inputs	Robot A/V stream from NUC
Outputs	Navigation and camera control from web server
	Remote user A/V stream
Functionality	Receives robot's surrounding A/V stream. Sends remote user A/V to allow for 2-way communication, also sends user inputs to control camera and robot's movement.

Table X above shows the functionality of the Web Server from the robot's point of view. This module is shown in Figure 12 at the bottom, entitled "Web Server."

5.3.2. Remote Interface

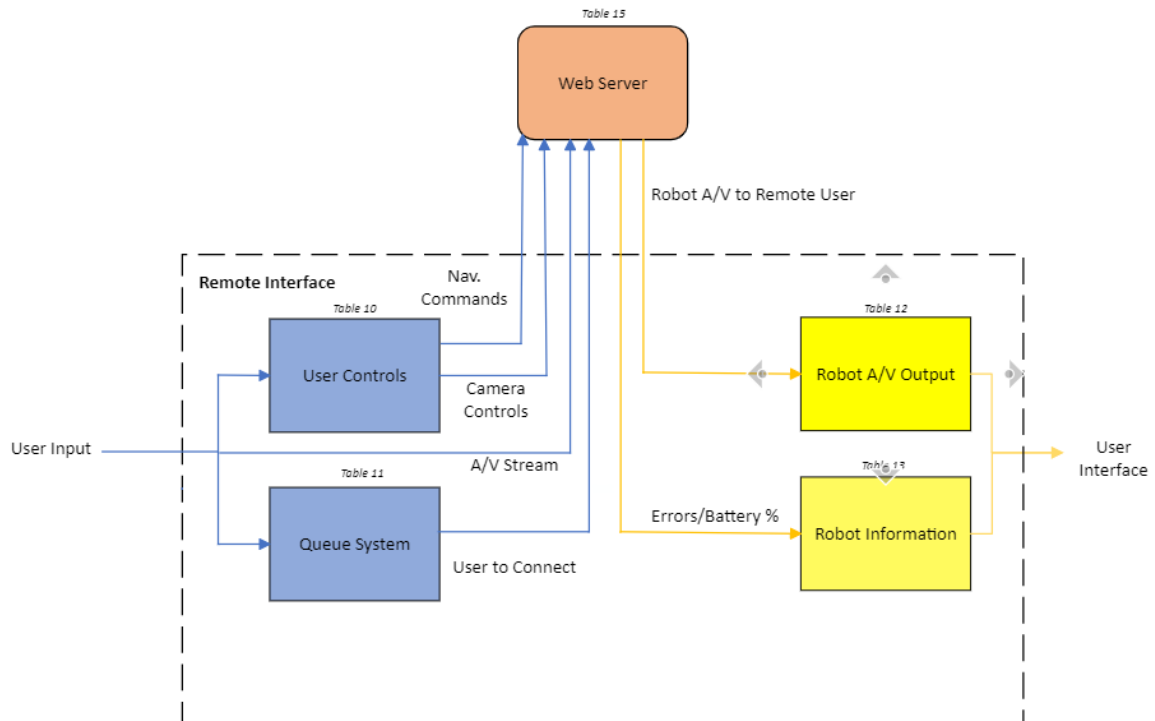


Figure 13 : Level 2 Functional Decomposition - Remote Interface

Shown in Figure 13 is the second half of our level 2 functional decomposition, focusing on the remote interface module. Similar to Figure 12, this gives a more in-depth look at what exactly the remote interface does with the inputs and outputs mentioned in our Level 1 Functional Decomposition shown in Figure 10. It also shows what information specifically is sent between the remote interface and the web server. Note that the web server block shown in Figures 12 and 13 is the same block, serving the same functionality, even though it is shown in both figures separately. This was done for improved readability, as well as focusing more on how both the robot and the remote interface interact with this web server.

Table 12 : User Controls Functionality

User Controls	
Inputs	User navigation control (Left, Right, Forward, Backwards, Stop)
Outputs	User navigation controls are sent to the computer on the physical robot via the web server
Functionality	Take the user control signals and send them to the main computing unit on the robot in order to be processed and executed. The Remote User will be able to decide if they want to manually control the robot, or to put the robot in “Automatic Drive”

Shown above in Table 12 is the functionality of the “User Controls” Module. This can be seen in Figure 13 in blue in the top left corner, entitled "User Controls.”

Table 13 : Robot A/V Output Functionality

Robot’s Audio/Visual Output	
Inputs	Audio signal captured from microphone on the robot
	Video signal captured from the 360° camera on the robot
Outputs	Audio signal captured from the microphone on the robot, received from the computer, displayed via the online interface to the remote user
	Video signal captured from the 360° camera on the robot, received from the computer, displayed via the online interface to the remote user
	Social interaction for the people surrounding the physical robot
Functionality	Take the Audio/Visual data captured by the physical robot, process it, and display it to the user on the interface

Shown above in Table 14 is the functionality of the Robot A/V Output module. This module is shown in yellow in Figure 13 in the top right corner entitled “Robot A/V Output.”

Table 14 : Robot Information Functionality

Robot Information	
Inputs	Battery level information from the computer
	Error messages that need to be displayed by the user
Outputs	Battery level displayed on the online interface
	Any error messages that pop up are displayed on the online interface
Functionality	To serve as a middleman between the computer and the online interface to receive and display certain information

Shown above in Table 15 is the functionality for the Robot Information module. This module is shown in yellow in the bottom right corner of Figure 13, entitled "Robot Information.”

Table 15 : Web Server Remote User Side Functionality

Web server	
Inputs	Nav commands user control
	User A/V from input
	User to connect from queue system
Outputs	Remote A/V from video audio system
	Battery error % to robot information
Functionality	Establishes connection between remote user and robot. Receives user input and A/V, controls queue system and delivers audio video to remote user

Shown above in Table 16 is the functionality for the Web Server, from the point of view of the remote interface. This is shown in Figure 13 at the very top in orange, entitled “Web Server.” Again note that this is the same block that was shown in the previous Level 2 breakdown, in Figure 12, but is shown again for readability’s sake.

5.4. Level 3

In this section, we will be discussing the different modules from our Level 2 Functional Decompositions from Figures 12 and 13 and breaking them down into even greater detail. We will be including a more detailed figure and another functionality table for each module.

5.4.1. Robot Communication Input

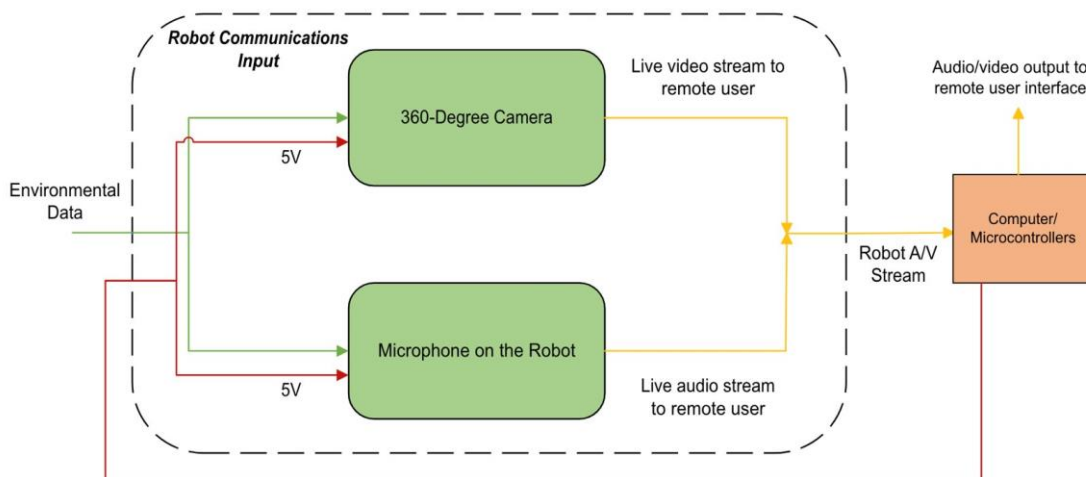


Figure 14 : Level 3 Robot Communication Input Functional Decomposition

Figure 14 is the level 3 functional decomposition for the Robot Communication Input Module, which is shown in the top left of the level 2 functional decomposition. In level 3, it is broken down further into the camera and the microphone blocks. This module is fundamental to this robot, as the entire purpose is social communication, and this module is essential for this. The purpose of this module is to take in the audio and visual data captured by the microphone and camera on the robot and send it to the NUC (the main computer in the robot) via 2 USB

cables, 1 for the camera and 1 for the microphone, for this data to be processed and sent out to the user interface.

Table 16 : 360-Degree Camera Functionality

360-Degree Camera	
Inputs	Environmental video
Outputs	Live video stream to remote user
Functionality	Incorporating the 360-degree camera on the robot allows a more immersive experience for the remote user connected to the robot, as well as giving that remote user increased awareness of the surroundings of the robot. This makes navigation easier as the remote user is able to see all surroundings of the robot, as opposed to only what is in front of the robot

Shown above in Table 17 is the functionality table for the 360 degree camera. This module can be seen in Figure 14 at the top labeled “360-Degree Camera.”

Table 17 : Microphone On The Robot

Microphone on The Robot	
Inputs	Environmental audio
Outputs	Live audio stream to remote user
Functionality	Record the audio around the robot so that it can be sent to the computer/microcontrollers, and eventually to the remote interface

Shown above in Table 18 is the functionality table for the microphone on the robot. This module can be seen in Figure 14 at the bottom labeled “Microphone on the Robot.”

Table 18 : Computers/Microcontrollers Functionality

Computer/Microcontrollers	
Inputs	Robot A/V stream from the camera and microphone
Outputs	Audio/video output to the remote user interface
Functionality	To process the video/audio coming from the robot and send it to the remote user interface through a web server

5.4.2. Robot Communication Output

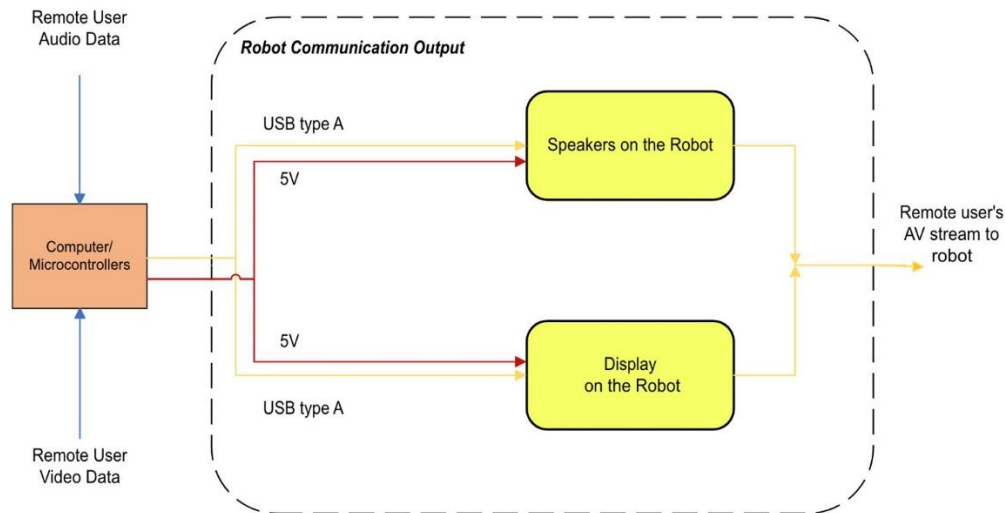


Figure 15 : Level 3 Robot Communication Output Functional Decomposition

Figure 15 is the level 3 functional decomposition for the Robot Communication Output Module, which is shown as a grey box on the right side of the level 2 decomposition. This module is broken down further in level 3 into the speakers and the display screen on the robot. The remote user's audio and visual information will have already been received by the NUC. The purpose of this module is to take in the remote user's audio and video data, which it received from the NUC via USB cables, and output them via the speakers and display screen on the robot. This module will help facilitate two-way communication and help achieve the main goal of the project.

Table 19 : Computer/Microcontrollers Functionality

Computer/Microcontrollers	
Inputs	Remote user audio data from the web server
	Remote user video data from the web server
Outputs	Processed remote user A/V data to speakers and display on robot
Functionality	To process the video/audio coming from the remote user and send it to the speakers and display

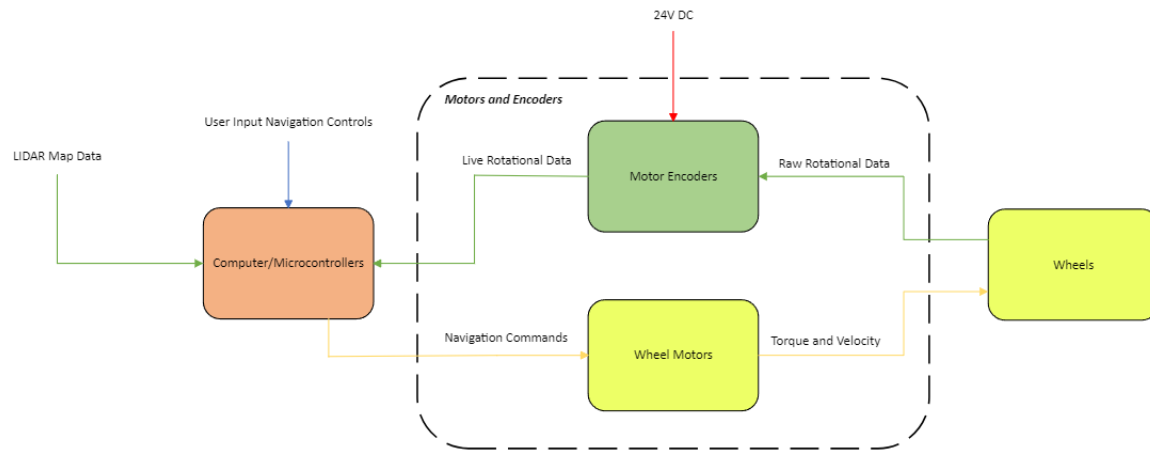
Table 20 : Speakers on The Robot Functionality

Speakers on The Robot	
Inputs	Processed remote user audio data from computer/microcontrollers
Outputs	Remote user audio stream to robot
Functionality	To take the remote user audio data and play it on the speakers for people around the robot to hear the remote user

Table 21 : Display on The Robot

Display on The Robot	
Inputs	Processed remote user video stream from computer/microcontrollers
Outputs	Remote user video stream to robot
Functionality	To take the remote user video data and play it on the display for people in front of the user to see a live stream of the remote user

5.4.3. Motors and Encoders

**Figure 16 : Level 3 Motors and Encoders Functional Decomposition**

Shown in Figure 16 is the level 3 functional decomposition for the Motors and Encoders Module. This module can be seen in the top right corner of the level 2 functional decomposition in Figure 12. In level 3, this module has been further broken down into 2 more blocks: the motor encoders and the wheel motors themselves. The purpose of this module is to help with the navigation and movement of the robot. This module will take the processed remote user's input navigation commands along with the processed sensor data and turn them into code that will move the wheel motors to correctly move the robot around the room. The motor encoders will be responsible for sending live rotational values to the NUC, which will be processed and sent back out to the wheel motors, which will in turn move the robot according to the remote user's desired inputs.

Table 22 : Computer/Microcontrollers Functionality

Computer/Microcontrollers	
Inputs	User input navigation controls from remote user
	Map data from lidar sensors
	Live rotational data on how much motors have moved from encoders
Outputs	Motor control instructions to wheel motors
Functionality	Use the user input navigation controls to control how the wheels move while also using the map data to stop the motors in the case that an object is

	detected. When returning to homebase, use the lidar map data (which has been enhanced by a camera detecting a fiducial) to semi-autonomously return.
--	--

Table 23 : Motors/Encoders Functionality

Motors/Encoders	
Inputs	None
Outputs	Live rotational data on how much motors have moved to computer/microcontrollers
Functionality	To keep track of how much the motors have moved so that we can keep track of where the robot is relative to the homebase

Table 24 : Wheel Motors Functionality

Wheel Motors	
Inputs	Motor control instructions from computer/microcontrollers
Outputs	Navigation of the physical robot
Functionality	To use the motor control instructions to move the wheels, enabling the movement of the robot

Table 25 : Wheel Motors Functionality

Wheel Motors	
Inputs	Motor control instructions from computer/microcontrollers
Outputs	Navigation of the physical robot
Functionality	To use the motor control instructions to move the wheels, enabling the movement of the robot

5.4.4. Navigation Systems

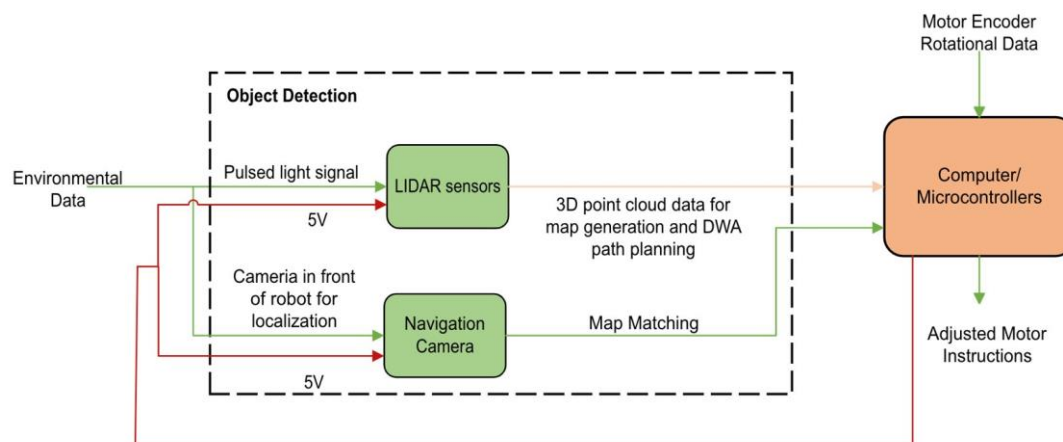


Figure 17 : Level 3 Navigation Systems Functional Decomposition

The navigation systems module for J.A.R.V.I.S. details the hardware and software that allows J.A.R.V.I.S. the ability to navigate itself within areas. One of the things included within

the navigation system is the ability to detect objects. The lidar sensor employs a technique called point cloud mapping to detect objects in its vicinity. This method involves emitting laser pulses and measuring the time it takes for each pulse to bounce back. By combining these measurements, the lidar generates a three-dimensional map of the environment, representing objects as clusters of points in space. This point cloud map serves as a source of spatial information. Objects appear as clusters, with their size, shape, and position accurately represented. A thread will perform post-processing on this information which is then used to implement a path planning algorithm which will send commands to the motors to move J.A.R.V.I.S.

The camera plays an integral role in the post-processing of data. Fiducials will be used on our home base to serve as a datum for localization of the robot. When instructed, the lidar thread will instruct the camera to look for fiducials within its field of view. The imagery from the camera combined with the spatial information will allow the robot to detect fiducials and other objects which may be undetectable by traditional simultaneous localization and mapping (SLAM) methods.

Table 26: Lidar Sensors Functionality

Lidar Sensors	
Inputs	Environmental data
Outputs	Distance, angle, and direction values to computer/microcontrollers
Functionality	To measure the distance between robot and objects so that the microcontroller knows to stop the motor if an object gets too close

Table 27: Computer/Microcontrollers Functionality

Computer/Microcontrollers	
Inputs	Point cloud values from lidar sensors
	Direction, angle, and direction values from lidar sensors
	Rotational data from motors/encoders
Outputs	Motor instructions to motors
Functionality	Using the data from the lidar, follow a path planning algorithm and then detect if an object is too close

5.4.5. Power Delivery

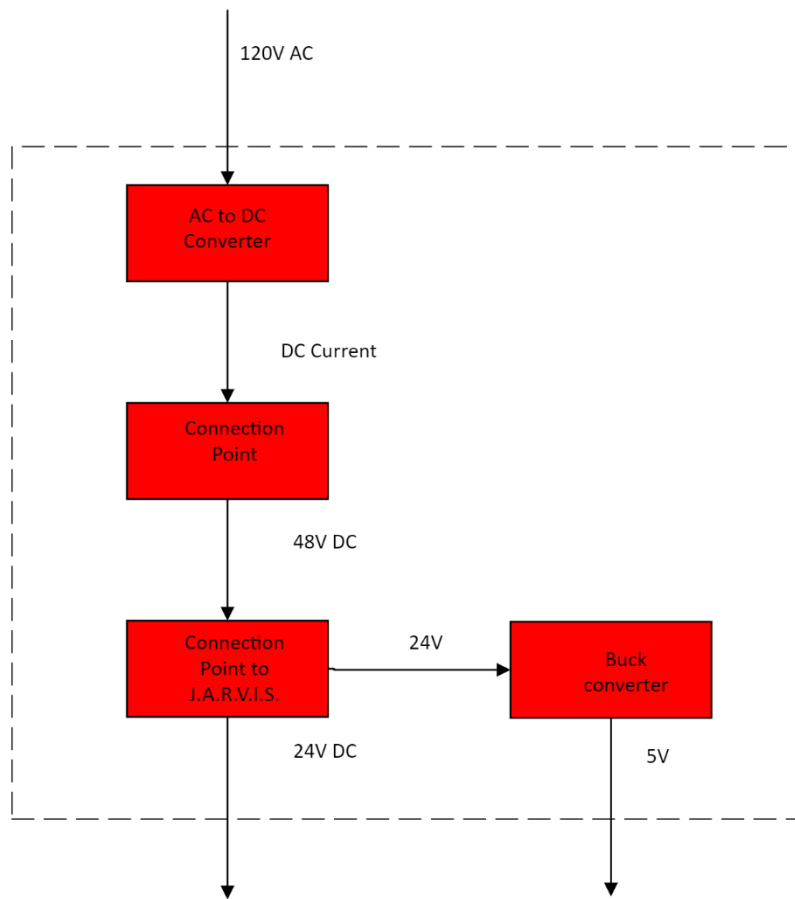


Figure 18 : Power Delivery Functional Decomposition

In our level 3 power delivery decomposition shown in Figure 18, the main power source is 120V AC power which will be the wall outlet. The main power source will be connected to our home base. AC power will be converted to DC power and that power will be delivered to the rechargeable battery by the two-connection point when the robot is driven to the home base and two connection points make contact. This will charge the 48V DC battery and then the DC power converted to 24V DC power from DC-to-DC converter which is installed beneath the base of the robot. We need to convert 48V DC to 24 V DC because the NUC on the robot uses 24V DC.

At the base of the robot, we also installed a USB hub which needs 5V DC power. For that reason, we have taken power from 24V DC converter and then converted to 5V with the help from buck converter to power the USB hub. The AC to DC converter, battery charge, DC to DC and USB hub functionality are shown in table 31, 32, 33, and 34.

Table 28 : AC to DC Converter Functionality

AC to DC Converter	
Inputs	120V AC power from Home Base
Outputs	Converted from AC to DC Power
Functionality	To supply DC power to the battery

Table 29 : Battery Charge Functionality

Battery Charge	
Inputs	DC power
Outputs	48V DC Power
Functionality	The stores variable 48V DC power

Table 30 : DC to DC Converter Functionality

DC to DC Converter	
Inputs	48V DC power
Outputs	24V DC Power
Functionality	Using DC to DC power converter to provide 24V DC power to the robot.

Table 31: USB Hub Functionality

DC to DC Converter	
Inputs	24V DC power
Outputs	5V DC Power
Functionality	Using buck converter to convert the 24V DC to 5V DC

5.4.6. Computers/Microcontrollers

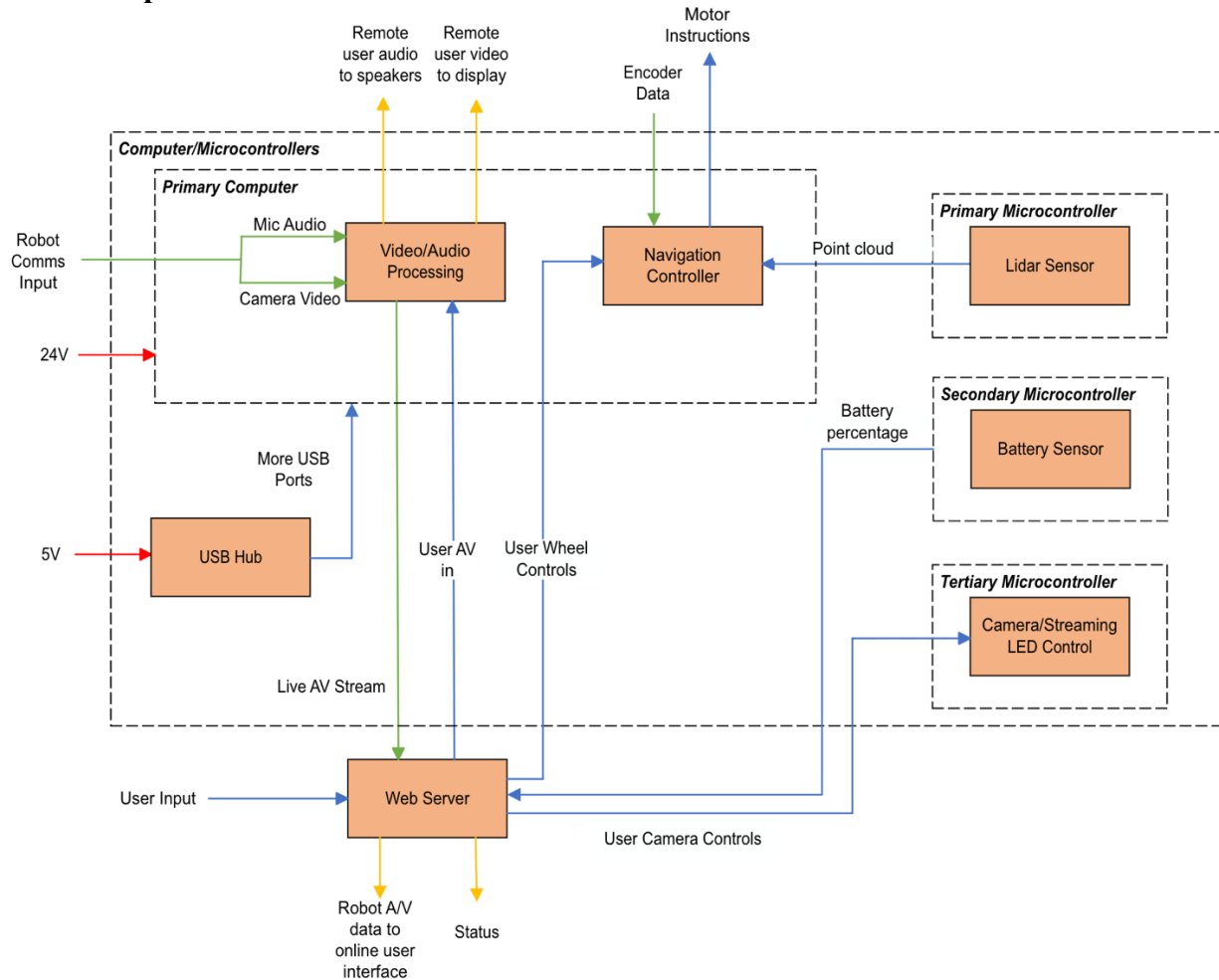


Figure 19 : Level 3 Computers/Microcontrollers Functional Decomposition

The primary computer is at the center of all the electronic systems in J.A.R.V.I.S., with a microcontroller aiding it with object avoidance. We chose to have a microcontroller controlling the object avoidance, as the microcontroller will be dedicated specifically to aid with object avoidance and the lidar sensor, while the primary computer will be taking the information from this microcontroller.

The primary controller houses the video/audio processing unit, and the navigation controller. It also hosts the web server which is used to transfer data back and forth between the remote user and the robot. The raw data from the user, which includes the user video, user audio, and navigation inputs, will be sent directly to the web server from the webpage that the remote user is connected to. The AV data will then go to the video/audio processing unit in the cloud, where it will first be processed and filtered before being output to the display and speakers on the robot. The video/audio processing unit also takes in data from the camera and microphone on the robot so it can process it and send it to the web server, where it will be shown to the remote user. In the case of any errors such as the battery being low or the sensors detecting an object in the way, the remote user will be notified. As for the navigation inputs, they will be sent to the navigation controller which determines where the robot will move. Although the user inputs are the main thing which determine where the robot will move, the navigation controller will also

use data from the sensors and the encoder to determine where to move. The way we plan on using the encoders is to keep track of the movement of the motors to determine the location of the robot on a 2D grid in relation to where it was first turned on, which will be (0, 0). This grid is how we plan to get the robot to move back to the home base on its own, which will be controlled by the navigation controller. The navigation controller will also use the sensor data from the secondary microcontroller to stop the motors in case an object is detected in front of the robot.

The secondary microcontroller serves to control the object avoidance. It will take in the raw data from the sensors so that it can calculate the distance from the robot to any objects it detects. If any objects are detected, it will then send a Boolean to the navigation controller so that we can immediately stop the movement of the motors.

Table 32 : Video/Audio Processing Unit Functionality

Video/Audio Processing Unit	
Inputs	Audio from microphone on robot
	Video from camera on robot
	Remote user audio/video data from web server
Outputs	Live audio/video stream goes to web server
	Remote user audio to speakers
	Remote user video to display
Functionality	Takes raw AV data and processes and filters it so that minimal background noise is noticeable

Table 33 : Web Server Functionality

Web Server	
Inputs	AV data and navigation controls from remote user
	Live audio/video stream of robot comes from video/audio processing unit
Outputs	User audio/video goes to video/audio processing unit
	User input data goes to navigation controller
	Robot audio/video data goes to online user interface
	Status of robot sent to online user interface to notify user that the robot is on, as well as any errors or warnings
Functionality	Takes in data from robot and sends it to the online user interface, in addition to taking in inputs from the user and sends it to the video/audio processing unit and navigation controller

Table 34 : Navigation Controller Functionality

Navigation Controller	
Inputs	User inputs from web server
	Encoder data from motors that move robot
	Detection Boolean from secondary microcontroller to determine whether object has been detected by sensors
Outputs	Motor instructions

Functionality	Controls motors based on inputs. Will primarily move solely based on user inputs, but will also make motors stop if object is detected and will take full control when navigating back to home base
---------------	---

Table 35 : Secondary Microcontroller (Object Avoidance) Functionality

Secondary Microcontroller (Object Avoidance)	
Inputs	Data from sensors
Outputs	Detection Boolean to navigation controller
Functionality	Uses data from sensors to calculate whether object is too close to robot. If so, send data to navigation controller to stop the motor from moving forward.

5.4.7. User Controls

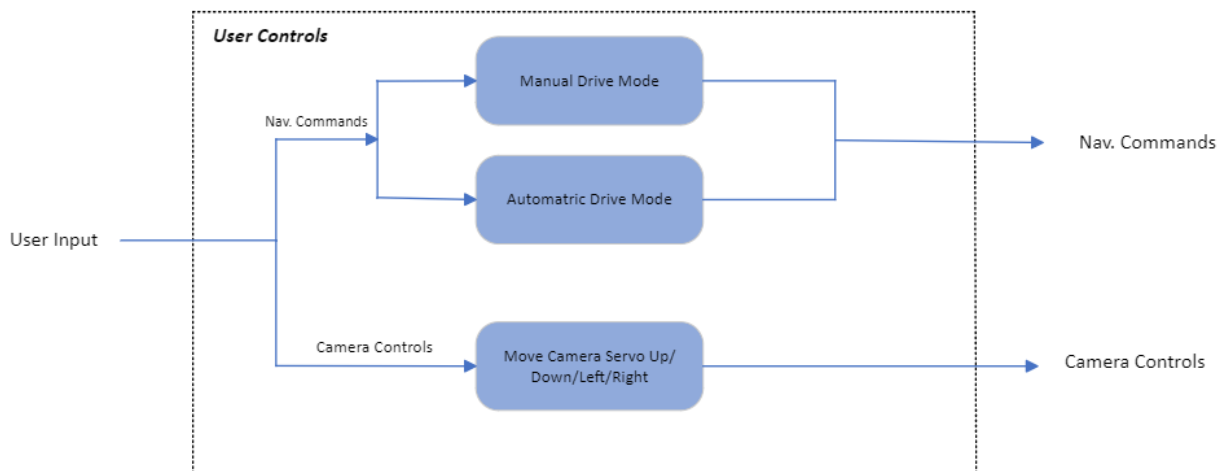


Figure 20: Level 3 User Controls Module

The User Controls Module is located within the remote interface module. This module is responsible for taking the controls entered by the remote user on the webpage and sending these commands to the NUC for handling. There will be two drive modes that the user is able to choose from: automatic and manual. The manual mode will be similar to the previous iteration of the robot, with D-Pad style controls, with buttons mapped to motor commands, to move the robot around the room. The new addition, automatic drive mode, will rely on the lidar sensor to navigate the robot around the room, while avoiding any obstacles that may be in the way. This user input thread is also responsible for taking the controls entered for moving the camera motor. The way the new camera is set up, it will display a 360 degree view of the surroundings of the robot. The issue with this view is that the user cannot see the below the robot monitor in the camera's field of view. Because of this, the camera will need to be on a controllable motor, so the remote user is able to point the camera to see a full-body picture of the person they are talking to. This module will take these commands and send it to the NUC to be sent out to the motors.

6. Subsystem Integration

In this section, the different subsystems will be discussed. Each subsystem will be described in detail, as well as how it dealt with the overall project. It will discuss who was responsible for each subsystem, discussion of the design and implementation, and also the integration into the entire system.

6.1. Subsystem Assignments

Ben Pudlowski is responsible for the online communication interface subsystem. This will include leading all tasks related to the communication between the online user interface and the NUC on the physical robot structure. Tasks associated with this subproject included adding to the previous user interface with things such as additional buttons, battery percentage, and a slightly different design to accommodate the new features in this iteration of the robot. This subsystem will also deal with the communication between the user-inputted navigation and camera controls, and the NUC and microcontrollers, as well as two-way communication between online interface and the communication elements on the robot, namely the display, microphone, speakers, and camera.

Bryce Grant is responsible for the navigation systems subsystem. This will include any tasks related to navigation control. This will include the basic things such as motor control but will also include things that the previous team was unable to accomplish, such as object avoidance and autonomous home navigation. This sub-system will work closely with the online communication interface and the hardware communication sub-systems.

Erick Calvillo is responsible for the hardware communication interface subsystem. This will include communication between the NUC and microcontrollers with the rest of the elements on the robot. He will be working with the online communication interface and navigation sub-systems.

Hasib Hasan is responsible for the power delivery and home base design subsystem. This will differ from the previous team because of our additions of the home charging location and the backup power source. He will be designing and implementing a new charging mechanism. He will work with the navigation subsystem lead on autonomous home location navigation.

Erick Calvillo and Bryce Grant share responsibility with optimizing the motors and encoders with Bryce Grant focusing on the sensor feedback system and Erick Calvillo implementing the motor response to objects and the environment.

6.2. Navigation Systems

```
Laser.setlidaropt(ydlidar.LidarPropMaxAngle, 90.0);
Laser.setlidaropt(ydlidar.LidarPropMinAngle, -90.0);
Laser.setlidaropt(ydlidar.LidarPropMaxRange, 32.0);
Laser.setlidaropt(ydlidar.LidarPropMinRange, 0.01);
scan = ydlidar.LaserScan()

def animate(num):

    r = Laser.doProcessSimple(scan);
    if r:
        angle = []
        ran = []
```

```

intensity = []
for point in scan.points:
    angle.append(point.angle);
    ran.append(point.range);
    intensity.append(point.intensity);
lidar_polar.clear()

```

Figure 21: Lidar Mapping Code

The navigation system is driven by lidar sensor. The lidar sensor will send light pulses to measure the distance to a target and develop a point cloud. This data will be filtered and cleaned before generating a 3D model of the environment. Software was developed while referencing the YDLidar SDK, including the code shown in Figure 21 which allows for control of the lidar scan arc and range, to process generated point clouds.

```

def go_forward(self, ser, speed):
    motor_speed = int(((100-speed)/100) * 128)
    print(f"Going Forwards... Before: {speed}, After: {motor_speed}")
    self.set_speed1(ser, motor_speed)
    self.set_speed2(ser, motor_speed)
def go_backward(self, ser, speed):
    motor_speed = int((speed/100) * 127) + 128
    print(f"Going Backwards... Before: {speed}, After: {motor_speed}")
    self.set_speed1(ser, motor_speed)

```

Figure 22: Motor Commands Code



Figure 23: Lidar 2D Point Cloud

Figure 22 showcases the motor_commands.py code, which serves as a component in executing precise movements through user control. The code enables forward, backwards, left, and right motions as well as left and right rotations with varying speeds, ensuring controlled and accurate navigation. Through rigorous testing, this motor control system has proven its reliability, allowing for manual remote navigation via a tkinter GUI, which is accessible through our [GitHub](#) repository.

Throughout the development process, the Lidar sensor has demonstrated its proficiency in generating accurate 2D point clouds, as depicted in Figure 23. The milestone of 2D point cloud generation not only provides a foundation for subsequent integration and testing but also acts as a precursor to the development of 3D point clouds. This transition ensures greater spatial

awareness, empowering J.A.R.V.I.S. to have a greater understanding of objects with varying heights and the environment.

This will allow the system to integrate a dynamic window approach (DWA) to refine the navigation. DWA harnesses data to dynamically adjust the robot's trajectory which will ensure that J.A.R.V.I.S. can make responsive decisions while navigating through complex environments. This functionality will allow the robot to map the area around it while it moves around.

The relevant marketing requirements to this subproject include MRs 3, 4, 5, and 8 and the relevant engineering requirements include the following:

1. Robot shall stop at 3 feet from the detected objects
3. Upon object detection, the robot shall disable forward moving navigation control for the user
7. Robot's battery shall be rechargeable using home base
9. Robot shall have a user-friendly interface for controlling the robot's movements and functions.
15. Robot should be able to store sensor data for further training and development
16. Feedback shall be provided to the user within 1 second under normal conditions if object detected
19. Robot shall be controllable through web interface using a keyboard and mouse
20. Camera shall be controllable through web interface using keyboard and mouse

As of writing, ERs 9, 19, 20 have been accomplished through the tkinter GUI that has verified accurate maneuverability of the robot. ERs 15 and 16 have been tested and verified through 2D point cloud generation of the lidar sensor.

6.2.2. Testing

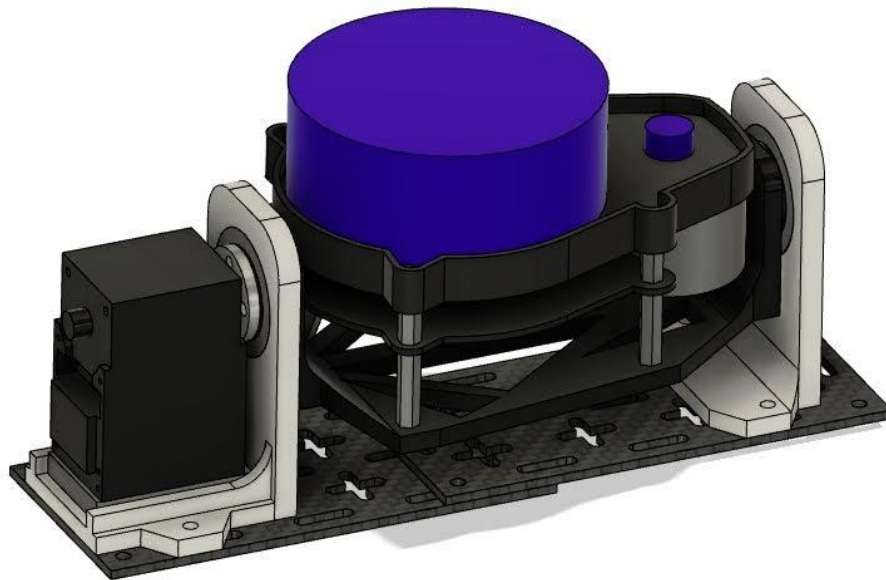


Figure 24: 3D Lidar Design

For the navigation systems, the testing of the lidar sensor will involve a comprehensive evaluation of its capabilities, focusing on 2D point cloud SLAM and 3D point cloud functionality with adjustment of the angle of the sensor through a servo motor shown in Figure 24. The 2D point clouds generated from tests have been plotted and assessed. However, we must now conduct controlled navigation tests within designated environments. This will allow the sensor to collect data to create accurate maps and evaluate J.A.R.V.I.S.'s ability to simultaneously locate itself and map the surroundings in real time.

For the 3D point cloud achieved through control of a servo motor, tests will be performed to examine the sensor's ability to capture detailed environmental data at various heights. The servo will be manipulated to vary the height of the lidar, allowing the ability to assess its adaptability to different scenarios and terrain.

To enhance the visualization and assessment of acquired data, we will use Foxglove, a visualization tool. This will ensure accuracy and clarity of the generated 3D point clouds and SLAM maps. Additionally, the fusion step involving the camera is employed, particularly the lidar's capability to sense fiducials representing the home base. These tests entail the strategic placement of fiducials within the environment, evaluating J.A.R.V.I.S.'s proficiency in detecting and navigating towards them using data from both lidar and camera sources. This fusion step plays a pivotal role in enabling semi-autonomous navigation back to the home base.

Finally, we will rigorously test the path planning and object detection algorithms by simulating various scenarios that challenge J.A.R.V.I.S.'s navigation capabilities. This testing will involve dynamic object placement and complex environments to ensure the capability of the motor control system and the accuracy of the point cloud map for both path planning and obstacle detection. Through these comprehensive tests, we aim to validate the effectiveness and reliability of the lidar sensor subsystem in enhancing J.A.R.V.I.S.'s navigation capabilities.

6.3. Hardware Communications

6.3.1. Summary

Hardware communications, encompasses both how the hardware devices on the robot communicate and send data to each other, as well as how the hardware interfaces with and sends data to the software. The hardware communication is primarily between the NUC, microcontrollers, speakers, monitor, microphone, and camera, which all need to be properly communicating with each other for the robot to function. The data being sent needs to be accurate and fast so that the remote user experience is not hampered in any way. This sub-project also includes implementing any additional hardware that has not already been installed onto the robot.

Although the software has been entirely overhauled, the hardware remains largely the same as in the 1.0 version of the project. As such, not many hardware design decisions needed to be made for this subsystem. However, we have made many additions such as the 360-degree camera, battery display, and streaming indicator LED, which will be discussed in this section. The relevant marketing requirements to this subproject include MRs 1, 2, 4, 6, 7, and 10, and the relevant engineering requirements include the following:

2. Camera viewing angles shall be 360 degrees.
7. User should be able to see remaining battery power available.
8. Robot should keep track of battery level within 3% accuracy
6. Robot should indicate video streaming is in progress via LED that is visible from at least 6 feet away
11. Wiring should be completely enclosed in a casing
12. Robot should have no sharp edges
13. Robot should weigh no more than 50 pounds
20. Camera shall be controllable through web interface using keyboard and mouse

The J5Create camera is connected to the NUC via USB and acts as a regular web camera that can be switched between a regular view and a 360-degree mode via a button on the camera. It was chosen for its low cost, size, and weight, with its lower quality of 720p being acceptable because Zoom automatically compresses video to 720p. Tables 56 through 60 show the AHPs which we used to choose which camera was the best for the robot. It was originally mounted to a servo motor but given that the motor got damaged during testing and the new camera requires the ability to look directly up to utilize the 360-degree view, we have opted to change the design. It has 2 stepper motors, one that controls the rotating base for horizontal movement and another that controls the rod for the vertical movement. The stepper motors will be connected to an Arduino which will receive inputs from the web server and move the camera through byte level commands.

The battery reading display uses an Arduino to get a voltage reading from the battery. Since Arduinos are only capable of reading up to 5V and the battery is 48V, the voltage is stepped down via a voltage divider, which is shown in Figure 25. We then used the formula shown in EQ 1 which calculates the charge in a 48V lithium-ion battery given the voltage across it. This battery charge reading is then sent to a 7-segment display near the battery so it easily visible to everyone around the robot, and it is shown on the remote user interface so that the remote user is aware of how much charge is left.

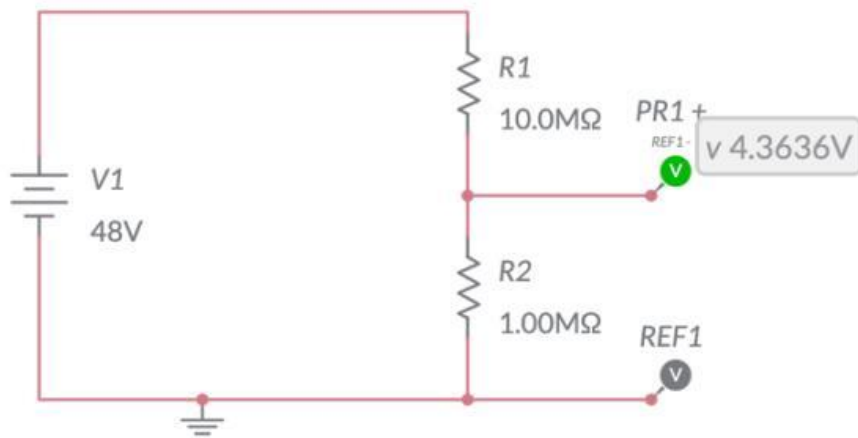


Figure 25: Voltage Divider Circuit for Battery Display

$$charge = (voltage - 39) * 6.41$$

EQ (1)

An LED will be connected to the Arduino, which will receive a bit from the web server that will be turned on if streaming is currently in progress. We will have to work to figure out how to seamlessly send the data from the web server to the Arduino. The LED will then slowly blink so that people nearby can easily tell that streaming is in progress. A screenshot of the code that is written is shown in Figure 26. All that needs to be done is for the streaming bit to be sent to the Arduino.

```

// Define the pin where the LED is connected
const int ledPin = 13; // Change this to actual pin

// Variable to store the streaming status (0 or 1)
int streamingStatus = 0;

void setup() {
    // Initialize the LED pin as an output
    pinMode(ledPin, OUTPUT);

    // Set the initial state of the LED to off
    digitalWrite(ledPin, LOW); // Initially off
}

void loop() {
    // Read the streamingStatus from Zoom
    // Toggle for the sake of testing
    streamingStatus = !streamingStatus;

    // Control the LED based on the streamingStatus
    if (streamingStatus == 1) {
        // If streamingStatus is 1, turn on the LED
        digitalWrite(ledPin, HIGH);
    } else {
        // If streamingStatus is 0, turn off the LED
        digitalWrite(ledPin, LOW);
    }

    // Delay for a while to see the LED status change
    delay(1000); // Delay for 1 second
}

```

Figure 26 : Streaming LED Indicator Code

We will now briefly touch on miscellaneous hardware pieces which have already been integrated onto the robot, as well as hardware pieces which only required minimal integration. Hardware pieces that were already integrated onto the robot include the NUC, microphone, motor drive, and motors. The NUC was already set up to run Ubuntu, the microphone connects to the NUC via Bluetooth, the motor drive connects to the NUC via USB and controls the motor by being sent byte level commands, and the motor is connected to the drive so that it can be controlled. The additional hardware pieces include a Bluetooth keyboard and mouse which we opted for so that people working on the robot have an easier time navigating through the NUC, as well as a USB hub. The USB hub is connected via USB, but it needs to be powered, so I will have to work with Hasib to design a way to get power from the battery to it. We also plan on adding a wooden board that the mouse and keyboard will be set on for ease of use.

6.3.2. Testing

There are 9 engineering requirements which are related to this subproject, so there are 9 separate tests that need to be done and successfully completed for this subproject to be

considered fully complete. This subsection will detail experiments I will perform to verify that parts of the subsystem are working, as well as which requirements are satisfied by that part of the subsystem.

The battery display is related to ERs 4 and 5. I have already designed and completed a test for this part in which I compared the battery charge gotten from the Arduino to the charge gotten from measuring the voltage across the battery using a voltmeter and calculating the remaining charge. The results of my testing is shown in table 39. As can be observed, the charge read from the Arduino is close to the charge gotten from the voltmeter, but it is not within 3%, so adjustments will have to be made to the voltage value read by the Arduino, as it is an average of 0.47 V lower than the one read from the voltmeter.

Table 36 : Battery Charge Reader Test Results

Time	Voltage Divider Voltage	Voltage Divider Charge	Voltmeter Voltage	Voltmeter Charge
11:31	54.41	99	54.74	100
11:41	54.44	99	54.84	100
11:51	54.44	99	54.82	100
12:01	54.38	99	55.07	100
12:11	54.35	98	55.07	100
12:21	54.34	98	54.97	100
12:31	54.23	98	54.87	100
12:41	54.35	98	54.37	99
12:51	54.07	97	54.33	99

The streaming indicator LED is related to ER 6. To test it once to program was been written in conjunction with Ben, I will begin streaming from the remote user interface and verify that if I am streaming, the LED is blinking and that the LED is off whenever it is not streaming. I will also verify that the LED is visible from 6 feet away to satisfy ER 6.

The 360-degree camera along with the camera mount is related to ERs 2 and 20. The camera has already been confirmed to be fully working, so next the design of the mount needs to be tested. I will do this by manually controlling the stepper motors through the Arduino and a rudimentary user interface just so that I know which movements work. Once I have gotten this part finished, I will work with Ben, who oversees the remote user interface, to implement the camera movements within the GUI so that the remote user is able to control the camera. Once this has been finished, I will go into the remote user interface and move the camera to ensure that I am able to see 360 degrees around.

The final 3 ERs listed, 11, 12, and 13, are not explicitly related to the hardware communications, but they are things that need to be kept in mind when making any physical changes to the robot. I need to make sure the wiring is all in an area that is enclosed enough that it can be completely covered with a reasonably sized casing, the hardware should not make it weigh more than 50 pounds, and I should avoid hardware that has sharp edges.

6.4. Remote User Interface

6.4.1 Summary

The remote user interface is the page responsible for allowing communication between the robot and the remote user that is connected to the robot. The interface will be a basic screen, with the Zoom meeting that the user is connected to displayed on the left side, and controls for the user to move the camera and the motors on the robot on the right side of the screen. Due to our switch away from the use of ROS, communication between the remote user and the robot itself will be different than the previous team. All code for the project can be found at:

<https://github.com/bryceag11/JARVIS/>

The remote interface is built from demo code in Zoom's SDK package. In order to allow the user to connect to the meeting, there are some files that need to be edited. In the JARVIS/Web_Server folder, the relevant files are located in Components/public. This contains one more folder called "tools" as well as 3 HTML document – `cdn.html`, `index.html`, and `nav.html` – and `cdn.css`. The `nav.html` file is the file deals with the remote landing page, allowing the remote user to join the Zoom meeting. This landing page was shown back in Figure 8 in the Design Summary, but for readability's sake, a screenshot of what the remote user sees upon connecting to this page is also shown below in Figure 27.

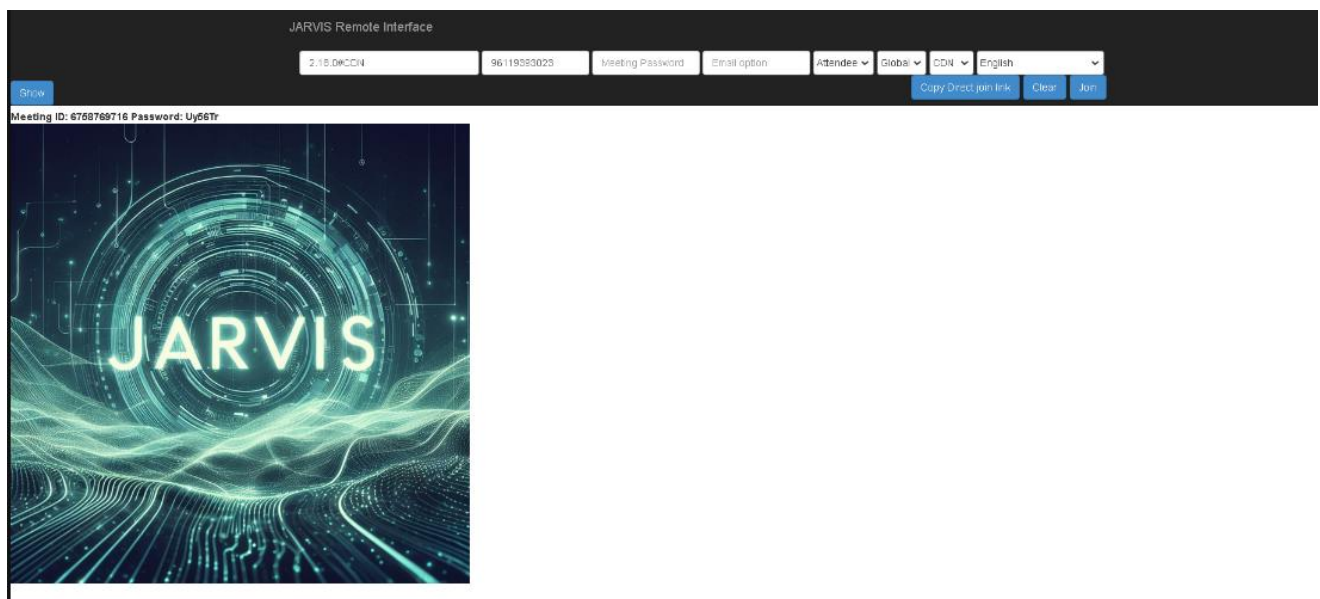


Figure 27 - Remote Landing Page

This landing page allows the user to enter in the Meeting ID to join the Zoom call hosted by the robot. This HTML file also has an associated `nav.js` Javascript file that is located in the tools folder. This Javascript file requires some updating from the demo package, as there are a few lines that need to be edited to allow for connection to the right address, hosted by the robot. On lines 16 and 23, the new `SDK_KEY` and `SDK_SECRET` needed to be updated to our team's Client ID and Client Secret ID given to us by Zoom.

Back in the Components/public folder, the `cdn.html` file is the main Remote Interface page, which also has an accompanying file in the tools folder called `cdn.js`. These are the files that allow the Zoom meeting and the controls to be shown in the browser. The HTML and Javascript files contain all of the functional code that allows the interface to behave the way it

should. The `cdn.html` is the most significant file in terms of the Web Server. It starts off with all the necessary source files. After a few lines regarding style, is the code for the buttons. This code is shown below in Figure 28.

```
42     <div class="side-buttons">
43         <div class="navigation-mode-title">Navigation Mode</div>
44         <button id="auto-btn" class="rectangular-button" onclick="Auto()">Autonomous Navigation</button>
45         <button id="man-btn" class="rectangular-button" onclick="Man()" disabled>Manual Navigation</button>
46     </div>
47
48     <div class="controls-container">
49         <!-- ... Existing code for autonomous and manual buttons ... -->
50
51         <div class="speed-section">
52             <div class="robot-speed-title">Robot Speed</div>
53             <button id="home-base-btn" class="speed-button" onclick="Home()">Home Base 5%</button>
54             <button id="low-btn" class="speed-button" onclick="Low()" disabled>Low 25%</button>
55             <button id="med-btn" class="speed-button" onclick="Med()">Med 50%</button>
56             <button id="high-btn" class="speed-button" onclick="High()">High 75%</button>
57         </div>
58     </div>
59 </div>
```

Figure 28 - Navigation Commands within `cdn.html`

There are similar controls for the Motor Controls and Camera Controls classes in the following lines. After the code for the buttons is implemented, the next step was the WebSocket code.

The WebSocket connection is what allows the webpage to communicate with the NUC, and there could be no communication without it. This code is shown below in Figure 29. The IP address of the NUC, and the port in which the server is hosted must be hard-coded into the function, otherwise the connection will not be established. In our case, the IP address of the NUC hosting the server was 10.47.244.128, and the port was 8000.

```
148     let socket = new WebSocket("ws://10.47.244.128:8000/");
149
150     socket.addEventListener("open", function(event) {
151         console.log("[open] Connection established");
152     });
```

Figure 29 - WebSocket Initialization

The next section of code is for sending a dynamic value of the battery remaining on the robot to the webpage. This code sets up an event listener on the WebSocket to listen for incoming messages from the server. When a message is received, it will log the received data to the console and update the battery percentage shown on the webpage. Once the function is called upon message reception, it uses “`document.getElementById`” to find the HTML element with the ID “`battery-value`,” which is found in line 39, updates this element to the value received, and

then changes the font color depending on the value of batteryValue. This code is shown below in Figure 30.

```
150     socket.addEventListener("open", function(event) {
151         console.log("[open] Connection established");
152     });
153
154     socket.addEventListener("message", function(event) {
155         console.log(`[message] Data received from server: ${event.data}`);
156         // Handle incoming messages from the server if necessary
157         // Example: Perform actions based on the received data
158         const batteryValue = document.getElementById('battery-value');
159         if (batteryValue) {
160             batteryValue.textContent = `Battery: ${event.data}%`;
161             if (event.data >= 75) {
162                 batteryValue.style.color = 'green'; // Set font color to green for high battery
163             }
164             else if (event.data >= 25) {
165                 batteryValue.style.color = 'orange'; // Set font color to orange for moderate battery
166             }
167             else {
168                 batteryValue.style.color = 'red'; // Set font color to red for low battery
169             }
170         }
171     });
```

Figure 30 - Battery Percentage Code

The next section of the code makes sure the socket connection is still open, and then declares the functions `isOpen()` and `wsSend()`. These functions are responsible for sending messages between client and server using the WebSocket. This section of code sets up another event listener to handle the opening and closing of the socket, as well as checking if the socket is open and sending messages if it is. There are also some lines for error handling. This code is shown below in Figure 31.


```

174     socket.addEventListener("close", function(event) {
175         console.log("socket closed", event);
176     });
177
178     function isOpen(socket) {
179         return socket.readyState === socket.OPEN;
180     }
181
182     function wsSend(msg) {
183         if (isOpen(socket)) {
184             socket.send(msg);
185             console.log("Message sent:", msg);
186         } else {
187             console.log("ERROR: Socket is closed");
188         }
189     }
190 }

```

Figure 31 - isOpen() and wsSend() Functions

The next chunk of code are the functions that send commands between the remote interface and the robot. The camera and motor commands use similar syntax, so they will be discussed simultaneously. Below, in Figure 32 is an example of the NavForward() and NavForwardClick() commands. When these functions are called, they send a log to the console to explain what happened, and then send a command using the setInterval and wsSend functions. These commands are then received through the WebSocket connection and handled for operation.

```

198     // Navigation Functions
199     function NavForward() {
200         console.log("Up button held");
201         sendInterval = setInterval(() => wsSend("W"), 100);
202     }
203     function NavForwardClick() {
204         console.log("Up button clicked");
205         wsSend("T");
206     }
207

```

Figure 32 - Navigation and Camera Commands

The next few functions deal with changing the speed of the robot. An example is shown below in Figure 33. The function used in this example is the Home() function. These functions

are called by clicking the different speed buttons on the remote interface: Home Base, Low, Medium, or High. These functions scan `cdn.html` for the associated button ID, which is `home-base-btn` in this case. The function also uses the WebSocket connection to send an associated command in order to properly change the speed of the robot.

```
// Speed Commands
function Home() {
  // Disable the 'Home' button and enable other buttons
  document.getElementById('home-base-btn').disabled = true;
  document.getElementById('low-btn').disabled = false;
  document.getElementById('med-btn').disabled = false;
  document.getElementById('high-btn').disabled = false;

  // Send WebSocket command
  wsSend("home");
}
```

Figure 33 - Changing Robot's Speed Commands

The final section of code in the `cdn.html` file are the functions for autonomous and manual navigation. These functions are shown below in Figure 34. These functions, similar to the speed commands, scan `cdn.html` for the button id and update them with values. This ensures that if a user is in Manual Navigation mode, that the Autonomous Navigation mode does not interfere with their desired controls. The functions end by sending a WebSocket command. Beneath these functions is code to create event listeners for the button clicks indicating which mode to operate in.


```

function Man() {
    document.getElementById('auto-btn').disabled = false; // Enable Autonomous Navigation button
    document.getElementById('man-btn').disabled = true; // Disable Manual Navigation button
    // Add your code to control the robot manually based on user input
    // For example, using your existing manual navigation functions
    wsSend("Man");
}

// Function to control the robot autonomously
function Auto() {
    document.getElementById('man-btn').disabled = false; // Enable Manual Navigation button
    document.getElementById('auto-btn').disabled = true; // Disable Autonomous Navigation button
    // Add your code for autonomous navigation here
    wsSend("Auto");
}

// Event listener for the Manual Navigation button click
document.getElementById('auto-btn').addEventListener('click', Auto);
document.getElementById('man-btn').addEventListener('click', Man);

```

Figure 34 - Autonomous and Manual Navigation Functions

As a reminder, these files work in tandem with those located in JARVIS/Main_App, JARVIS/SLAM_Visualization, and JARVIS/YDLidar_SDK in order to correctly handle the WebSocket functions to ensure the robot behaves as the remote user intends.

This subsystem is vital in dealing with the integration of the project as a whole. It deals with launching the server and the WebSocket connection that allows for two-way communication between the robot and the remote user. The code used in this subsection is integrated with the code for the navigation of the robot which was discussed in Section 6.2, Navigation Systems. The integration of these two systems allows for the robot to operate under safe conditions, while giving the remote user freedom and flexibility to operate the robot how they see fit. The integration of this subsystem was also important in dealing with Erick's subsystem, Hardware Communications, which was discussed in Section 6.3. The WebSocket and serial connections that are established in the code associated with this subsystem allow communication between the remote interface and hardware elements on the robot, such as the camera or the microcontroller which reads the battery. Successful communication between these two subsystems allows the user to see the remaining battery percentage, and control the camera on the robot, which helps satisfy some of the Engineering Requirements and Marketing Requirements that guided the construction of this robot. There are several Marketing and Engineering Requirements that rely on the completion and integration of this subsystem. These will be discussed in the Testing and Validation section of this report.

6.5. Home Base and Power Delivery

6.5.1. Summary Home Base

The home base is constructed of a square shaped box made of wood. The dimension of the box is 15 x 15 x 15 inch where the back side of the box is open. We have selected a comparatively large box because when the robot contacts the charging station, the robot does not displace the charging station from its designated place. The power supply unit will be placed inside of the box which can be accessed from the back. In Figure 35, the front side of the design of the charging station is shown.

Our team's primary objective is to enhance the telepresence robot created by the previous group. One key focus area for improvement is the robot's semi-autonomous movement with the home base. To achieve this, we are implementing a charging station where the telepresence robot can effortlessly navigate to recharge its battery. This station, referred to as a home base, will guarantee a consistent and uninterrupted power supply. It will be equipped with an AC to DC power converter, which will receive 120V of AC power from a wall outlet and convert it to DC power, enabling the lithium-ion battery to charge.



Figure 35: Front Side of Charging Station

The relevant marketing requirements for this sub project are 6, 12, and the relevant engineering requirements for this sub project are 4, 5, and 19.

The front of the charging station is a connection point that will hold two metal bars large enough so that the connection point has enough forgiveness for the other connection point (mounted on the base of the robot) to contact robot for charging the battery. Two metal bars will be housed inside a 3D plastic box is shown in Figure 36.

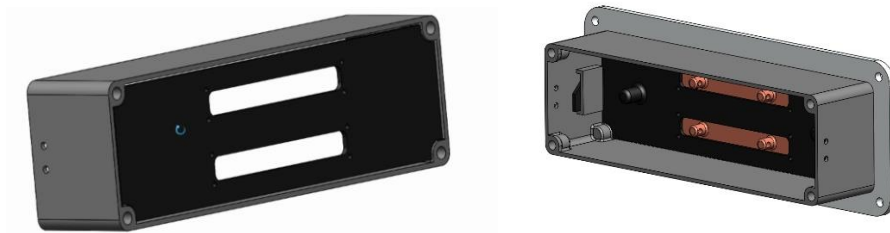


Figure 36: Front and Back View of Connection Point

The plastic part that will hold two metal bars made of aluminum will have two flanges as shown in the design Figure: 37.

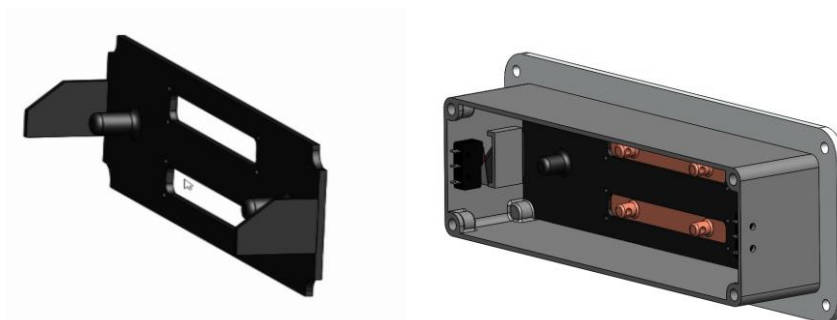


Figure 37: Flange of 3D Print (left) and Limit Switch (right)

When the base of the robot's connection point will contact the home base connection point then the flange will push inside and trigger the micro limit switch which will make connection to the power supply unit complete, the power can flow from the charging station to the battery. These 3D printed plastic box (Figure: 36) will hold two metal bars made of aluminum and the design of the metal bars shown in the Figure 38

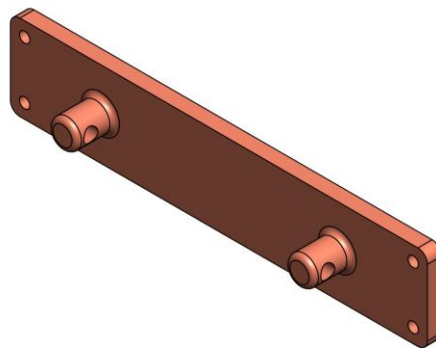


Figure 38: Home Base Metal Bar Connection Point

The back of the metal bars is the place where wires will be connected. One metal bar will be connected to the negative side of power supply unit and positive side of the power supply first will be connected to the micro limit switch's always open or NO side and the COM side will be

connected to the other metal bar. The micro limit switch is a mechanical switch shown in Figure: 35. When the wire is connected to the NO side of the limit switch the circuit remains open and if the limit switch is activated by pushing the metal bar shown in Figure 39 the switch becomes activated and circuit becomes closed. Then the power will flow through the switch by the wire to the metal bars. In our case the switch will be activated when the connection part attached to the robot base contacts home base connection part then the flange will push inside the box and activate the limit switch.

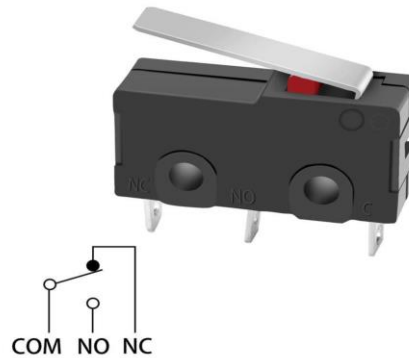


Figure 39: Micro Limit Switch

The 3D plastic with the flange shown in figure 33 also has two columns expanded towards the z-axis. These two columns will be attached to two springs, and the other end of the springs will be attached to the back panel of the box that is shown in figure 40. When the robot disengages after the charge the back plastic plate spring will push back the front plastic plate to its original state as a result the limit switch will on the NO side that will make the circuit open again. When the circuit is open no power will flow through the metal bars, and it will be completely safe.

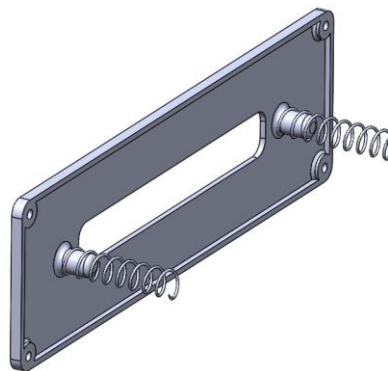


Figure 40: Back Plate of Home Base Connection Point

The second part of the connection point will be attached to the base of the robot. The original plan was to attach the connection point to the back of the robot's base but after much deliberation we decided to attach it in front of the base of the robot. For that reason, we had to

make some changes to the design. For connection point we will use two aluminum metal bars that are designed shown in figure 41.

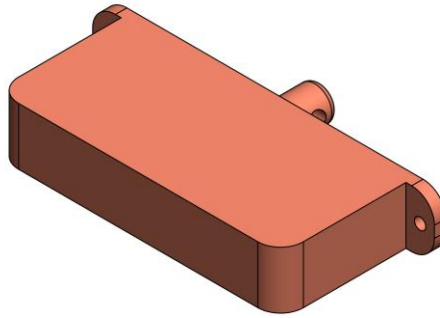


Figure 41: Robot Base Connection Part

These two metal bars will be held together by 3D printed rectangular shape box that is shown in figure 42.

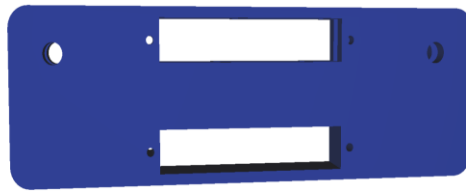


Figure 42: 3D Printed Plastic Part Robot Side

The back side of the metal parts will be wired to the rechargeable battery with the help from a barrel jack. To attach to the front side of the robot we took a piece of wood and hollowed out in the middle, so it looked like half side of a rectangular shaped box. The reason we hollowed out the middle so that wire can run to the battery. We also had to consider the front side of the wheel and make the z-axis of the wood a little long. In figure 43 shows the top and front view of connection point of the robot side.



Figure 43: Top and Front View of Robot Connection Point

Metal bars are attached to the 3D printed plastic box with the help of screws and the plastic part is attached to the wood with the help from plastic to metal adhesive. The whole mechanism is then glued to the front base side of the robot as shown in figure 43.

In the process of building the charging station we made some calculation error regarding making an entrance at the front side of the charging station. As a result, we faced difficulties setting up the rectangular shaped plastic box from the inside of the wooden box. We decided to use adhesive to attach the plastic part in front of the charging station as shown in figure 44. The 3D printed plastic box parts were supposed to be screwed together but we found that screw holes were not the same size so at the last minute the backplate was glued. The micro limit inside the box is glued also.



Figure 44: Front Side Charging Station

6.5.2 Summary Power Delivery

In terms of the power supply, our requirement is a unit that can deliver 48V DC current, as we are utilizing the previous team's 48VDC, 10000mAh VISET lithium battery. To fulfill this, I conducted thorough research on two potential power supply products: the MEAN WELL LRS-350-48 and the DROK 48V power supply.

To make an informed decision, we performed AHP and listed the results in a table, considering factors such as compatibility, safety, reliability, and affordability. Each of these factors are calculated and included in appendix section C Table 52 to Table 55. Through pairwise comparisons and AHP methodology, we derived the weights for our evaluation criteria. The tables displaying the geometric mean and respective weights can be found in table 40.

Table 37 : DROK 48V VS MEAN WELL Power Supply

Criteria	DROK 48V	Mean Well LRS-350-48
Compatibility	0.503	0.497
Safety	0.495	0.505
Reliability	0.410	0.590

Criteria	DROK 48V	Mean Well LRS-350-48
Affordability	0.375	0.625
Geometric Mean	0.431	0.535
Weight	0.191	0.809

The Mean Well LRS-350-48 surpasses the DROK 24V power supply in terms of safety, dependability, and cost-effectiveness. Although the DROK 48V exhibits a slightly higher compatibility score, overall, the Mean Well LRS-350-48 emerges as the superior choice according to the AHP methodology, with higher geometric mean and weight scores.

Both devices feature compatibility in terms of connector types and voltage ranges, rendering them suitable for a wide array of applications. However, the Mean Well LRS-350-48 boasts a higher current capacity, making it more versatile compared to the DROK 48V. Additional safety measures, including overvoltage and overcurrent protection, set the Mean Well LRS-350-48 apart by providing enhanced security for both the device and the user. Conversely, the DROK 48V lacks these protective features. In terms of reliability, the Mean Well LRS-350-48 outperforms the DROK 48V due to its higher MTBF (mean time between failures) and longer lifespan. While the Mean Well LRS-350-48 comes at a slightly higher cost than the DROK 48V, its superior performance in other aspects makes it a more advantageous long-term investment.

In conclusion, despite both the DROK 48V and Mean Well LRS-350-48 power supplies sharing similar connector types and voltage ranges, the AHP strongly recommends the Mean Well LRS-350-48 due to its higher ratings in safety, reliability, and cost-effectiveness.

Another critical aspect of the power delivery system is the plug connectors that facilitate the connection between the charging battery and the power supply point of the charging station. These connection points should offer enough flexibility to enable the user to easily guide the robot to the charging station, allowing for smooth docking or contact with the home base for recharging. To achieve this, we propose utilizing rectangular metal plates, such as copper, which will be attached to both ends of the charging battery and the charging station.

When the user guides the robot to the charging station and establishes contact, the battery will initiate the recharging process from the station. This forms the core of our charging point design concept, and we plan to implement this approach in building the connection points for the robot.

Testing 6.5.3.

Power Supply Unit

The Mean Well power supply unit comes with voltage regulator. We used the voltage regulator to set the power supply to 48V and the used multimeter to measure voltage. We were able to confirm that voltage reading was 48V. We took several readings to make sure that voltage is correct. For power delivery, we have set the power supply voltage to 54.5V because according to rechargeable battery charge converter the cut off voltage is 54.6V and we wanted to be close to the value.

Limit Switch

We wired up the micro limit switch with power supply unit and the metal bars of home connection point. We connect the NO side of the limit switch with the positive side of the power supply and other end to positive marked metal bar. We also then connect the negative side power

supply with the other metal bar as shown in figure 45, where red wire indicates the positive and green wire as negative. After making the connection we tested by measuring the voltage across the metal bars. When connected and micro limit switch was not activated there was no voltage across the metal bars. After pressing the micro switch, we were able to read voltage in our multimeter.

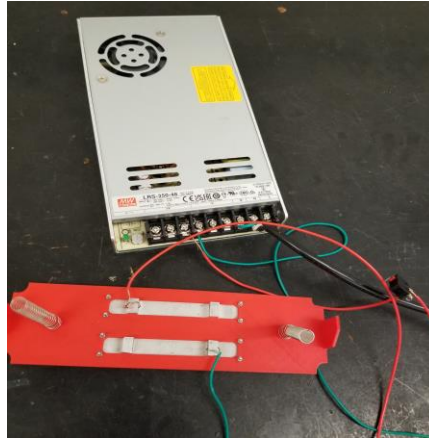


Figure 45: Wire Connection Home Base

Robot Base

Same way we also wired up the robot side with the connection point to the rechargeable battery and able confirm that the metal bars carry 48V. After that we contacted the home base connection point and robot base connection point to measure the current. We used DC multimeter to confirm when the robot base connection point pressed against the home base connection point and the limit switch is activated then there were DC current in the connection point and the DC multimeter showed 1.5mA current reading. We concluded that our charging mechanism is working.

Final Testing

We set up our home base to wall outlet and try to move the robot to the home base to see if the connection point makes contact. We were able to make full contact with the home base and concluded it was successful.

7. Presentation of Final Design

This section will describe the final design of our prototype. This will discuss the final hardware and the software design of the project. This will include a section for the robot structure, the home base structure, and the software design.

7.1. Robot Structure

Shown below in Figure 46 is the final prototype of the Robot Structure. We did not change too much from the previous team's robot structure. Most of our changes were aesthetic changes. The most notable change made is that we painted the main pole and the peripherals to improve the appearance of the robot.



Figure 46 - Final Robot Structure

We added a basket on the front of the robot in case the robot would need to carry anything, as there was no longer room to store anything on the base of the robot. We also added a USB hub on the back side of the main pole, as adding new features required more ports than we initially had available. This is shown in Figure 47.

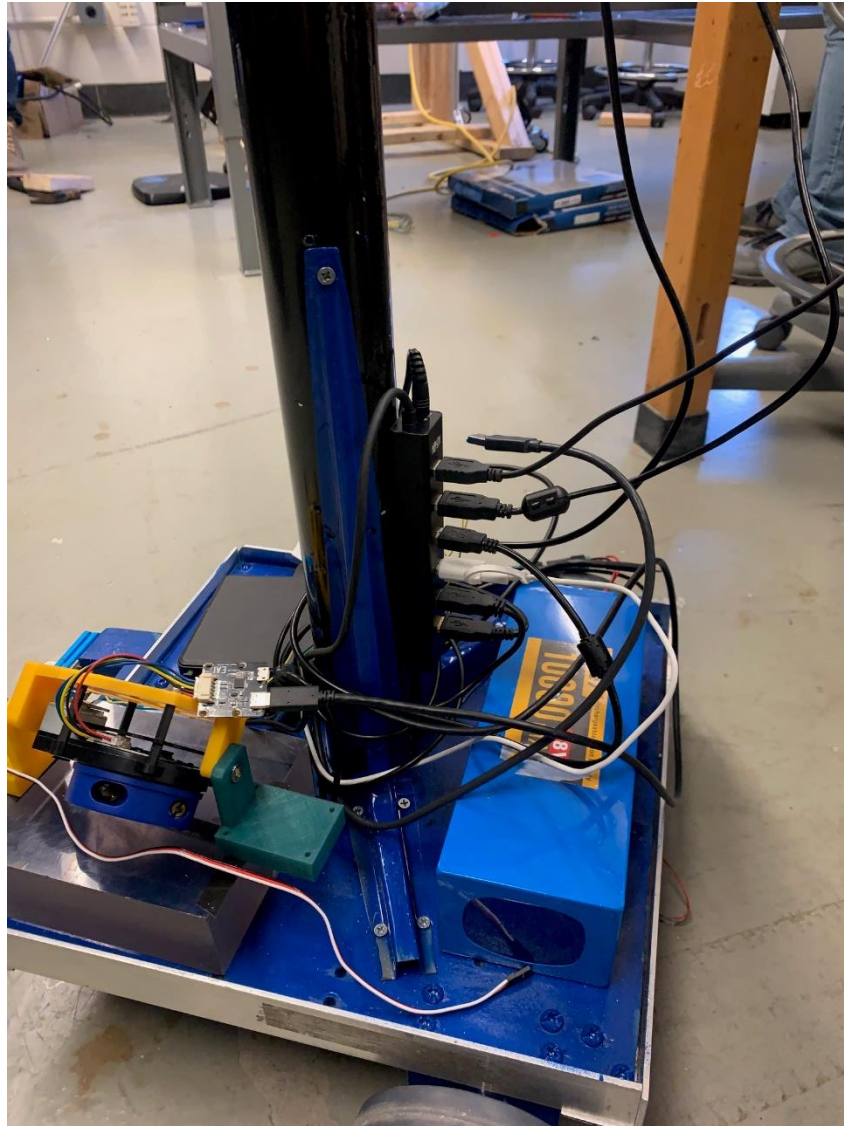


Figure 47 - USB Hub

Another change we made to the robot structure was the addition of the home base charging leads. These leads come in contact with the leads on the home base in order to trigger the limit switch and begin charging. The leads on the robot are shown below in Figure 48.

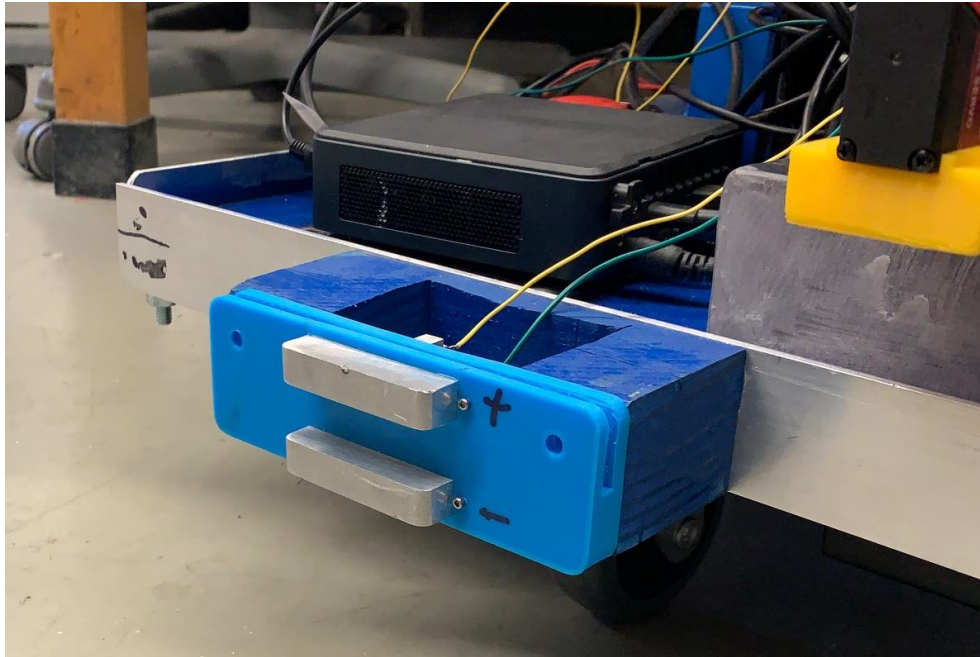


Figure 48 - Robot Charging Leads

The final large change made to the robot structure is the modifications to the top camera. As we wanted to not only give the user a 360-Degree camera, we also wanted to provide them with 360-Degrees of motion, accomplished with stepper motors. Creating this new apparatus required us to create an entirely new top piece to hold the mount as well as the wiring needed to operate it. This new structure is shown below in Figure 49.

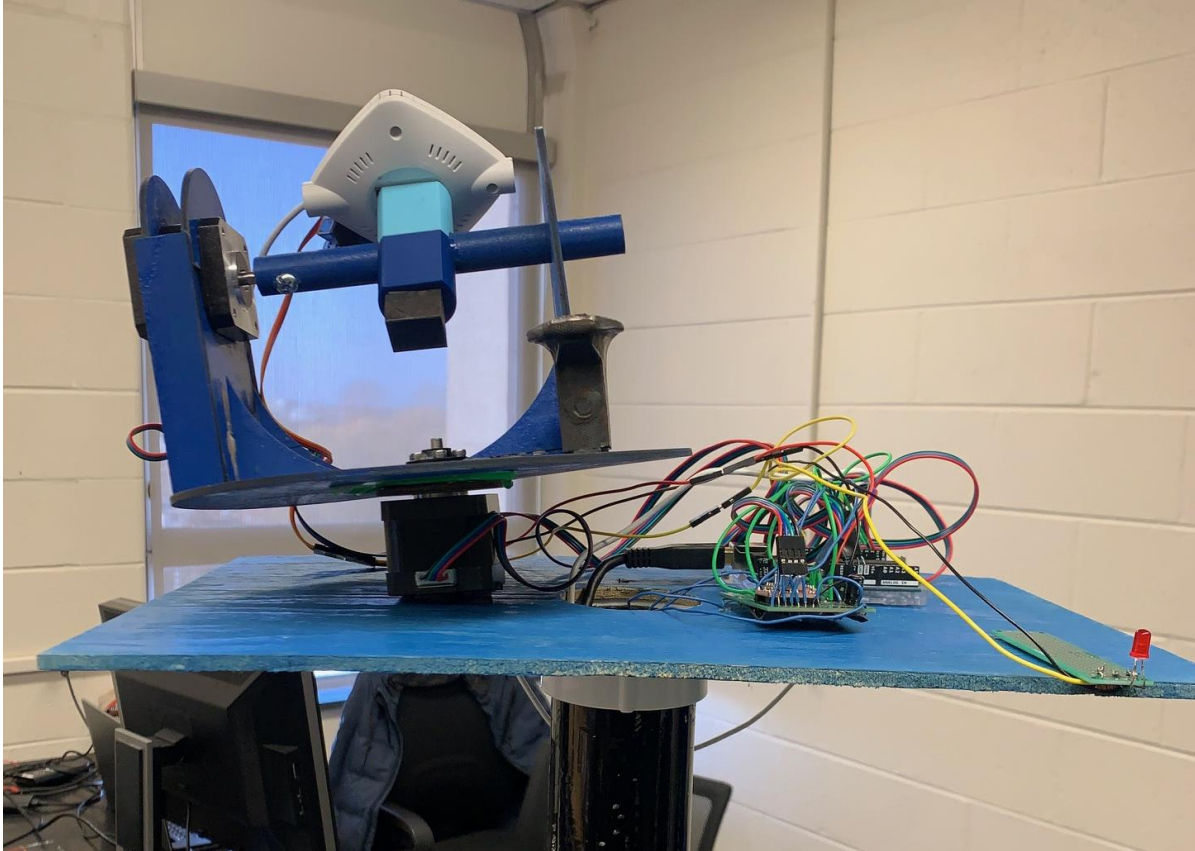


Figure 49 - Camera Mount Structure

7.2. Home Base Structure

The home base as we have discussed in section 6.5.1 is a cube shaped box made of wood and the charging mechanism is attached in front of the box. The home box back side is open, and the power supply unit is housed inside of the square shaped box shown in figure 50. The power supply unit will be connected to the wall outlet to 120V AC



Figure 50: Home Base Setup Back View

power source. Once the thick black cable is plugged into the wall, the charging mechanism is ready for use. As previously mentioned, the robot will drive up to the home base, and the leads shown in Figure 48 will come in contact with the leads attached to the front side of the home base shown below in Figure 51. Next, the limit switch will be triggered, and the charging will begin.



Figure 51 - Home Base Charging Leads

7.3. Software

All of the code for the software used in our version of the project can be found at :

<https://github.com/bryceag11/JARVIS/>

This code has been discussed in detail throughout this report. The github repository also includes an operation manual that details the steps that need to be taken to launch the server and access it from other devices on the network. To give a short summary:

1. Clone the github repository to your personal machine or download the folder as a .zip file.
2. Start a Zoom Meeting from the robot. Zoom is already installed, one would just need to start a meeting.
3. Navigate to the JARVIS/Web_Server/Components
 - a. Use the cd commands in the terminal to navigate to the correct directory.
4. Use the “npm start” command to launch the web server
 - a. The necessary URL will be given in the terminal similar to as shown below.

```
PS C:\Users\bmpud\Downloads\meetingsdk-web-sample-master\meetingsdk-web-sample-master> cd Components
PS C:\Users\bmpud\Downloads\meetingsdk-web-sample-master\meetingsdk-web-sample-master\Components> npm start

Compiled successfully!

You can now view websdk-component-demo in the browser.

Local:      http://localhost:3000
On Your Network:  http://172.20.112.1:3000
```

Figure 52 - Launching the Server

5. The remote user will now be able to navigate to the address specified.
 - a. In this example, the user wanting to connect would enter in <http://172.20.112.1:3000> in their browser.
6. Once the user reaches the landing page, they will enter the details of the meeting, including ID, password, and name they would like to join the meeting under, and then click “Join.”
7. Clicking “Join” will bring the user to the main interface page, which will enable them to control all available robot functions.
8. Please see the user manual located in the GitHub repository for further details regarding operation of the robot.

8. Testing and Validation

8.1. Marketing Requirements

ER #	Engineering Requirement	Testability / Verification
1	Robot shall be able to clearly communicate with the individuals surrounding it	Validated by connecting to the web interface as the remote user and communicating to nearby people.
2	Robot shall display a real-time 3D viewing of its surroundings to the remote user	Validated by connecting as the remote user and being able to look around with the camera for a real-time 3D viewing.
3	Robot shall be able to automatically avoid any obstructions or hazards that are in its path	Successfully validated by controlling the robot and ensuring that it does not run into objects even when the user attempts to.
4	Remote user shall be able to control the movement of the robot through an online user interface	Successfully validated through joining a zoom meeting from the local web server and controlling the robot.
5	Robot shall pose no safety risks to its surroundings	Validated by ensuring that all our safety measures put on the robot work as intended to prevent potential injury to others.

6	Robot shall have a 3-hour long battery charge	Validated by running the robot for over 3 hours at a time without any charge being lost.
7	Robot shall clearly indicate when video streaming is in progress	Validated by ensuring streaming LED works when streaming is in progress.
8	Robot should have a user-friendly interface for setup and user operation	Validated by allowing third party to use the interface to control the robot and confirm that it is intuitive.
9	Robot should implement a queuing system for users to wait for remote access	This marketing requirement was unsuccessfully validated as the queue system was not able to be implemented.
10	Robot should have a practical design and appearance	Partially validated since most of robot's practical design goals were met, but wiring is still exposed.
11	Robot should be competitively priced with similar products	The cost of the robot is \$2,838 which is significantly cheaper than Omnilab's \$6,000 telepresence robot and cheaper than Double Robotics' \$4,500 telepresence robot
12	Robot shall have a designated charging location that automatically charges the robot	This marketing requirement was validated through measuring the voltage and current through the closed circuit between the robot and the home base when it is plugged in and connected.

8.2. Engineering Requirements

ER #	Engineering Requirement	Testability / Verification
1	Robot shall redirect or stop at 3 feet from the detected objects	Successfully tested by driving the robot towards an object and measuring the distance at which the robot stops (3 feet).
2	Camera viewing angle shall be 360 degrees	Successfully tested by connecting camera to NUC and verify that the 360-degree panoramic view works properly
3	Upon object detection, the robot shall disable forward moving navigation control for the user	Successfully tested by driving the robot towards an object and verifying that once an object is detected, forward navigation is disabled

4	User should see the remaining battery power available	Successfully tested by logging onto the web interface and verify that battery charge remaining is seen.
5	Robot should keep track of battery level within 3% accuracy	Unsuccessfully tested as our current battery reader only has a precision of 10%.
6	Robot should indicate video streaming is in progress via LED that is visible from at least 6 feet away	Successfully tested by stepping 6 feet away from the robot while streaming is in progress and verifying that the LED can be seen, as well as verifying that it is off when streaming is not in progress.
7	Robot should contain an operation manual	Successfully tested, manual has been created and third party could use robot with only manual
8	Robot shall have a user-friendly interface for controlling the robot's movements and functions	Successfully tested by allowing third party with no knowledge of the project to connect to user interface and confirm its ease-of-use
9	Web interface should provide a queue for up to 10 users to wait for connection access	Unsuccessfully tested, as no queue implementation was made.
10	Robot should have no sharp edges	Tested by rubbing skin on robot and confirming that no damage is done.
11	Robot should weigh no more than 50 pounds	Successfully tested by placing robot on a scale and measuring its weight. It weighed 40.2 pounds.
12	Robot should be able to carry an object up to 5 pounds and up to 6x6x6 inches for remote user with proper stability	Successfully tested by putting items of varying weights in the basket to verify the maximum load that can be carried is at least 5 pounds, and by measuring the dimensions of the inside of the basket, which are 4x9x7.5.
13	Robot shall be able to store sensor data for further training and development	Successfully tested by recording and exporting point cloud data and using open3D to successfully visualize the data
14	Object Detection feedback shall be provided to the user within 1 second under normal conditions if object detected	Successfully tested by placing an object in front of robot and recording time it takes for user to receive feedback, which was under 1 second.
15	Robot should cost no more than \$3000	Successfully tested by measuring costs and verifying that all parts, including those inherited, is no more than \$3000.

		The cost of our inherited parts is \$1731.53. The cost of our purchased parts is \$1106.90. This brings the total cost of reproduction to \$2838.43
16	Real-time video streaming to user shall have a maximum latency of 150ms	Successfully tested by joining zoom meeting as remote user and checking streaming latency through Zoom (avg. 41ms)
17	Robot shall be controllable through web interface using a keyboard and mouse	Successfully tested by using the interface and verifying that the robot can be moved in all directions.
18	Camera shall be controllable through web interface using a keyboard and mouse	Successfully tested by logging in as remote user and checking that all directions can be used, and that the mode can be changed
19	Robot shall be rechargeable using home base	Tested through measuring a current of 1.5mA when connected and the home base connector was actuated and a voltage of 54V between the connector and robot.
20	Wiring should be completely enclosed in a casing	Unsuccessfully tested as all wires are still exposed.

8.3.Requirements Summary

For our Marketing Requirements, we had 8 “Shall” requirements and 4 “Should” Requirements. We successfully validated all 8 “Shall” Requirements. For our “Should” Marketing Requirements, we successfully validated 2, and unsuccessfully validated the other 2. The two we did not successfully validate were “Robot should have a practical design and appearance,” and “Robot should implement a queueing system.” We were unable to successfully validate “Robot should be able to have a practical design and appearance” as there were still wires exposed, and this was one of our Engineering Requirements, and it was not met. The other Marketing Requirement was unable to be met as we ran out of time to implement the queueing system. We spent our time making sure the robot and interface were usable for a single user before we had time to spend time optimizing it for multiple users.

For our Engineering Requirements, we had 10 “Shall” requirements, and 10 “Should” requirements. We were able to successfully test and verify all 10 of our “Shall” requirements. For our “Should” requirements, we were able to successfully test and verify 7 of these requirements, leaving 3 unsuccessfully tested. The ones that were unsuccessful were “Wiring should be completely enclosed in a casing,” “Web interface should provide a queue for up to 10 users to wait for connection for access,” and “Robot should keep track of battery level within 3% accuracy.” We were unable to complete the wiring being enclosed in a casing because we were working with the wires and cables up to the deadline, and did not have time to cover them up in some sort of casing. We were unable to complete the queue requirement as described in the previous paragraph. We were unable to keep track of the battery within 3% accuracy because of

the way we chose to measure the battery. Due to our choice in measuring voltage of the battery, and converting these to battery percent measurements, we were only able to obtain a reading within 10% accuracy.

For all requirements not met, we will discuss some recommendations for future teams in Section 10 of this report. We will discuss some things we wish we would have done differently in order to successfully accomplish these requirements. Overall, we had a 100% completion rate for our “Shall” requirements on both Engineering and Marketing Requirements. This shows that we did well in prioritizing our work focus and achieving the goals we set out to achieve with the highest priority. We had about a 64% completion rate of our “Should” requirements. Although we were unable to successfully accomplish these requirements for the reasons discussed above, we are still happy with the final result we were able to produce, and have provided discussion in Section 10 on how to approach these requirements differently.

9. Final Project Plan

9.1. Work Breakdown Structure

Table 38 : Work Breakdown Structure

ID	Task	Start Date	End Date	People	Predecessors
1	Navigation Systems			Bryce Grant	N/A
1.1	Navigation System Design (HW)	04/03	09/22	Bryce Grant	N/A
1.2	Lidar Object Detection (Software)	08/24	09/24	Bryce Grant	1.1
1.3	Beacon Mapping Sub-System Design (HW)	08/24	10/26	Hasib Hasan	N/A
1.4	Autonomous Path Planning	08/24	10/12	Bryce Grant	1.2
1.5	Semi-Autonomous Navigation to Base (SW)	09/07	10/26	Bryce Grant	1.3, 1.4, 1.2
1.6	Lidar Data Collection (SW)	04/17	09/24	Bryce Grant	1.1
1.8	Motor Feedback System (HW)	02/20	09/13	Bryce Grant	N/A
1.9	Manual Navigation of Robot (SW)	02/20	09/13	Bryce Grant	1.8
2	Motors & Encoders				N/A
2.1	Motor Feedback System	02/20	09/13	Bryce Grant	N/A

2.2	Manual Navigation of Robot (SW)	02/20	09/13	Bryce Grant	2.1
2.3	Lidar Feedback System (SW)	9/13	10/12	Bryce Grant	2.1, 1.2, 1.4
3	Power Delivery				N/A
3.1	Research Power Delivery System Components	03/02	03/13	Hasib Hasan	N/A
3.2	Develop Preliminary Design of Power Delivery System	03/14	03/17	Hasib Hasan	3.1
3.3	Develop Preliminary Design of Home Base	03/20	04/03	Hasib Hasan	N/A
3.4	Develop Preliminary Design of Connection Point	04/04	09/25	Hasib Hasan	N/A
3.5	Research AC to DC power converter for Home Base	04/05	04/10	Hasib Hasan	3.1,3.3
3.6	Select AC to DC Converter for Home Base	04/12	09/29	Hasib Hasan	3.1,3.3
3.7	Research on Safety System for Home Base	04/13	04/17	Hasib Hasan	3.3
3.8	Develop preliminary Design of Safety System	04/17	04/18	Hasib Hasan	3.7
3.11	Design the home base	04/11	10/04	Hasib Hasan	3.3
3.12	Build prototype Home Base	20		Hasib Hasan	3.11
3.13	Build Home Base	10/10		Hasib Hasan	3.12
4	Online User Interface				
4.1	Version 1 Code review	2/20	2/22	Ben Pudlowski	N/A
4.2	Battery Level Display Research and Design (SW)	2/23	2/28	Ben Pudlowski	N/A
4.3	Control of 360-degree camera Research and Design (SW)	9/4	9/8	Ben Pudlowski	N/A

4.4	Zoom Web SDK Integration Research and Design (SW)	9/19	10/16	Ben Pudlowski	4.1
4.5	Remote Navigation Control Research and Design (SW)	9/11	9/19	Ben Pudlowski	N/A
4.6	UI Prototype	10/16	10/23	Ben Pudlowski	4.1
5	Hardware Communication Interface				
5.1	360 Degree Camera Physical Implementation (HW)	26	10/13	Erick Calvillo	N/A
5.2	360 Degree Camera Programming (SW)	26	10/20	Erick Calvillo	5.1
5.3	Interface With Online UI (SW)	33	11/3	Erick Calvillo	N/A
5.4	Battery Level Reading (HW)	26	10/20	Erick Calvillo	N/A
6	System Integration				
6.1	SDK/Remote Interface with Navigation Controls	10/16	10/23	Ben Pudlowski, Bryce Grant, Erick Calvillo	
6.2	Navigation through interface Testing & Verification	10/24	10/24	Ben Pudlowski, Bryce Grant	1, 4
6.3	Object Detection & Navigation Testing	3		Bryce Grant	1, 2
6.4	Remote Interface Battery Testing & Verification	10/25	10/25	Ben Pudlowski, Erick Calvillo	4, 5.4
6.5	Localized Navigation to Home Base Troubleshooting & Testing	10/26	11/2	Hasib Hasan, Bryce Grant	1.5, 2.4, 3
7	Final Submission Work				

7.1	Final Presentation/Demonstration preparation	3		All Team Members	
7.2	Final Poster Design	3		All Team Members	
7.3	Final Report Updates	5		All Team Members	
7.4	Last Minute finalizations	2		All Team Members	

9.2. Gantt Chart

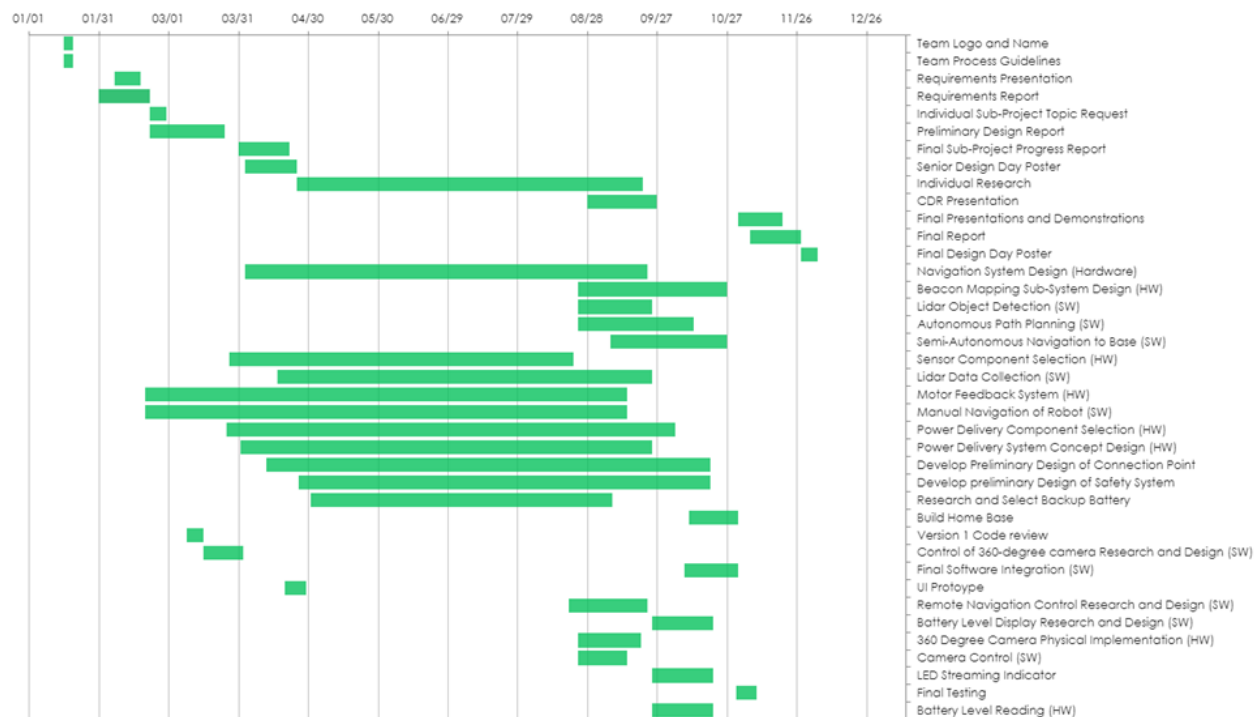


Figure 53 : Gantt Chart

9.3. Final Costs

Several of the costs that would normally be associated with the creation of a robot we will not have to consider, due to our inheritance of the previous robot. As previously mentioned, the team who developed the first robot already made some design decisions that we will be keeping. Shown in Table 42 is the bill of materials from the initial version of this robot. This includes all purchases made to create the first version of the robot, including materials to build the physical structure.

Table 39 : Original Bill of Materials/Inherited Parts

<i>Version 1</i>			
Purchased Parts			
<i>Category</i>	<i>Part</i>	<i>Quantity</i>	<i>Price</i>
Battery	VISET Ebike Battery	1	\$185.99
Motors	Devantech RD03 Motor Kit	1	\$235.44
Camera	Logitech 4K Pro Webcam	1	\$149.99
Microphone	ReSpeaker USB Mic Array	1	\$69.99
Display	ASUS 15.6" Portable Monitor	1	\$236.99
Display Mount	VIVO Universal VESA Mount	1	\$24.99
Speakers	Creative Pebble V3	1	\$39.99
NUC (Main Computer)	Intel NUC11PAHI5	1	\$439.99
Voltage Converter	48V -> 24V DC to DC Converter	1	\$170.00
Robot Structure	1-1/2 in. x 96 in. angle bar at 1/2 in. thick	1	\$52.27
Robot Structure	1/4 in. x 2ft. x 2 ft. plywood board	1	\$7.99
Robot Structure	3/4 in. x 2 ft. x 2 ft. plywood board	1	\$18.99
Robot Structure	1/4 in.-20 x 1/2 in. machine screw (100-pack)	1	\$9.97
Robot Structure	1/4 in.-20 hex nut (100-pack)	1	\$7.65
Robot Structure	1/4 in. lock washer (100-pack)	1	\$16.53
Robot Structure	2 in. x 10 ft. PVC schedule 40 Pipe	1	\$19.72
Robot Structure	2 in. PVC Pipe cap (4-pack)	1	\$12.32
Cables	10 ft. USB-C to USB-C power/video cable	1	\$18.99
Cables	6 ft. USB-A to USB Micro-B Cable	1	\$6.99
Cables	10 ft. USB-A to USB-B cable	1	\$5.99
Navigation/Sensor	Adafruit time of flight distance sensor	3	\$44.85
Emergency Shut-Off	Emergency stop push botton	1	\$69.51
Total (Purchased)			\$1,845.14

Due to having an inherited project, we need to include the parts purchased by the previous team, whose parts we inherited. Although we didn't spend this money ourselves, the parts still contribute to the total cost of the production of the robot. Shown in the table above are the parts that were purchased by the group who had the first iteration of the project. Table 43 shows the parts that we needed to purchase to produce the second iteration of the robot.

Table 40 : Purchased Parts

<i>Category</i>	<i>Part</i>	<i>Quantity</i>	<i>Price</i>
Motor	Devantech RD03 Motor Kit	1	\$235.44
Display	ASUS 15.6" Portable Monitor	1	\$209.99
Peripherals	HP Bluetooth Mouse	1	\$ 19.99
Object Detection	youyeetoo Lidar Sensor	1	\$ 79.99
Camera	j5Create 3D Camera	1	\$ 89.99
Peripherals	Spectrum Diversified Basket	1	\$ 12.08
Peripherals	Logitech Bluetooth Keyboard	1	\$ 29.99
Peripherals	Uxcell 7 Segment Display	10	\$ 6.50
Peripherals	Arduino Microcontroller	2	\$ 28.99
Peripherals	Stepper NEMA Motor (5pk)	1	\$ 8.99
Peripherals	Stepper Motor Drive Kit	1	\$ 4.95
Peripherals	Adafruit Digital Potentiometer	1	\$ 10.19
Peripherals	Assorted Capacitor Kit	1	\$ 9.99
Power Delivery	Power Switching Unit	1	\$ 34.80
Peripherals	WaveShare 30KG Serial Bus Servo	1	\$ 30.90
Peripherals	10pcs F684ZZ Mini Ball Bearings Double	1	\$ 6.68
Peripherals	Series Transistors MOSFET Assortment Kit	1	\$ 18.99
Total			\$838.45

Due to most necessary parts already being purchased, our bill of materials is limited to the changes we are making to the previous version. The two most expensive parts we had to buy, the Devantech Motor Kit and the ASUS Portable Monitor had to be re-purchased by our team, as the original parts were damaged during testing. The cost of our purchased parts are shown in Table 43. The total cost of our purchased parts and our inherited parts is \$2683.59. This is under our target budget of \$3000. Also, if this robot was to be re-produced, the cost of the second monitor and drive kit could be subtracted, bringing the total cost down to \$2238.16, which is well under the target budget.

9.4. Final Team Member Contributions

Bryce oversaw all work dealing with the operation of the motors and encoders and the LiDAR sensor, used for object detection. This included the physical components as well as the software design that was associated with these elements. He also worked with Erick and Ben in order to successfully integrate these components with the Hardware Communications and Remote Interface subprojects.

Ben was responsible for all things dealing with the remote user interface and the web server. This was almost entirely a software subsystem, only dealing with hardware when it came to integration with other subsystems, such as the motors or the camera. This included overhauling the previous team's code and redesigning the web server and the different functions on the user interface, in order to incorporate the new and existing features without the use of R.O.S.

Erick handled the use of microcontrollers, which dealt with controlling the stepper motors of the camera mount, the streaming LED, and the battery reader. This also included designing the

camera mount itself, which combined with integration, was how most of the time on this subproject was spent. During integration, he worked with Ben and Bryant to make sure that the communication between the microcontrollers and the NUC was seamless.

Hasib was responsible for the power systems, which includes powering any additions made to the robot as well as designing the home base. Things that were added to the robot that needed to be powered include the buck converter, the USB hub, the stepper motors for the camera mount, and the battery reader. However, the majority of his time was spent working on the home base, which includes designing the electrical components inside of the home base, the housing of the home base, and the leads on the robot which allows it to connect to the home base.

10. Final Thoughts

10.1. Future Recommendations

After completing the second iteration of this project, we believe that there is still room for improvement, should the instructor team and our sponsor, Dr. Cheung, choose to assign this project for a third iteration.

The first recommendation we would make to a future team is to use coulomb counting instead of voltage readings for the battery reader. We decided to use voltage readings, and use a conversion table to find an associated battery percentage to display to the user. We believe that if we had used a coulomb counter, we would have been able to get more precise and consistent readings.

The next recommendation we would make is to 3D print the camera holder on top of the robot, as opposed to using the pieces of wood we used. We believe this would have made a sturdier design that was easier to work with, and that could tolerate heavier usage. The file we used to design what we used is in our github repository if a future team would like to use it. The file is located in JARVIS/Microcontroller_Stuff and is called "Camera Mount.stl"

A third recommendation we would make would be to add some sort of counterweight to the base of the robot. Incorporating this would increase the stability of the robot. One of our Engineering Requirements was that the robot should weigh no more than 50 lbs. We had about 10 lbs to spare, and if we had used this extra weight to make a counterweight for the base of the robot, we believe the robot would have been able to drive at a more steady pace, and mitigate some of the rocking that the robot currently experiences.

Another recommendation we would make to a future team is to add some sort of casing to cover the exposed wires. After adding this as a requirement, we realized that it would be difficult to accomplish, as we were constantly adjusting wires and cable connections all the way up to our presentation time. Although future teams may need to access ports on the NUC and the USB hub, we believe they could incorporate some sort of casing for the wires underneath the robot as well as those on top of the robot dealing with the camera servo.

An additional recommendation we would make is for the basket to be re-attached. This was done hastily by our team, so we believe there is room for improvement for both aesthetics and functionality.

As far as software additions, we believe there is room for improvement in this area as well. One addition that would be cool to see that was mentioned to us by Dr. Cheung would be the incorporation of virtual reality. After some research, we discovered that Zoom's SDK can be used with the Unreal Engine 4 to incorporate virtual reality with Zoom Meetings. We were using a different SDK package than the one that was compatible with Unreal Engine. However, if a team were to start the year with this additional feature in mind, we think they would be able to implement this cool feature. Additionally, the implementation of a queue for users would be a beneficial feature to add. We did not have enough time to implement this feature, but think it would be a good addition in a future iteration.

We also believe some more autonomy could be added in the next version of the robot. We removed the ability to have a predefined area of operation that would allow the user to click a point on a map generated by the LiDAR point cloud and have the robot navigate itself to that point. We think this

would be a cool feature to add. Another autonomous feature that would be helpful is full autonomous navigation with respect to navigating back to the home base. We think that a good feature would be the ability to automatically return to the base, which we did not have time to implement.

10.2. Reflection

In thinking back on the last 12 months and the development of this second iteration of the robot, there are a few things that we wish, as a team, we would have done differently that we believe would have put in in a better position for Senior Design Day at the end of our second semester. The biggest mistake we believe we made as a group was not scrapping the previous software design sooner. We spent the better part of the first semester trying to work with the previous team's code and software setup. Looking back, we should have realized the lack of organization and documentation in the previous team's work and been more decisive at an earlier stage in choosing to move away from R.O.S., which they used to operate their robot.

Another thing we would have changed was limiting our goals based on our experience with web programming and design. At the beginning of our first semester with this project, we had very lofty goals and expectations for the project. At the time, we did not realize how C.S. and web design intensive this project would be. As a group, we were not the most well versed in web programming, and because of this, had to spend a lot of time learning on the fly about how to accomplish the things we wanted to accomplish. Looking back, if we had realized how much we would have to learn about software design before being able to accomplish what we wanted to accomplish, we would have tailored our Marketing and Engineering Requirements around this fact, and likely would have completed a higher percentage of our requirements.

10.3. Conclusion

To finish the project, we are very proud of our final design for the physical structures of the robot and home base, as well as the software work that went into making things functional. There were several obstacles and roadblocks that came with the nature of having an inherited project like poor communication and documentation from the previous team. There were things we wish had been taken care of differently, but overall, we are proud of the way we worked as a team to navigate through these issues and provide our sponsor with a product that is able to be used. Hearing our sponsor say in our Final Presentation that this was the type of robot he imagined when he came up with the project idea was a rewarding feeling. Each team member was able to learn a lot in regards to working with a team, overcoming many challenges, learning new topics and developing our understanding of the things we thought we knew, and completing a major project like this one.

11. References

- [1] U. S. C. Bureau, "The number of people primarily working from home tripled between 2019 and 2021," *Census.gov*, 16-Sep-2022. [Online]. Available: <https://www.census.gov/newsroom/press-releases/2022/people-working-from-home.html#:~:text=SEPT.%2015%2C%202022%20%E2%80%93%20Between,by%20the%20U.S.%20Census%20Bureau>. [Accessed: 02-Mar-2023].
- [2] "Padbot telepresence robot: Best Virtual Remote Presence: Mobile Robots," Padbot Telepresence Robot | Best Virtual Remote Presence | Mobile Robots. [Online]. Available: <https://www.padbot.com/padbot3>. [Accessed: 21-Feb-2023].
- [3] "VGO Robotic Telepresence for Healthcare, education and business," Two-way audio/video mobile communications for applications in enterprise, manufacturing, education and healthcare.[Online]. Available: <https://www.vgocom.com/>. [Accessed: 21-Feb-2023].
- [4] "Telepresence robot market size, share, Opportunities & Forecast," *Verified Market Research*, 05-Jul-2022. [Online]. Available: <https://www.verifiedmarketresearch.com/product/telepresence-robot-market/>. [Accessed: 01-Feb-2023].

- [4] “Office telepresence,” *Double Robotics - Telepresence Robot for the Hybrid Office*. [Online]. Available: <https://www.doublerobotics.com/pricing.html>. [Accessed: 02-Feb-2023].
- [5] V. Ahumada-Newhart and M. Warschauer, “Virtual Inclusion via Telepresence Robots in the Classroom: An Exploratory Case Study,” *The International Journal of Technologies in Learning*, vol. 23, no. 4, pp. 18–18, Jun. 2016.
- [6] M. Bélanger-Barrette, “Which ISO standards are made for Collaborative Robots,” Blog. [Online]. Available: <https://blog.robotiq.com/which-iso-standards-are-made-for-collaborative-robots>. [Accessed: 19-Feb-2023].
- [7] “What is IEEE 802.11? - definition from Techopedia,” Techopedia.com. [Online]. Available: <https://www.techopedia.com/definition/24967/ieee-80211>. [Accessed: 19-Feb-2023].

Appendix A - Analytical Hierarchy Process for Objective Tree Weights

In our Appendix A, we will be showing how our team came up with the weights for each of the leaves under the 4 main categories, as well as the main categories themselves. Shown in Table 41 is our decision matrix for the main categories, followed by the more detailed matrix for each of the general categories showing how each requirement inside that category was weighed. For our main categories, we decided to look at our marketing requirements and see what the best way to group them was. We decided the best way to break up the project was into 4 sections: human interaction, interface/connectivity, safety/privacy, and product. To determine our weights, we used pair-wise comparisons in partnership with the Analytical Hierarchy Process, or AHP. The main deciding factor in rating each cell’s importance was how important it was to our sponsor in our initial meeting. He made it clear that the main improvements should be more autonomy, including object avoidance, as well as navigation to a charging location, so our ratings of importance reflect this. Each matrix shows a cell’s importance compared with another. It can be read as “Row title is __ times more important than column title.” For example, in Table A1, Interface and Connectivity is ½ as important as human interaction.

Table 41 : Decision Matrix for General Categories

Overall Tree	Human Interaction	Interface/Connectivity	Safety/Privacy	Product	Geometric Mean	Weight
Human Interaction	1.00	2.00	0.67	2.00	1.28	0.29
Interface/Connectivity	0.50	1.00	0.33	0.33	0.49	0.11
Safety/Privacy	1.50	3.00	1.00	2.00	1.73	0.38
Product	0.50	3.00	0.50	1.00	0.93	0.22

For the decision matrix shown in Table 2, we found that safety/privacy was the most important part of the robot since this device could both hurt someone physically and invade someone’s privacy by recording them without their permission. We decided that human interaction was the 2nd most important since talking to other people is the main purpose of the robot. After that, we thought product was the most important since we want to create a quality product that is not only functional but also marketable. Finally, we decided that

interface/connectivity was the least important since most of the interface has already been completed by the previous team.

Table 42 : Decision Matrix for Human Interaction

Human Interaction	Two-Way communication	3D-Viewing	Warning Messages	Geometric Mean	Weight
Two-Way communication	1.00	0.50	3.00	1.14	0.32
3D-viewing	2.00	1.00	4.00	2.00	0.56
Warning Messages	0.33	0.25	1.00	0.12	0.12

For the decision matrix shown in Table 3, we decided to make 3D-viewing the most important part since the 3D camera is one of the main things our sponsor told us to add. We decided to make two-way communication the 2nd most important since it's vital the robot's main functionality, but the previous group already has that part mostly working, so we figured that it didn't need much changed. We agreed that the warning messages were the least important by a wide margin since it was not requested by the sponsor and is just a functionality that we will add if we have enough time.

Table 43 : Decision Matrix for Interface/Connectivity

Interface/Connectivity	Physical User Interface	Web Interface	Queue	Geometric Mean	Weight
Physical User Interface	1.00	4.00	2.50	2.15	0.58
Web Interface	0.25	1.00	3.00	0.91	0.27
Queue	0.40	0.33	1.00	0.51	0.15

For the decision matrix shown in Table 4, we decided that the user physical interface is the most important since it is important among interface/connectivity matrix. It is our opinion that physical interface is important because it is important that the telepresence robot is easy to maintain and upkeep. We decided that the second most important is web interface. Friendly and accessible web interface will make the user experience pleasant. Finally, we decided that queue system is the least important system in connectivity matrix because the queue system is built in system that is already available with the zoom, we just need to incorporate with our system.

Table 44 : Decision Matrix for Safety/Privacy

Safety/Privacy	No Safety Risks	Object Avoidance	Stay In Predefined Area	Video Streaming Indicator	Geometric Mean	Weight
No Safety Risks	1.00	1.50	3.00	9.00	2.52	0.44
Object Avoidance	0.67	1.00	3.00	7.00	1.93	0.34

Stay In Predefined Area	0.33	0.33	1.00	8.00	0.97	0.18
Video Streaming Indicator	0.11	0.13	0.13	1.00	0.20	0.04

For the decision matrix shown in Table 5, we found that taking no safety risks was the most important and that we should always consider any potential safety risks first when adding anything new. After that, we added object avoidance because that is one of the high priority functionalities that our sponsor wants us to add. Next is staying in a predefined area, which we believe is a good feature to add, but a relatively low priority one. We found the video streaming indicator to be the least important since that is simple feature which is already functioning.

Table 45 : Decision Matrix for Product









Product	3-Hour Battery Life	Home Charging Station	Competitive Pricing	Physical Appearance	Geometric Mean	Weight
3-Hour Battery Life	1.00	0.67	5.00	3.00	1.78	0.34
Home Charging Station	1.50	1.00	6.00	4.00	2.45	0.47
Competitive Pricing	0.20	0.17	1.00	0.50	0.36	0.07
Physical Appearance	0.33	0.25	2.00	1.00	0.64	0.12









For the decision matrix shown in Table 6, we found that the home charging station was the most important one since it is the key component of one of the most important features being asked of us by our sponsor. We decided that the 3-hour battery life was the next most important since all the components that we are adding to the robot will draw more current from the battery, so we need to make sure the battery life is still above the 3-hour mark. We decided that the pricing and physical appearance were the least important parts since they don't affect the functionalities of the robot, but still add to the overall quality of the product.









Appendix B – Previously Completed Engineering Requirements









To avoid confusion, engineering requirements the previous team had in their report designated as completed have been given their own table and numbering system. These engineering requirements are designated with the prefix P to differentiate them from the engineering requirements not completed in section 3.3 of this report. Although these requirements have been previously satisfied, it is imperative that our additions to the design still maintain the integrity of the original design, so we will run tests to verify that these requirements are still satisfied.





Table 46 : Previously Completed Engineering Requirements

ER #	Engineering Requirement	MR #	Rationale / Justification	Testability / Verification
P-1 	Robot shall be able to capture audio within a 90-degree radius directly in front of it	1	The robot needs to hear what individuals in front of it say to enable remote conversation	Speak to robot from various angles within 90 degrees and verify robot can capture audio from all angles
P-2 	Robot shall be able to capture audio from at least 6 feet away	1	To respect personal space, the robot should be able to hear individuals speaking to it 6 feet away	Speak to robot from various points up to 6 feet away and verify it can capture that sound
P-3 	Robot shall be able to capture audio between 40 and 60dB	1	Robots should hear people speaking at various volume levels	Produce sounds between 40 and 60dB around the robot and verify the robot can detect them
P-4 	Robot shall be able to capture audio at a minimum sampling rate of 8kHz	1	The frequency range of a human voice 80 Hz – 2 kHz and most phone conversations are around 8 kHz	Listen to the captured audio and verify it provides sufficient quality audio for clear conversation
P-5 	Robot shall be able to output audio in at least a 90-degree radius from its center	1	The individuals standing in front of the robot should be able to hear what the remote user is saying to enable conversation	Stand in front of robot at various angles within 90 degrees and verify audio output can be heard
P-6 	Robot shall be able to output audio to at least 6 feet away	1	To respect personal space, the robot should be able to speak to individuals 6 feet away	Stand at various points up to 6 feet and verify audio output can be heard
P-7 	Robot shall be able to output audio between 40 and 60 dB	1	Individuals should be able to hear the robot at various volumes depending on the environment	Measure sound from robot at different volumes and verify it can output between 40 and 60dB
P-8 	Robot shall be able to output audio at a	1	8 kHz sampling rate gives a maximum frequency	Listen to a remote user speaking through the robot and verify it

	minimum sampling rate of 8 kHz		response of 4 kHz, which is more than enough to output	provides sufficient quality
P-9 	Communication devices shall be at a height between 5 and 5.5 feet	1	To allow for better communication, devices should be located near the head height of the average person	Measure the various communication devices and verify each is between 5 and 5.5 feet off the ground
P-10 	Robot shall have a physical method to decrease and increase volume	1	There needs to be a way for the volume to be changed either for initial setup at an event or dynamically during operation	Measure the dB of robot at various volumes to verify it is being changed correctly
P-11 	Camera viewing angle shall be at least 180 degrees horizontally	1, 2	The remote user needs to see what is around the robot to navigate effectively	Connect as a remote user and verify objects within the entire 180 degree viewing angle from left to right are visible
P-12 	The vertical camera view angle should be at least 90 degrees.	1, 2	Remote users must have the ability to look at the participant at eye level to give the sense of one-to-one conversation.	Connect as a remote user to see if we can identify objects vertically from top to bottom.
P-13 	Real-time video streaming to user shall have a maximum latency of 150ms	1	Low latency allows the remote user to respond to changes in the environment quickly	Use secondary software to determine average latency between multiple connected users
P-14 	Robot shall stop at least 1.5 feet from detected objects in front of it	3, 5	Stopping at least 1.5 feet away from people allows them adequate personal space and room to move	Navigate robot towards objects and verify it stops at least 1.5 away from them
P-15 	Robot shall override user control upon object detection before robot can travel 1 foot	3, 5	In the worst case, the robot should be able to stop within 1 foot of a detected object to prevent collisions	Manually trigger an object detection event through software and measure how far the robot travels
P-16 	Robot shall disable forward navigation upon object detection and only allow remote user ability to turn until	4	To prevent long idle times, the robot should allow the remote user to turn away from a detected object and continue navigation	Place an object in front of robot and verify a remote user can turn away from the object and continue navigation

	the object is out of its path			
P-17 	Robot shall be controllable through web interface using a keyboard and mouse	4	Keyboard and mouse are widely accessible, and any user should be comfortable using them	Use the interface and verify keyboard and mouse can be used to fully control the robot
P-18 	Web interface shall provide robot's battery level to users	4	Users should know how much battery a robot has left so they can decide on whether to use it	Access the web interface and verify robot's battery level is displayed and is accurate
P-19 	Feedback shall be provided to the user within 5 seconds if object detected	4	Quick feedback to user about loss of control keeps them informed and less frustrated	Place object in front of robot and verify it sends feedback to user within 5 seconds
P-20 	Robot shall have an emergency shut off that stops movement before it can travel 1 foot	5	In case of emergency, the robot should be able to be stopped by a physical shut off within a very short distance of travel	Activate the shut off in various room conditions and measure distance traveled before complete stop
P-21 	Robot shall move no faster than 1 mile per hour or about 45 centimeters per second	5	Based on an average human walking speed of 3 miles per hour, the robot should move slower to prevent damage in a collision	Move the robot through the online interface as fast as it will go and verify it does not travel faster than 2 miles per hour
P-22 	Robot shall have a battery that can last 3 hours on a single charge	6	The robot needs to be able to last for a typical meeting or event	Record how long it takes the battery to go from 100% to 10% charge while running continuously, then verify it is at least 3 hours
P-23 	Robot's battery shall be rechargeable using a standard 120V outlet	6	A rechargeable battery allows the robot to be reused many times using standard outlets found in any room	Plug robot into a standard outlet and ensure that it can charge to 100%
P-24 	Robot's battery shall be able to be charged from 0% to 100% within an 8-hour period	6	Robot will likely be left to charge overnight during the typical non-work hours of 12 to 8am	Measure time it takes for battery to go from 0% to 100% charge and verify it is within 8 hours

P-25 	Robot shall keep track of battery level within 1% accuracy	6	Accurate battery information allows users to know how much time they have left or whether to connect in the first place	While robot is being used, take a battery level measurement every 10 minutes and compare to the displayed value in the interface
P-26 	Robot shall have a visual indicator when streaming that can be seen from at least 6 feet away	7	For privacy purposes, people around the robot need to be informed that it is streaming video, so they have a chance to leave or watch what they are saying	Have a user connect to the robot and verify visual indicator turns on and is visible from at least 6 feet away
P-27 	Robot's home location should be able to be changed or edited	8	In case the room layout changes, or another location is better suited, the home location will need to change	Set various home locations on the robot and verify for each that the robot can navigate home successfully
P-28 	Robot should be able to autonomously navigate to home location within a 1-foot-radius of the set home location	8	Since home location will be against a wall near an outlet, the robot will need to return near enough to the location to be out of the way and reach the outlet for charging	After placing the robot in various positions within a room, verify it can successfully navigate home from each
P-29 	Robot's boundary of operation should be able to be changed or edited	9	If the robot gets moved to another room or the layout is changed, the boundary will need to be changed	Test robot within multiple rooms and boundaries and verify the robot adheres to them
P-30 	Robot should not travel more than 1 foot out of the set operational boundary	9	To protect others and the robot, it should not be allowed to navigate beyond the boundary where stairs, bathrooms, or other private areas might be	Navigate the robot across the boundary at various points and verify it will not allow any part of itself to travel over 1 foot from the line
P-31 	Robot should keep track of current location within operational boundary accurate within 1 foot	9	To return home safely and accurately, the robot needs to know where it currently is	Place the robot in various locations and verify it can travel home
P-32 	Robot should have a physical interface for connecting to wireless networks and displaying	10	For initial setup and troubleshooting, the robot will need a way of connecting to different wireless networks and displaying basic info	Connect to a new wireless network using the interface and verify the robot can be connected to

	relevant information			
P-33 	Web-based interface should provide a queue for up to 10 users to wait for access	11	Multiple people might want access to the robot and need some way of waiting in a line for access	Have more than 10 users attempt to connect to the queue and verify 10 of them can successfully take control of robot when it is their turn
P-34 	Robot should not have any sharp edges or loose wiring	12	Sharp edges or loose wiring causes safety hazards to people and objects around the robot	Check all wiring and exposed parts of robot and verify nothing is sharp or loose
P-35 	Robot base should be no larger than 2 square feet	12	A small base size allows the robot to fit through standard doorways which span at least 3 feet	Measure the base of the robot and verify it is not larger than 2 feet and small enough to fit through standard doors
P-36 	Robot should cost no more than \$3000	13	Costs of similar robots range between \$2500 and \$5000 and our total budget is \$3000	All associated manufacturing cost of robot should remain under \$3000

Appendix C – Analytical Hierarchy Process for Subsystem Design

Power Systems AHPs

We used the AHP tool to decide which one will best serve our objective. Table 52 shows our decision matrix for the major categories. through matrix for each of the basic categories that shows how each need within the category was weighted. We have decided that the best way to break up the sub project power delivery system into four sections: compatibility, safety, reliability, and affordability. We employed pairwise comparison coupled with the AHP to obtain our weights. The numbers in table 52 demonstrate a pairwise comparison of each criterion where 1 indicates equal relevance and the other numbers indicate how many times more important one criterion is than the other. For example, in Table 52 the first-row compatibility is rated three times higher than safety and four times higher than safety. For the decision matrix shown in Table 52 we found that compatibility is the most important part of the power system because the power that will be delivered to the robot must be usable by all the subsystems of the robot. For this reason, selecting the power supply we had to make sure the power supply has 48V DC output because the lithium-ion battery can only be charged by 48V.

Table 47 : Decision Matrix for Power Supply

Criteria	Compatibility	Safety	Reliability	Affordability	Geometric Mean	Weight
Compatibility	1	3	4	2	2.213	0.284

Criteria	Compatibility	Safety	Reliability	Affordability	Geometric Mean	Weight
Safety	0.33	1	2	3	1.394	0.179
Reliability	0.25	0.50	1	2	1.000	0.128
Affordability	0.50	0.33	0.50	1	1.058	0.136

Table 48 : Safety Matrix for Power Supply

Criteria	Electrical Safety	Fire Safety	Overload Protection	Geometric Mean	Weight
Electrical Safety	1	5	3	2.466	0.529
Fire Safety	0.2	1	0.22	0.564	0.121
Overload Protection	0.33	3	1	1.817	0.390

Table 49 : Reliability Matrix for Power Supply

Criteria	Battery Life	Charging Time	Efficiency	Geometric Mean	Weight
Battery Life	1	3	2	1.817	0.408
Charging Time	0.33	1	0.33	0.693	0.155
Efficiency	0.5	3	1	1.225	0.437

Table 50 : Affordability for Power Supply

Criteria	Initial Cost	Maintenance Cost	Operating Cost	Geometric Mean	Weight
Initial Cost	1	3	3	2.080	0.568
Maintenance Cost	0.33	1	1	1.000	0.273
Operating Cost	0.33	1	1	1.000	0.159

Hardware Communications AHPs

When choosing the 360-degree camera, there are several factors which must be considered. One is cost. We have a budget of \$2000 and this will likely be the single most expensive thing that we buy, so it is important that we do not waste a large portion of our budget on one camera. Another factor to consider is size. We want to make sure that the camera is not too heavy or large, as it needs to sit on the top of the robot. A third factor to consider is overall quality. We want the robot to be a high-quality product, so the components that make up the robot should be as well. The quality includes camera quality, build quality, and design. We also need to consider any additional features that the camera may have. If the camera has any

additional features that we believe could improve the overall user experience, that will only serve to improve the camera in our eyes.

Table 51 : AHP for 360-Degree Camera

360-Degree Camera	Cost	Size	Quality	Additional Features	Geometric Mean	Weight
Cost	1.00	4.00	1.25	2.00	1.78	0.39
Size	0.25	1.00	0.33	0.40	0.43	0.09
Quality	0.80	3.00	1.00	2.00	1.48	0.32
Additional Features	0.50	2.50	0.50	1.00	0.89	0.20

Table 52 : AHP for Camera Cost

Cost	Insta360	KanDao	J5create	Geometric Mean	Weight
Insta360	1.00	1.33	0.20	0.64	0.15
KanDao	0.75	1.00	0.15	0.48	0.11
J5create	5.00	6.67	1.00	3.22	0.74

Table 53 : AHP for Camera Quality

Quality	Insta360	KanDao	J5create	Geometric Mean	Weight
Insta360	1.00	2.00	3.00	1.82	0.54
KanDao	0.50	1.00	2.00	1.00	0.30
J5create	0.33	0.50	1.00	0.55	0.16

Table 54 : AHP for Additional Camera Features

Additional Features	Insta360	KanDao	J5create	Geometric Mean	Weight
Insta360	1.00	0.67	0.50	0.69	0.22
KanDao	1.50	1.00	0.67	1.00	0.32
J5create	2.00	1.50	1.00	1.44	0.46

Table 55 : AHP for Camera Size

Size	Insta360	KanDao	J5create	Geometric Mean	Weight
------	----------	--------	----------	----------------	--------

Insta360	1.00	2.00	0.40	0.93	0.25
KanDao	0.50	1.00	0.20	0.46	0.13
J5create	2.50	5.00	1.00	2.32	0.63

Table 56 : Final Decision Matrix for Camera

Camera	Final Score
Insta360	0.30
KanDao	0.21
J5create	0.49

Navigation Systems AHPs

To proceed with designing the navigation systems, it was imperative to decide upon the components which would be used for the subsystem. Lidar sensors were compared against other proximity sensors. However, for our needs, we were able to find a cheap lidar sensor that provided more adaptability in dynamic environments while optimizing precision and accuracy of collected data to drive autonomous movement and object avoidance.

Table 57: Object Detection Matrix

Sensors	Cost	Accuracy	Range	Response Time	Final Priorities
Cost	1.00	1.50	3.00	2.00	0.39
Accuracy	0.67	1.00	3.00	2.00	0.32
Range	0.33	0.33	1.00	0.67	0.12
Response Time	0.50	0.50	1.50	1.00	0.18
Sum	2.50	3.33	8.50	5.67	1.00

Table 58: Sensor Type Final Priority Matrix

Proximity Sensors	Final Priority
Lidar	0.527870474
Ultrasonic	0.237180936
Inductive	0.23494859

Table 59: Lidar Sensor Final Priority

Lidar Sensors	Final Priority
Velodyne	0.32286993
YDLidar X4	0.353903251
Benewake	0.323226819

Table 60: IMU Final Priority Matrix

Inertial Measurement Units	Final Priority
AdaFruit	0.341180387
Pololu	0.330619146
Razor	0.328200466

Table 61: BLE Beacon Parameter Final Priority

BLE	Cost	Battery	Range	Final Priorities
Cost	1.00	2.00	4.00	0.57
Battery	0.50	1.00	2.00	0.29
Range	0.25	0.50	1.00	0.14
Sum	1.75	3.50	7.00	1.00

Table 62: BLE Beacon Final Priority

Inertial Measurement Units	Final Priority
Estimote	0.400928841
Kontakt	0.319645045
Gimbal	0.279426115

Online User Interface AHPs

For the Software AHP, there was not many decisions to be made. Due to the project being inherited, and the team's decision to continue with the use of Zoom's SDK package, most of the software decisions had already been made for us. We still considered other options, most importantly, if we were going to continue using ROS. We ended up deciding to stray away from ROS in favor of other options, which are shown in Tables 65-67. Based on the criteria and weights set in Tables 65 and 66, we ended up deciding to move away from ROS in favor of writing a regular Python program.

Table 63 : Software AHP Matrix (Un-Normalized)

Software	Ease of Use	Library Size	Software Integration	Additional Resources
Ease of Use	1.00	2.00	1.25	3.00
Library Size	0.50	1.00	0.33	2.00
Software Integration	0.40	3.00	1.00	2.00
Additional Resources	0.33	0.50	0.50	1.00
Sum	2.63	6.50	3.08	8.00

Table 64 : Software AHP (Normalized)

	Ease of Use	Library Size	Software Integration	Additional Resources	Final Priorities
Ease of Use	0.38	0.31	0.41	0.38	0.37
Library Size	0.19	0.15	0.11	0.25	0.18
Software Integration	0.30	0.46	0.32	0.25	0.33
Additional Resources	0.13	0.08	0.16	0.13	0.12
Normalized Sum	1.00	1.00	1.00	1.00	1.00

Table 65 : Final Software AHP Matrix

Software Type	Final Priority
ROS	0.47
Python Program	0.53