Free Flowing Furniture

Final Design Specification

Ledingham Inc. Design Group - Team 20
Kevin Chow, Tejus Gangadharaiah, Michael Gelfand, Zachary Lau, Bobby Zhou

Instructor: Denis Onen
Teaching Assistants: Abbas Mahbod

Acknowledgements

Ledingham Inc. Design Group would like to acknowledge the assistance of all individuals and groups who supported the project. The team would like to express their gratitude towards Dr. Denis Onen, as well as the ENEL 400 lab technicians and teaching assistance, who's contributions all helped bring the team to where it is today. Special thanks is also extended to the University of Calgary, Makerspace, and the Schulich UAV club, for providing access to necessary facilities and materials. Special thanks to Tyler Sawatzky for ordering the PCBs used in the final design.

Table of Contents

1. Introduction

The following report provides a detailed overview of Ledingham Inc. Design Group's Free Flowing Furniture product. This report includes some of the predecessor works that served as an inspiration for the final product. Also included are system block diagrams to illustrate the hardware connections and the software design, as well as detailed descriptions for both. Following this is an outline of the engineering analysis that the team used in the development of the project. There is also a detailed description of the product's enclosure and PCB. The report contains a list of hardware and software design choices, possible future iterations that could be made to the design, and a list of regulatory codes that the product must meet.

1.1 Acronyms and Abbreviations

CAD - Computer Aided Design
COTS - Commercial Off-the-Shelf
GUI - Graphical User Interface
IMU - Inertial Measurement Unit
IP - Internet Protocol
MISO - Master In Slave Out
MVC - Model View Controller
PCB - Printed Circuit Board
SCK - Serial Clock
SPI - Serial Peripheral Interface
TCP - Transmission Control Protocol
UML - Unified Modelling Language

2. Predecessor Works

Inspiration for this project came from multiple sources. The Nissan parking chairs is the product that has the strongest correlation to our project. These chairs would use motion capture cameras to identify the locations of the chairs and then send the information to a computer to determine the optimal route to their destination location. Mechanically they were controlled by 4 omni-wheels. This project has taken some of the same concepts such as roof-mounted cameras for identification, along with a single computer to communicate to each robot.

However, the use of motion capture cameras and the use of a large computer would prove to be too expensive for this project. This is where the work of Amazon’s autonomous robots would come into play. These robots would deliver packages around warehouses by the use of the identification makers on the floor to localize each robot. The idea of using a marker for each robot has become an important asset and component for this project. It has allowed us to easily send commands to different robots and plan separate routes without the hassle of trying to determine the location and orientation of a robot with more complex methods.

The most notable predecessor work that has been done in the field of autonomous navigation was from the 2005 DARPA Grand Challenge. The winner of the challenge was a car built by the Stanford Racing Team led by Professor by the name of Sebastian Thrun. Stanley was fed numerous data and was able to determine the best route to travel to its destination locations. Two of the solutions for autonomous driving were PID control and A* path finding. Both of these concepts would be used in the project.

3. System Block Diagram

3.1 Hardware

Fig. 1 below shows a high level hardware diagram for the FreeFlowing Furniture system. This design is further divided up into multiple modules, specifically a motor module, communication module, motor controller module and user input module.The motor module receives power and signal to transfer to motor output moving the furniture. The communication module consists of a network of radio modules to send commands between the Raspberry Pi and each piece of furniture. The motor controller module consists of a battery pack and a 5V regulator supplying power to an Arduino Nano. The Arduino Nano handles all the information received by the communication module to process into motor commands. Finally the image processing block consists of a Lifecam hd-3000 camera, Raspberry Pi and computer. The computer handles all of the user’s requests from the GUI and sends them to the Raspberry Pi. The Raspberry Pi prompts the camera for images and uses these images to develop new motor commands to send to each of the furnitures.

One important aspect of this project is the fact that it’s a proof of concept for a solution which takes up whole rooms. Therefore we designed this solution to be scalable and modular enough to succeed in a real life environment. This is easily shown as each module can be replaced with larger components as the size increases. For example the motor module can be replaced with large 24V motors to provide enough power for an actual chair, or the battery pack can be replaced with rechargeable lithium ion batteries.

The modularity of the design also lends itself well to adding additional modules for future improvements in design. Setting up communication over Wi-Fi and using a microcontroller provides a large help for this. In this current design the microcontroller is simply converting the received commands into motor commands leaving a lot of space for components such as IMU sensors for more accurate feedback. The Wi-Fi communication is especially helpful as most devices and systems have Wi-Fi capabilities allowing for large improvements, especially on the user interface side.

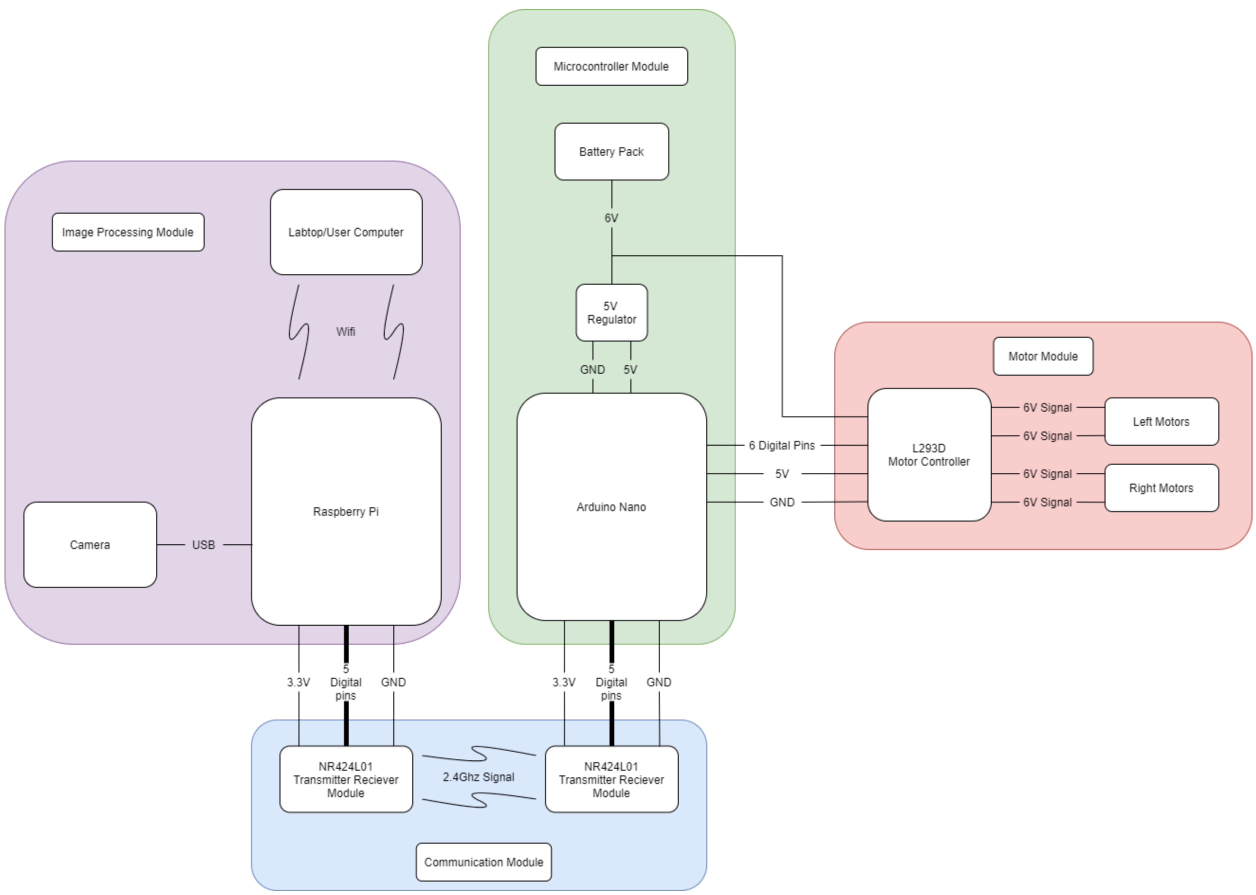


Figure 1: Overall Hardware System Block Diagram

3.2 Software

The software for Free Flowing Furniture is split into three parts:

1. A client application to be run on a desktop or laptop
2. A server-side application to be run on a Raspberry Pi
3. Firmware to be run on each robotic table

When designing the software, our first priority was extensibility; because this project serves as a prototype for larger projects it is important that other users can easily add their own functionality. Thus a modified Model-View-Controller (MVC) design pattern was chosen. By separating business logic from GUI logic the project becomes easier to port and to build on.

We also chose to make the firmware on the bots extremely light. Three factors motivated this decision. First, the processing power of each individual robot is limited by the cost of the processor that it can carry, so it makes more sense to do most of the processing on a more powerful central processor. Second, central processing would make it easier for developers trying to automate certain actions to coordinate movements amongst several robots. Lastly, more advanced sensors systems--for example to accomplish inertial navigation--would add increased cost to each robot that would exceed our budget.

The package diagram in Fig 2. gives a general overview of the design. The client and server each contain a controller package that handles communication between different modules. In addition they each store their own version of the data model with updates communicated between the client and server through TCP. The client contains a view package to interact with the user.

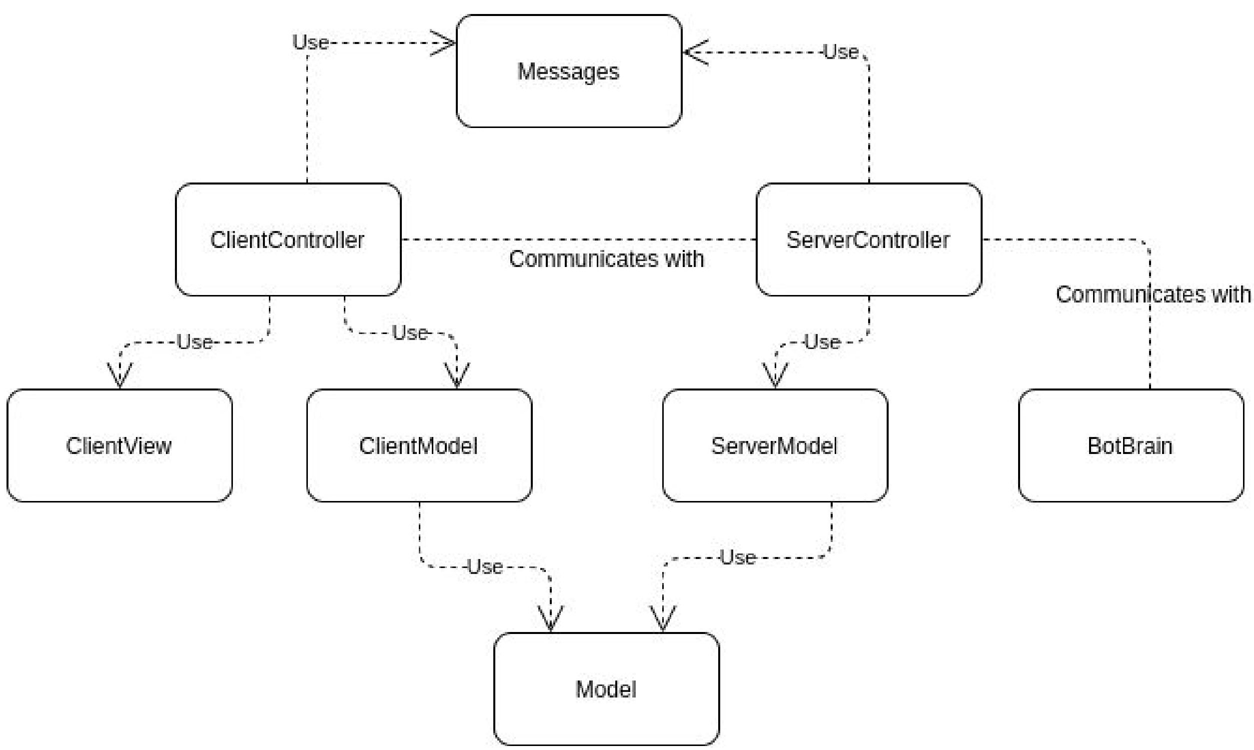


Figure 2: Package diagram of all software required for a remote robotic control system

Partial progress has been made in implementing this design and this work is available at https://github.com/zach-lau/FreeFlowingFurniture. Unfortunately work was suspended prior to completion due to COVID-19.

4. Block Descriptions

4.1 Hardware Blocks

4.1.1 Image Processing Module

The Image Processing module begins with the GUI on a computer handling all of the user input. The user's choices are communicated over wifi to a Raspberry Pi on the same local area network. The Raspberry Pi will then prompt the Lifecam hd-3000 camera for an image whenever necessary and use this new image to determine the current location of all the furniture. While the furniture is moving the Raspberry Pi can prompt the camera for new images to continually gain its location to determine the new motor commands for each robot.

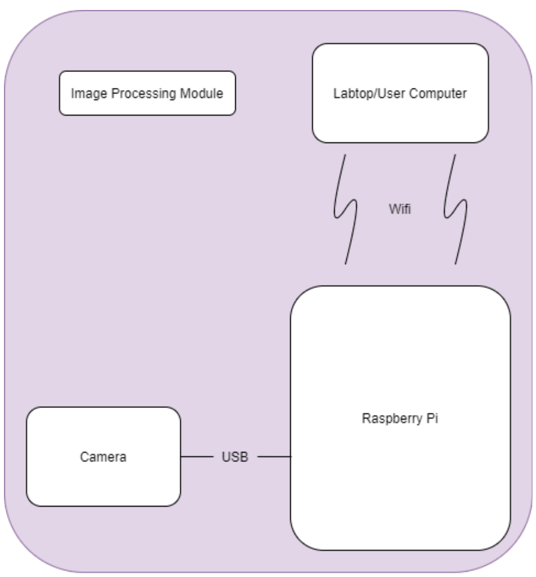


Figure 3: Image Processing Module Diagram

4.1.2 Communication Module

The NRF24L01 receiver and transmitter module is the core component used for communication to each furniture. The NRF24L01 modules communicated over a 2.4Ghz radio signal providing enough range for an entire room, while still having a low current draw making it ideal for this small scale design. The NRF24L01 modules communicate with microcontrollers through SPI lines asking as the slave allowing for easy implementation for cross platform communication, such as between a Raspberry Pi and Arduino Nano. Fig. 5 shows a more detailed circuit of the NRF24L01 module specifically showing the SCK clock input for SPI connection alongside the 16MHz quartz clock ensuring each chip transmits and receives at the same clock speed. The NRF24L01 also comes equipped with an interrupt pin allowing for instant recognition of change in software.

Each module can open and close channels at varying frequencies around 2.4Ghz allowing a single chip to send signals to multiple distinct chips without interference as required for the network of furniture. The NRF24L01 chips also send feedback each time a message is sent letting the transmitter know if the message was received properly. Therefore if the message fails to send the Raspberry Pi can continue to send messages as well as know the furniture may be unresponsive. The NRF24L01 however requires an extremely reliable 3.3V power source or else it is prone to inconsistent behaviour, even with power coming from the arduinos regulated source. Therefore when placed on the PCB a capacitor must be added between the 3.3V and GND lines ensuring stable power.

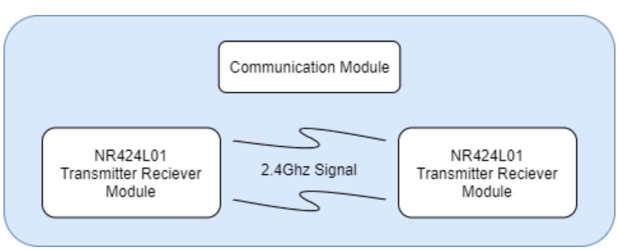


Figure 4: Communication Module Diagram

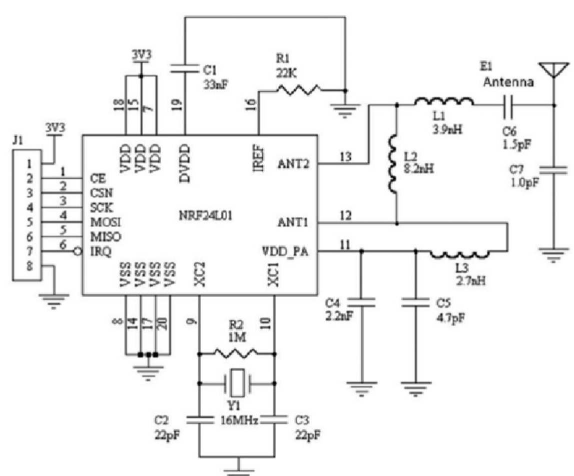


Figure 5: NRF24L01 Circuit Diagram
Source: [1]

4.1.3 Microcontroller Module

The microcontroller module consists of a power source, voltage regulator and an Arduino Nano microcontroller all combined on the PCB to handle all the processing requirements for each furniture. The arduino also powers 2 LEDs to demonstrate the current status of the furniture, namely communication and motor status. This only requires 8 digital pins and 5 analog pins leaving a majority of the arduino’s pins open for future improvements.

Each piece of furniture contains a battery pack consisting of four AA batteries providing 5-6 V DC at approximately 2A with a 8000 mAh capacity. Using AA batteries allows for an easily replaceable system once batteries become depleted or need to be recharged since all the connections are soldered together. The arduino is powered through it’s V_{in} pin which requires a regulated 5V signal supplied through the MP1584EN DC Buck converter. The arduino then provides a 3.3V output to power the NRF24L01 transmitter receiver module.

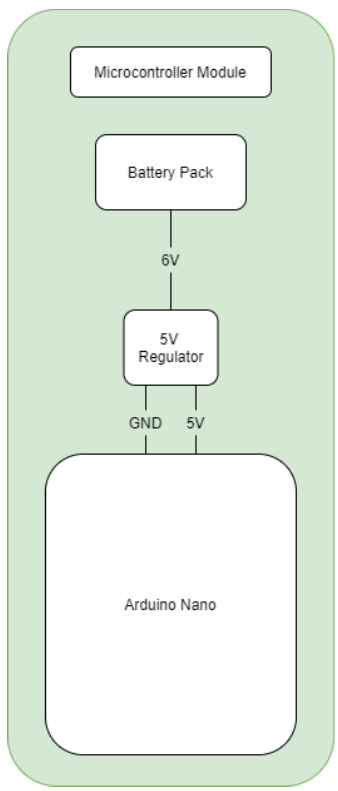


Figure 6: Microcontroller Module Diagram

4.1.4 Motor Module

The motor module uses the L293D motor controller which acts as four half h-bridges to combine the signal from the arduino with the battery power. Each piece of furniture moves using tank drive meaning each wheel on either side of the robot moves synchronously like a tank. This provides more precise on the spot turning. As shown in Fig. 8 the L293D circuit simply takes in an analog value from 0 to V_{ss} and maps it from 0 to V_s. For example an input of $0.35V_{ss}$ leads to an output of $0.25V_s$, therefore the L293D can be used as 2 full h-bridges by pairing the outputs together sending each pair to a single motor. Each channel is also supplied with an enable input which determines if the amplifier for that channel is active, allowing for more control on the microcontrollers side.

The L293D has a wide supply voltage of 4.5V to 36V lending itself well to increasing the battery voltage. Each channel allows for a peak current supply of 1.2A providing more than enough current to power the four 6V Adafruit Industries 711 DC motors which functions in a range from 4.5V to 9V. These motors however provide very little torque, which is why they are fitted with a 1:48 reduction gear ratio.

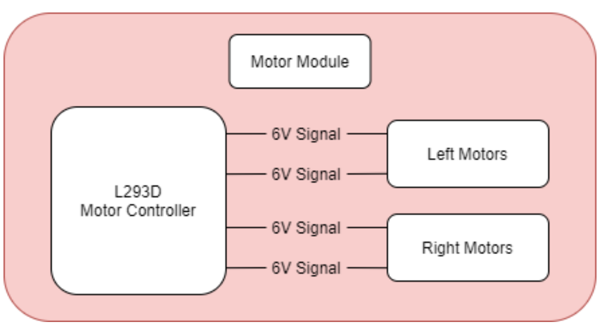


Figure 7: Motor Module Diagram

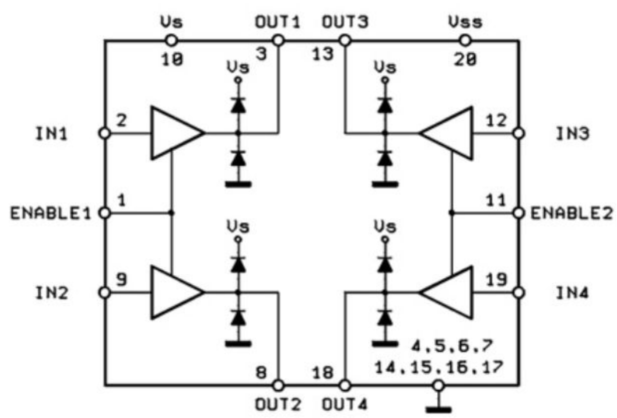


Figure 8: L293D Motor Controller
Source: Adapted from [2]

4.2 Software Blocks

4.2.1 GUI

The client GUI provides the user the ability to interact with the system through one of two main methods.

1. Through manually driving a robot using the keyboard
2. Instructing one or more robots to follow a pre-set path.

Prior to suspending the project due to COVID-19, we implemented a simple command line interface that would allow a user on their laptop to wirelessly select and control one of up to 255 robots specified by a unique robot ID. A finished GUI would look like Fig. 9, and allow the user to create new paths, remove paths and view information about each of the individual robots. Fig. 10 shows a UML diagram of how this would be implemented in TkInter, a python library for event-based GUI's.

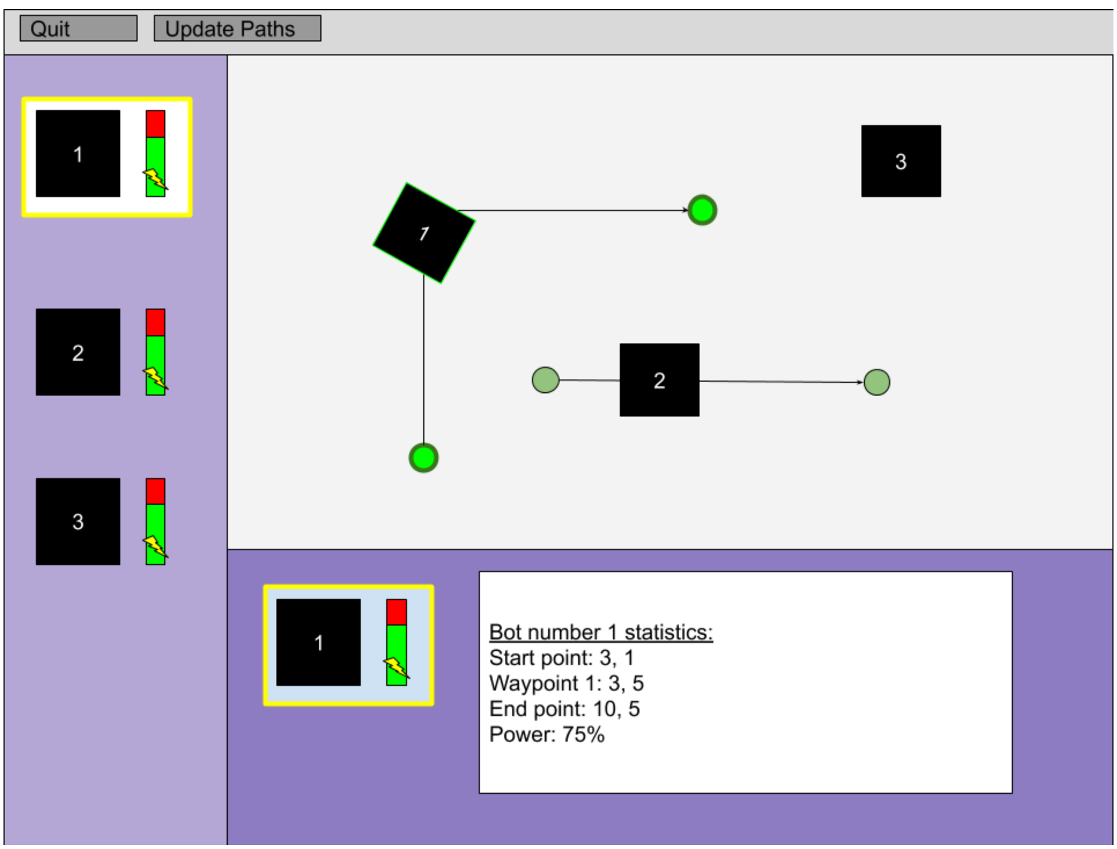


Figure 9: Example GUI for a remote robotic furniture controller. Each dark square represents a robot that the user can control. Green circles represent way points and green lines represent the paths the user has instructed the robots to follow.

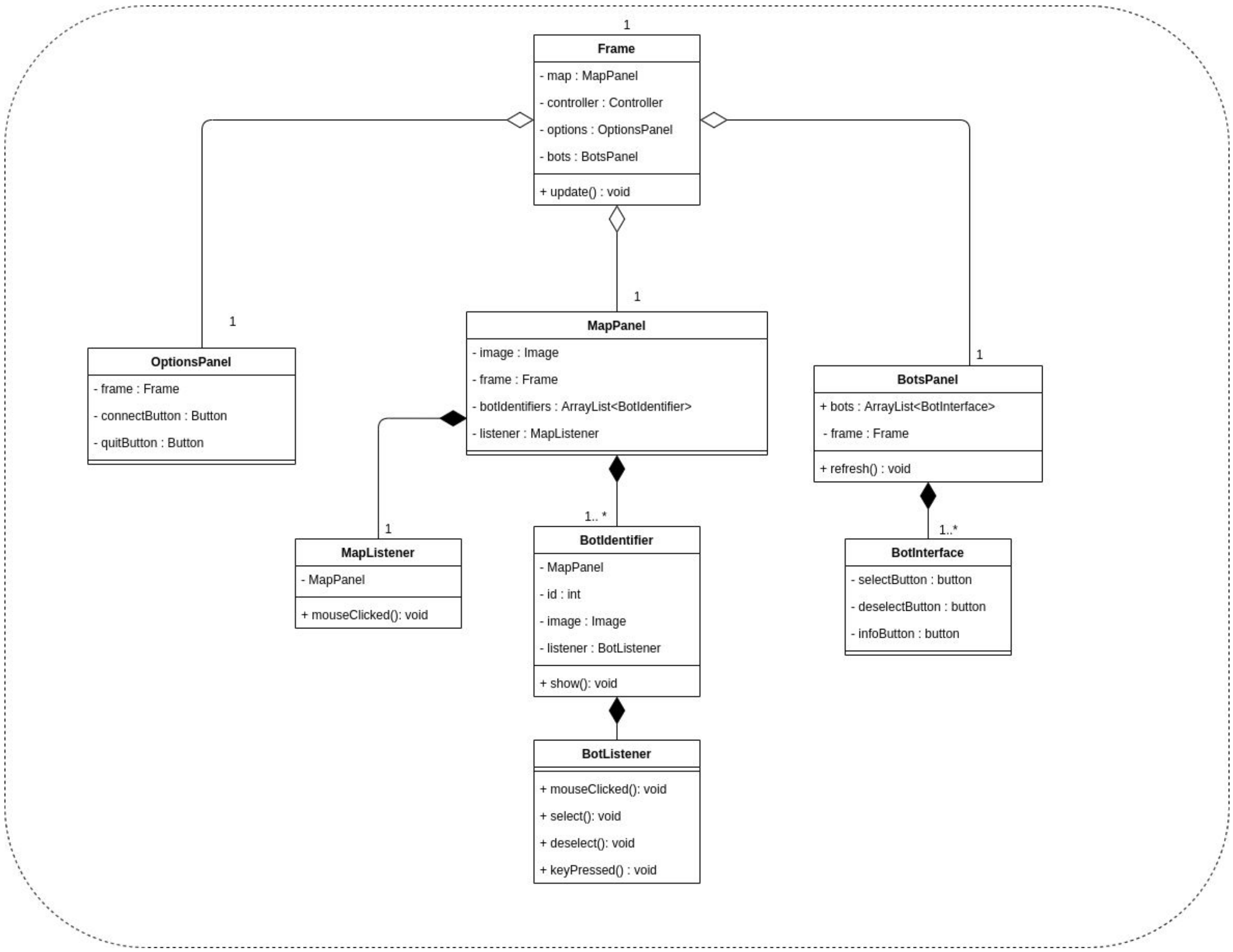


Figure 10: UML Diagram of the GUI for a remote robotic control system

4.2.2 Data Model

The model contains recorded information about the bots, their paths and any obstacles that the system might detect. Both the client and the server have different versions of the model that build on top of the general model framework shown in Fig. 11.

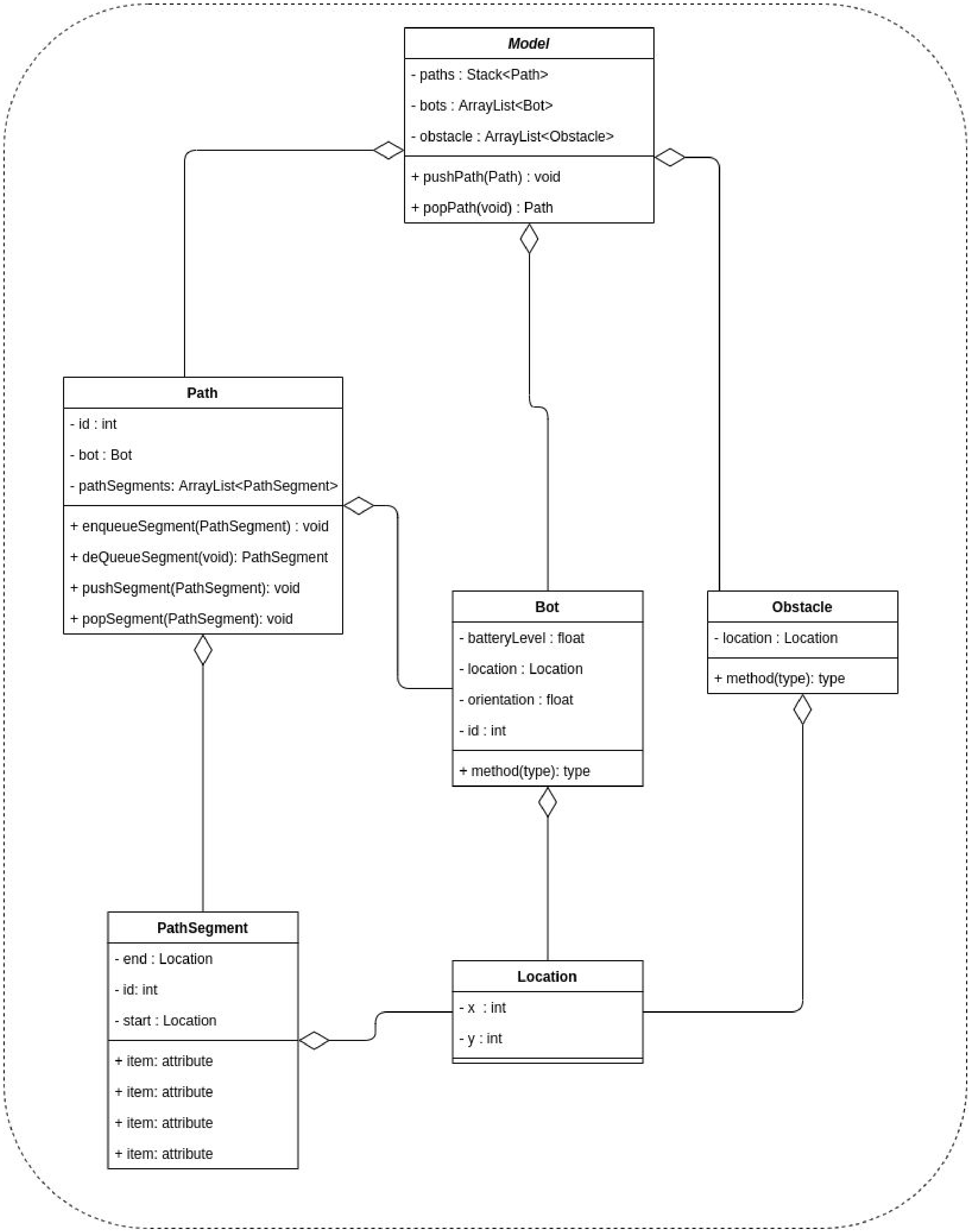


Figure 11: Software model of remote robot environment

In addition to containing paths, robots and obstacles, the client model will also keep track of useful user input such as which bot the user has currently selected or what the most recently added path was. These help to render the GUI. Fig. 12 expands on this functionality.

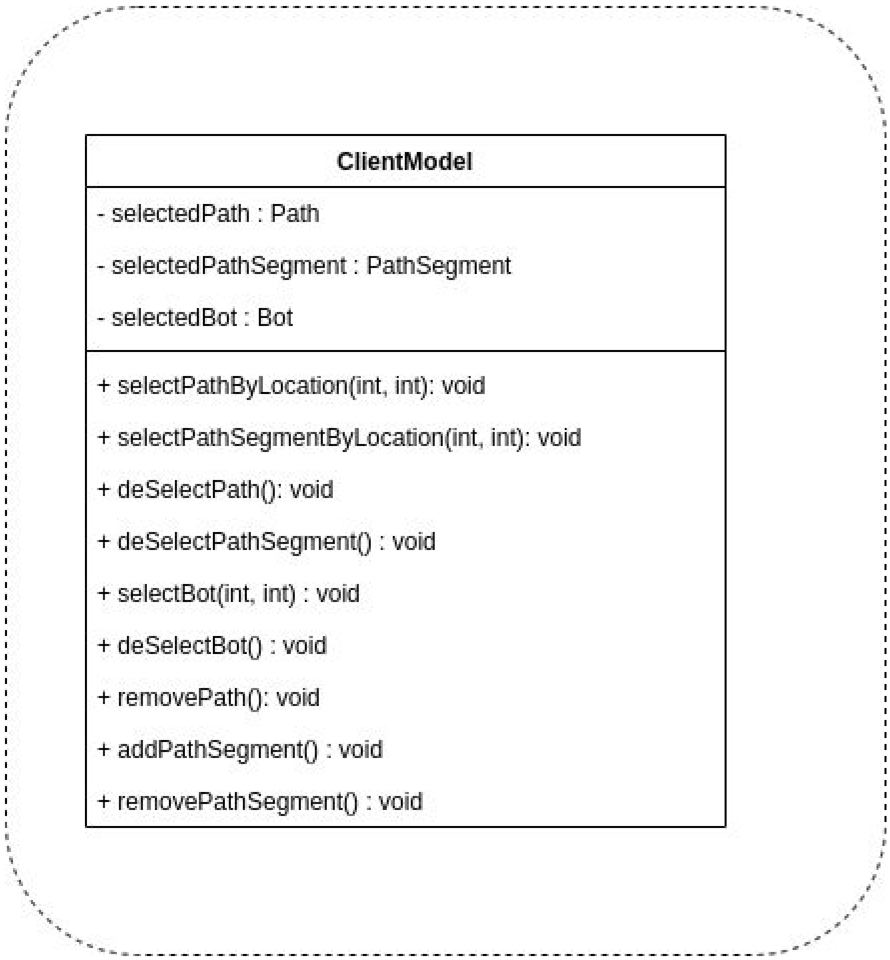


Figure 12: Client layer on top of model for remote robot control system

The server model is more complex because it handles calculations to keep each of the robots on track as well as collecting and processing image data from the camera. This is done in a pipelined process described below.

1. Get the image. The server will ask the camera for a raw image showing the room containing the robots.
2. Extract locations. Each robot is tagged with an aruco marker which is associated with an id. Using OpenCV the server can identify the location and orientation of each bot.

3. Update bot position. Based on the locations extracted from the image in question, the server updates the location of the bots in its model. When the client requests these positions they will be provided to it.
4. Update paths. This is an optional step and something to be built upon after the main design has been built and tested. Using a multi-agent pathfinding algorithm such as a modified A* algorithm, waypoints for each robot will be added, removed or adjusted to ensure that robots all reach their final destination in the fastest time possible without running into any obstacles. This is quite difficult to implement and thus a “horizon goal” for the project.
5. Calculate commands. The server compares the position of each bot to it’s intended trajectory. If it is deviating left or right from the line that is expected to travel, this will be corrected by a PID control loop. If the bot has reached its end point it will be told to stop. If the bot has completely left the path it will be directed to the nearest point on the intended path.

The data model does not, however, handle sending data to the bots. This is the job of the communication module described later. For a graphical overview of the server specific software blocks see Fig. 13.

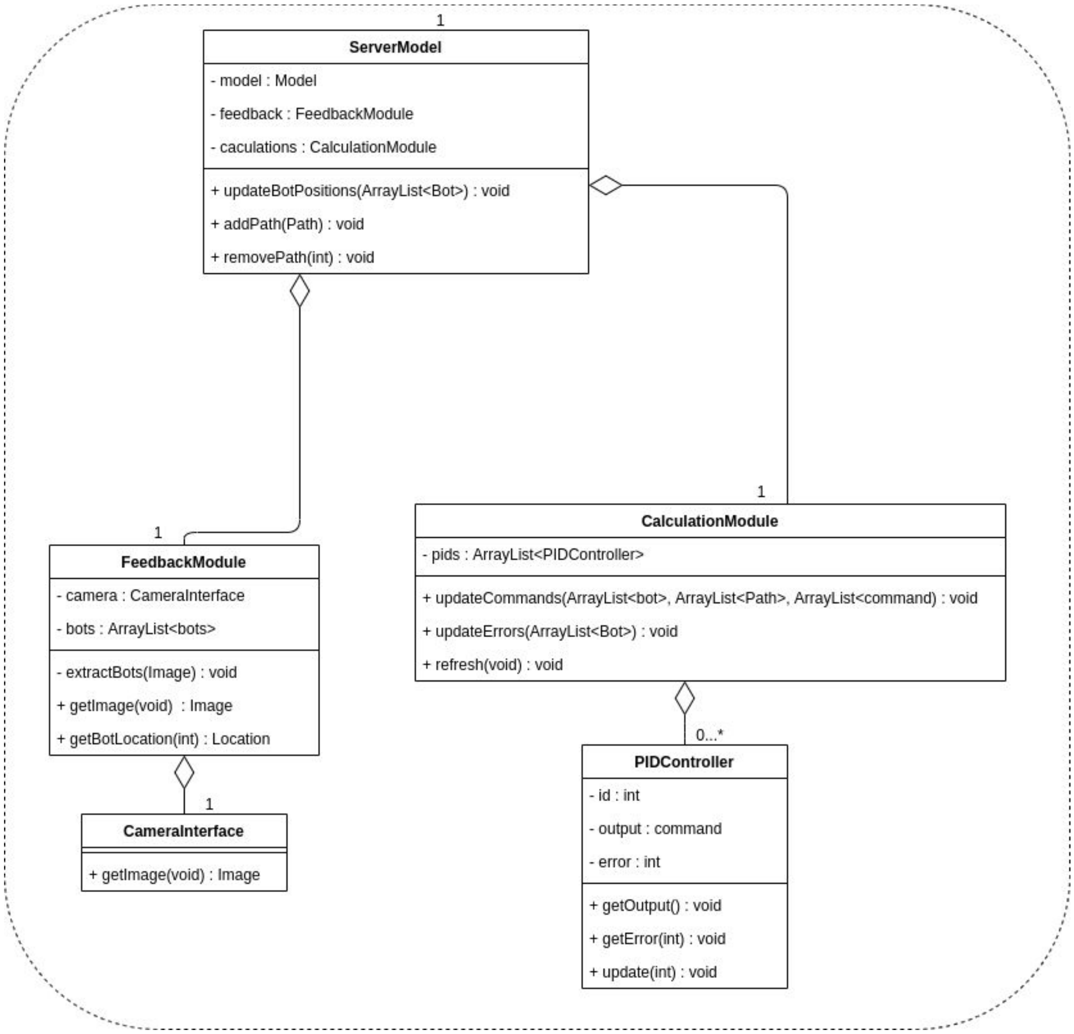


Figure 13: Server-specific modules in a client-server software system designed to remotely control furniture

4.2.3 Robot Firmware

As mentioned earlier the furniture firmware was kept purposefully simple as to use the small and inexpensive Arduino Nano microcontroller. Each Furniture contains its own ID to determine which instructions are meant solely for that robot. Each message contains motor commands for each motor, once received the robot simply sends these same commands into its digital output pins to move the robot.

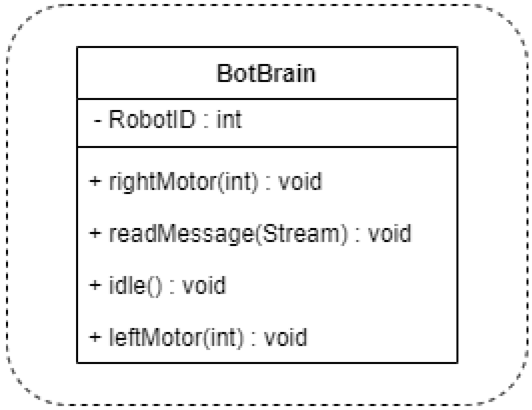


Figure 14: Furniture firmware module to receive commands and move the furniture appropriately

4.2.4 Client-Server Communication

Client-Server communication is done through TCP/IP and this connection is handled at the socket level. The Raspberry Pi should first be assigned a static IP address, and then the client can connect if it is on the same local area network. Connecting from a different network is also possible but requires port-forwarding and access to the router. To date the server functionality has been implemented but it has not yet been multithreaded, which is important to handle more than one user at a time.

Data is sent back and forth between the server and the client using “UserMessages” and “ServerMessages” (see Figure 15). In the final design, these messages are sent as .json files.

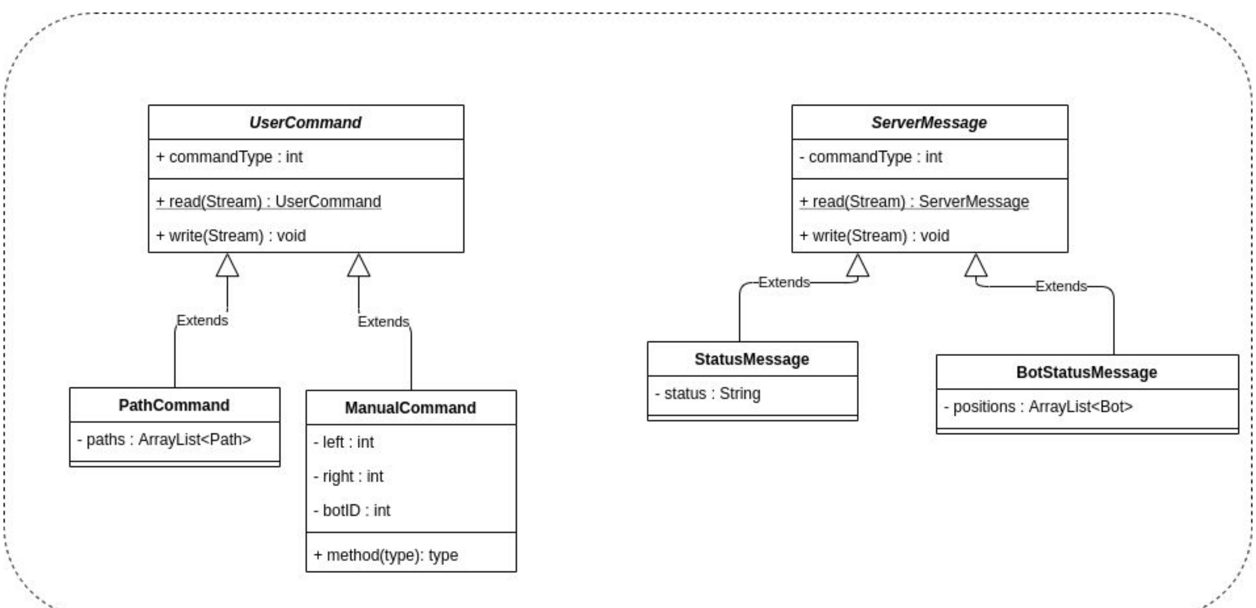


Figure 15: Message types for client-server communication in a remote furniture control system

4.2.5 Controllers

The server and the client each have control blocks. This is a concept from MVC that allows the user to partition the communication from business logic and the program view.

In figure 16 we can see that the client has one overarching controller which contains both a model controller to manipulate the model and a server controller to communicate with the server. In figure 17 the server module is solely responsible for listening for incoming connections. Once a connection is established it is handed off to a server thread which manipulates the server model and also communicates with each bot by writing the commands as a byte stream to the NRF24I01 module.

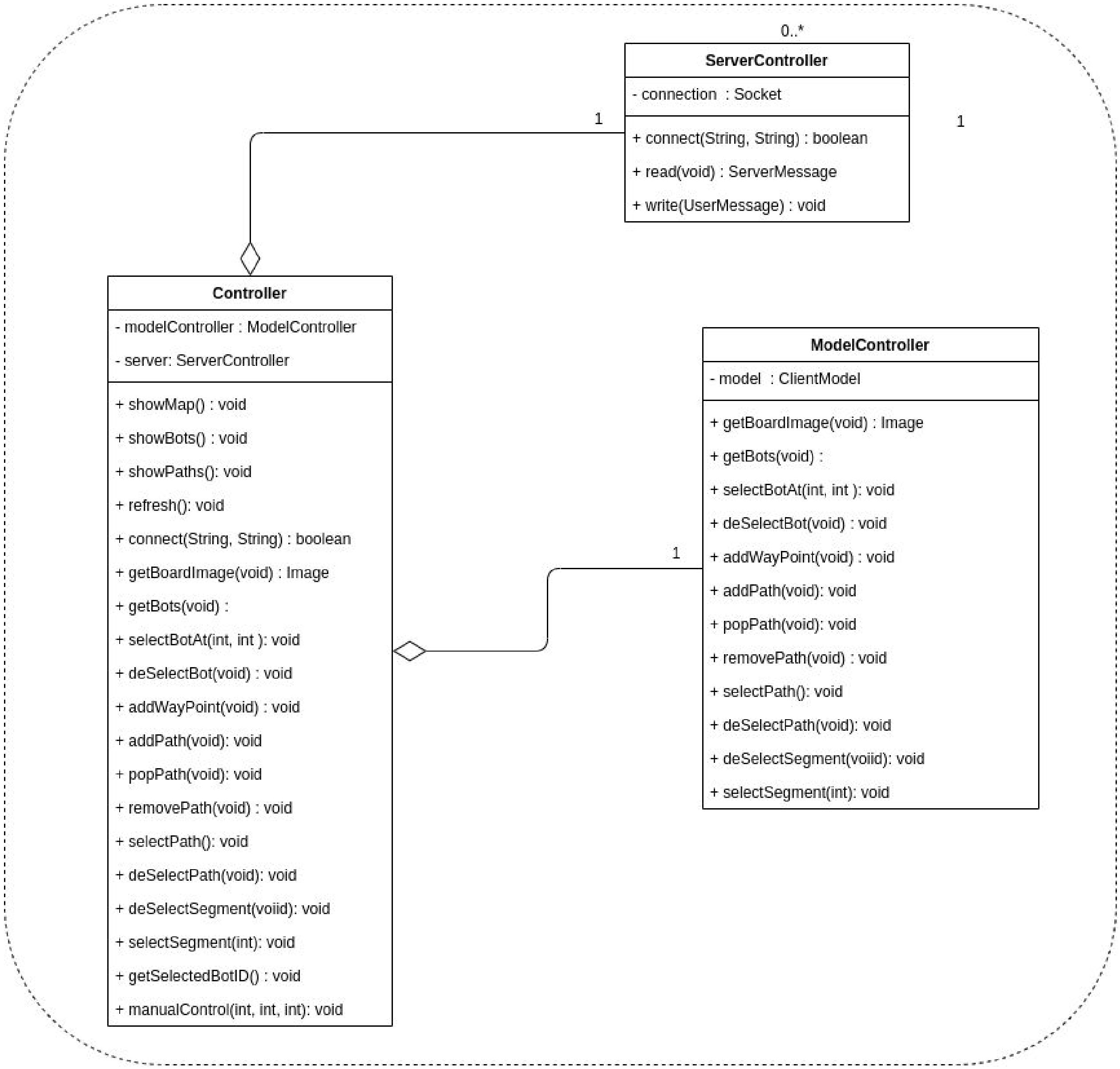


Figure 16: Client controller for a remote furniture manipulation system

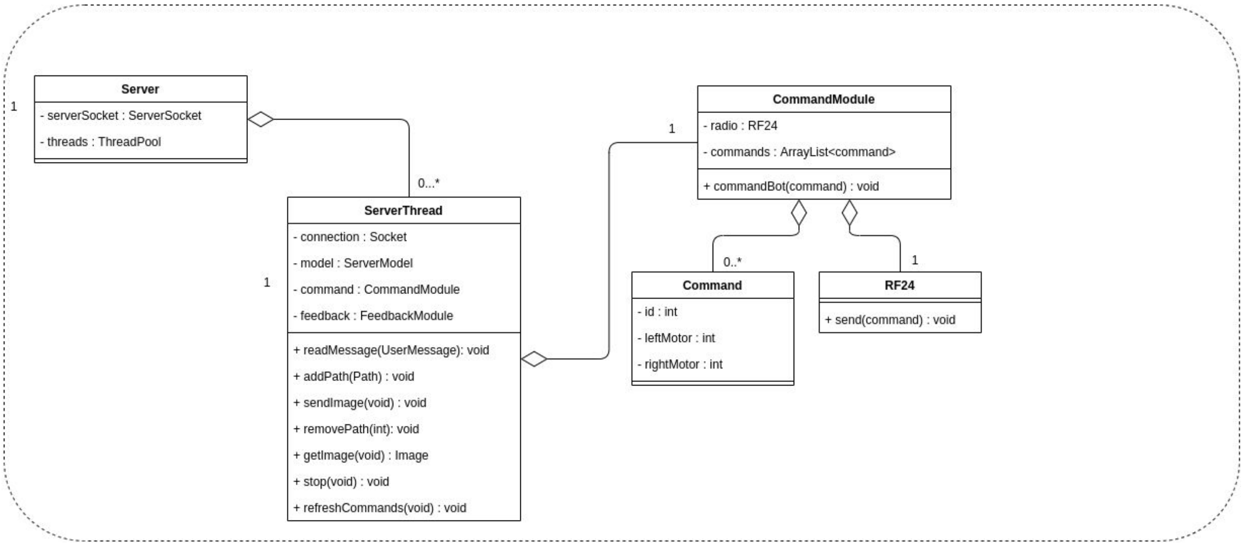


Figure 17: Server controller for a remote furniture manipulation system

5. Engineering Analysis

To ensure that our software would execute properly on the Raspberry Pi, we chose the 4GB model, which proved to be more than enough RAM. Testing of the device showed that while running the program, the Pi was only using 1.5GB of memory. These tests however were executed without implementation of the image processing and with only two robots running. The group is hopeful that upon implementation of the remaining software, that 4GB of ram will be more than enough to have the system run smoothly.

Another important factor when choosing a device to run our software on was the wireless capabilities. In order for the product to function properly, a clean uninterrupted signal is required to ensure consistent communication between the Pi and client. The Raspberry Pi comes with a built in WiFi card while the Arduino Nano microcontrollers did not. In order to establish a connection between the microcontroller and the Pi, NRF24l01 transceiver modules were used. The group had no issues with the wireless connectivities of the product.

To ensure each Furniture will remain active for a long enough period of time a simple power budget was made based on all the existing components in each furniture. As mentioned before, the batteries operate at 6V with a 8000mAh capacity. Assuming the furniture is constantly in use we find a total power loss of 4.93W as shown in Table 1.

Table 1: Power Draw of each component on a single Furniture

Component	Arduino Nano	NRF24l01	DC Motors	L293D	MP1584EN	Total
Power Draw (W)	0.25	0.045	3.60	0.84	0.19	4.93

Therefore the total time of operation can be easily calculated with the following formula:

$$Total\ Time = (Voltage * mAh)/(Power\ Draw * 1000)$$
$$Total\ Time = (6V * 8000mAh)/(4.93W * 1000)$$
$$Total\ Time = 9.74\ hours$$

This same calculation can be done for the furniture operating in “low power mode” or with the motors disabled through the motor controller. This cuts down on the power of each of the components to roughly 1.325W, leading to:

$$Total\ Time = (Voltage * mAh)/(Power\ Draw * 1000)$$
$$Total\ Time = (6V * 8000mAh)/(1.325W * 1000)$$
$$Total\ Time = 36.2\ hours$$

Therefore when not receiving any commands, the furniture can last a very long time in standby mode. Applying this to the scaled version of the solution, the battery life will be vastly extended. Given the ability to charge the furniture overnight while in their “home” positions this leaves no issues in terms of power usage.

Finally when the enclosure and robots were modeled a CAD of the components was also made. This allowed us to precisely place mounting holes and other key details. This helped reduce the number of prints needed and the mechanical work to make the parts fit together. A pair of calipers was used to measure the size of different parts for their individual CADed components.

6. Enclosure

This project was composed of 3 main enclosures. The first being the robot, the camera/image processing unit and the testing apparatus. Each component has a different function and shape.

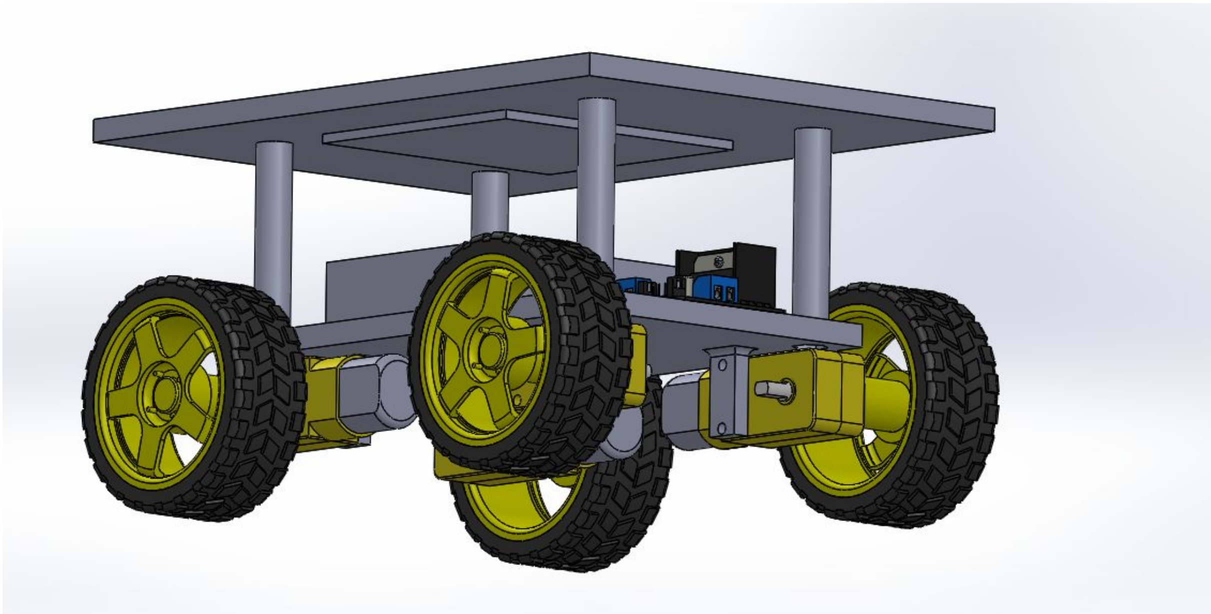


Figure 18: 3D Render of Robot

The robot was composed of 3 parts each of which was 3D printed. It took a total time of 16 hours to print the robot which is purely composed of PLA. The footprint of the robot is 16.5 x 20 cm. On the first iteration of the robot the motors were mounted with glue however this proved to be too weak and less forgiving. The base of the robot was given jigs to mount the motors onto using screws and bolts. This allowed for easy mounting of the motors. Standoffs were then printed and indents were added to the top plate giving a flush finish. A total of 3 sets of robots were printed before the impact of COVID-19 delayed the development of the project.

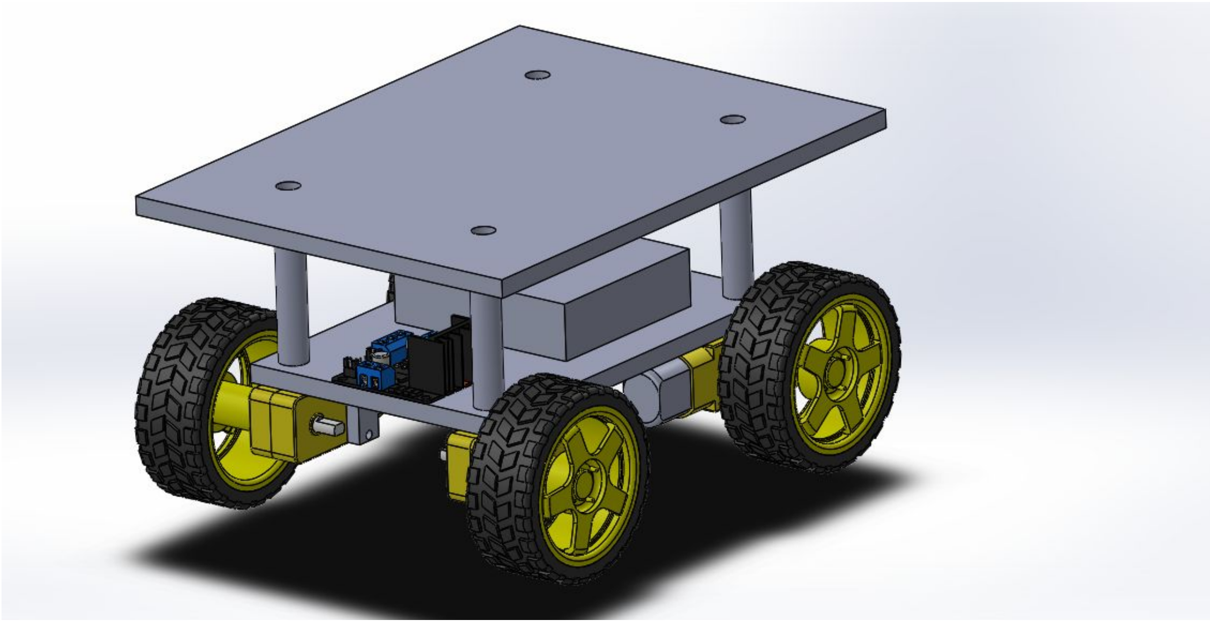


Figure 19: Robot from different angle

The camera mount was only able to reach the CAD phase of its development and was not printed. Concepts of the mount would have “wings” on the side of the enclosure to allow the use of tape-velcro or tape to mount it to the ceiling. These wings can be seen in the figure below. A hole was left in the bottom of the mount for the placement of the camera. The enclosure was big enough for both the storage of the camera and of the Raspberry Pi 4.

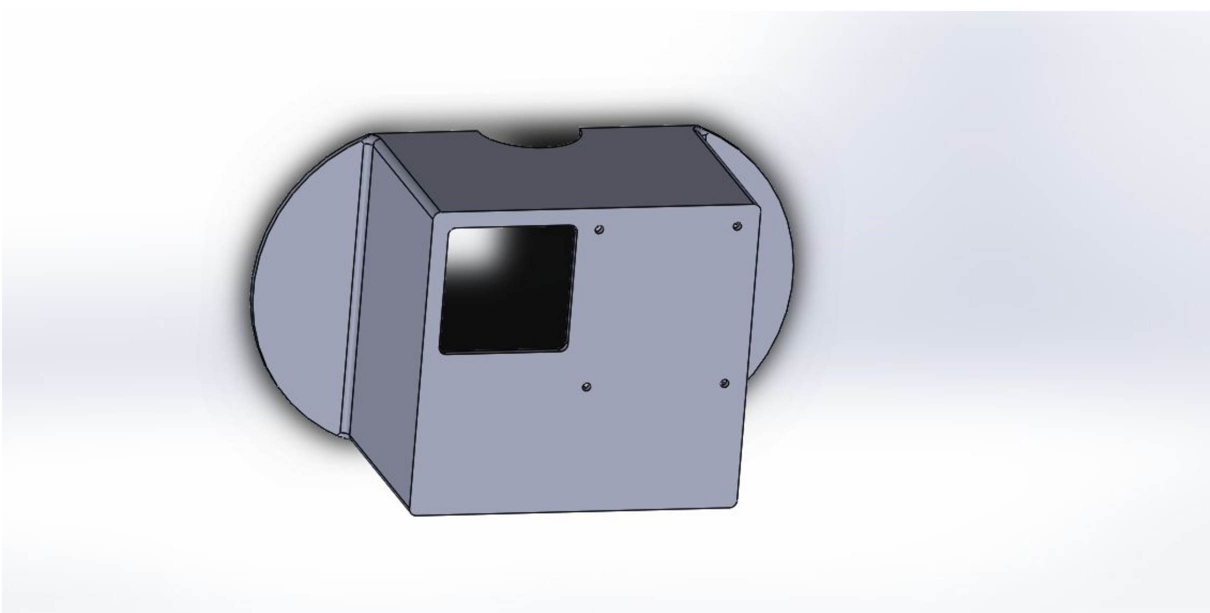


Figure 20: Camera mount

The testing apparatus was made from 2x4 wood pieces. This apparatus would span over the testing area providing a pseudo-ceiling for the testing environment. With the size of 1m x 1m it would cover a large enough area for the development of path planning with the robots.

7. Printed Circuit Board

The Free Flowing Furniture PCB as shown in Appendix 2 has an overall size of 85mm by 70mm with 2 layers. This PCB houses all the components required for each Furniture, specifically the Arduino Nano, MP1584EN DC Buck converter, NRF24L01 receiver transmitter module and L293D motor controller. Alongside these components are 3 status LEDs with their required current limiting resistors, a capacitor to smooth the 3.3v input to the NRF24L01 module and a switch to control the power. The board also has pad connections for the power supply and motor output which are connected externally.

This PCB design is the second iteration as a prototype PCB was initially produced and ordered. The original prototype lacked a switch for the power and used a secondary motor controller separate from the PCB. The second order also fixed routing errors in the original prototype, however it couldn't be ordered due to the unfortunate circumstances surrounding Covid-19.

Both the schematic and PCB design were produced with the software EAGLE. The prototype PCB was ordered through JLC PCB for $18.17 and shipping costs were split with another order. The PCB took 8 days to manufacture and arrive once ordered.

8. Regulatory Codes

To ensure that Free Flowing Furniture can be successfully pushed to consumer markets, codes laid out by the Standards Council of Canada would have to be considered. Below is a list of standards, domestic and international, that must be considered.

8.1 International Organization of Standardization

8.1.1 ISO 12100 [3] - Safety of Machinery - Ensure the protection of the consumer.

8.1.2 ISO 9001 [4] - Quality Management Systems- To ensure the product meets the quality standards expected by the consumer.

8.1.3 ISO 27701 [5] - Information Security and Privacy Protection - Ensure the user's data is handled appropriately and that their privacy rights are not infringed upon.

8.1.4 ISO 10218-1 [6] - Safety requirements for industrial robots - Safety requirements for industrial robots can be utilized for our non-industrial robot.

8.1.5 ISO 90003 [7] - Software Engineering - Guidelines for development and distribution of software.

8.2 CSA Group

8.2.1 CSA-Z434-14 [8] - Safety of Robotics - To ensure the safety of the consumer as well as their property.

8.3 Government of Canada

8.3.1 ICES-Gen [9] - Interference-causing Equipment Guidelines - To ensure that the device meets wireless noise standards.

8.3.2 RSS-216 [10] - Wireless Power Guidelines - To ensure that if wireless charging were implemented, satisfactory noise levels are met.

In order to gain regulatory approval, each committee and code outlines a process that companies can use to meet necessary requirements. For ISO 9001 [11], one requirement for approval is that the company must present a number of documents, such as a quality policy and quality objectives. These documents are not generic and must be company specific. Once these documents exist, the policies and procedures outlined within them must be implemented throughout the company. Afterwards, the company is expected to perform an internal audit to ensure proper implementation. The last step in achieving ISO 9001 certification is completed by having an audit performed by an independent third party auditor who will also see if the proper policies and procedures are in place. Upon completion of this audit, if there are no significant problems, the company will receive an ISO 9001 certificate.

9. Design Alternative Choices

9.1. Hardware

Most of the hardware choices for this project were COTS. The COTS products have provided a time-effective solution to this project. Decisions were made in which it was deemed better to over-shoot the ability of processors to make sure there would not be issues when programming the project.

Three processor boards were considered for the image processing for this project. Each having a different level of processing power and speed. The first consideration was an Ardunio. This would provide a small and compact solution along with a cheap cost. However, it was feared that setting up OpenCV on an arduino would prove to be too difficult and processing power behind the device would be too small. The next option was an Nvidia Jetson TX1. This board was made to run OpenCV for image processing and provided a computer like UI. The drawbacks were that the board itself and power supply were quite large. Additionally the cost of the board was much more ~$600 USD which does not fit in our budget. This is when the Raspberry Pi option appeared. The small single board computer fit all the requirements in terms of processing and had the added benefit of easy UI similar to the Jetson TX1. There was also documentation behind using OpenCV on Raspberry Pi’s which gave the team an added benefit.

Once we decided to use an Arduino board, we looked into using the ATMega328 microprocessor separately. A complete furniture was completed using the ATMega328 microprocessor on a breadboard, however a PCB was not created before prototyping was completed. To use this microprocessor a second MP1584EN DC buck converter was needed alongside and a 16Mhz quartz clock to run properly while an arduino uno was used to bootload the software.

9.2. Software

For the localization makers two different libraries were considered, April Tags and Aruco markers. April Tags have been used by NASA for their 3D localization and have a BSD license. Identification range is farther with April Tags and is more computationally efficient compared to Aruco markers. Aruco markers on the other hand are easy to set up, online generators are easily accessible, and when given default parameters whas fewer false detections. This means that the amount of a setup time for the markers would be less however at the cost of quality. The decision was made to use the aruco maker library because it would only need the use of OpenCV without the need of external dependencies. In addition, the extra time spared instead of setting up was used to dedicate to other parts of the project.

10. Future Work

There are multiple aspects of this project that can be improved upon. The most important would be moving to the project to a full scale demonstration. Due to costs and space accessibility this project had to be scaled down which has both benefits and drawbacks. To scale up the project the type of motors and wheels would have to be reconsidered. For example instead of using small 5V DC motors larger 12V motors with larger torque and speed characteristics would be needed. This would most likely require a change in power supplies. However, some of our devices require 5V power such as the Raspberry Pi. The types of wheels would also have to be changed. All the wheels used in the small scale model were tractions wheels. Since the mass of the robot was light enough there was not enough scrub to prohibit the robot from turning. In a full scale version scrub would be a much larger issue and the use of omni wheels could be used. A different building material for the tables would also be needed instead of PLA as PLA would be too weak.

Another improvement to be made would be the inclusion of an IMU and encoders. When trying to callback the video stream from the camera would slow down the system. This would limit the response time of movement commands sent to the robot. By having an IMU the camera could take a snapshot of the locations and orientation of the robots. The Raspberry Pi could then send setpoints to each robot and then the robots could run their own error reduction. This would free up the signal space and allow for faster response time of the robots. It would also remove the need for a higher frame rate camera.

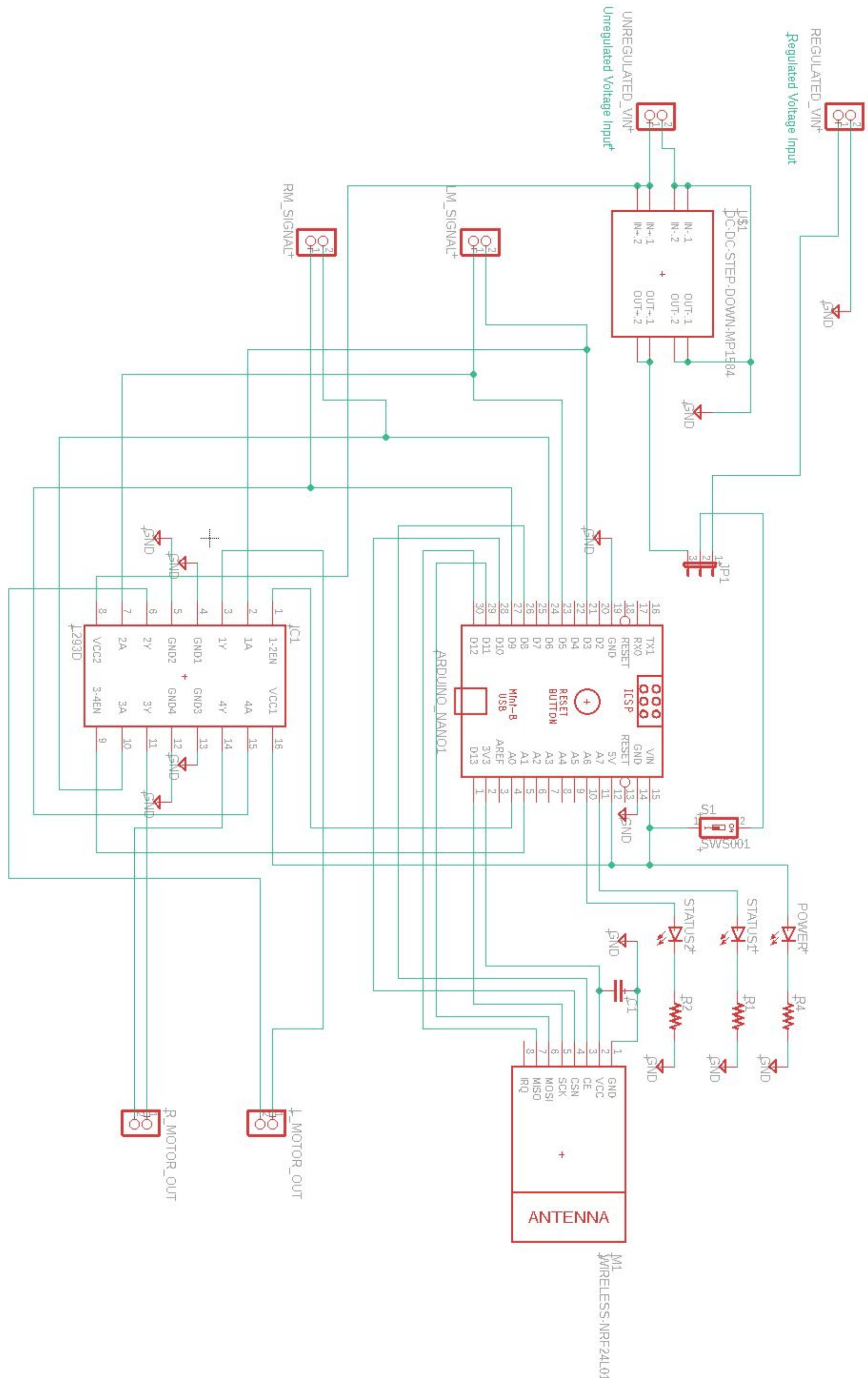
The final consideration of this project is the use of cloud processing. It would help spread the load of the work done by the processor. Cloud processing would allow for easy scalability to other rooms and possibly keep the price of the project lower as an individual processor for each room would not be needed.

11. References

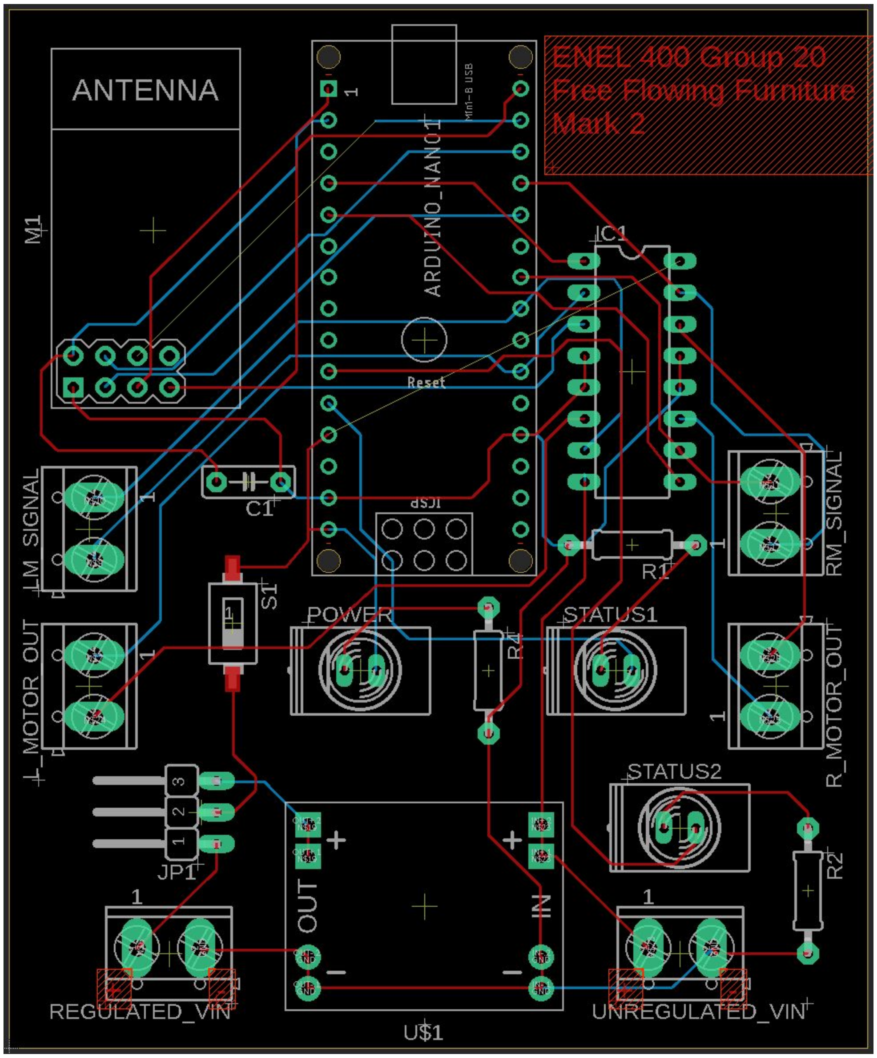
[1] A. Márquez-Espinoza, J. G. Mercado-Rojas, G. Vega-Martínez, and C. Alvarado-Serrano, "ECG Ambulatory System for Long Term Monitoring of Heart Rate Dynamics," in *Practical Applications and Solutions Using LabVIEW™ Software*, S. F. Folea, Ed. InTech, 2011, pp. 201–226.
[2] "L293x Quadruple Half-HDrivers," Jan-2016. [Online]. Available: http://www.ti.com/lit/ds/symlink/l293.pdf. [Accessed: 10-Apr-2020].
[3] *Safety of machinery — General principles for design — Risk assessment and risk reduction*, ISO 12100:2010, Nov. 2010.
[4] *Quality management systems — Requirements,* ISO 9001:2015, Sep. 2015.
[5] *Security techniques — Extension to ISO/IEC 27001 and ISO/IEC 27002 for privacy information management — Requirements and guidelines* ISO/IEC 27701:2019, Aug. 2019.
[6] *Robots and robotic devices — Safety requirements for industrial robots — Part 1: Robots,* ISO 10218-1:2011, Jul. 2011.
[7] *Software engineering — Guidelines for the application of ISO 9001:2015 to computer software*, ISO/IEC/IEEE 90003:2018, Nov. 2018.
[8] *Industrial robots and robot systems*, CAN/CSA-Z434-14, Nov. 2014.
[9] *General Requirements for Compliance of Interference-Causing Equipment*, ICES-Gen, Jul. 2018.
[10] *Wireless Power Transfer Devices*, RSS-216, Jan. 2016.
[11] "How to get ISO 9001 Certification | Step-by-Step Guide | 9001Simplified," *9001Simplified*. [Online]. Available: https://www.9001simplified.com/iso-9001-certification-steps.php. [Accessed: 10-Apr-2020].
[12] K. Chow, T. Gangadharaiah, M. Gelfand, Z. Lau, and B. Zhou, "Free Flowing Furniture Project Execution Plan," Version 1, Ledingham Inc. Design Group, Schulich School of Engineering, University of Calgary, Canada, 2020.

12. Appendices

Appendix 1: Schematics



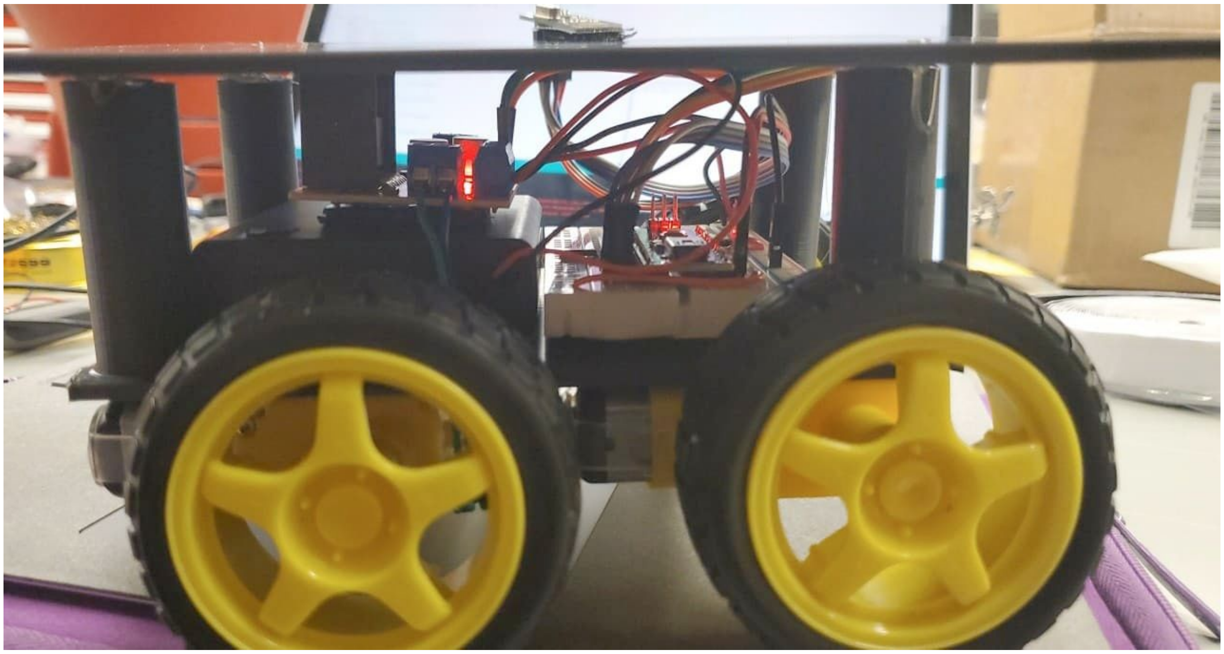
Appendix 2: PCB Layout



Appendix 3: Bill of Materials (BOM)

Item	Schematic Number	Manufacturer part number	Manufacturer	Shipping agency	Price per part ($)	Quantity	Total Price
MicroController	ARDUINO_NANO	ATMEGA328P-PU	Microchip Technology	Digikey	2.08	10	20.80
300Ω Resistor	R1	N/A	N/A	Provided	0.00	1	0.00
300Ω Resistor	R2	N/A	N/A	Provided	0.00	1	0.00
300Ω Resistor	R3	N/A	N/A	Provided	0.00	1	0.00
10uF Capacitor	C1	N/A	N/A	Provided	0.00	1	0.00
Inertial measurement unit	N/A	MPU-9250	DSD TECH	Amazon	9.99	4	39.96
Voltage regulator	US1	L7805CV	STMicroelectronics	DIgikey	0.76	10	7.60
Raspberry Pi	N/A	BCM2711	Raspberry Pi Foundation	Already bought	0.00	N/A	0.00
Lifecam hd-3000 Camera	N/A	T4H-00002	Raspberry Pi Foundation	Already bought	0.00	N/A	0.00
Arduino Uno	N/A	A000066	Arduino	Amazon	23.00	1	23.00
Transciever Reciever Modules	M1	NRF24L01+	Longruner	Amazon	16.99	4	67.96
Clock	N/A	HC-49S	Bolsen	Amazon	15.87	1	15.87
Batteries	N/A	6LR61-24	AmazonBasics	Amazon	18.66	1	18.66
Battery Holder	N/A	EB-CA-477	Ancable	Amazon	11.23	1	11.23
Motor Kit	N/A	L298N	KeeYees	Amazon	13.99	1	13.99
2x4 Lumber	N/A	090214009272	Home Depot	Bought in-store	4.18	3	12.54
Wood Screws	N/A	622412018895	Home Depot	Bought in-store	3.12	1	3.12
Corner Brackets	N/A	622412342914	Home Depot	Bought in-store	2.23	4	8.92
Nuts	N/A	3202	Home Depot	Bought in-store	0.13	16	2.08
Bolts	N/A	3032	Home Depot	Bought in-store	0.88	16	14.08
						Grand Total	259.81

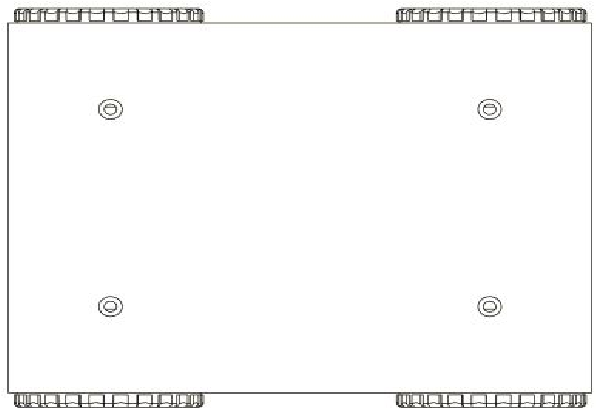
Appendix 4: Pictures of Prototype



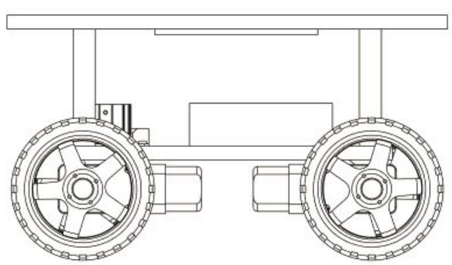
Note: Due to COVID-19, this is the only picture of the prototype. The group was unable to get any pictures of the enclosure.

Appendix 5: 3D Models of Printed Components

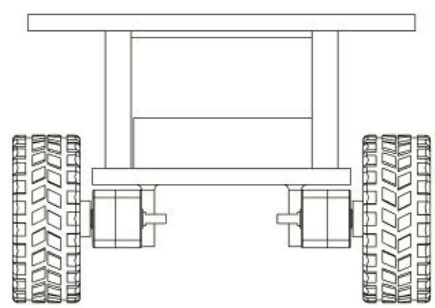
Robot Drawing



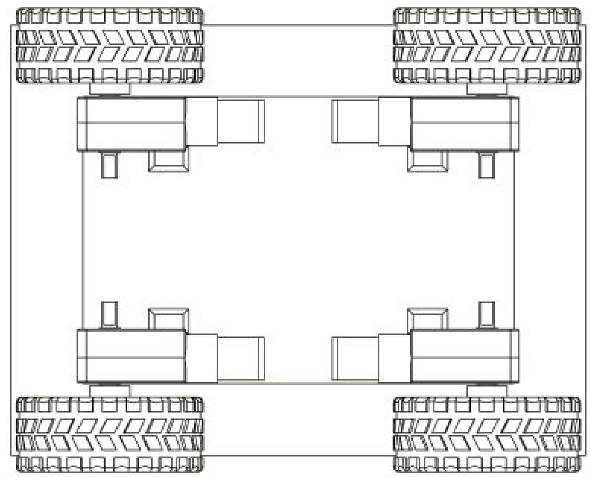
Top



Front

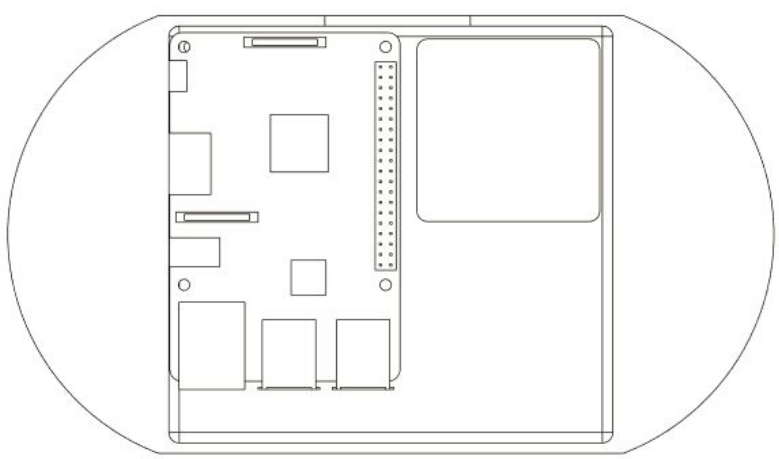


Right

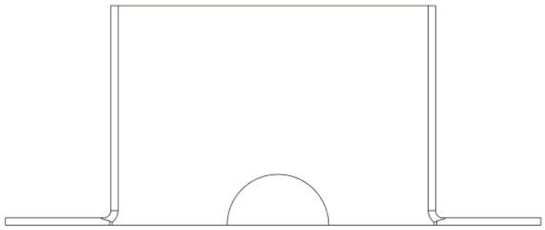


Bottom

Camera Mount Drawing

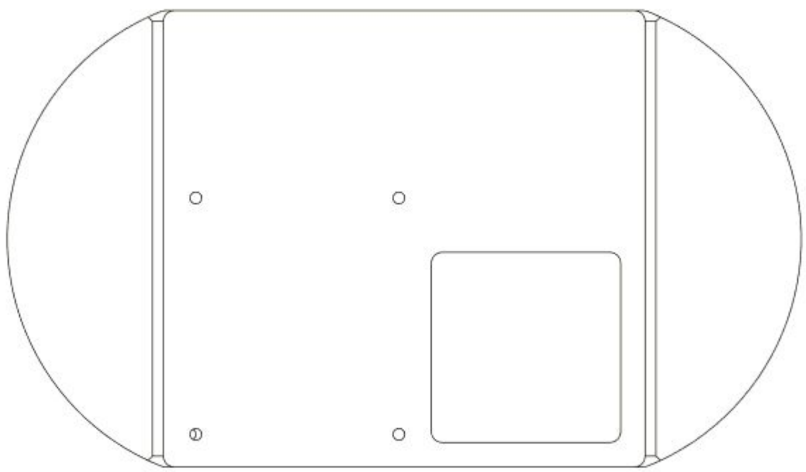


Top



Front

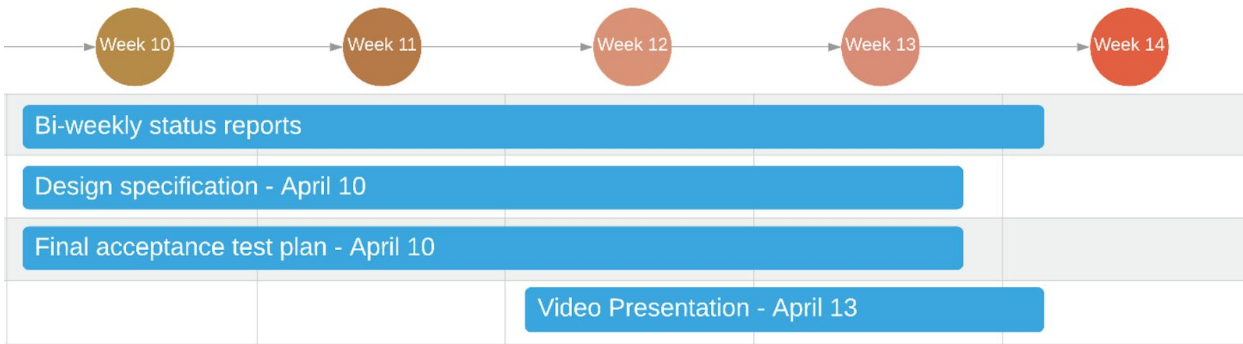
Right



Bottom

Appendix 6: Variance in Schedule and Expenditures

Schedule



Revised schedule

Overall, all deliverables in the originally planned schedule as seen in Appendix A of the project execution plan [12], has been completed on time and on schedule up until the Coronavirus situation which occured in the end of week 9 which resulted in the canceling of all in person deliverables. Before the cancelling, the only major variance was in the PCB component assembly, WBS 1.1.4.. This variance was due to the PCB that we ordered being inaccurately assembled by the manufacturer, which was partially due to an error in our PCB design. If the project has continued as planned, we would have had to reorder our PCB, which would have delayed that component of the project. However, all other previously required deliverables have been completed on time. Due to the change in deliverables format due to social distancing measures, much of the deliverables have been moved online. The revised schedule is shown above. Thus far, the new schedule has also been adhered to accurately with minimal variance. However, this schedule is very different from the originally planned schedule. The work done on the poster presentation in week 8 and 9 has been integrated into the video presentation and the final presentation and acceptance test report have all been cancelled. Our groups prototyping and design of physical components have also been cancelled as a result of this schedule change.

Expenditures

Part	Planned Price ($)	Actual price ($)
Furniture Components	60.00	29.89
PCB Fabrication	50.00	18.17
PCB Components	145.00	173.31
Camera System	105.00	0.00
Testing	0.00	40.74
Estimated Cost	360.00	262.11

For expenditures, all in all, we ended up using less of our total budget as well as our planned budget. This is partially due to already having some of the parts we needed for the project. A big variance was that we did not anticipate testing to have a cost associated with it, therefore we did not plan for it in our initial budget. Luckily, furniture components as well as PCB fabrication both came far under budget. One major factor that caused our expenditures to be under the planned expenditure was the cancellation of physical deliverables of the project in week 9, which meant that all physical development of the project came to a halt. Had this not occured, the additional PCB ordering that would have been required, as well as the additional furniture that would have been assembled, may have increased the cost beyond the estimate. Much of the work on the physical components of the camera would have also been done in the later stages of the project, which meant that had the physical components of the project not been cancelled, there would have been an associated cost there as well. However, had the physical deliverables proceeded as normal, while we would have expected the expenditure to go above the initial estimate, we still would not have expected to exceed the maximum expenditure of $500.00.