PongPal

Authors: Seung Yun Lee, Michael Bahner, Alex Kireeff
Affiliation: Electrical and Computer Engineering, Carnegie Mellon University

***Abstract*—While the game of water pong is widely enjoyed, there are no existing solutions that allows someone to play the game remotely. We designed PongPal to be an accessible platform for anyone to enjoy the game. The system consists of a robot that a user can control through a web application, and utilizes computer vision to detect the game state and give feedback on the user's performance.**

***Index Terms*—Computer Vision, Networks, PID, Robotics, Water Pong**

1 INTRODUCTION

Water pong is a game people all over the world love to play together, but problems can arise when it comes to accessibility. The goal of this project is to provide an alternative way to play water pong with your friends regardless of these issues that may arise.

While there has been pong playing robots in the past, [1], they've been mostly focused on automatic playing agent. PongPal allows someone to remotely control a pong robot through a website, and get feedback when the state of the game changes. The website allows the user to set the vertical angle of the barrel, its radial angle, and how much pressure the cannon will use when launching the ball. In response to their shot, the website will update a figure of the current rack of cups, and tell the user if their ball went into a cup or where it missed if that was the case. Through this set up, anyone who wants to join the game can go to the website and start playing, even if there are real world barriers preventing them from physically playing.

2 USE-CASE REQUIREMENTS

PongPal is primarily an entertainment platform, with the goal of spreading the joy of playing the game of water pong and lowering the barrier to entry. With that overarching goal, the gameplay experience with PongPal has to be stress-free and enjoyable. From this, three use-case requirements naturally follows.

2.1 Accessibility

The biggest use-case requirement for our project is accessibility. There are two user groups to our product: the ones who are controlling the robot, and the ones who are setting up the robot. The first user group are likely isolated from their friends and don't have much experience playing the game. For them, it is crucial that the UI is intuitive so that they can play the game without any guidance, so the experience is not frustrating or confusing. To that end, **at least 90%** of the users who try the product for the first time should be able to play the full game without outside interference. For the other group, setting up and storage of the robot should be simple: this means that the robot cannot be too heavy, and the form factor should be reasonably small. The robot should have a **maximum form factor of 40cm * 40cm * 50cm** when stored, and the weight should be **less than 5kg**.

2.2 Repeatability

The user should experience minimal variance while playing the game. The drift of an individual shot is inevitable and adds entertainment factor stemming from unpredictability. However, if the variability of each shot is too high, then it completely eliminates the skill factor of the game, which makes the game not fun to play. Assuming the same control settings, the shot's depth variance **should not exceed 5cm** and the horizontal variance **should not exceed 2cm**. This is motivated from the fact that the radius of the standard pong cup is 5cm, so if the user is aiming at dead center of the cup, they should reliably make it into the cup. This accuracy guarantee is valid only if the distance between the ping pong ball's stationary location and the intended target is **1.5m to 3m away** - this is the typical distance between the cups and an individual when one is playing the game.

2.3 Accuracy

The second biggest component of PongPal is the feedback system. The user should be able to gain insight from previous shots they've shot, and make appropriate adjustments. Instead of showing a video stream, PongPal will detect the game state and display them to the user. The platform will detect the trajectory of the shot and predict where the ball will land, and detect the cups' locations. This detection must be accurate **within 5cm** of the ground truth. The motivation for this number is the same as the previous requirement in which the radius of the cup is 5cm.

2.4 Responsiveness

The last use case requirement is the responsiveness of the system. Water pong is a fast-paced game, and if the interval between the shots drag on, it will not be a pleasant experience for everyone involved in the game. After the shot, the platform must be able to detect the cups' location, ball's landing spot, and if the ball went into the cup all **within 5 seconds**, so that the game play loop can be kept speedy.

3 ARCHITECTURE AND/OR PRINCIPLE OF OPERATION

PongPal can be divided up into 4 major sub-systems: (1) the mechanical aiming system, (2) the computer vision system, (3) the pressure system, and (4) the user interface. The physical layout of all the subsystems can be found below (Fig 1), and the data flow between the parts is illustrated in the block diagram (Fig 5).

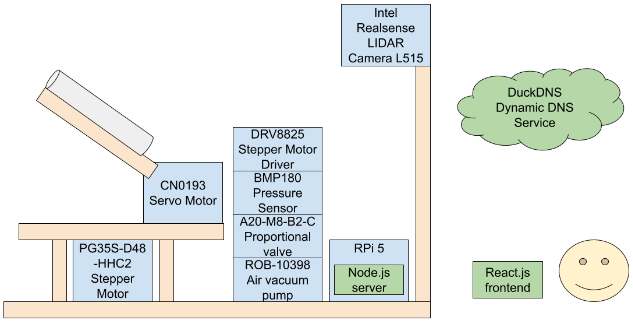

Figure 1: Architecture Overview

3.1 Mechanical Aiming System

The mechanical aiming system has two sub-components: the vertical aiming system and the radial aiming system. For vertical aiming, a servo motor is directly attached to the wooden arm. When the barrel is integrated to the aiming system, the PVC pipe that the ball will shoot out from will sit on top of this wooden arm. The radial aiming uses a lazy susan plate design, in which a circular plate is directly driven by a stepper motor, which sits under the plate. The servo motor will be directly communicate with the micro-controller, and the stepper motor will have a motor driver in between so that the micro-controller.

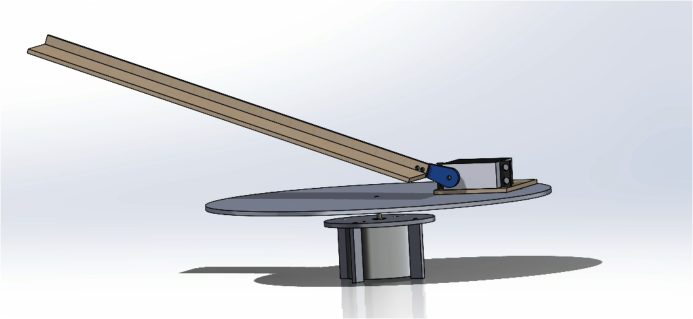

Figure 2: Mechanical aiming system rendering

3.2 Computer Vision System

The computer vision subsystem is in charge of detecting and localizing the ball after it is fired and calculating its trajectory to display on the website. Additionally, it is in charge of detecting and locating the cups on the table. All detection for the computer vision subsystem is done with an intel realsense L515 LIDAR which comes with a LIDAR, a camera, and an IMU (inertial measurement unit). The intel realsense L515 LIDAR will directly communicate with the Raspberry Pi which will extract the location and ultimately trajectory of the ball after firing as shown below.

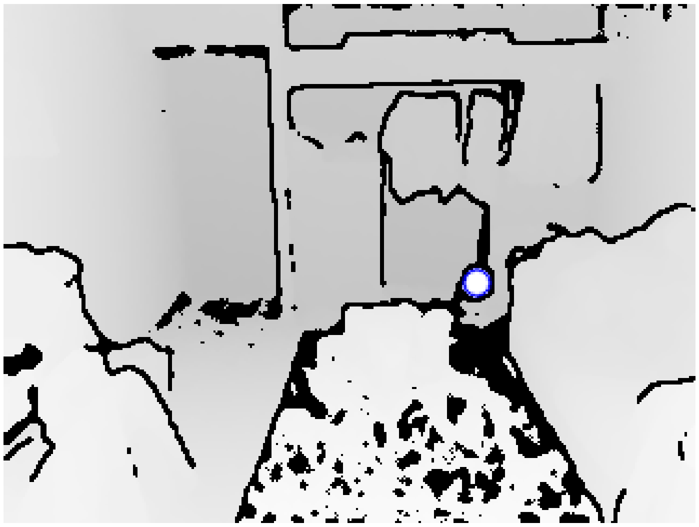

Figure 3: Example frame for the ball detection subtask

3.3 Launching System

The launching system consists of a barrel, a pressure chamber, a pump, a proportional solenoid valve and a regular solenoid valve. The proportional solenoid valve is used to release air when the pressure is too high whereas the regular solenoid valve is actuated when the user wants to fire. We will be using a PID controller (implemented by the Raspberry Pi) to control the pressure inside the solenoid. Further details about the launching system are in system implementation section.

3.4 User Interface

The user interface is intuitive and helps the user make informed decisions without the need for guidance while they play. On the left side of the UI, it will first prompt the user to select their radial angle for the cannon by dragging a slider. Then, they are able to select the vertical angle of the barrel with another slider. Lastly, they are able to select the power at which the ball is fired with another slider. When they choose to hit the fire button, it will send a signal to the robot to shoot the ball, and after processing the robot will send the ball location data back. In return, the right side of the UI will display where the ball landed in relation to the cups, or mark a cup if the ball went in. The way we set up the sliders makes playing setting the cannon and shooting the ball simple and self explanatory in order to make an intuitive system that responds to user as quickly and efficiently as possible. This whole system can be seen in figure 4.

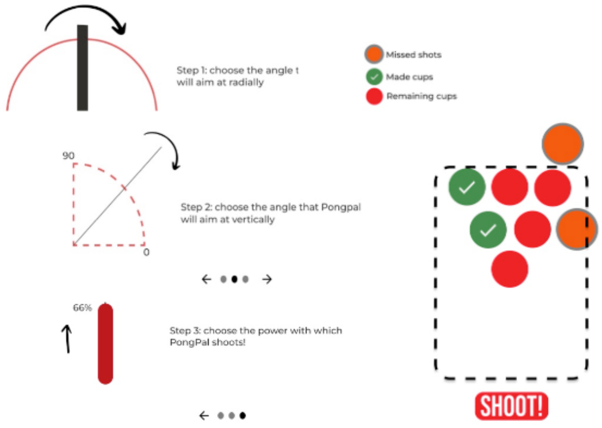

Figure 4: Graphical depiction of planned UI

4 DESIGN REQUIREMENTS

Based on the use-case requirements we've identified on the previous section, we have derived few design requirements on the performance of the individual sub-systems.

4.1 Aiming Granularity

The aiming system should be capable of providing a diverse range of angular granularity. We can consider the $5cm * 2cm$ bounding box identified in the use case requirement as our basic unit of granularity. That is, we should be able to draw a grid of $5cm * 2cm$ boxes in the valid range, and the aiming system should be capable of aiming at every corners. We also observe that the valid target range is from $1.5m$ to $3m$ vertically, and $74cm$ horizontally (which is the width of a standard pong table.

For the horizontal angular aiming, we can derive the required minimal granularity by finding the angle formed by the two points on the imaginary grid and the origin in which the robot is placed. The minimal angular difference can be found near the maximal range of the robot and the point right below it, as shown in figure 6. This angle turns out to be

$$arctan(\frac{74}{300}) - arctan(\frac{74}{295}) = 0.14^\circ \tag{1}$$

Therefore, the stepper motor that is responsible for radial aiming should be able to provide **at least** $360/0.14 = 2572$ **granularity per rotation**, which can be achieved either through taking a full step or a half step.

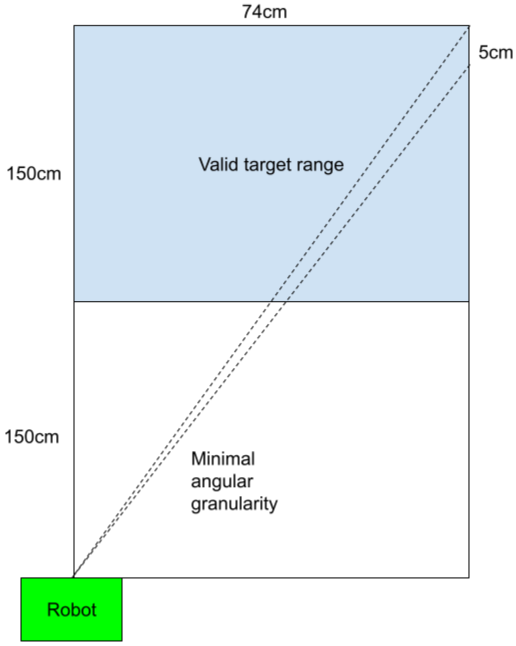

Figure 6: Minimal horizontal aiming granularity

The vertical aiming has a more relaxed granularity requirement, as two different subsystems are responsible for the distance in which the ball travels: the vertical aiming subsystem and the pressure subsystem. Together, they should be able to provide **at least 30 different vertical distance options 5cm apart from each other**, since the supported vertical range is 1.5m long.

4.2 Vision Hardware's Performance

Since the game of pong is played in various lighting environment, the vision hardware has to be resilient to lighting variations. The most adverse environment would have the color and the intensity of the background lighting changing multiple times in a second. Therefore, the camera should provide a sensor data that is resilient to varying lighting conditions. Since LIDAR camera would provide such resiliency, we've decided to use it for our project.

This LIDAR camera should provide a good depth accuracy for both cup detection and the ball trajectory detection. The pingpong ball has a diameter of 40mm, so having a variance greater than that would lead to inaccurate trajectory detection. Therefore, the depth accuracy of the camera should be **at most 40mm**. Moreover, when measured by hand, the average flight time of a ping pong ball during the game is around $0.5s$. For accurate trajectory calculation, we would need at least 10 frames of images, which requires the camera to have a **minimum fps of 20fps**.

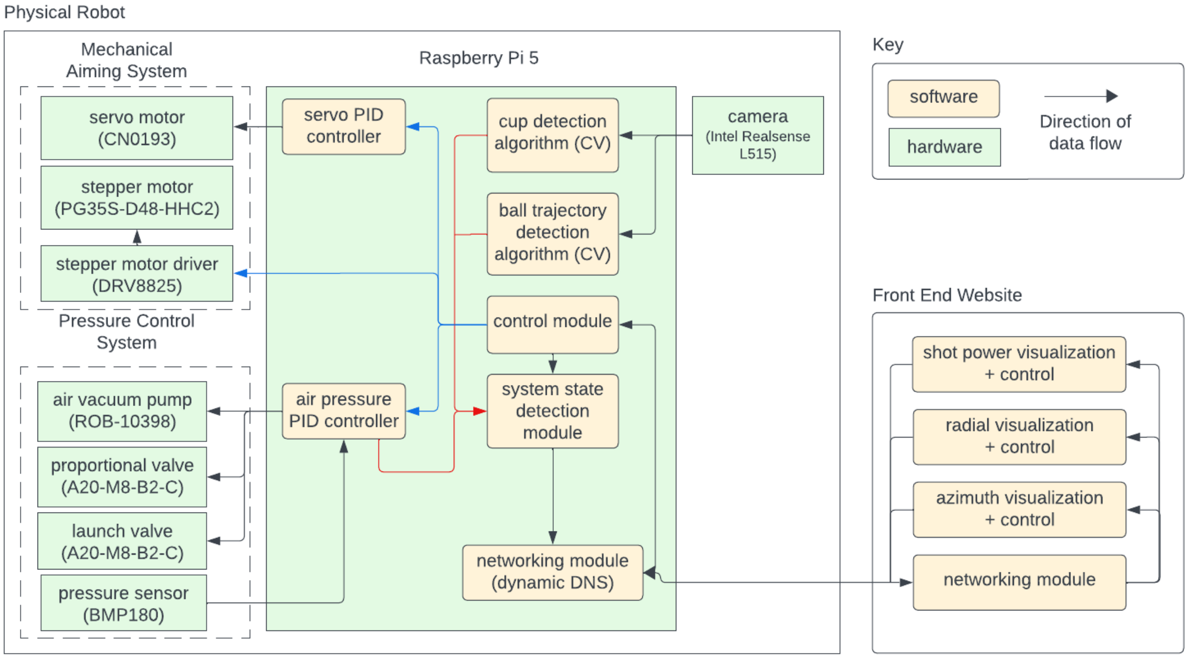

Figure 5: Block diagram of data flow

4.3 Pressure Requirements

sThe launching subsystem should be capable of launching a ping pong ball 0.3 to 3 meters. To do so we need a pressure vessel that can hold the required amount of air at the required pressure, we also require a motor that can reach said required pressure. The math to calculate this is as follows:

$$\begin{aligned}
\text{ping pong ball mass} &= 2.7 \text{ grams} \\
\text{max shot distance} &= 3 \text{ meters} \\
\text{max shot relative apex} &= 1 \text{ meter} \\
\text{max shot speed} &= 3 \text{ meters per second} \\
\text{barrel length} &= 0.2 \text{ meters} \\
\text{barrel radius} &= 0.04 \text{ meters} \\
\text{total energy} &= \ \text{max shot relative apex} \\
&\times \text{ping pong ball mass} \\
&\times \text{gravity} \\
&+ \frac{1}{2} \\
&\times \text{ping pong ball mass} \\
&\times \text{max shot speed}^2 \\
\text{barrel volume} &= \text{barrel length} \\
&\times \pi \times \text{barrel radius}^2 \\
\text{min volume} &= \frac{\text{total energy}}{\text{approximate pressure}} \\
&+ \text{barrel volume}
\end{aligned}$$

Since the motor we bought can go up to more than 200000 pascals, let us set the approximate pressure of the pressure vessel to be 100000 pascal, leading us to get:

$$\begin{aligned}
\text{minimum volume} &= 0.001005703 \text{ meters}^3 \\
&= 1005.703 \text{ centimeters}^3
\end{aligned}$$

For reference, a 2 liter coke bottle is about 2000 centimeters^3. We would like to make sure the reader is aware that this is a rough upper bound of the energy required for our launching system and we expect significant energy losses due to friction with the barrel and air escaping between the ping pong ball and the barrel. In all, we can clearly see that our design parts meet the required parameters to launch a ping pong ball 3 meters.

5 DESIGN TRADE STUDIES

Over the course of the design, we've considered different implementations of each subsystems, and their pros and cons. Below are some of the major design decisions we've made and their justifications.

5.1 Local vs Remote Image Analysis

Engaging in image analysis poses a considerable demand on computational resources, prompting us to explore the feasibility of conducting this task locally. This involved subjecting the Raspberry Pi to a series of tests, revealing its capability to proficiently analyze 300 frames (equivalent to 10 seconds of frames) within 4.3 seconds. Remarkably, this falls well within the confines of our specified constraints, affirming the practicality of local image analysis.

An alternative consideration was the prospect of offloading the image analysis to another computer. However, our evaluations unveiled that the same number of frames could be analyzed in 1.4 seconds, indicating a discernible increase in speed. Despite this apparent advantage, the decision was reached to maintain local image analysis. The rationale behind this choice stems from the recognition that outsourcing the task to another computer would introduce unnecessary complexity without offering any tangible benefits aligned with our specific use-case requirements.

5.2 Dynamic DNS vs Backend Server

Initially we had though about using a middle backend server for DNS management. This server would be our intermediary by receiving DNS update requests from our points of access, receiving and forwarding requests. While this approach could've worked, it added unnecessary complexity, as well as points of failure into our system, with the need for extra software and hardware maintenance, as well as more latency then what we ended up going with, dynamic DNS [2]. By using Duck DNS, a free dynamic DNS hosting service, we are able to point a fixed hostname to a dynamic DNS, bypassing the need for extra support from using a middle backend server. It makes the JSON communication between our robot and server much simpler and reliable, so we are able to focus more efforts on other areas of the project.

5.3 Linear Actuator vs Servo Motor

While designing the vertical aiming system, we've initially considered a linear actuator design that was attached on the top of the pole to pull in the barrel, as sketched in the figure 9

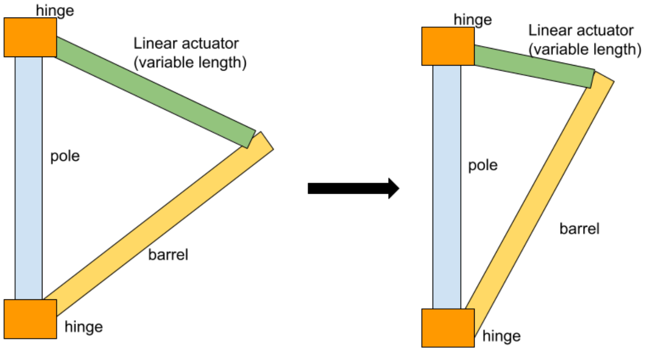

Figure 7: Alternative vertical aiming using linear actuator

This design has an advantage of allowing us to precisely control the vertical angle. If using a linear actuator with high gear ratio and accuracy, we can achieve high angular granularity. This would also be an improvement from a design which uses a stepper motor to "reel in" a string or a fishing line to control the angle, as that would introduce errors from the thickness of the string.

However, we decided to use servo motor to directly drive the barrel instead. First, the vertical aiming is mainly concerned with the vertical trajectory of the ball, and its granularity is not as important compared to the radial aiming. Since the distance the ball travels can additionally be controlled with a pressure value (using a PID controller), the vertical angle can have a coarser granularity. Second, the linear actuator design introduces more mechanical complexity that could render the system unreliable. Lastly, the cost of the linear actuator that at least has a range of $100mm$ is cost prohibitive. The cheapest one we could find was around \$100, which is unnecessarily expensive compared to the servo motor which was only around \$15. For these reasons, we've decided to use a servo motor instead of a linear actuator.

6 SYSTEM IMPLEMENTATION

6.1 Mechanical Aiming System

As shown in figure 4, the mechanical aiming system has two parts: the vertical aiming system and the radial aiming system. For the vertical aiming system a servo motor will be directly attached to the barrel in which the PVC pipe will be placed. We will be using a CN0193 Servo Motor, which has a rated torque value of $2010mNm$. The length of the barrel support is $200mm$, and assuming the center of mass is at the center of the support, we can calculate the maximum weight it can lift. First, to find out the maximum force it can apply,

$$F = \frac{T}{d} = \frac{2.01Nm}{0.1m} = 20.1N \tag{2}$$

From this, we derive the maximum weight it can lift as

$$m = \frac{F}{g} = \frac{20.1N}{9.81m/s^2} = 2.05kg \tag{3}$$

The barrel support is made out of plywood, and has the dimension of $50mm * 200mm * 3mm$. Given that the maximum density of a plywood is $700kg/m^3$, the weight of the vertical aiming support is

$$0.05m * 0.2m * 0.003m * 700kg/m^3 = 0.21kg \tag{4}$$

Therefore, the system can support $2.05kg - 0.21kg = 1.84kg$, which is more than enough for the PVC pipe barrel to sit on top of.

For the radial aiming system, the stepper motor that will be placed at the bottom is PG35S-D48-HHC2. This will be connected to a stepper motor driver DRV8825. The stepper motor we're using has 1698 steps per revolution, which does not meet the mimimum steps of 2572 as outlined in the design requirement. However, since we're using a stepper motor driver, we can use microstepping to boost the resolution. The driver support at maximum 1/32 steps, which means that the 1698 steps now has the granularity of $1698 * 32 = 54336$ steps per revolution, which is more than enough for radial aiming purposes.

The servo motor will be directly connected to the Raspberry Pi 5 micro-controller, and the stepper motor will first connect to the motor driver, which will then connect to the micro-controller. The RPi5 will communicate using GPIO to control these motors.

6.2 Computer Vision System

As discussed above, the point of the computer vision system is to:

1. Detect the ping pong ball landing location

2. Detect the cup locations

The computer vision subsystem will do this using an Intel Realsense L515 LIDAR [3]. Due to the difficulty in consistently detecting a ping pong ball before it lands in a cup, we have opted to detect the ping pong ball for a couple of frames while it is in the air and then calculate its trajectory to determine where it has landed. This requires that we accurately detect the ping pong ball and its location, which would normally be even more difficult, but thanks to the L515 this problem becomes tractable. We do this by recording frames from the L515 while the ping pong ball is in the air. Then we preprocess the frames to make the LIDAR noise less perceptable. Next, we use a blob detector to detect close circular objects. So far the above processing stack works exceptionally well and we only get a few extraneous detections when there is no ball in view while consistently getting around 3 frames of data on the location of the ball while it is traveling through the air. After that, we will process each of the detected circles to find the location using the depth sensor as well as the location in frame to map it onto a relative 3d space. Once the circles are mapped into this 3d space, we can use basic kinematic equations to determine if a set of circles is feasible according to the laws of physics. If a set of circles is not feasible we plan to discard it, if it is feasible we plan to keep the sequence of circles that move the most. Finally, with this trajectory, we can solve for when the ping pong ball will hit the plane that is created by the tops of the cups and send that to the front end to be displayed.

We plan to make the cup detection algorithm much simpler by simply doing a convolution over circular objects using the L515's built in camera. This will allow us to easily detect the cups as they will always be ellipses from the point of view of the L515. Once we have detected the cups, finding their exact location is trivial as all we have to do is map them into our 3d space, similar to how the circles are mapped into the 3d space in the previous algorithm.

6.3 Launching System

After calculating all the necessary parameters for the launching system, we can see that it isn't very complex in the figure below. The launching system is designed to fill up a pressure chamber with the needed amount of pressure using the ROB-10398 air pump. Then any excess pressure is relieved using PID control from the RPi5 that is attached to the USS2-00012 proportional valve. Finally the ping pong ball is launched by actuating the kl04000 solenoid valve which releases all the built up pressure, launching the ping pong ball through the mea230928ee000323 PVC pipe.

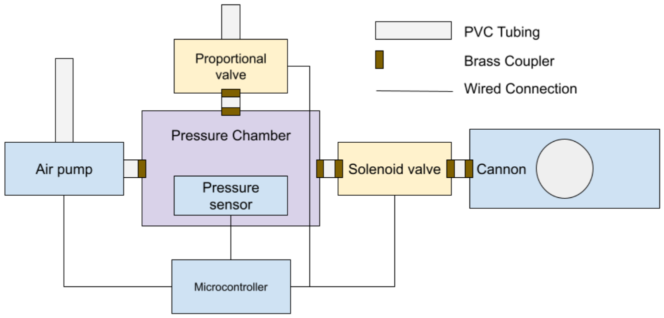

Figure 8: Launching System

6.4 User Interface

For the front end of the user interface, we are using react to make an intuitive interface that lets the user easily set the setting of the robot and fire. The final product of UI intends to look similar to that of figure 4. This front end built on react will communicate with a backend on the Raspberry Pi. Using express with Node.js, the Raspberry Pi will receive data from the front end with a line of connection via Duck DNS, which is a free dynamic DNS service that allows our devices to communicate over the internet despite changing IPs by making a static domain name they can communicate through. When the user makes changes on the bots settings, a post request will be send to the RPi server. In return the RPi confirms that the information was received with an HTTP 200 (OK). If the user presses fire, a different post request will be sent to the robot, and if it is able to fire it will send an HTTP 200, or if their is an issue it will send a 409 (conflict). After the ball is fired, the RPi will process where the ball landed and if it entered a cup, and the client will send a GET request in order to see the status of the game, allowing the UI to be updated with the current game state. This system of communication between the Client and RPi can be seen in Figure 8.

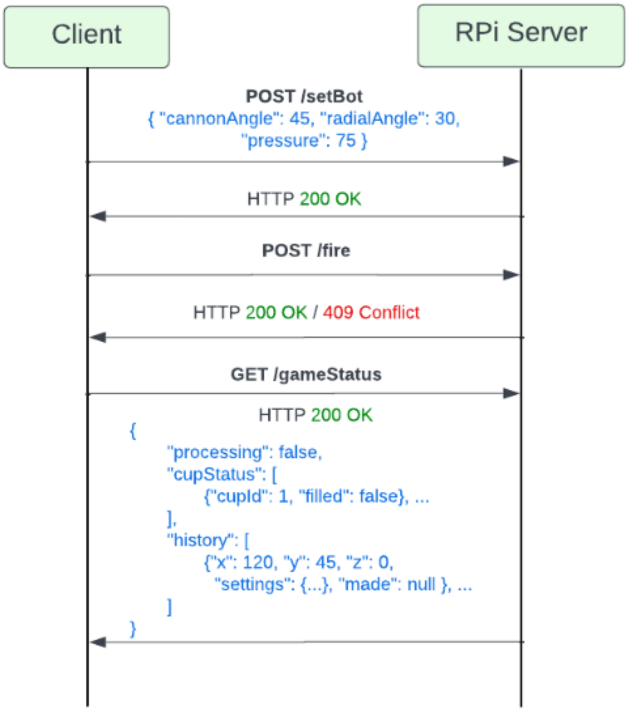

Figure 9: Communication between client and RPi server

7 TEST & VALIDATION

7.1 Tests for Repeatability

For repeatability we are aiming to minimize the variation in distance between separate shots with the settings of the robot being held constant. To test this, we will fire 10 shots into sand from both 1.5m and 3m away, and we will record the locations of the different shots. In order to meet use-case requirements, we want all of the shots for each test to land in a box that is 5cm by 2cm. The small amount of variability is acceptable as we don't want shots to be completely reproducible, but somewhat similar to help users stay engaged with an experience that isn't set in stone.

7.2 Tests for Accuracy

For accuracy, we want our CV to be consistent in detecting the cups. To test this, we will place the robot and the cups in a known position, and measure the cup detection's deviation from ground truth. This test will be repeated three times, with outdoor lighting, indoor lighting, and indoor flashing lights. In order to meet use-case requirements, we want the average deviation from the ground truth to be less than 5cm. The integrity of the feedback system lies in the system's ability to reliably track the game. In order for users to gain proper insight from previous shots, it is essential we reach this goal.

7.3 Tests for Responsive

For responsiveness, we want a real-time feedback loop for the user by being able to detect and process the balls location as quickly as possible. To test this, we will preform 10 shots, and measure the processing time to detect the ball's trajectory and the landing spot for each shot. In order to meet use-case requirements, we want the average processing time to be less than 5 seconds. If the system is responsive, it is more intuitive and engaging for users, so it is important that we make our system at least this responsive.

7.4 Tests for Accessibility

For accessibility, we want our website to have an intuitive UI for user to control the robot. We want simple controls that are both responsive and require minimal guidance to use. To test this, we will have a randomly selected individual control the robot using the UI, and see how much guidance they need to use. In order to meet use case requirements, we want 90% of test subjects to play the whole game without the need for guidance. If we can confirm that the system is user friendly, it would mean people who can't play water pong for any given reason will have an easy time joining by using our system, which is our ultimate goal for accessibility.

8 PROJECT MANAGEMENT

8.1 Schedule

Our schedule is set up to minimize the amount of time different members of the group need to work together. We have a couple milestones:

- March 15th - Finish the interface subsystem
- March 21st - Finish the launching subsystem
- March 23rd - Finish the aiming subsystem
- March 29th - Finish the computer vision subsystem
- April 1st - Somewhat integrated system
- April 15th - Fully integrated and working system

The schedule is shown in Fig. 10.

8.2 Team Member Responsibilities

Seung Yun is in charge of the aiming subsystem (both yaw and pitch). Alex is in charge of the launching subsystem and the computer vision subsystem. Mike is in charge of the human interfacing subsystem. Mike, Seung Yun and Alex are all in charge of integration once their subsystems are done.

8.3 Bill of Materials and Budget

The bill of materials is shown in Table 1.

Table 1: Bill of materials

Description	Model #	Manufacturer	Quantity	Cost @	Total
Stepper Motor	PG35S-D48-HHC2	NMB Technologies Corporation	1	$37.34	$37.34
Servo Motor	CN0193	SunFounder	1	$15.56	$15.56
Stepper Motor Driver	DRV8825	Texas Instruments	1	$7.99	$7.99
Driver Expansion Board	Jeanokozqgaw394kh	Jeanoko	1	$7.49	$7.49
2ft * 1ft 3mm plywood	N/A	N/A	5	$5.10	$25.50
Tubing Splitter	aww-12310009	ANPTGHT	3	$3.98	$11.94
Pressure Sensor	LEPAZA60117	Walfront	1	$15.36	$15.36
Solenoid Valve	kl04000	beduan	1	$9.99	$9.99
Proportional Valve	USS2-00012	U.S. Solid	1	$34.90	$34.90
Proportional Valve 2	HSH-Flo	HSH-Flo	1	$51.99	$51.99
Brass Coupler	HF39184805	LTWFITTING	2	$14.99	$29.98
Brass Coupler 2	HF39184205	LTWFITTING	1	$7.99	$7.99
Female Coupler	3300*A	Legines	1	$5.99	$5.99
PVC Pipe	mea230928ee000323	MECCANIXITY	1	$12.99	$12.99
Acrylic Pipe	a20102800ux0220	uxcell	1	$14.49	$14.49
Pipe Sealing Tape	B095YCMHNX	DOPKUSS	1	$3.99	$3.99
Tubing	B07PY4KM8C	EZ-FLO	1	$3.98	$3.98
Air Pump	ROB-10398	SparkFun Electronics	2	$18.44	$36.88
Raspberry Pi 5	SC1112	Raspberry Pi	1	$0	$0
LIDAR Camera	L515	Intel	1	$0	$0
					$272.56

8.4 Risk Mitigation Plans

Our primary focus for our risk mitigation plans lies in ensuring the repeatability of both our aiming and launching subsystems. Recognizing that yaw accuracy is pivotal, we've identified potential challenges in this domain. In the event that thare are discrepancies between our requirements and the robot's capabilities we have created a thorough risk mitigation plan.

If our yaw accuracy is found to be lacking, we plan to evaluate the lazy susan mechanism's load-bearing capacity by conducting tests with varying weight configurations. Should the weight distribution prove to be a factor, we plan to introduce additional load-bearing wheels or rollers. Alternatively, if the culprit lies in the stepper motor's fidelity, we are prepared to upgrade to a more precise stepper motor. If the yaw issue is due to the stepper motor skipping steps, we will arrange to use a servo motor instead as they have inbuilt encoders that prevent step skipping. Parallelly, we are concerned about pitch accuracy, where potential remedies include investing in a more accurate stepper motor or, if the issue stems from the firing tube's weight, considering counterweight additions or opting for a higher-torque motor. While significant time and effort have been dedicated to designing and researching the construction of our launching subsystem, the inherent complexity poses a risk of failure due to our relative lack of experience. We also have a risk mitigation plan in the event that our launching accuracy requirements are not met. This involves isolating different components and assessing their performance. For instance, should concerns arise about the pressure chamber expanding beyond a certain threshold, we will systematically investigate pressure changes during controlled increases. If power deficiencies impede the launch force, potential solutions include acquiring a larger motor, pressure chamber, or extending the barrel length to facilitate the transfer of additional power to the ping pong ball. Through this comprehensive approach, we are committed to addressing challenges and ensuring the optimal performance of our water pong robot.

9 RELATED WORK

While our project is one-of-a-kind in its precise functionality, it's worth noting that there exist several projects with similar undertakings. As mentioned in the introduction, there is one case of someone building an automated pong ball launcher [1], but their main goal was to select a cup and then shoot the ball into the cup, without any human skill component. A notable precedent is the work of the ECE Capstone team, Team E4, in Spring 2021. Their creation, the Automatic Gentleman [4], featured an automated pong ball launcher with targeting capabilities; however, it differed significantly from our approach. Unlike our emphasis on accessibility and our utilization of an air pressure-based launching system, their focus and mechanism deviated from these aspects. During our design phase, we drew inspiration and insights from Team E4's work, particularly their final report and weekly status reports. The decision to employ an air pressure-based launching system in our design was a deliberate response to the observed inaccuracies in their launching mechanism. This strategic departure ensures that our project not only stands out in its unique features but also addresses ethical and societal issues instead of simply being an interesting technical challenge.

10 SUMMARY

Pongpal presents a innovative solution to the accessibility challenges faced by water pong enthusiasts around the world. By introducing a remote-controlled water pong robot accessible through a user-friendly website, the barriers of distance, disabilities, and health concerns are effectively addressed. We are in the process of building a compressed air cannon to launch the ping pong balls, utilizing a propotional valve to control the pressure inside the pressure chamber. Our aiming subsystem consists of a stepper motor for radial aiming and a servo for pitch aiming. The computer vision subsystem utilizes an intel realsense L515 LIDAR and we have had great progress in ball tracking and are now moving to cup detection. The interface subsystem is currently being developed for an optimal user web experience. While it is possible that we experience issues in actually building the mechanical hardware to control the robot, we have meticulously created our risk mitigation plan and have thoroughly researched what we needed to do during the design phase. With this project, we hope the joy of playing water pong can be extended to a broader audience, fostering inclusivity and shared enjoyment regardless of physical limitations or geographical constraints. Pongpal stands as a testament to the power of technology in overcoming barriers and bringing people together in the spirit of play.

References

[1] Niklas Bommersbach. *I built a robot that DOMINATES at Beer Pong*. 2023. URL: `https://www.youtube.com/watch?v=Xm2Y8mjOVtc&ab_channel=NiklasBommersbach`.

[2] ClouDNS. *Getting Started with Dynamic DNS*. 2023. URL: `https://www.cloudns.net/wiki/article/364/` (visited on 02/27/2024).

[3] *Intel® RealSense™ LiDAR Camera L515*. Intel® RealSense™ Depth and Tracking Cameras. URL: `https://www.intelrealsense.com/lidar-camera-l515/`.

[4] Nelson J., Kojiro L., and Botello J. "Team E4: Automatic Gentleman". In: (May 2021). URL: `https://course.ece.cmu.edu/~ece500/projects/s21-teame4/` (visited on 03/01/2024).

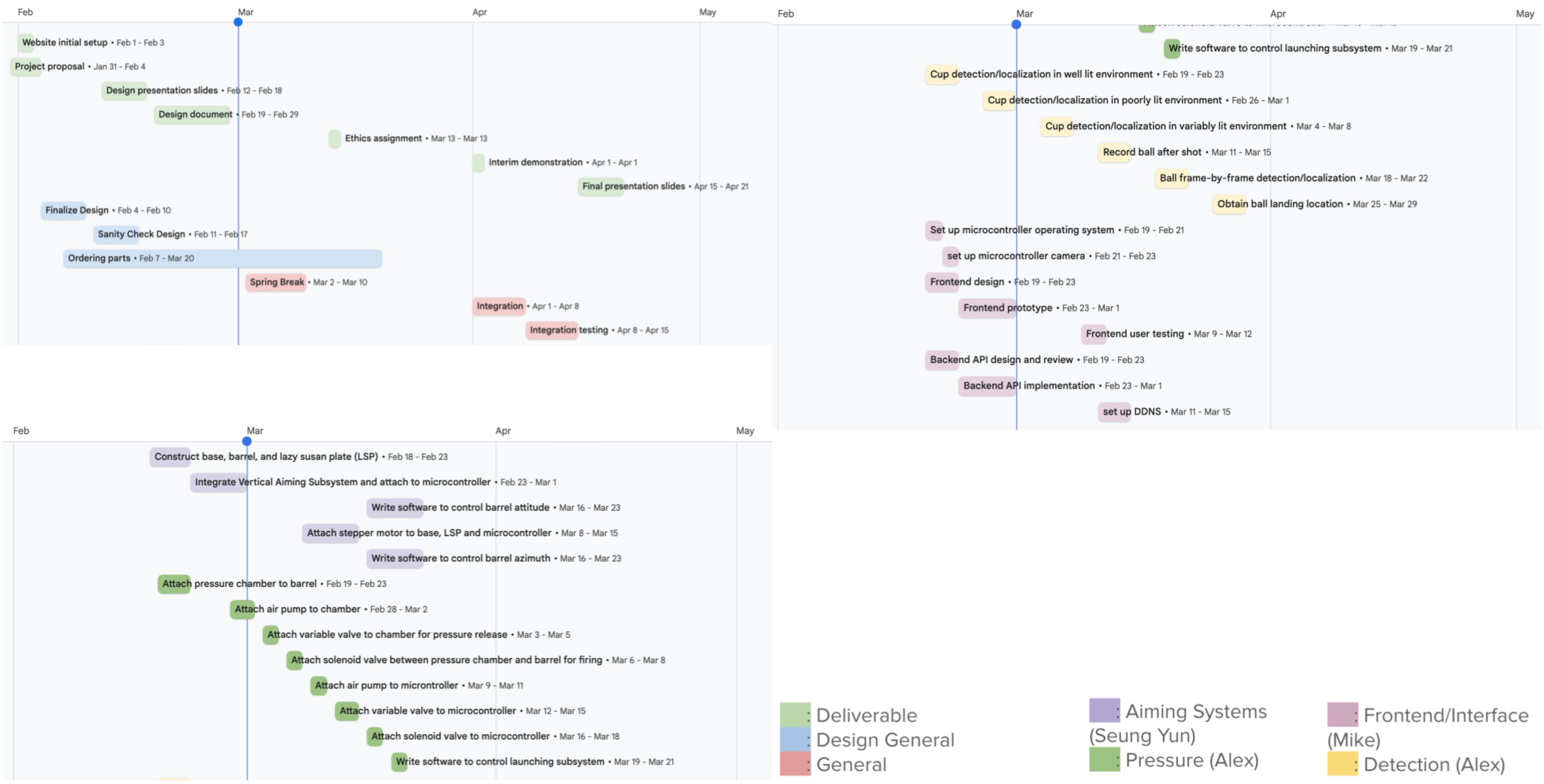

Figure 10: Gantt Chart