MA4012: Mechatronics Engineering Design

Semester 1 AY 2022/23

Final Report

Group 2

Laboratory: Mechatronics Lab

Location: N3.2-B1-03

School of Mechanical and Aerospace Engineering [MAE]

Group Members

1. CHUA JUN YANG CHALMERS (U2020149K)
2. KHOO TIONG HEE (U1920430J)
3. OOI JIA MIN (U1920404A)
4. TANG CHI ONN (U1920463J)
5. TEOH CHEE SERN (U1920324L)
6. WONG SHANG YI (U1920523C)
7. WU GUAN JIE (U1920993K)

Abstract

Automation has grown in relevance to the operations of organizations across many different industrial areas with the introduction of Industry 4.0. Robotics has thus established itself as a crucial field that may assist businesses in automating processes, streamlining operations, cutting costs, and facilitating disruptions in the market. Emerging technologies, including self-driving robots, are already widely used and have completely changed the industry structure.

As future robotics and mechatronics engineers, it is critical that we have the skills necessary to meet industry demands and add value to organizations. Thus, our team created an autonomous tennis ball recovery robot over the course of 13 weeks. MYVI (our robot's name) won the first prize in the competition.

This report outlines the methodical approach the team used to first distinguish the design specifications from the project requirements. A conceptual and embodiment design that most closely resembled the design requirements was produced. The detailed design part of the process then described how the conceptual design was carried out iteratively to create the finished robot. The programming design is then described in detail followed by a summary of the control strategy the team used to carry out the required job of ball recovery in the competitive context. Finally, the team discusses some key lessons learned from the project.

Content

Table of Figures

List of Tables

1 Introduction

This chapter begins by describing the background information and objective for the project as well as the design process that our group went through. It then goes over our team's organization and timeline, as well as a list of provided components.

1.1 Background Information

The goal of this project was to create a small autonomous vehicle that could perform the following functions in descending order of importance.

- Search and Retrieval
- Edge Detection

Robots from two teams competed against one another in the track arena shown in Figure 1 to collect three tennis balls within three minutes. The BO3 (Best of 3) format was used for the competition. Three balls would be placed at random in the arena. To warn the robot not to cross the boundary, the arena's edges are marked with yellow reflecting taps. The delivery area has a boundary wall that the ball must be lifted over in order to be delivered. The robot will be disqualified if it leaves the arena.

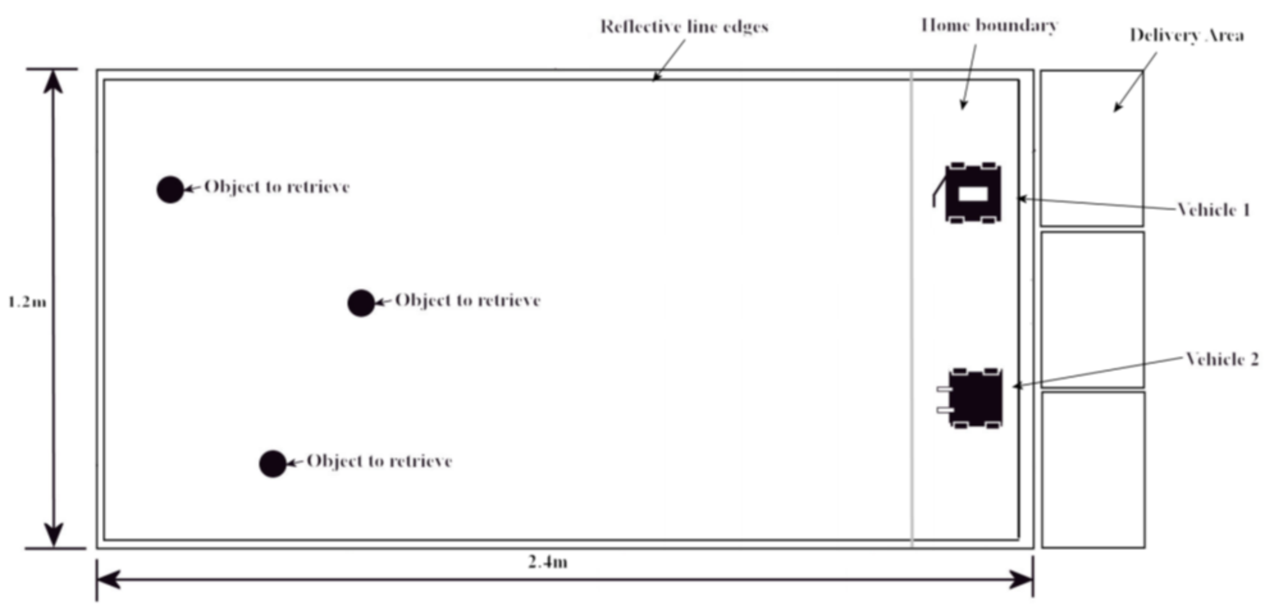


Figure 1. Competition Stage

Materials from VEX Robotics were provided – which includes the VEX ARM cortex-based microcontroller, motor modules, steel structures and a hardware kit. Furthermore, line sensors, Sharp distance sensors, limit switches, an Arduino Mega microcontroller and a digital compass were given. These components as shown in Figure 2 are described in more detail in [Chapter 1.5 Components Provided].

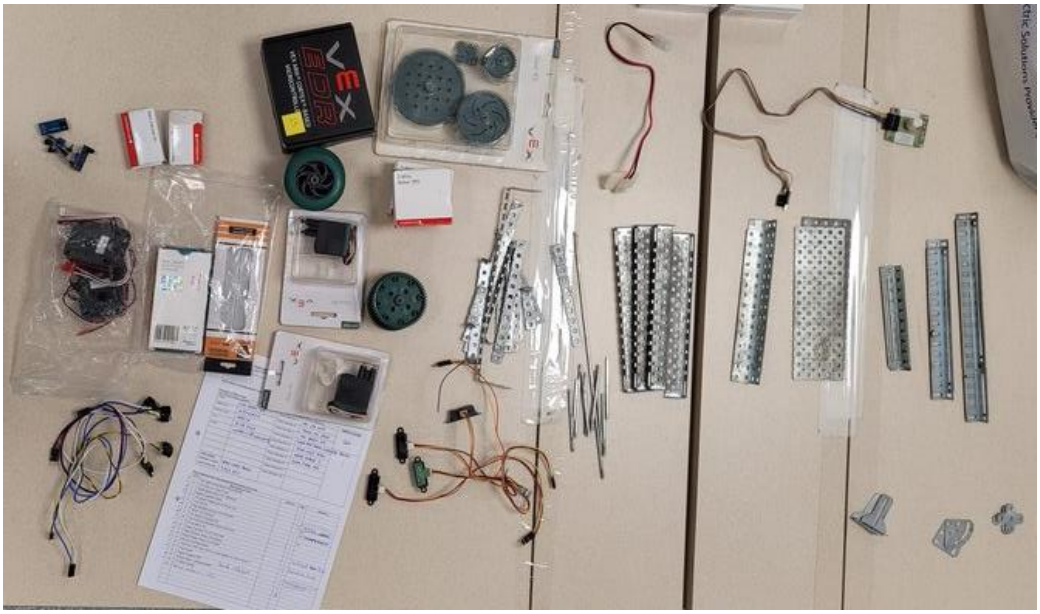


Figure 2. Components Provided

1.2 Objective

The goal of this project is to train students on applying design thinking in a systematic manner to the development of autonomous vehicles.

1.3 Design Process

Our group used a systematic approach to develop our robot in accordance with what we have learned in class and with the usage of the Miro tool. Several steps make up the design process, as seen in Figure 3.

This strategy was developed to give each team member a chance to share their thoughts and participate in each design job. Our team was split into the hardware team and the software team to move forward with the detailed design after the design tasks were finished. The specifics of each phase are covered in the following chapters.

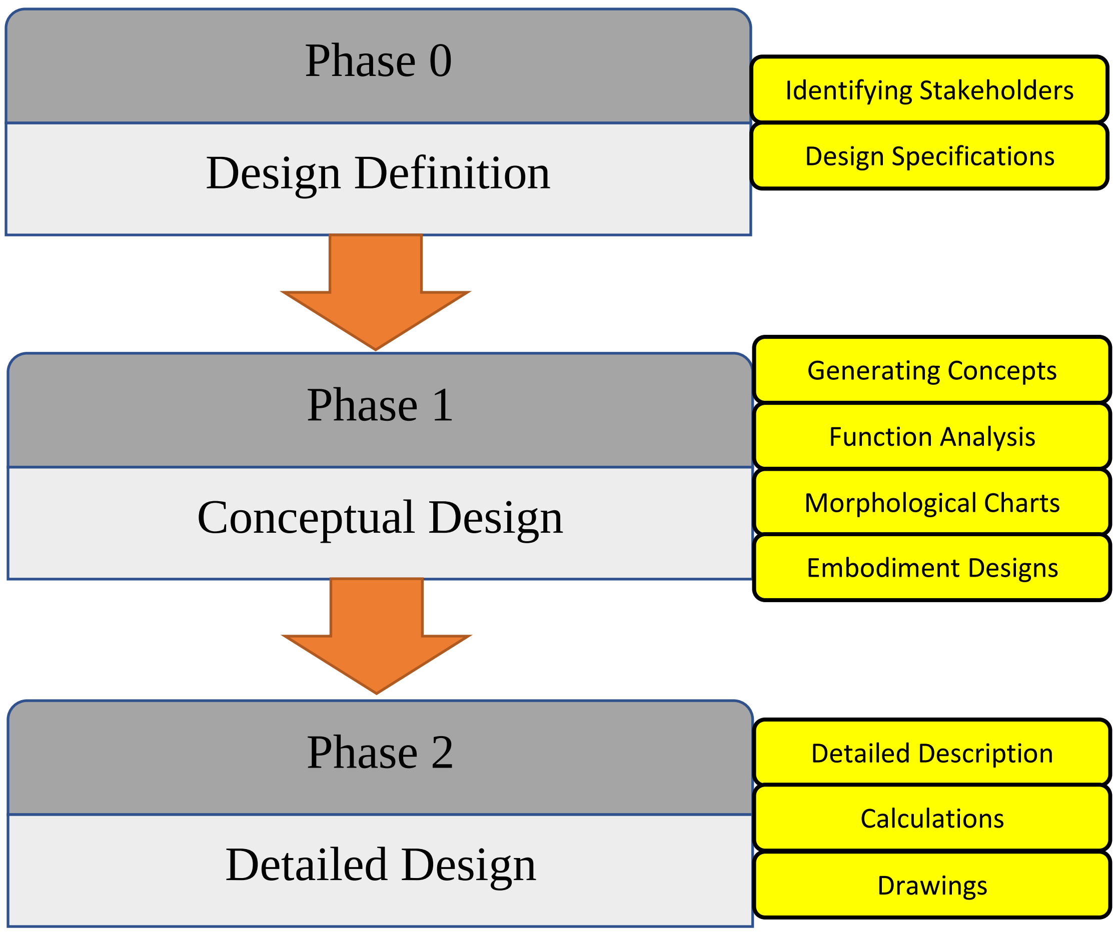


Figure 3. Design Process

1.4 Team Organization and Project Timeline

Following the completion and finalization of the conceptual designs for each of the processes, our group of seven was divided into two main sub-groups based on the various aspects of the robot, as shown in Table 1 below:

Mechanical Team	Electrical & Software Team
Structure Design	Sensors' Placement and Calibration
Collector Mechanism Design	Electrical Design
Ramp System Design	Search Algorithm Design
Drive System Design	

Table 1. Sub-group tasks

Our team members and our beloved Dr John Heng are shown in Figure 4:

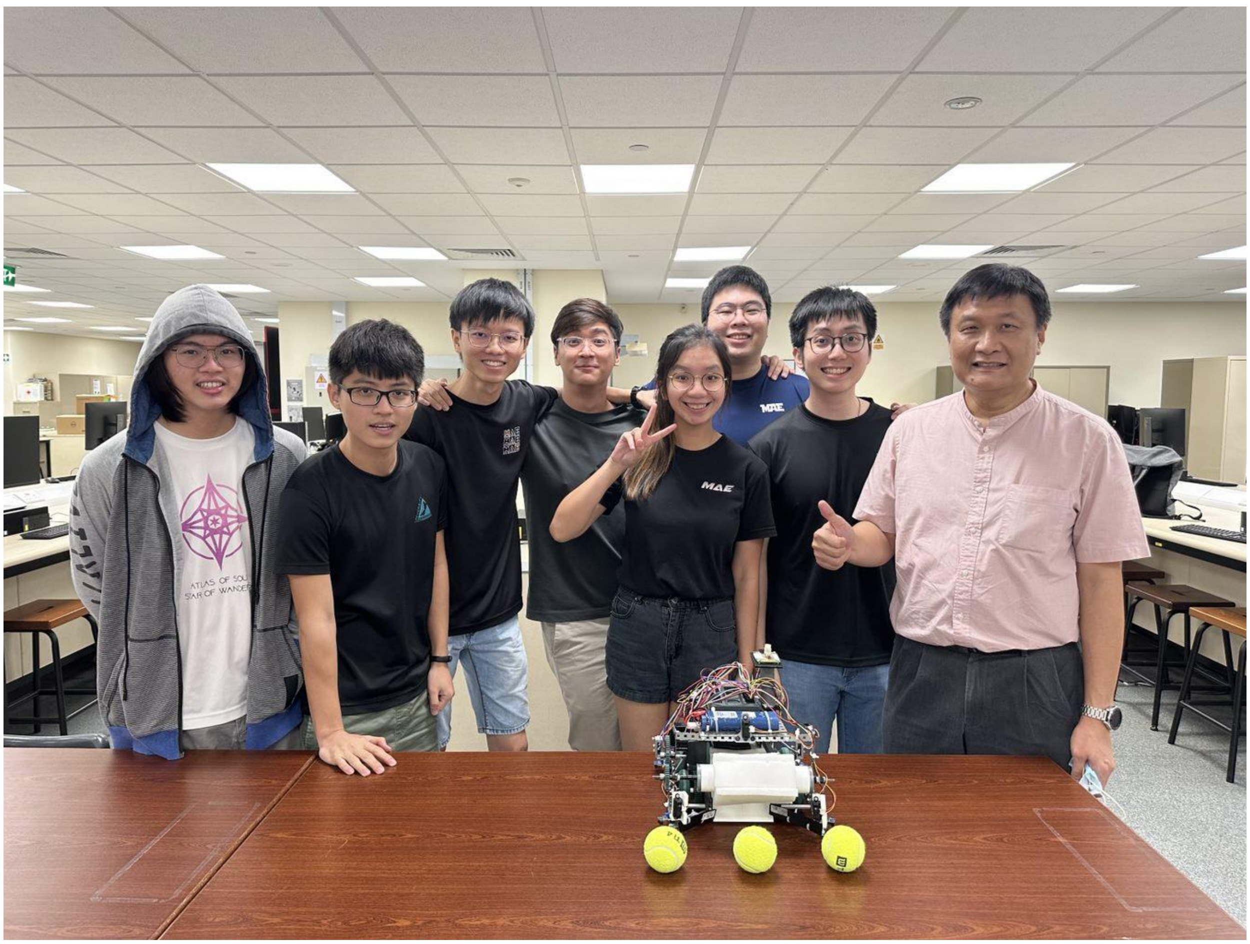


Figure 4. Group Photo

Following the selection of the robot design, both teams worked in parallel to increase productivity, but we are still collaborating actively in many aspects especially in the area of sensor placement.

Table 2 below provides a brief chronology of our project:

Week	Mechanical Team	Electrical & Software Team
2-7	Idea Generation and Conceptual Design	
7-9	• Design calculation • CAD design of ramp system, sensors mounting • Construct collector system and robot frame structure	• Testing and calibration of sensors and motors • Design of software architecture and search strategies
9-10	• Refinement of mechanical design • 3D print and assembly of the mechanical system	• Mounting of sensors • Testing and calibration of sensors
11-13	• Assisted software team to resolve hardware issue during testing and tuning • CAD assembly of whole robot	• Testing and refinement of various functions, search algorithms and strategize for edge detection • Testing and tuning of sensors and parameter • Refinement of software design

Table 2. Project Timeline

1.5 Components Provided

Component	No.	Description
VEX ARM Cortex-based Microcontroller	1	A microcontroller with a STMicroelectronics ARM Cortex-M3 user processor. It coordinates the flow of all information and power on the robot. All other electronic components (motors, sensors, etc.) interface with the microcontroller
Arduino Mega Board	1	A microcontroller board based on the ATmega2560
VEX Metal and Hardware Kit	1	Contains structural elements to build our robot, such as metal plates, bars, shafts, gussets, screws and nuts, etc.
VEX Gear Kit	1	Contains gears of multiple sizes which are used for power transmission from the motors
High-strength Gear Kit	1	Contains gears of multiple sizes which can withstand higher forces than the ones present in the VEX Gear Kit
Low-friction Wheel	4	Wheels of 3.5cm radius, made of rubber.
VEX Continuous Rotation Motor (276-2163)	2	A motor which rotates continuously based on the duty cycle sent through the Pulse Width Modulation (PWM) signals sent from the VEX microcontroller.
2-Wire Motor 393	2	Similar to the VEX Continuous Rotation Motor but must be connected to a motor controller to be driven by PWM signals.
Motor Controller 29	2	Uses a standard PWM signal to drive the 2-Wire Motors
Sharp Distance Sensor (10 to 80 cm)	3	Sharp distance sensor is a great way to add obstacle avoidance or motion sensing to robot or any other project. With a detection range of 10 cm to 80 cm.
Sharp Distance Sensor (4 to 30 cm)	1	Sharp distance sensor features a detection range of 4 cm to 30 cm. The shorter range gives you higher resolution measurements, and the lower minimum detection distance makes this sensor great for detecting very close objects.
IR Line Tracking Modules	4	Detects the boundary based on the signal reflected from the IR emitter

Limit Switch	4	Contact sensing devices used for detecting the presence or absence of an object (depressed or not)
Digital Compass 1490	1	A sensor which can be used to detect the four cardinal directions (N, S, E, W) and the intermediate directions (NE, NW, SE, SW). It uses a solid-state hall effect device and is sensitive enough to detect the Earth's weak magnetic field
NiMH Battery Pack	1	Rechargeable battery with specifications of 7.2V and 3000mAH
Vex Shaft Encoder	2	Used to measure both relative position of and rotational distance travelled by a shaft

Table 3. Hardware Components and Description

The hardware parts that were given to our group for this project are listed in Table 3 above. These pieces served as both a starting point and a hardware constraint, as no additional motors or sensors could be used and at least 50% of the vehicle had to be made up of the parts that were given.

2 Design Definition

The definition phase of the design process is described in this chapter, along with the stakeholders who affected the design’s considerations and the design requirements that served as a roadmap for further stages.

2.1 Stakeholders

It is crucial for us to first consider the different users engaged in this project and their needs before developing our design criteria for the robot. As a result, Table 4 provides a list of the many stakeholders and the effects that each stakeholder had on the robot’s design considerations:

Stakeholder	Implications on Design Consideration
Judge	• Comply to regulations of competition • Design constraints • Determine the area where the tennis ball will be placed
Electrical Engineer	• Ease of wiring • Ease of repair/replacement
Mechanical Engineer	• Ease of assembly • Ease of repair/replacement/adjustment of parts
Robot Handler	• Ease of operation (reset)
Software Engineer	• Ease of programming • Ease of debug/diagnose and troubleshoot problems
Opponent	• Strategies to avoid collision with opponent

Table 4. Stakeholders

2.2 Design Specifications

The design requirements for our robot are shown in Table 5 below, keeping in mind the abovementioned stakeholders as well as additional factors that can affect the robot's efficiency in the competition.

Task: Tennis Ball Collection Vehicle	
Product Description	A vehicle capable of searching for tennis balls while avoiding opponents and returning the ball to the collection area without human intervention.
Benefit Description	• Able to retrieve the ball quickly and reliably • Able to avoid collision with opponent and boundaries
Product Requirements	• High maneuverability • Within size limit of $30 \times 30 \times 30cm$ • Does not carry multiple balls at the same time & able to release extra ball(s) • Able to lift the ball above the height of the ball collection area wall (10cm) • No extra actuators/active sensors • Able to resist collision • High speed of ball picking – does not require stopping • High allowable margin of error while picking up balls • Able to distinguish ball and opponent and ball collection area quickly • Able to avoid moving out of field • Require low complexity of programming

Table 5. Design Specification

Some product requirements and benefit description are divided into groups below based on various tasks for the robot.

2.2.1 Ball Retrieval Mechanism

- The capture area should be large enough to allow the tennis ball to pass through.
- The collection area must be wide enough to capture and store the ball.
- It is critical that the collection mechanism retrieves only one ball at a time. Contingency plans must be in place to ensure that the second ball is dropped if it is picked up. This can be accomplished by designing the storage area so that only one ball can be stored in it. Also, the ball collecting mechanism will only start operating after the ball collected is being delivered to the designated area.
- The sensors should be installed and calibrated to accurately find tennis balls.
- The vehicle ought to be capable of detecting the presence of a tennis ball in the storage space.
- The tennis ball that the vehicle has just retrieved needs to be elevated high enough to be deposited into the delivery location.
- The vehicle should orient itself to proceed to the delivery location as soon as it detects the existence of tennis balls inside the storage area.
- Ball delivery should be swift and reliable once the vehicle arrives at the delivery location.

2.2.2 Navigation Functions

- The vehicle must be capable of taking consistently low-curvature spins while moving in a straight line.
- The vehicle should advance to pick up the ball after the sensors have located it.
- The vehicle should turn around or advance to search its surroundings for the ball if sensors fail to detect anything.
- To prevent the vehicle from leaving the arena, line detection sensors should be installed at its perimeter.
- The vehicle should utilize the compass to find its way back to the delivery place when navigating the arena.

To enhance the vehicle's performance, these two fundamental functions should be refined while taking other factors into account.

2.2.3 Other Considerations

- To perform maintenance quickly and effectively, the vehicle should be modular.
- Cable management is crucial to avoid any cable as it could potentially block the signals of the sensors, entangled with moving components (gears, motors, connecting rod).
- The vehicle should be built to withstand collisions and forces from another moving vehicle.
- To make it simple to add new features and conduct debugging, software architecture should be modular.

3 Conceptual Design

The conceptual step of the design process is described in this chapter. During lab sessions, we used the Miro application to develop ideas and do functional analysis in pairs. We then created a morphological chart to compare the many concepts generated and then assessed them to produce a final concept.

3.1 Idea Generation

We used intuitive techniques to come up with concepts. We divided ourselves into pairs and brainstormed ideas. Some of us looked into past designs for inspiration while others researched in websites like YouTube, Google, and other robotic competition websites. We gathered and created original concepts and research. While deciding on the necessary functions, which will be covered in the following sections, we kept track of all the intriguing and viable options. After the pair-level discussions were completed, we gathered to discuss voted on the best designs to incorporate into our robot.

3.2 Functional Analysis

Functional analysis is a technique in determining the crucial components of a proposed design. By breaking down and simplifying design difficulties, functional analysis enabled us to comprehend how we envisioned the system to operate. The design is then broken down into smaller functions: key sub-functions and auxiliary functions that assist and enable the sub-functions to perform their intended functions. To ensure clarity amongst the designers, sub-functions are described using verb-noun pairs. By inquiring about the purpose of the product before deciding on how to design it, functional analysis compels designers to step back and look at the big picture from another angle. This approach promotes creative analysis and possibly a breakthrough to conventional problems or current solutions.

Our team conducted a functional analysis for this design project, which is shown below in Figures 5 and 6. In this study, the sub-functions were connected via the flow of energy, materials, and signals. Each sub-function is listed in Table 6 along with its related description.

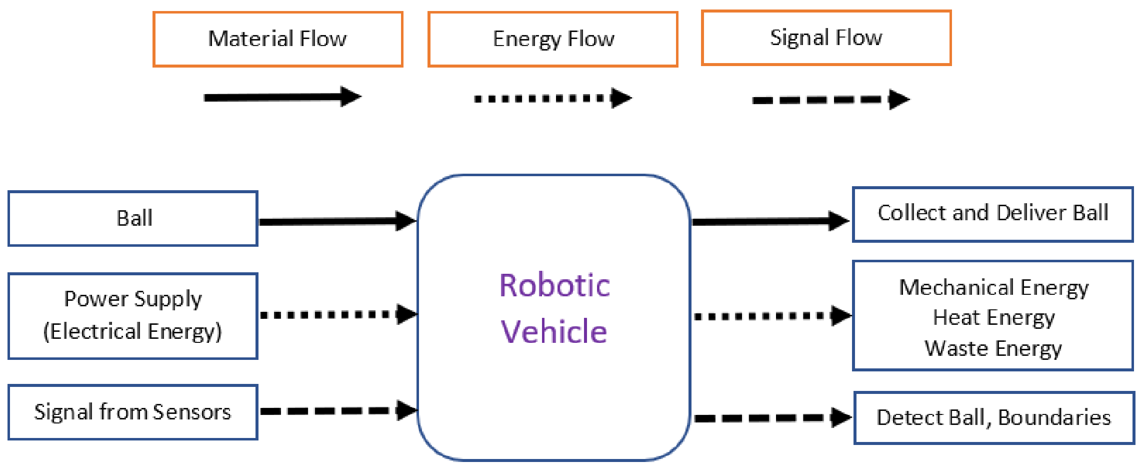


Figure 5. Overall Functional Analysis Diagram

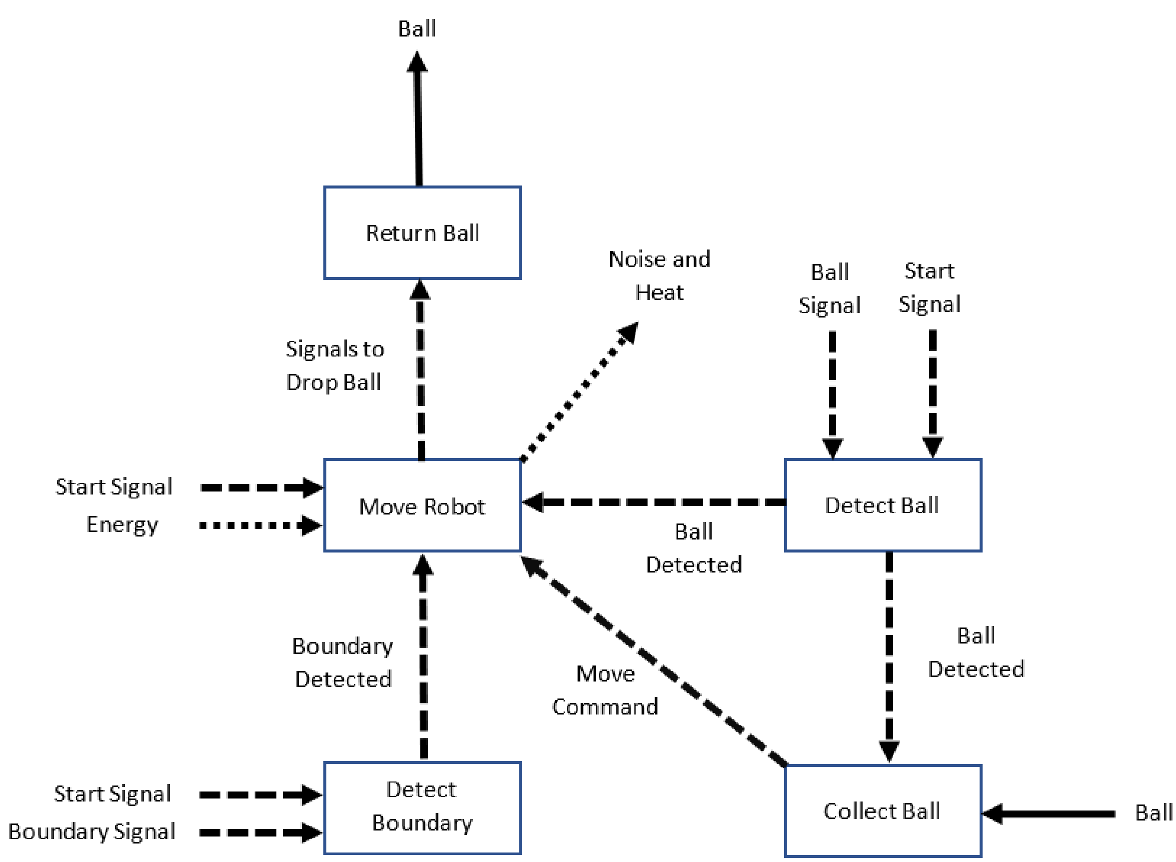


Figure 6. Functional Analysis Diagram

Sub-function	Explanation
Move Robot	The robot should begin moving independently to look for the ball as soon as the start signal is sent. When a ball is found, the robot should move in a straight line towards the detected direction. This sub-function also compels the robot to return to the starting line upon retrieving the ball.
Detect Object	The ball or obstacles (opponent’s robot /ball return area) could be detected. For this purpose, the robot should be designed in such a way that it can distinguish between the ball, the opponents, and the ball return area wall.
Detect Boundary	When the boundary of the arena is detected, the robot should move away from it to avoid going beyond the edges of the arena. Because the ball's position is within the arena, the robot's design should always ensure that it is constantly within the boundary.
Retrieve Ball	When the ball is found, the retrieval mechanism is activated to collect the ball. The robot then reverses back to the delivery area, releasing the ball collected.
Return Ball	The ball should be returned to the ball delivery area once the robot has reached the ball collection area.

Table 6. Sub-functions

Before developing the robot, various considerations were made using the functional analysis diagram.

Hardware Consideration	
Power	To achieve the goal, three processes must be followed. Moving the robot, retrieving the ball, and returning the ball are the three tasks. These functions require electrical energy from the battery in order to be carried out. Hence, the power capacity of the battery should be capable of supplying enough power for the robot to perform all functions repeatedly.
Speed	The robot is placed in a competitive environment. High movement speed and efficient ball retrieval are paramount for success. The robot must complete the task in the shortest amount of time possible.

Table 7. Hardware Consideration

Software Consideration	
Sensor	The use of sensors, sensor placement, and sensor accuracy are all important factors in determining the success finding and delivering the ball while staying within the arena.
Code	The robot must be able to function without additional human assistance. Therefore, the program must ensure that the robot can function independently at each stage of the code. When the sensor captures data, the robot must act quickly and with minimal error.

Table 8. Software Consideration

With these factors in mind, the team was prepared to look for operating principles to fulfil the sub-functions.

3.3 Morphological Chart

The morphological charts divide the potential designs into many functional segments. Following the segmentation of the designs, we choose the combinations within each segment to create a finished design for the robot. We have two final conceptual designs of the robot to choose from at the conclusion of the design. The structure, move robot, detect obstacles, retrieve ball, and return ball are the key sub-function.

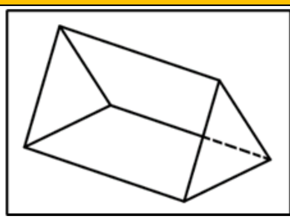
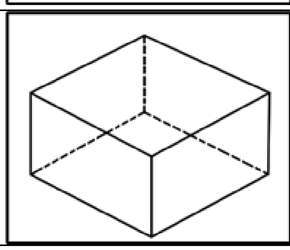
Structure		
1.		<u>Prism shaped</u> **Pros:** Relatively more stable, where components are closer to the center of mass. **Cons:** Spaces for hardware components and sensors are limited
2.		<u>Cube shaped</u> **Pros:** Solid, sturdy and more spaces for components and sensors **Cons:** Huge and bulky

Table 9. Structure

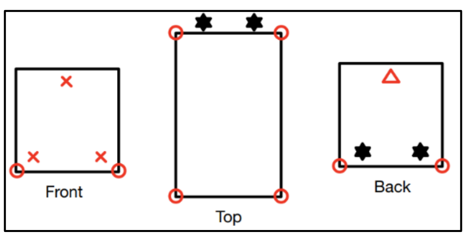
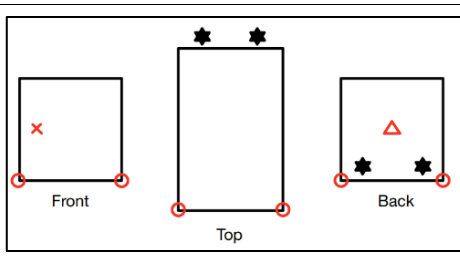
Detect Obstacles		
1.		<u>Symmetrical Sensors</u> 1 long distance sensor at the top front, 2 long distance sensors at the bottom front, 4 line sensors at the each corner of the vehicle, 1 short distance sensor at the rear top, 2 limit switch at the rear bottom
2.		<u>Skewed Sensors</u> 1 long distance sensor at middle front, 2 line sensors at the each front corner of the vehicle, 1 short distance sensor at the rear middle, 2 limit switch at the rear bottom

Table 10. Detect Obstacles

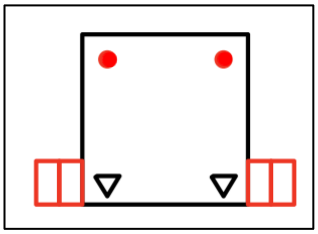
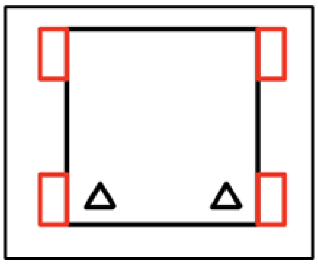
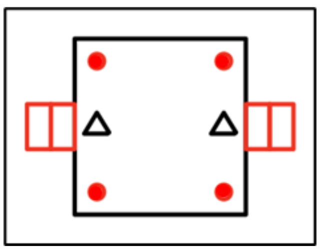
Move Robot		
1.		2 dual wheels at the rear of the vehicle, 2 motor at the rear each driving a dual wheel, 2 castor wheels in front **Pros:** High stability at the rear of the vehicle **Cons:** Hard to control the dual wheel, center of gravity of the vehicle must be placed at the rear
2.		4 wheels at each corner of the vehicle, 2 motors at the rear **Pros:** Good resistant to collision from opponent **Cons:** The robot is less agile and less maneuverable, Omni wheel has been considered but not used due to low friction which leads to low gripping capability.
3.		2 dual wheels at the middle of the vehicle, 2 motor in the middle each driving a dual wheel, 4 castor wheels at 4 corners of the vehicle **Pros:** Agile and the most maneuverable **Cons:** Not resistant to collision from opponent as turning torque in a collision cannot be countered effectively.

Table 11. Move Robot

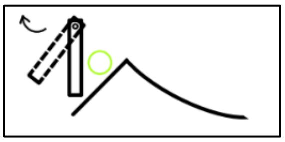
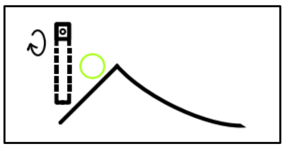
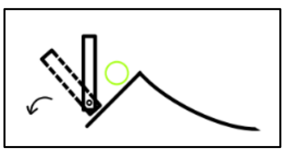
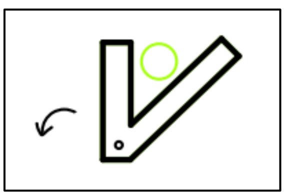
Return Ball		
1.		<u>Open gate up mechanism</u> **Pros:** Ball can fall automatically due to gravitational force **Cons:** Motor is placed at the highest level as compared to other design, making the robot least stable, the gate might block the ball from falling out if not completely open
2.		<u>Open gate left/right mechanism</u> **Pros:** Ball can fall automatically due to gravitational effect **Cons:** The motor is placed relatively high, the vehicle is relatively less stable, the gate might block the ball from falling out if not completely open
3.		<u>Open gate down mechanism</u> **Pros:** Easy mechanism, the motor is placed at the lowest position which lower the centre of gravity of the robot **Cons:** Gate might block the ball from falling out if not completely open
4.		<u>Cup shaped mechanism</u> **Pros:** Able to secure the balls with the cup shaped design during delivery Cons: Ball might not be able to deliver to correct position as the ball will rotate with the motor

Table 12. Return Ball

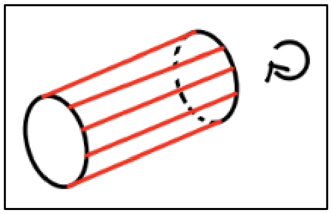
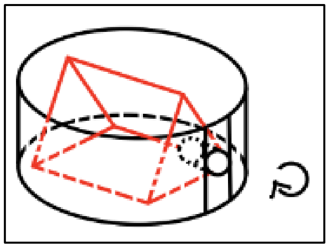
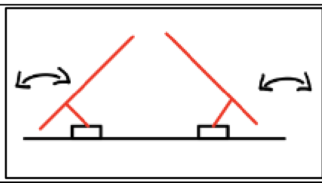
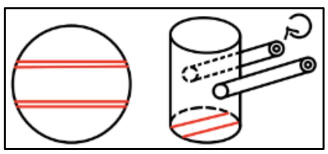
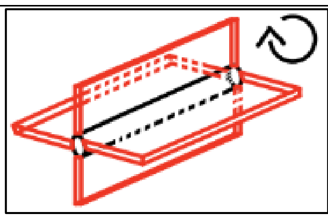
Retrieve Ball		
1.		<u>Roller collector</u> **Pros:** Ease of collecting, unlikely to push ball away, high allowance for ball detection error **Cons:** Elastic material (rubber band) might snap due to fatigue of material or during collision with opponent
2.		<u>Barrier collector</u> The barrier is lowered around the ball with another mechanism inside to turn and scoop the ball **Pros:** Stable design that ensure opponents unable to steal the ball away **Cons:** Bulky design, need a strong motor to lift the barrier
3.		<u>Wiper collector</u> Wiper will force the ball in the middle **Pros:** Can secure the ball tightly **Cons:** If one motor fails, whole mechanism will fail
4.		<u>Cylinder collector</u> A cylinder to capture the ball and squeeze the ball through the rubber band at the bottom **Pros:** Easy to manufacture, decreased complexity (combination of collection and dispensing mechanism.) **Cons:** Elastic material (rubber band) might snap, low margin of error in ball detection
5.		<u>Lawnmower collector</u> Similar to lawnmower mechanism **Pros:** high allowance for ball detection error **Cons:** Blades might be damaged when ball stucked

Table 13. Retrieve Robot

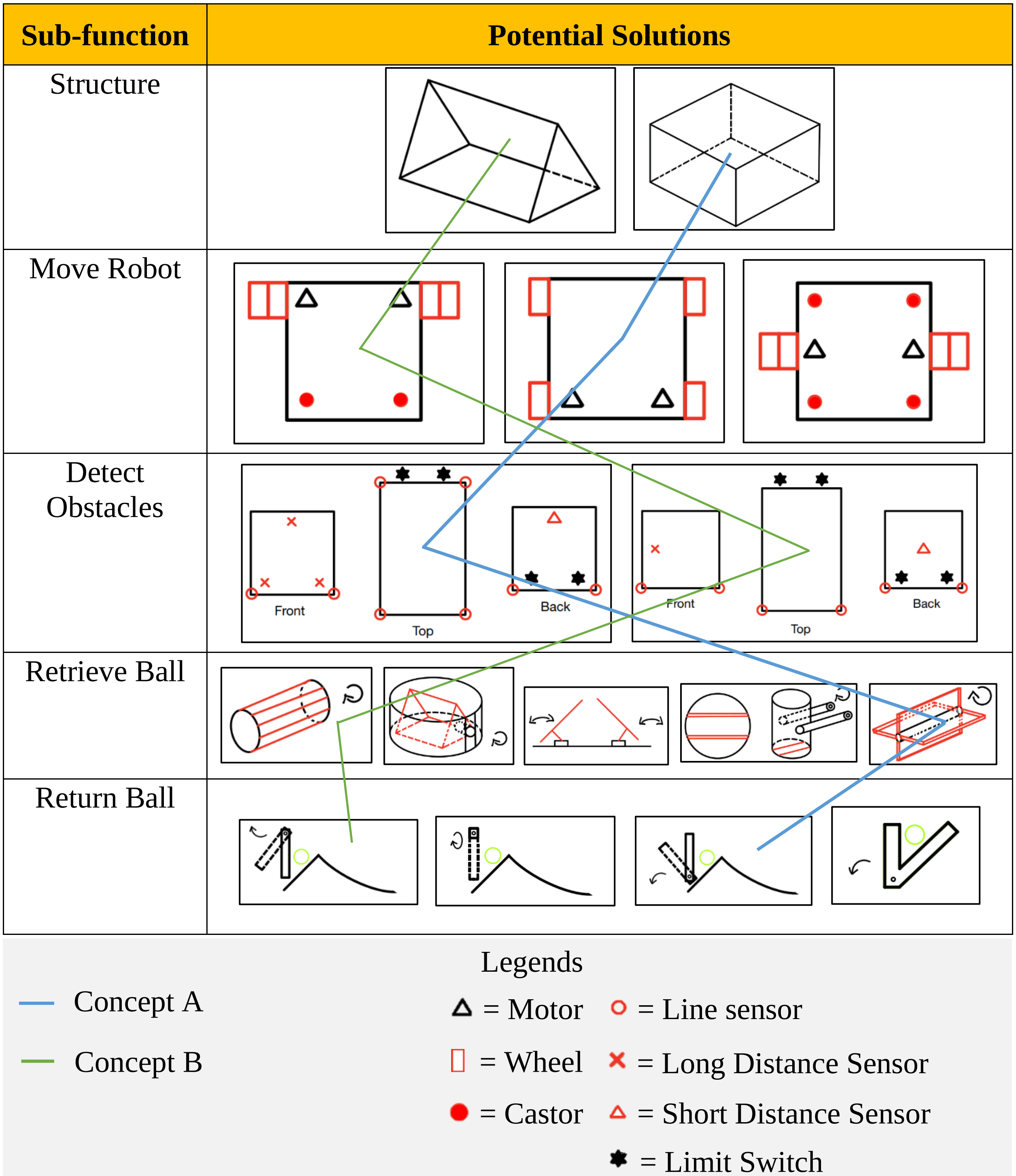


Figure 7. Morphological Chart

3.3.1 Concept A

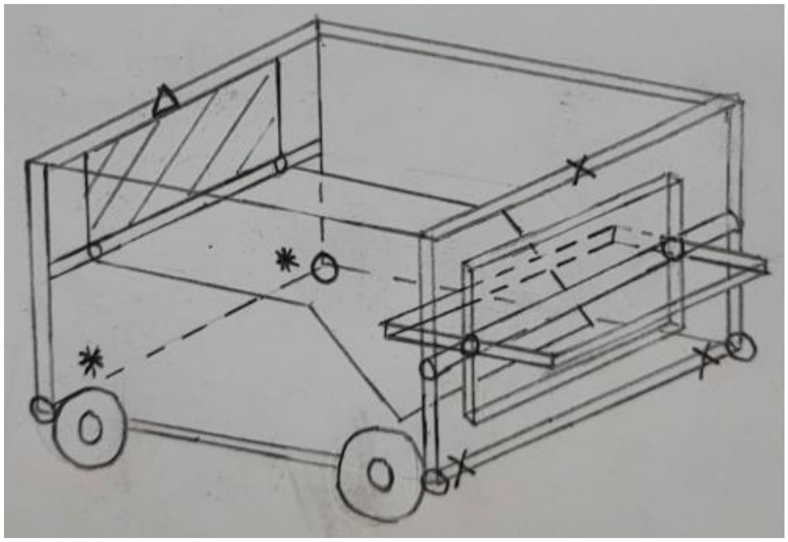


Figure 8. Concept A Drawing

Sub Function	Solution	Concept Description
Structure	Cube shaped	High stability and high volume for components and sensors.
Move Robot	4 wheels at each corner of the vehicle, 2 motors at the rear	Front and rear wheels of each side are connected via gears. Two motors each drive two wheels on one side. This design synchronizes the rotation of the wheels.
Detect Obstacle	Symmetrical Sensors	Four line sensors at the vehicle's 4 bottom corners are utilized for edge detection, while three long distance sensors up front are employed to detect the ball. One short distance sensor and two limit switches are employed at the top and bottom of the back, respectively, to detect the delivery area.
Retrieve Ball	Lawnmower collector	With the lawnmower design, the ball would be swiped into the car and up the ramp.
Return Ball	Open gate down mechanism	Gate opens downward to let the ball to roll down the gate into the delivery area due to gravity.

Table 14. Concept A Analysis

Pros:

- It allows large margin of error during ball collection
- It allows the vehicle to pick up the ball without the need to stop
- It is stable and resistant to collision from opponent

Cons:

- Higher mass might decrease the maximum acceleration and velocity of the vehicle
- Gate might block the ball from falling out if not completely open
- Decreased maneuverability and longer turning time.

3.3.2 Concept B

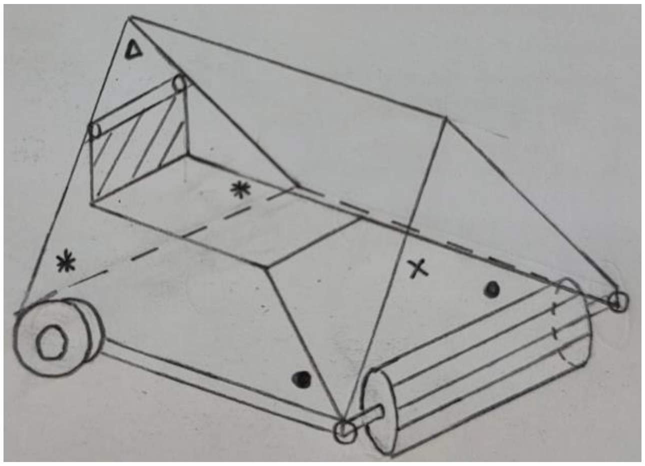


Figure 9. Concept B Drawing

Sub Function	Solution	Concept Description
Structure	Prism shaped	Due to its low centre of gravity, a triangle structure configuration offers the most stable base.
Move Robot	2 dual wheels at the rear of the vehicle, 2 motor in the rear each driving a dual wheel, 2 castor wheels in front	A differential drive is made possible by two dual wheels. By setting each side of the motor to rotate in the opposite direction, the vehicle is able to rotate on the spot. The car is supported while the castor can move freely in all direction.
Detect Obstacle	Skewed Sensors	One long range distance sensor at the front is used to detect ball, two line sensors at the bottom are used to detect edges, one short distance sensor and two limit switches are used to detect the delivery area at the back of the vehicle.
Retrieve Ball	Roller collector	Rubber bands wrapped around the rotating wheel conform to the shape and size of the ball, providing traction and reaction forces to the ball. Thus, with the torque from the rotating wheel, the ball would be pushed up the ramp and into the ball catchment area.

Return Ball	Open gate up mechanism	Gate open up to allow the ball to roll down and enter the delivery area by gravitational force

Table 15. Concept B Analysis

Pros:

- Low center of gravity, stable and mass
- More maneuverable, as the castor wheels can rotate freely in all direction compared to rubber wheels

Cons:

- Rubber bands at the collector might deteriorate and break over time
- Entanglement with opponents might happen during collision at the collector
- The tennis ball has a tendency to be trapped within the collector

3.4 Concept Evaluation

3.4.1 Evaluation Criteria Weight

Several criteria and weighting factors were used to evaluate each concept design. Each criterion was assigned a weighting factor value based on its importance in the design. Table 11 shows the evaluation criteria and assigned weighting factor.

Criteria	Explanation	Weightage (%)
Retrieval Capability	Whether the design allows for a large margin of error	35
Maneuverability	Depends on the turning radius and time taken for one complete rotation	20
Retrieval Speed	Determined by how quickly the robot can detect the ball, retrieve it, and return it to the ball depository area.	20
Complexity	Determined by the robot's ease of construction and programming.	15
Collision Resistance	Depends on the structural integrity of the robot to withstand collision and the ability to counter external torque	10

Table 16. Evaluation Criteria and Weightage

3.4.2 Evaluation Table

A weighted value score was calculated for each concept. This value was computed by summing each weightage and multiplying it by the criteria rating for each concept. The concept that received the highest total weighted value score was chosen. Table 12 summarizes the evaluation for each module:

Evaluation Criterion		Weightage (%)	Criteria Rating for Concept	Concept A		Concept B	
				V	W.V	V	W.V
1	Retrieval Capability	35	Scale of 1 to 5: 1 - worst 5 - best	5	175	5	175
2	Maneuverability	20		4	80	3	60
3	Retrieval Speed	20		5	100	3	60
4	Complexity	15		4	60	3	45
5	Collision Resistance	10		4	40	2	20
Total				Total W.V = 455		Total W.V = 360	

Table 17. Evaluation Table

3.4.3 Evaluation of Results

Retrieval Capability

The larger the margin of error allowed in retrieving the ball, the stronger the retrieval capability. Because the goal is to successfully search for, retrieve, and return tennis balls, retrieval capability is given the highest weight among all the criteria.

Both concept A and B have 175 points for this. Both designs retrieve the ball while approaching it. If a ball is detected nearby, both collectors will spin and engulf the ball into the vehicle. As a result, they both scored similarly on this criterion.

Maneuverability

The gear systems used to synchronize the front and back wheels of each side of the vehicle allow Concept A to move and turn properly. Concept B, which has a differential drive system, can also maneuver smoothly. However, because the vehicle's wheels are at the rear, its turning radius isn't quite ideal.

Retrieval Speed

The time it takes to search and return the ball can be used to gauge retrieval speed. The concept that achieves the overall goal in the shortest amount of time will be given the greatest credit. A similar retrieval mechanism is used in both Concepts A and B. However, Concept B's roller collector might have a ball lodged between the rubber band, which significantly lowered the credit that Concept B received for this criterion.

Complexity

Complexity has a 15% weighting in the selection criteria, 7.5% for programming, and 7.5% for building. As both Concepts A and B have a comparable retrieval mechanism except for the collector, it is thought that their construction challenges will be similar. However, concept A is simpler to program than concept B since it includes more sensors to overcome any problems that may arise. As a result, concept A received a higher rating for this criterion.

Collision Resistance

Concept A is more likely to endure more impact because it is more robust than Concept B. In concept B, the head of the vehicle will deviate easily from its original path during a side collision.

3.4.4 Concept Finalization

The reasons listed in 3.4.2 contributed to Concept A having the highest weighted value. As a result of our weighted criteria, Concept A was chosen as our final design.

4 Embodiment Design

One of the important stages in the product design process is embodiment design. The product, starting from a chosen concept, is further developed to accommodate requirements relating to the desired economic and technical functionality and feasibility of the robot throughout the embodiment design stage. To create a reliable robotic system, we adhere to the three universal rules of embodied design: safety, simplicity, and clarity.

4.1 Safety

The most important rule of embodiment design is safety. For a robot to be safe and not pose a threat to the various elements interacting with it during its lifetime, it must be designed carefully. We combine direct and indirect mechanisms into the robot design to reduce risk and assure safe operations.

4.1.1 Direct Safety

The goal of direct safety is to build components into our robot that help us prevent failure before it occurs. We have implemented the following in our robot:

1. <u>Metal Frame</u> – Metal is used to construct and surround the robot entirely
2. <u>Safeguarding of essential components</u> – To protect them from damaging in the event of a collision, exposed sensors are covered with shells or protectors with screws locking them in place.
3. <u>Insulation of electronic components</u> – To prevent short-circuits, electrical tapes or 3D-printed components are used between the sensors and metal frames
4. <u>Distribution of wheels</u> – 4 wheels are equally spread on 4 corners of the robot to maintain stability

4.1.2 Indirect Safety

The objective of indirect safety design is to limit the extent of damage in the event of failure. This is accomplished by installing a spacer on the roller shaft in the event that it moved out of place. Additionally, a shield is built at the ball collector region to stop the ball from dropping on the adjacent gears.

4.2 Simplicity

The robot's function can be divided into two tasks (navigation and retrieval mechanism). It is also possible to divide the navigation task into smaller tasks (edge detection and path planning). Likewise, there are three parts to the retrieval mechanism: seek, find, and return ball. We may create a program that is modular and effective by decomposing the functionality into these straightforward, mutually exclusive parts. Also, electrical, mechanical, and computing logic components are combined to form the robot itself. This straightforward classification makes it simpler to assemble the robot.

4.3 Clarity

The robot must have a straightforward and unambiguous functionality in order to function with the fewest interruptions possible and to make maintenance and repairs simple. Some distinct functionality for various component:

1. Wheels + Gears: Basis of motion and maneuvering
2. External Frames: Provide mounting points and safeguard the components onboard.
3. Collector Mechanism: Collect the ball and push it into storage area
4. Ramp: Provide friction force in the ball collecting process
5. Storage Zone: Storage area for the ball before it is being disposed
6. Battery: Provide power supply to the robot
7. Line sensors: Edge detection
8. Limit switches: Determine if the ball is collected, if the ball dispensing gate is closed, if the robot has hit the metal plate at ball delivering area.
9. Distance sensors: Detecting ball and opponents
10. Digital compass: Provide orientation information to the robot

4.4 Principles

The design principles that being applied are force transmission, task division and stability.

4.4.1 Force Transmission

Uniform Stress

The battery and the vex microcontroller are placed at the center the top plate to evenly distribute the stress between the 4 bars that support the top plate.

4.4.2 Division of tasks

To ensure mutually exclusive compatibility between the various components, each component serves a specific purpose. The ramp to route the ball to the storage where it will be waiting for release, the collector to gather balls, and the wheels and gears are built for motion and maneuverability.

4.4.3 Stability

All the 4 wheels are placed symmetrically to provide the widest base of the robot. Also, the vex microcontroller and batteries are placed closed to the center of the base as much as possible so that the load can be distributed evenly on the 4 wheels.

5 Detailed Design

This section goes into detail about the chosen concept, Concept A. First, detailed justifications, implementations, and improvements are made for each hardware component (Structure, Drive System, and Retrieval Mechanism). Following that, calculations for the drive system and retrieval mechanism will be discussed, followed by overall assembly and part engineering drawings. Finally, the assembly of each electrical component (Motors, Sensors, and Limit Switches) is detailed, followed by an overall wiring diagram.

5.1 Hardware Design

5.1.1 Structure

The chassis' size restriction of 30 cm in length and 30 cm in width was the primary factor to be taken into account. A square base frame 22.5 x 30.0 cm with a hollow structure was built using metal plates and bars to accommodate the ramp. In order to prevent collisions, the wheels, motors, and gears with the exception of the collector gears are all housed inside the base frame. The remaining 7.5 cm is set aside for the installation of our collector. Our original plan was to conceal the battery, vex controller, and compass beneath the ramp. However, due to a lack of space, all of this has been moved to the robot's top plate. Our robot has a cuboid design, which is similar to what was selected for the conceptual design phase.

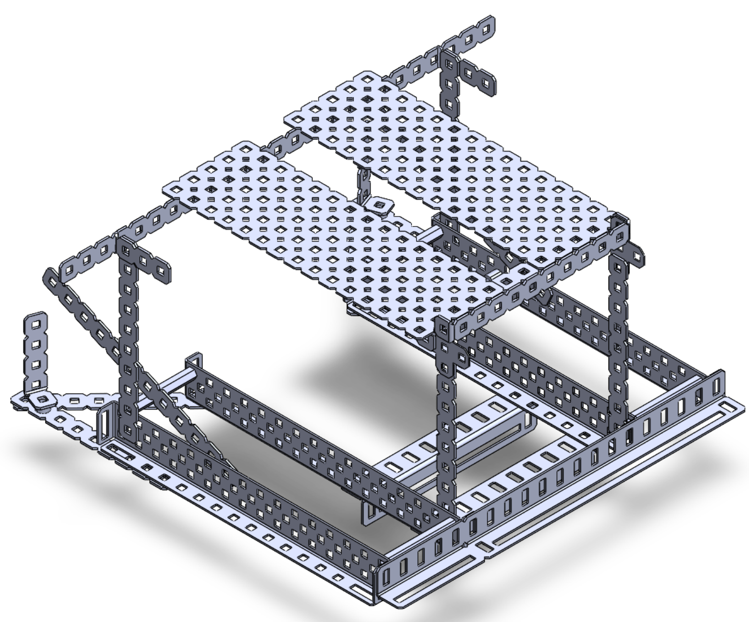


Figure 10. Robot Frame Design

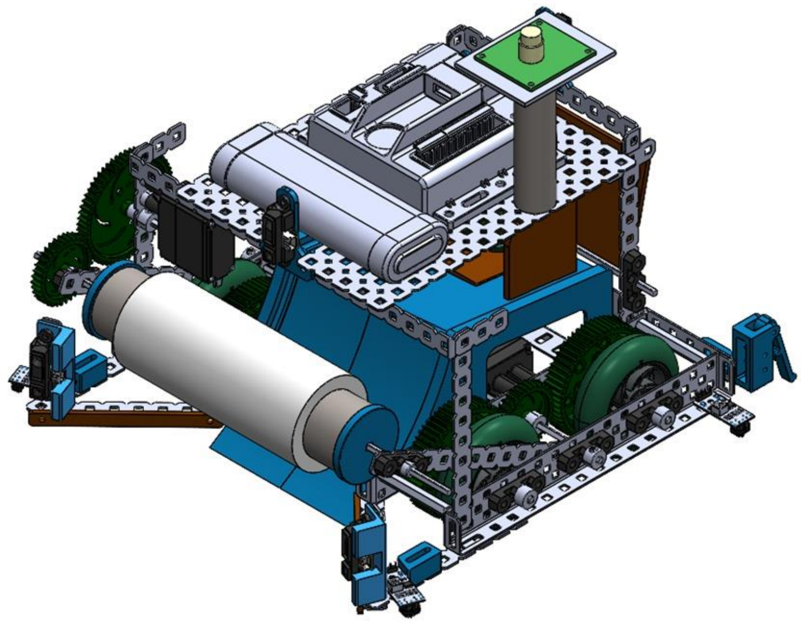


Figure 11. Overall Robot CAD Design

Improvisation

After some testing, we discovered that some sensors that are mounted on the robot's sides and corners are vulnerable to collision. To enclose and safeguard them, housings are 3D printed. Additionally, the limit switches are not sufficiently extruded to function. The implementation of an extension cover design enables the proper operation of the limit switches.

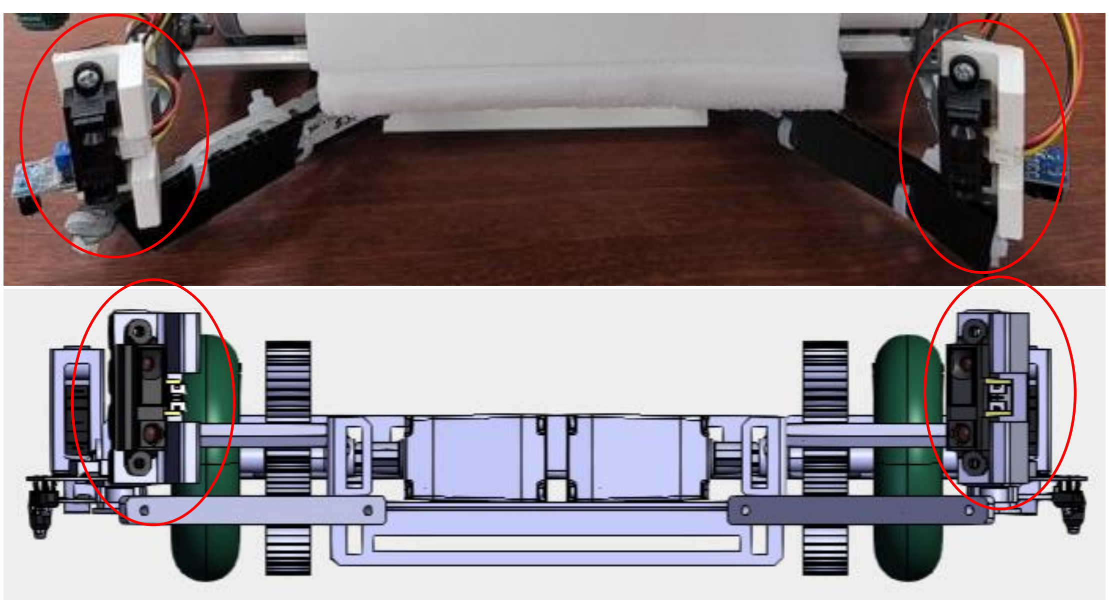


Figure 12. Robot Front Distance Sensors with Housing

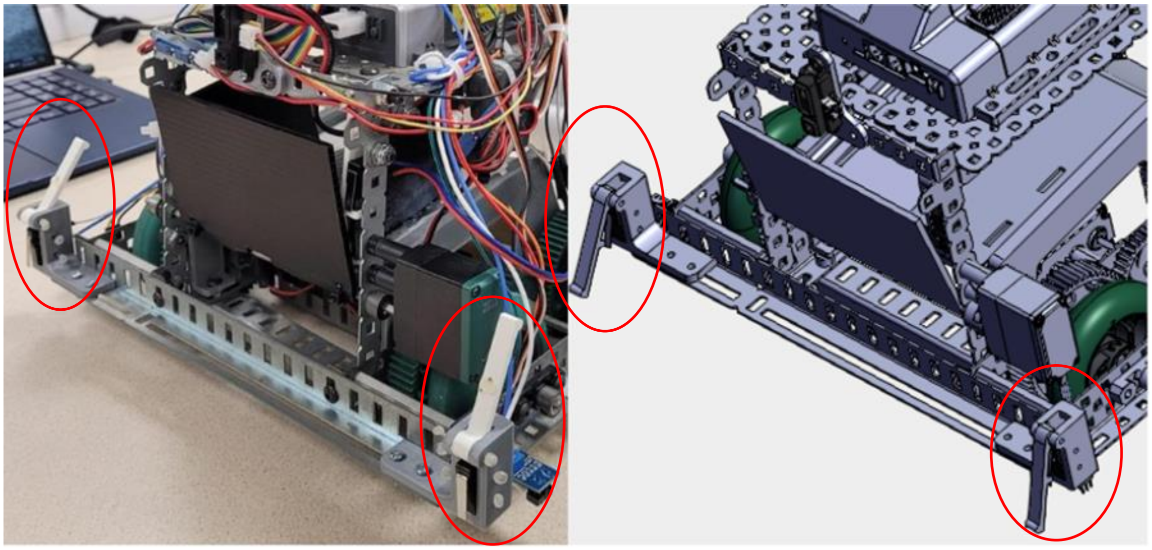


Figure 13. Robot Limit Switches with Extensions

5.1.2 Drive System

The maneuverability and collision resistance, which complement our goal of collecting and delivering the tennis ball, were our key concerns.

Having good control over speed and turning speed is what is meant by maneuverability. We could synchronize the two wheels to rotate at the correct speed by using gears to connect the two wheels on the same sides. Additionally, because the robot's maneuverability is controlled by just two motors, its turning speed can be managed similarly to a differential drive system.

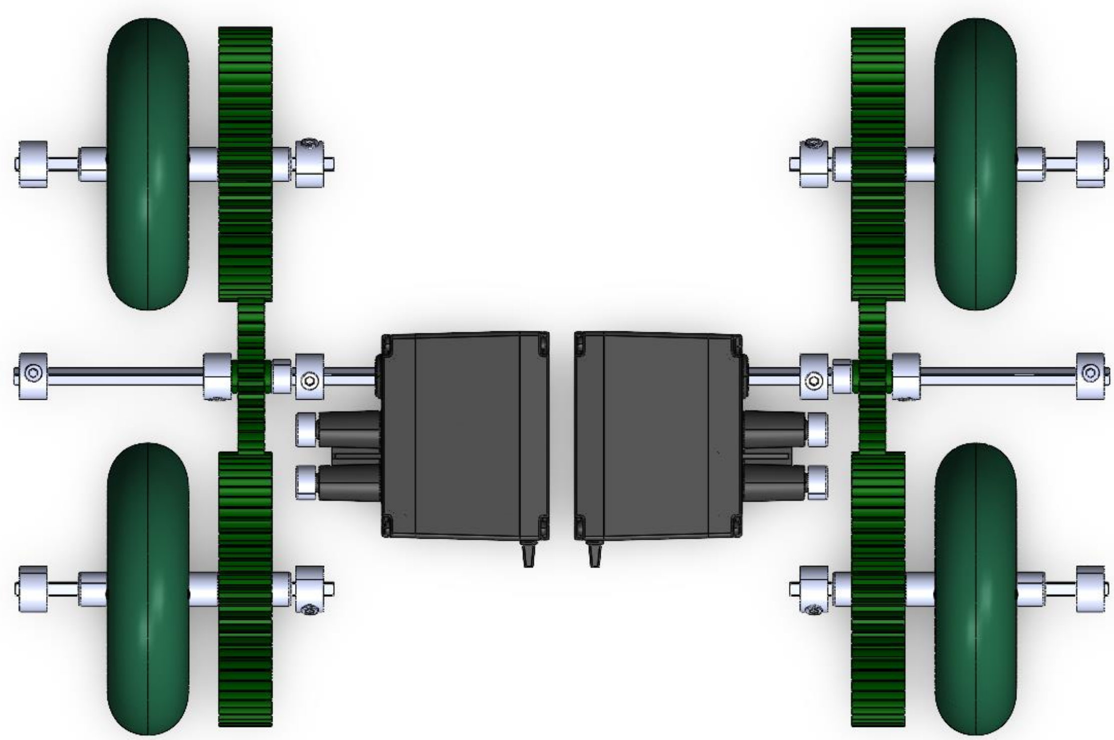


Figure 14. Robot Wheels and Gears System

Collision resistance refers to the ability to stay on its own path during a collision with the opponent’s vehicle. As previously indicated, the base frame of the robot completely encloses the gear and wheel system that powers it.

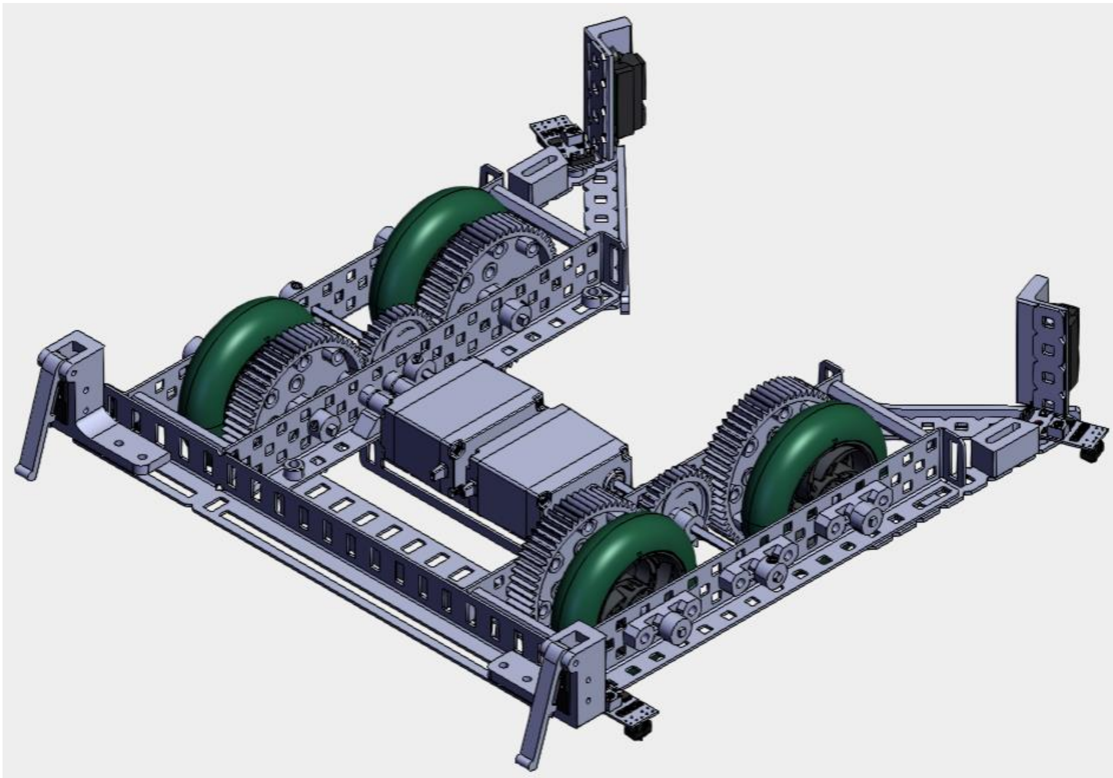


Figure 15. Robot Wheels and Gears Enclosed Inside the Frame

5.1.3 Retrieval Mechanism

This section details the methods and implementations of the robot collecting the tennis balls, which are placed randomly by the lab technicians, as well as to deposit the tennis ball into the respective collection area found in the arena. The retrieval mechanism is split into three different components, the Ball Collector, the Ramp section to guide the ball into the carriage system in the robot and the Ball Disposal system to deposit the ball into the collection area.

5.1.3.1 Ball Collector

The Ball Collector is the front of robot which is made of a 40mm diameter PVC pipe of 18cm length with sponge foam hot-glued onto the pipe. The sponge is cut and glued in such a manner to allow the minor protrusion of 2mm of sponge to push the ball up the ramp system into the ball carriage area. Sponge was chosen for the collector as the team decided that compression force as well as the roughness of the sponge surface would greatly help to push and guide the ball up the ramp.

In order to facilitate the control and direction of rotation of the collector, two holder parts are designed, and 3-D printed to allow a 32cm square bar to pass through and to be connected to a 2-Wire Motor. Black plastic spacers of 1cm have been added to fix the collector position in place and prevent it from wobbling and moving about the metal bar.

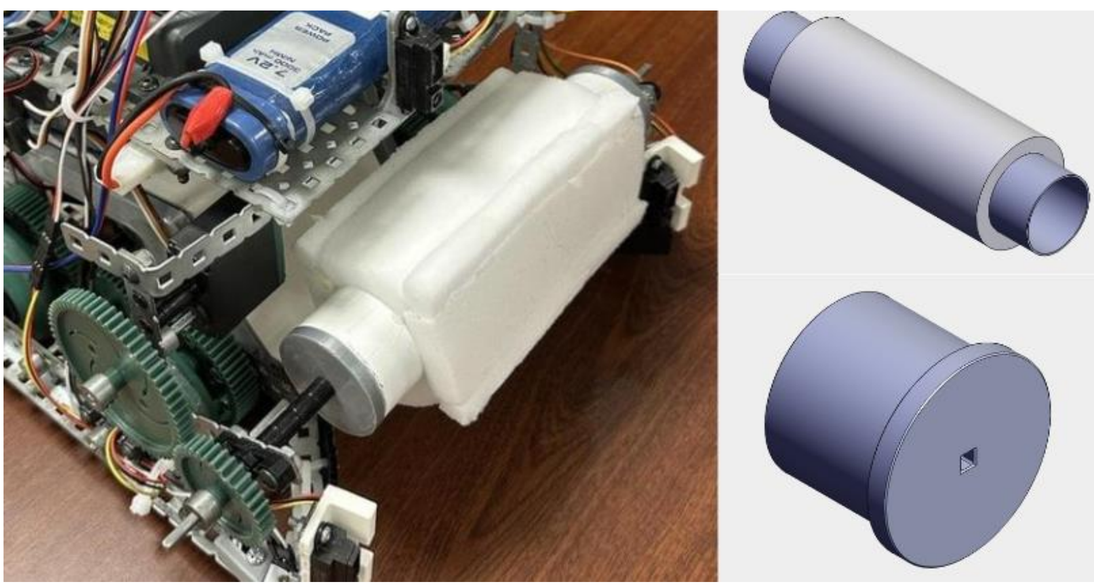


Figure 16: Close up of Robot's Collector with its Respective 3D Printed Parts

5.1.3.2 Ramp System

The Ramp System acts as the intermediary between the Ball Collector and the Ball Carriage systems. It aims to use the compression force and the torque of the motor moving the collector to push the ball up the ramp into the ball carriage system.

The ramp is made with four different 3D printed parts and merged with superglue. The ramp is designed to have a height of 6cm at the end of the ramp in order to match the height of the collection area's box. The ramp supports that are located on the side of the ramps are designed to have cut-outs to allow wires to pass through the ramp from the motors and sensors etc. into the vex controller to aid cable management of the robot.

The ramp features 2 slots of the left and right to allow installation of guards as a preventive measure for the chance of the ball exiting the robots from the side instead of the back of the robot as well as a slot in the middle to house a limit switch to detect if the ball have exited the ramp and entered the ball carriage system. The ramp is also designed to be sloped slightly towards the middle of the ramp to further facilitate the guidance of the ball towards the middle of the ramp onto the limit switch.

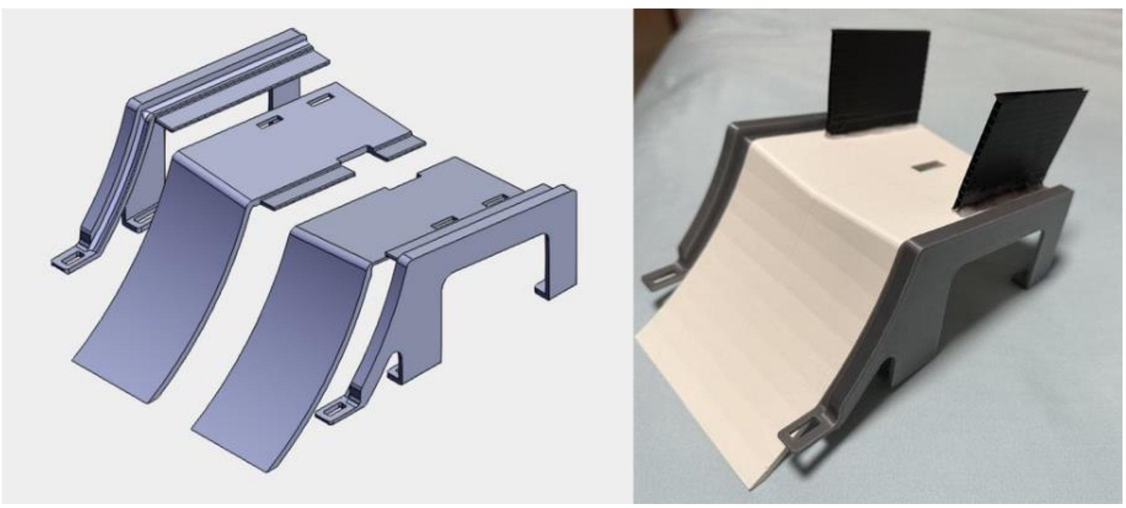


Figure 17: Comparison of Exploded View of the Ramp (Left) and Constructed Ramp of the Robot (Right)

5.1.3.3 Ball Disposal System

The Ball Disposal System is responsible for unloading the ball when the robot reaches the collection point of the arena upon collecting a tennis ball. The Ball Disposal System consists of a 2-Wire Motor attached to a square bar across the machine and a gate made of corrugated board of 10cm by 6cm (LxH) is attached to the square bar using hot-glue.

When the robot collected the ball and reached the collection area via the activation of the limit switches located at the back of the robot, the motor controlling the gate is then activated and rotates the ramp 100° clockwise onto the collection area. With the aid of the sloped ramp, the ball would naturally roll down and into the collection area before rotating counterclockwise to its original position. The robot would then confirm that the gate is truly closed when the gate touches the limit gate on the robot.

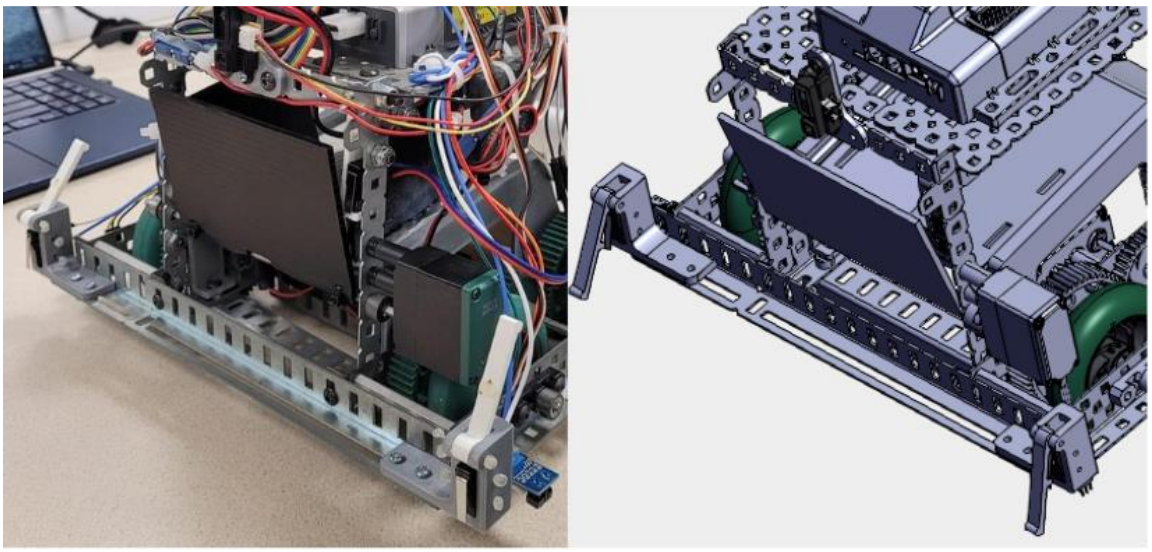


Figure 18: Comparison of Constructed Part and Modelling of the Ball Disposal System

5.2 Design Calculations

5.2.1 Motor Specifications

Referring to the datasheet of the 2 motors provided in the Appendix C (3-Wire Motor and 2-Wire Motor), the following table summarizes the specification of the 2. 2-Wire Motor 393 can be configured into "High Speed" or "Turbo" modes. However, the standard version is sufficient for the application for our project.

Types of Motors	**Free Speed (rpm)**	**Stall Torque (Nm)**	**Free Current (A)**	**Stall Current (A)**
3-Wire Motor (276-2163)	100 at 7.5V	0.7344 (6.5 in lbs)	0.14	1.6
2-Wire Motor (393)	100 at 7.2V (Standard mode)	1.67 (14.76 in lbs)	0.37	4.8

Table 18: Motor Specifications

Both motors can only operate in continuous mode. In this project, 3-wire motors were used in ball collection and ball dispensing mechanism while 2-wire motors were used for the drivetrain of the vehicle.

Since there is no internal motor controller in the 2-wire motor 393, 2 motor controllers (Motor Controller 29) were used between the Vex microcontroller and 2-wire motors 393 to regulate the speed of 2-wire motor.

5.2.2 Power System

As there are 3 rounds of competition, with each round lasting for 3 mins, the following calculations are performed to ensure that there is sufficient battery power to operate the robot throughout the competition.

Total duration of competition = $3\ rounds \times 3\ mins = 9mins$
To allow some allowance for trial run, it is assumed that the robot will operate for 15 mins with a fully charged battery ($3000mAh, 7.2V$):

$$Battery\ power = \frac{3000\ mAh \times 10^{-3} \times 7.2V}{\frac{15\ mins}{60}} = 86.4W$$

The following table tabulates the power used by each electrical component (referring to their respective data sheets):

	No	Current Consumption (A)	Voltage (V)	Power consumed per device (W)	Total Power Consumed (W)
2-Wire Motor	2	0.37	7.2	2.66	5.32
3-Wire Motor	2	0.14	7.5	1.05	2.10
Motor Controller	2	0.012	8.5	0.10	0.20
Sharp Distance Sensor (Long)	3	0.030	5.0	0.15	0.45
Sharp Distance Sensor (Short)	1	0.012	5.0	0.06	0.06
IR Line Tracking Module	4	0.020	5.0	0.10	0.40
Digital Compass	1	0.020	5.0	0.10	0.10

Table 19: Power Consumption of each component

$Total\ power\ consumption$

$= 5.32 + 2.10 + 0.20 + 0.45 + 0.06 + 0.40 + 0.10$

$= 8.63\text{W}$

Since total power consumption << battery power, the robot is operating well within the scope of the battery life.

5.2.3 Drive System

The following table summarizes the mass of the components of the robot, to facilitate for the calculation of force and torque needed to drive the robot.

Component	No.	Mass (g)	Total Mass (g)
Vex Microcontroller	1	137	137
Battery (276-1491)	1	312	312
2-Wire Motor	2	87	174
3-Wire Motor	2	95	190
Chassis*	N/A	2000	2000
Collector (Roller)*	1	80	80
Sharp Distance Sensor*	4	15	60
Wheels	4	50	200
Gear System*	N/A	180	180
3D Printed Ramp	1	120	120
Sensors (IR Line Tracking Module, Limit Switches, Compass)*	N/A	90	90
3D Printed Parts (Sensor Mounts and Spacers)	N/A	80	80
Others (Screws, Nuts, Cables etc.)*	N/A	50	50
Tennis Ball	1	58	58

Table 20 Mass of Each Component of Robot

*The mass of the components is roughly estimated for calculation purposes

$$Robot\ Mass = Sum\ of\ the\ mass\ of\ the\ components = 3731g = 3.73kg$$

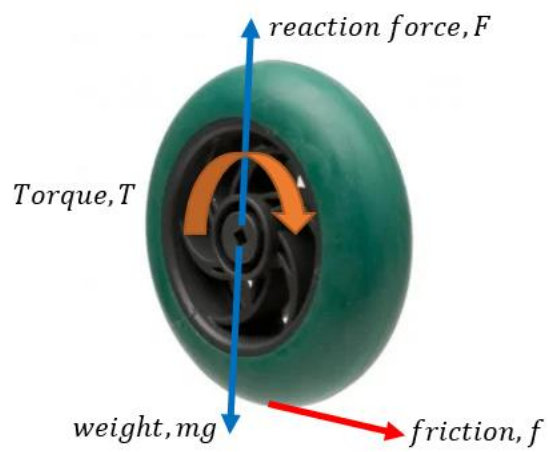


Figure 19 Force Analysis of the Robot Wheel

(A) Force Analysis

Given that mass of the vehicle, $m = 3.73k\ g$; radius of the wheel, $r = 3.5 \times 10^{-2}\ m$; coefficient of friction between the wheel and a smooth wooden surface, $\mu = 0.5$

Since the robot is a 4-wheel drive system, we assume that the weight of the robot is evenly distributed among 4 wheels:

$$Reaction\ force, F = mg = \frac{3.73}{4}\ kg \times 9.81\ m\ s^{-2} = 9.15\ N$$

$$Friction, f = \mu F = 0.5 \times 9.15\ N = 4.575\ N$$

Torque required to drive the wheel, $T = f \times r = 4.575\ N \times 3.5 \times 10^{-2}\ m$

$$= 0.160\ Nm$$

(B) Torque to Drive the Wheel

The following diagram shows one side of the gear system of the drive system.

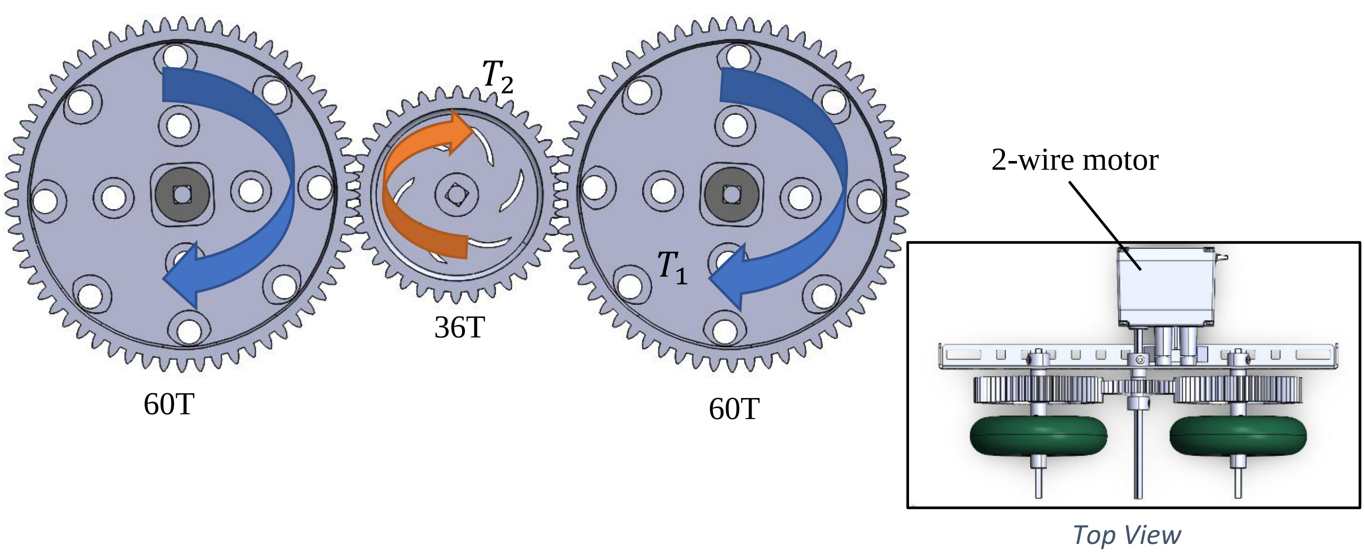


Figure 20: Gear Train of the Wheel Drive System

As we can observe from the top view of the gear train, the 36T gear is connected to the 2-wire motor and is meshed with 2 60T gears to drive the wheel. It is assumed that the gear train has the efficiency of 90%.

Under constant velocity operation:

To find the torque needed for the motor:

$$\sum T = 0$$

$$T_2 = 2\,T_1 = 2(0.160) = 0.320\ Nm$$

With 90% efficiency, the torque needed for motor, $T_{motor} = \frac{0.320}{0.9} = 0.356\ Nm$

Under acceleration operation:

During the competition, we operate the robot to drive at the speed of $0.173\ m\ s^{-1}$. Hence, $\omega_{motor} = 78.7 rpm$. *(Refer (C) Robot Operating Speed for detailed calculations)* Assuming the motor accelerates to the angular velocity of $78.7 rpm$ within $0.5s$,

$$angular\ acceleration, \alpha = \frac{78.7 \times \frac{2\pi}{60}}{0.5} = 16.48\ rad\ s^{-2}$$

$$T_{accelerating} = J\alpha$$

To calculate the moment of inertia, J:

We assume the screws, lock nuts and shafts have negligible mass compared to the wheels and gears. The following table summarizes the moment of inertia of each component obtained from SolidWorks imported from Vex Website (refer to Appendix D):

Components	**Moment of Inertia/J ($g\ mm^2$)**	**Moment of Inertia/J (k$g\ m^2$)**
36T Gear	1192	1.192×10^{-6}
60T Gear	14234	1.4234×10^{-5}
Wheel	28972	2.8972×10^{-5}
2-Wire Motor	39164	3.9164×10^{-5}

Table 21: Moment of Inertia of Each Component of Robot

The moment of inertia on one side of a gear mechanism is in the ratio of the inverse square of the gear ratio, r when seen from the other side, and the efficiency η of the gear mechanism must also be considered.

$$J_{reflected} = \frac{J}{r^2\eta}$$

To calculate the moment of inertia, J:

$$J = J_{motor} + J_{36T\ Gear} + 2\left(J_{60T\ Gear,reflected} + J_{wheel,reflected}\right)$$

$$J = 3.9164 \times 10^{-5} + 1.192 \times 10^{-6} + 2\left[\frac{1.4234 \times 10^{-5}}{0.9\left(\frac{36}{60}\right)^2} + \frac{2.8972 \times 10^{-5}}{0.9\left(\frac{36}{60}\right)^2}\right]$$

$$J = 3.071 \times 10^{-4}\ kg\ m^2$$

Hence, to calculate the required motor torque to accelerate:

$$T_{accelerating} = J\alpha = 3.071 \times 10^{-4}(16.48) = 0.0051\ Nm$$

Total torque required by the motor:

$$T_{total} = T_{accelerating} + T_{steady\ state} = 0.0051\ Nm + 0.356\ Nm = 0.361\ Nm$$

Under breakaway operation:

Referring to the lecture notes, since the motor shaft is connected to the load with sleeve bearings and the motor has been used regularly, we assume the breakaway torque to be 140% of the running torque:

$$Breakaway\ Torque = 1.4 \times 0.356 = 0.498\ Nm$$

Conclusion:

Breakaway	**Accelerating**	**Running (Steady State)**
$0.498\ Nm$	$0.361\ Nm$	$0.356\ Nm$

Since $0.498\ Nm \ll 1.67\ Nm$ (stall torque of the motor), the 2-Wire Motor is a good fit for the drive system.

(C) Robot Operating Speed

With known stall torque ($1.67\ Nm$) and free angular speed ($100rpm\ at\ 7.2V$) of the 2-wire motor, we plotted a linear graph of torque against angular velocity as shown in the figure below:

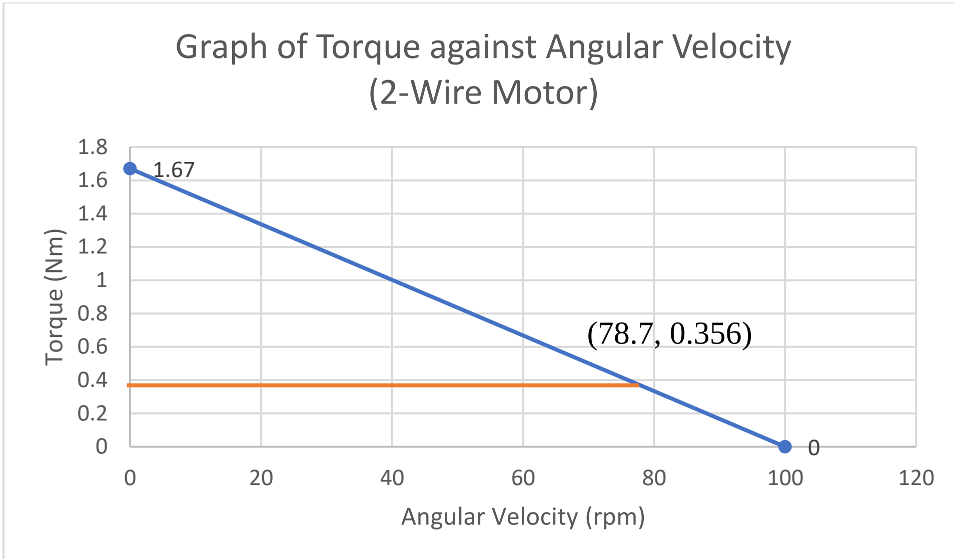


Figure 21: Graph of Torque against Angular Velocity (2-Wire Motor)

$$T = -0.0167\omega + 1.67$$

At $T_{motor} = 0.356\ Nm,\ \omega_{motor} = \omega_{36T} = 78.7\ rpm$

$$Gear\ Ratio = \frac{N_{60T}}{N_{36T}} = \frac{\omega_{36T}}{\omega_{60T}} = \frac{60}{36} = \frac{5}{3}$$

$$\frac{5}{3} = \frac{78.7}{\omega_{60T}}$$

Hence, $\omega_{60T} = 47.22\ rpm$

We know that $\omega_{wheel} = \omega_{60T} = 47.22\ rpm$ and $r_{wheel} = 3.5 \times 10^{-2} m$

$$v = r_{wheel} \times \omega_{wheel} = \frac{47.22\ rpm \times 2\pi}{60} \times 3.5 \times 10^{-2} m = 0.173\ m\ s^{-1}$$

Since the competition area is $2.4m \times 1.2m$,

We use $v = \frac{d}{t}$

$$0.173 = \frac{2(2.4)}{t}$$

Hence, $t = 27.74s$

The robot can reach the end of arena and return to ball dispensing area within 30s, which is deemed sufficient for the winning condition of collecting 2 balls to in 3 minutes.

5.2.4 Retrieval Mechanism

(A) Ball Collection

<u>(I) Force Analysis</u>

The following assumptions are made for the force analysis of the ball up the ramp:

1. The ball and the roller does not deform when compressed

2. There is always only one point of contact between the ball and the ramp

3. The ball does not slip as it rolls

4. During the first time of contact between the ball and the ramp, the centre of the roller, the bottom tip of the ramp and the centre of the ball forms a straight line (Refer to Figure 22 below)

5. The coefficient of friction between the roller and the 3D-printed ramp, $\mu = 0.60$.

The figure below shows the force analysis of the ball rolling up the ramp:

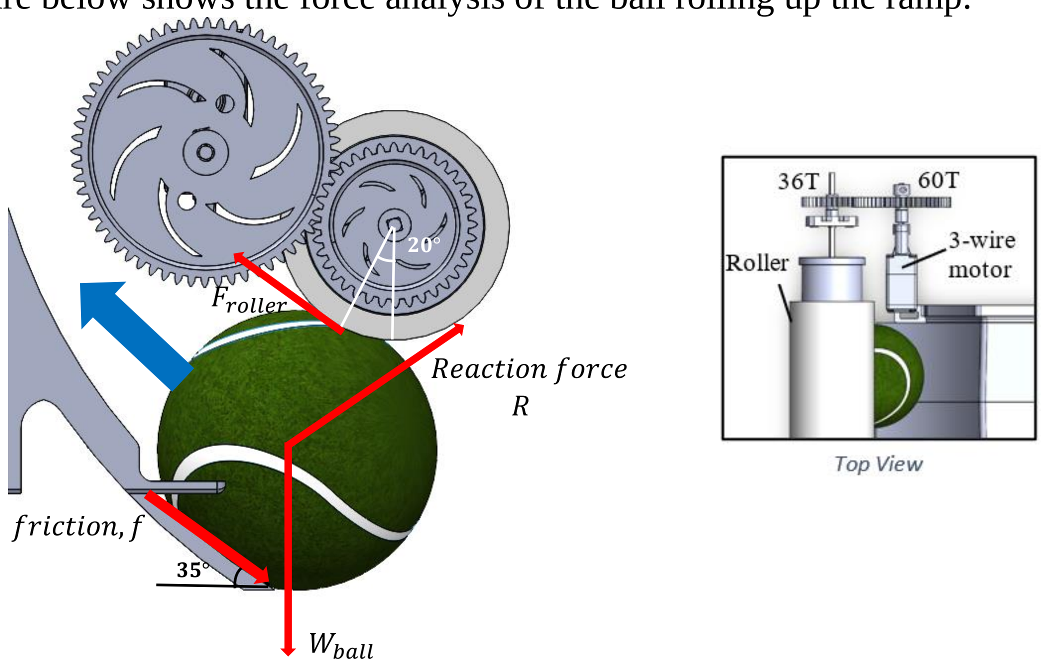


Figure 22: Force Analysis of Tennis Ball up the Ramp

We know that $friction, f = \mu R = 0.6R$; $m_{ball} = 58 \times 10^{-3}\ kg$ hence $W_{ball} = m_{ball} g = 58 \times 10^{-3}\ (9.81) = 0.569N$

Consider horizontal component of the forces:

$F_{roller} \cos 20 = R \sin 35 + f \cos 35$

$F_{roller} \cos 20 = (\sin 35 + 0.6 \cos 35)\, R$ ---- (1)

Consider vertical component of the forces:

$F_{roller} \sin 20 + R \cos 35 = f \sin 35 + \ W_{ball}$

$R\,(\cos 35 - 0.6 \sin 35) = 0.569 - F_{roller} \sin 20$ --- (2)

By solving the simultaneous equations (1) and (2) above, we obtain the following:

$R = 0.6596\, N$

$F_{roller} = 0.7476\, N$

We know that the radius of the roller, $r = 28 \times 10^{-3}\ m$; Hence to find the torque required for the roller to drive up the ball,

$T_{roller} = 0.7476(28 \times 10^{-3}) = 0.021\ Nm$

<u>(II) Torque to Drive the Collector</u>

The following diagram shows the gear system of the collector mechanism

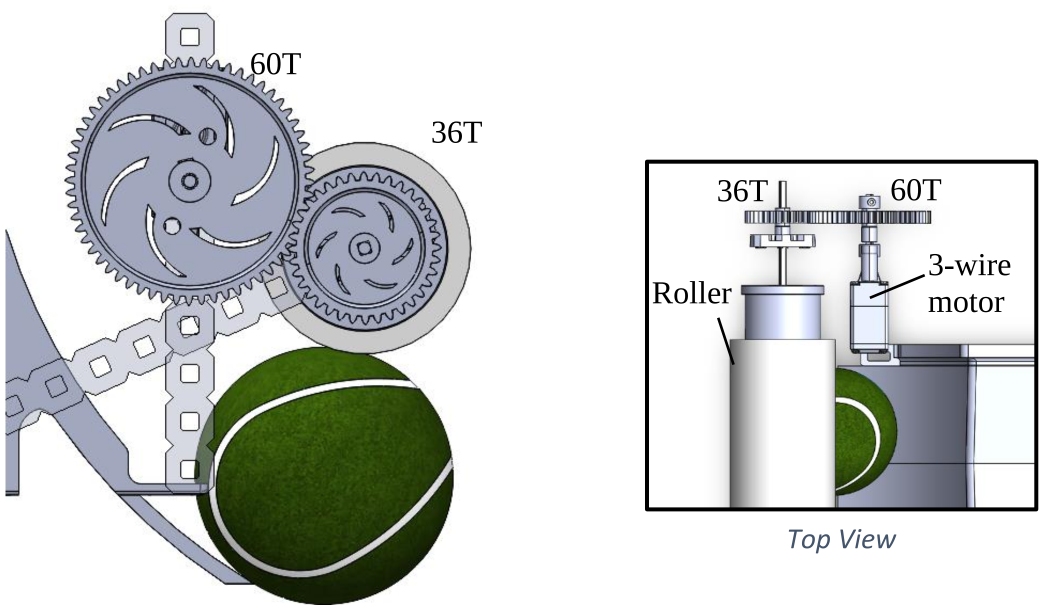


Figure 23: Gear Train of the Collector Mechanism

As we can observe from the top view of the gear train, the 60T gear is connected to the 3-wire motor and is meshed with the 36T gears which then drive the roller. It is assumed that the gear train has the efficiency of 90%.

Under constant velocity operation:

Since the 36T gear is used to drive the roller, we know that:

$$T_{36T} = T_{roller} = 0.021\ Nm$$

$$\frac{T_{60T}}{T_{36T}} = \frac{\omega_{36T}}{\omega_{60T}} = \frac{N_{60T}}{N_{36T}} = \frac{60}{36}$$

$$T_{60T} = \frac{60}{36}(0.021\ Nm) = 0.035\ Nm$$

Since the 60T gear is driven by the 3-wire motor, we know that:

$$T_{motor} = T_{36T} = 0.035\ Nm$$

With 90% efficiency, the torque needed for motor, $T_{motor} = \frac{0.035}{0.9} = 0.039\ Nm$

Under acceleration operation:

During the competition, we operate the motor to drive the roller at angular velocity, $\omega_{motor} = 94.7rpm.$ *(Refer (III) Collector Motor Speed for detailed calculations)*
Assuming the motor accelerates to the angular velocity of $94.7rpm$ within $0.5s$,

$$angular\ acceleration, \alpha = \frac{94.7 \times \frac{2\pi}{60}}{0.5} = 19.83\ rad\ s^{-2}$$

$$T_{accelerating} = J\alpha$$

To calculate the moment of inertia, J:

We assume the screws, lock nuts and shafts have negligible mass compared to the roller and gears. The following table summarizes the moment of inertia of each component obtained from SolidWorks (refer to Appendix D):

Components	Moment of Inertia/J ($g\ mm^2$)	Moment of Inertia/J ($kg\ m^2$)
36T Gear	1192	1.192×10^{-6}
60T Gear	28972	2.8972×10^{-5}
Roller(including PVC pipe, foam and coupler)	28476	2.8476×10^{-5}
3-Wire Motor	31842	3.1842×10^{-5}

Table 22: Moment of Inertia of Each Component of Robot

$$J = J_{motor} + J_{60T\ Gear} + J_{36T\ Gear,reflected} + J_{Roller,reflected}$$

$$J = 3.1842 \times 10^{-5} + 2.8972 \times 10^{-5} + \frac{1.192 \times 10^{-6}}{0.9\left(\frac{60}{36}\right)^2} + \frac{2.8476 \times 10^{-5}}{0.9\left(\frac{60}{36}\right)^2}$$

$$J = 7.268 \times 10^{-5}\ kg\ m^2$$

Hence, to calculate the required motor torque to accelerate:

$$T_{accelerating} = J\alpha = 7.268 \times 10^{-5}(19.83) = 0.0014\ Nm$$

Total torque required by the motor:

$$T_{total} = T_{accelerating} + T_{steady\ state} = 0.0014\ Nm + 0.039\ Nm = 0.041\ Nm$$

<u>Under breakaway operation:</u>

Referring to the lecture notes, since the motor shaft is connected to the load with sleeve bearings and the motor has been used regularly, we assume the breakaway torque to be 140% of the running torque:

$$Breakaway\ Torque = 1.4 \times 0.039 = 0.055\ Nm$$

<u>Conclusion:</u>

Breakaway	Accelerating	Running (Steady State)
$0.055\ Nm$	$0.041\ Nm$	$0.039\ Nm$

Since $0.055\ Nm \ll 0.7344\ Nm$ (stall torque of the motor), the 3-Wire Motor is a good fit for the collector system.

(III) Collector Motor Speed

With known stall torque ($0.7344\ Nm$) and free angular speed ($100 rpm\ at\ 7.5V$) of the 3-wire motor, we plotted a linear graph of torque against angular velocity as shown in the figure below:

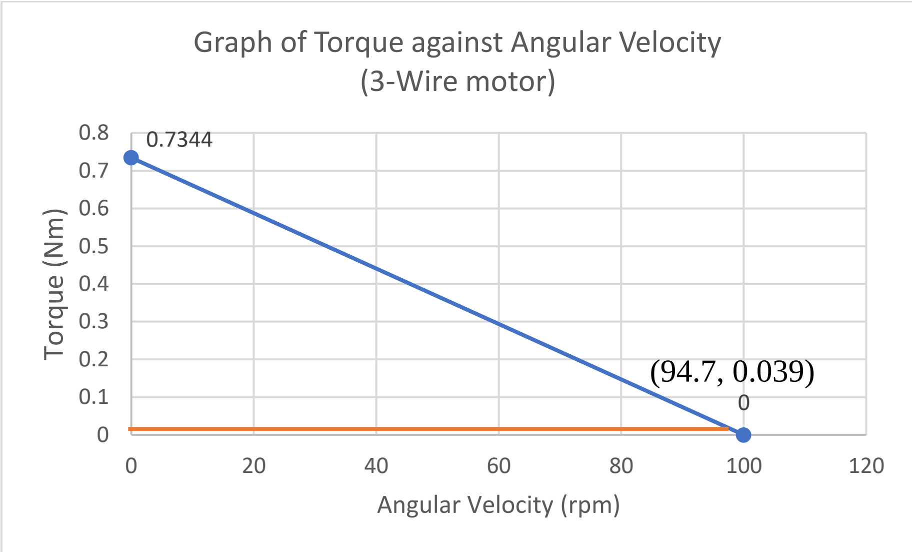


Figure 24: Graph of torque against Angular Velocity (3-Wire Motor)

$$T = -0.007344\omega + 0.7344$$

At $T_{motor} = 0.039\ Nm$, $\omega_{motor} = \omega_{60T} = 94.7\ rpm$

The maximum angular velocity of the motor, $\omega_{motor} = 94.7\ rpm$ The roller is able to collect the ball up the ramp as long as $\omega_{motor} \leq 94.7\ rpm$.

$$Gear\ Ratio = \frac{N_{60T}}{N_{36T}} = \frac{\omega_{36T}}{\omega_{60T}} = \frac{60}{36} = \frac{5}{3}$$

$$\frac{5}{3} = \frac{\omega_{36T}}{94.7\ rpm}$$

Hence, $\omega_{roller} = \omega_{36T} = 157.83\ rpm$

The angular velocity of the roller can go up to $157.83\ rpm$.

(B) Ball Dispensing

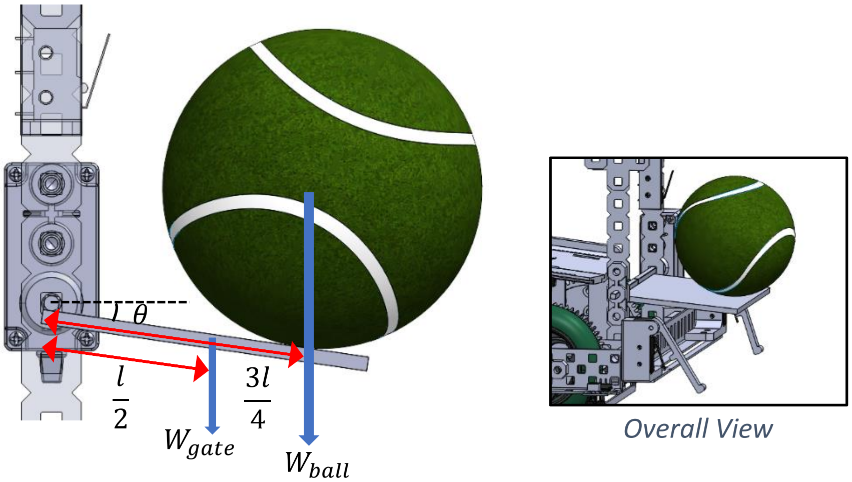


Figure 25: Force Analysis for Ball Dispensing Mechanism

Given that the $m_{ball} = 58 \times 10^{-3} kg$; $m_{gate} = 5 \times 10^{-3}\ kg$;$\theta = 10°$ and length of the gate, $l = 70 \times 10^{-3} m$.

We assume that the friction between the tennis ball and the gate is negligible, to calculate the torque required by the motor to actuate the gate:

$$T_{motor} = m_{gate} g \left(\frac{l}{2}\ cos\theta\right) + m_{ball} g \left(\frac{3l}{4}\ cos\theta\right)$$

$$T_{motor} = 5 \times 10^{-3}(9.81)\left(\frac{70 \times 10^{-3}}{2}\ cos10\right) + 58 \times 10^{-3}(9.81)\left(\frac{3(70 \times 10^{-3})}{4}\ cos10\right)$$

$$T_{motor} = 0.0311\ Nm$$

Referring to the graph of torque against angular velocity of the 3-wire motor plotted,

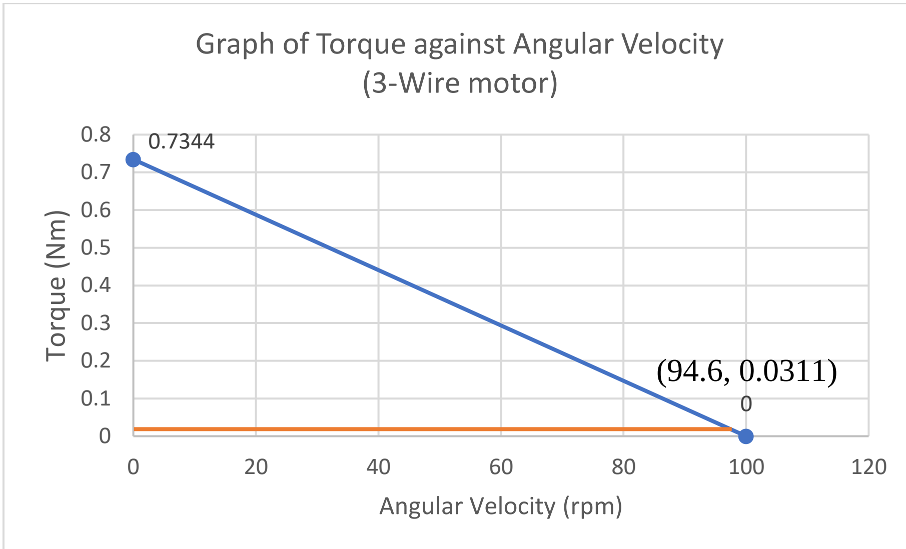


Figure 26: Graph of Torque against Angular Velocity (3-wire motor)

$$T = -0.007344\omega + 0.7344$$

At $T_{motor} = 0.0311\ Nm, \omega_{motor} = 94.6\ rpm$

The maximum angular velocity of the motor, $\omega = 94.6\ rpm$. The motor is able to flap and dispense the ball as long as $\omega \leq 94.6\ rpm$.

5.3 Engineering Drawings

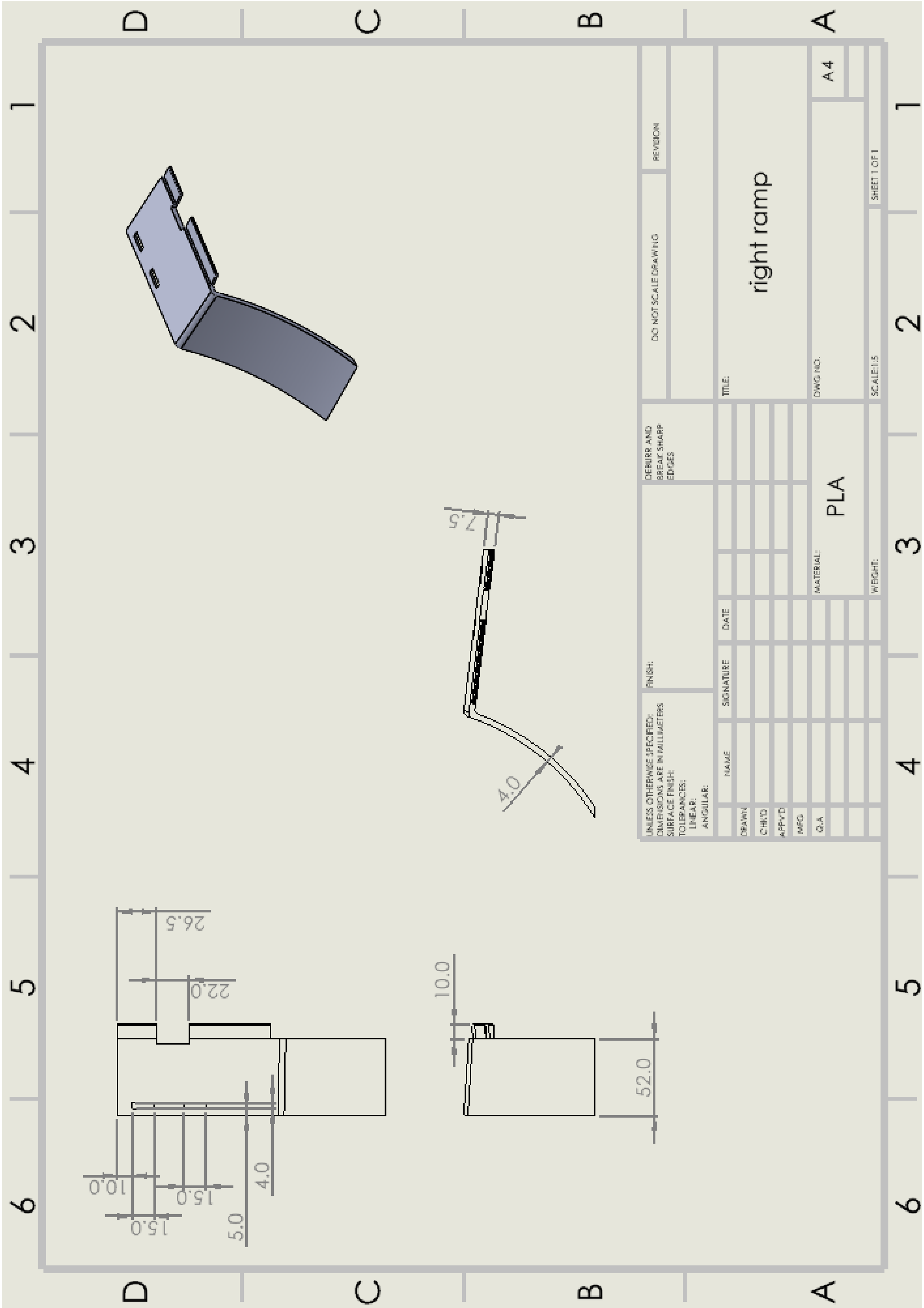


Figure 27: Right ramp

Figure 28: Left ramp

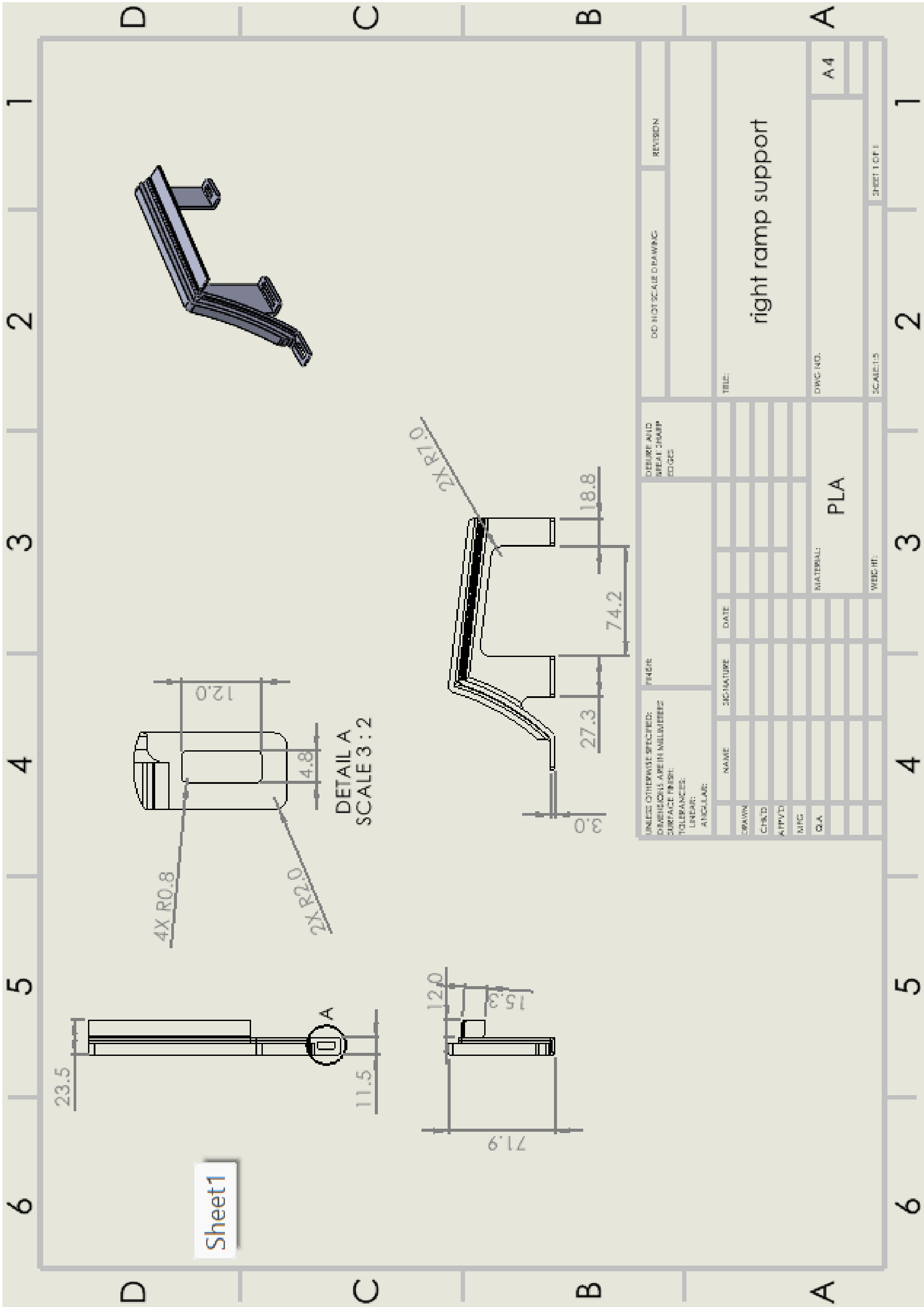


Figure 29: Right ramp support drawing

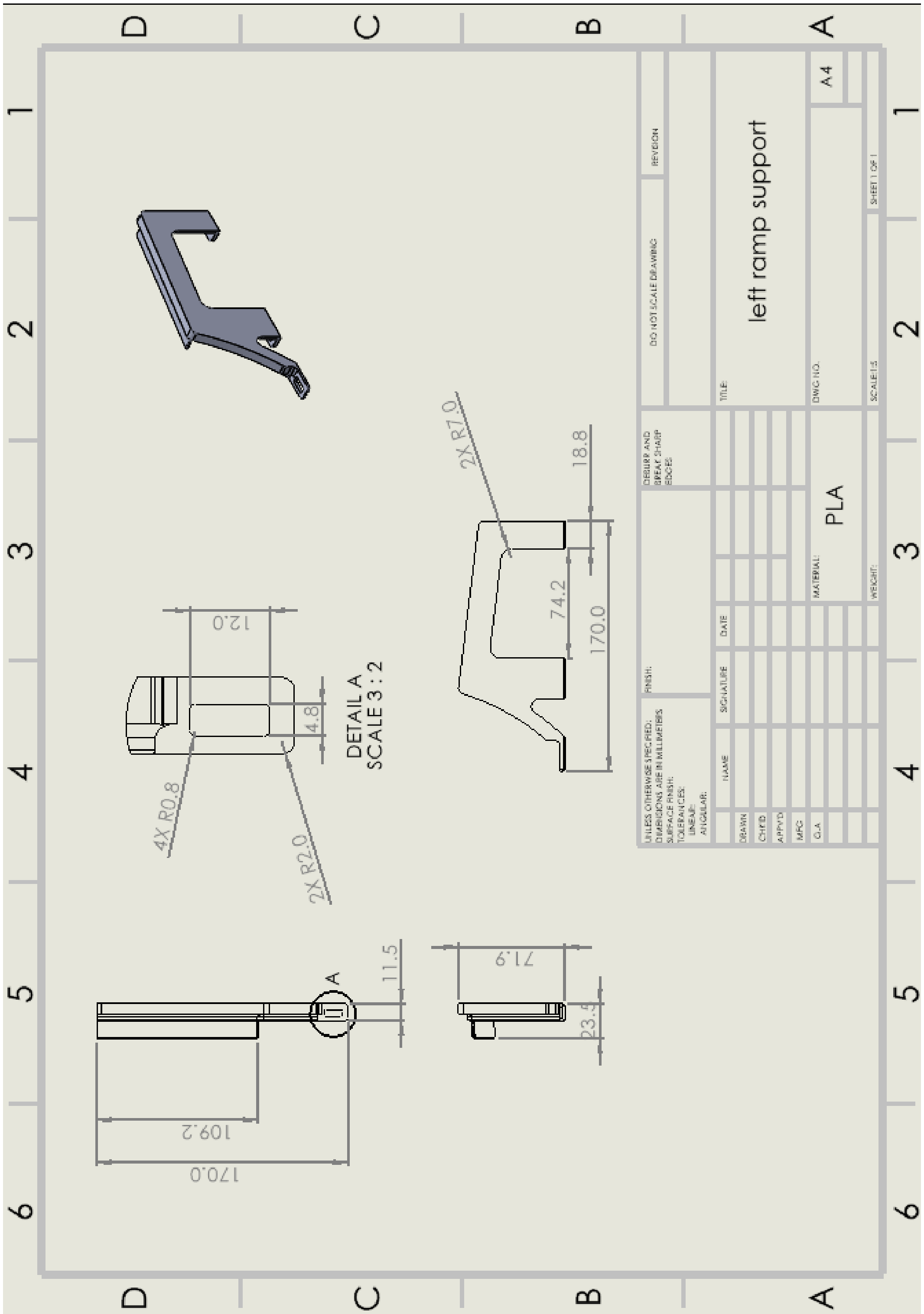


Figure 30: Left ramp support drawing

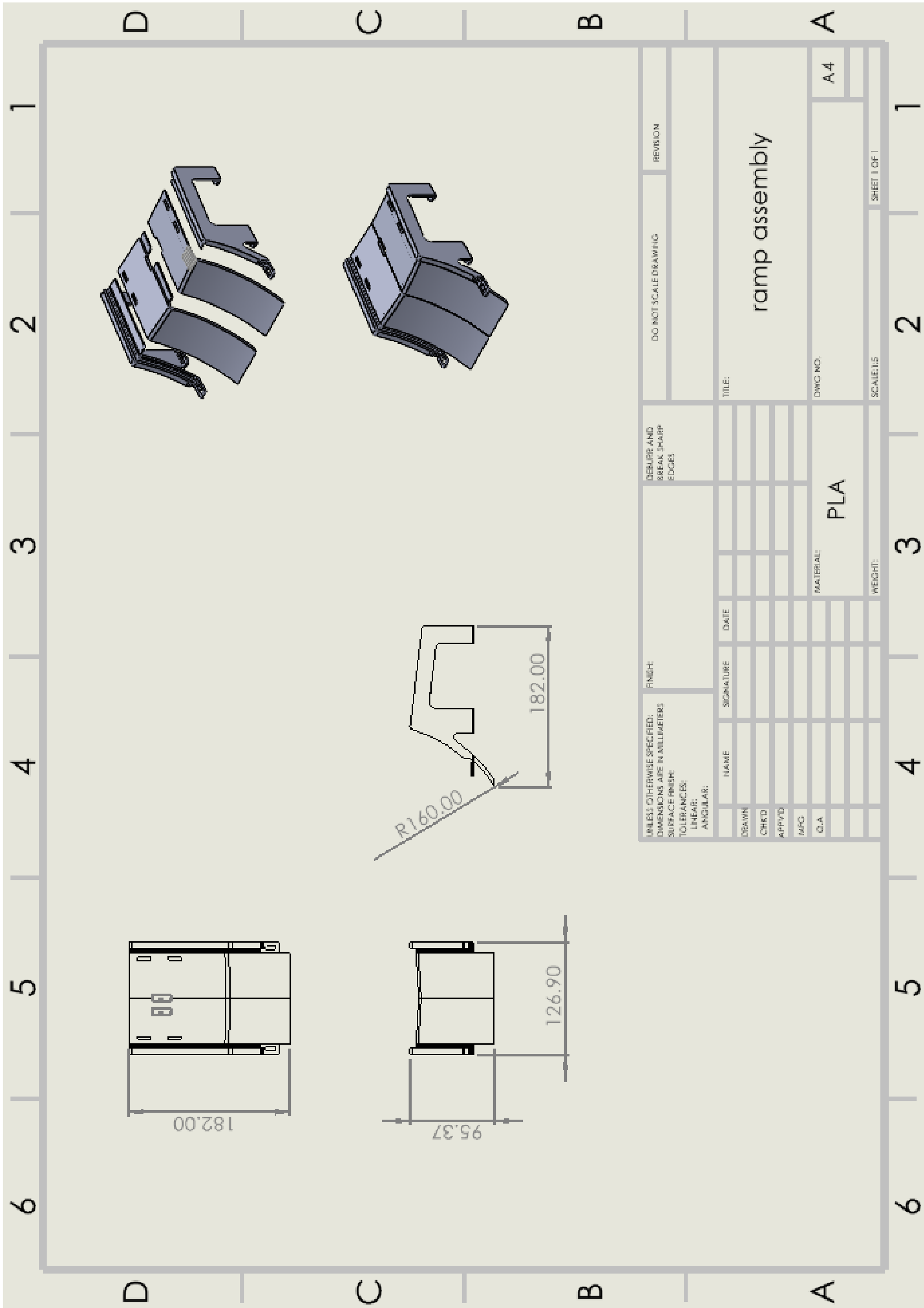


Figure 31: Ramp assembly drawing

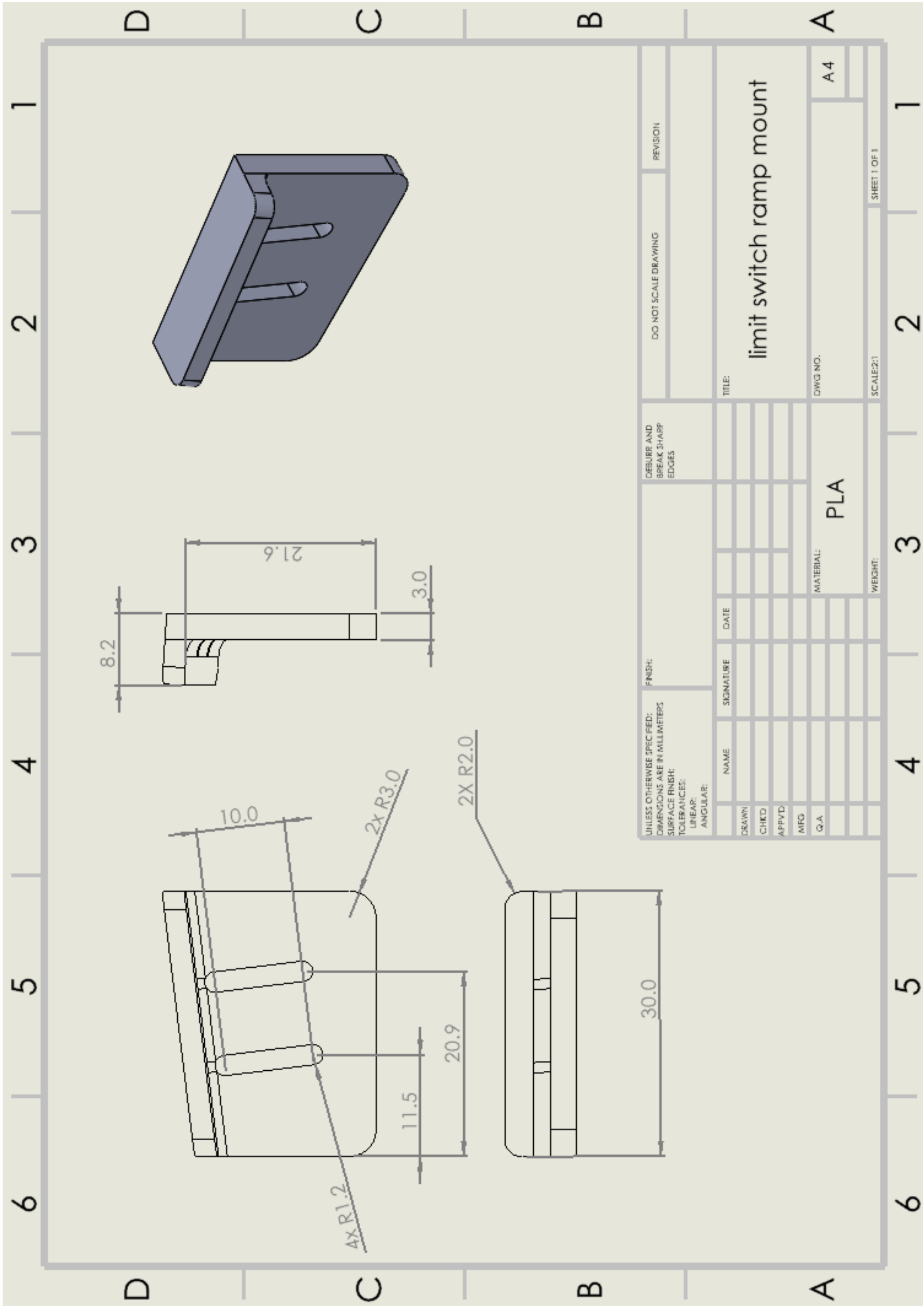


Figure 32: Limit switch ramp mount drawing

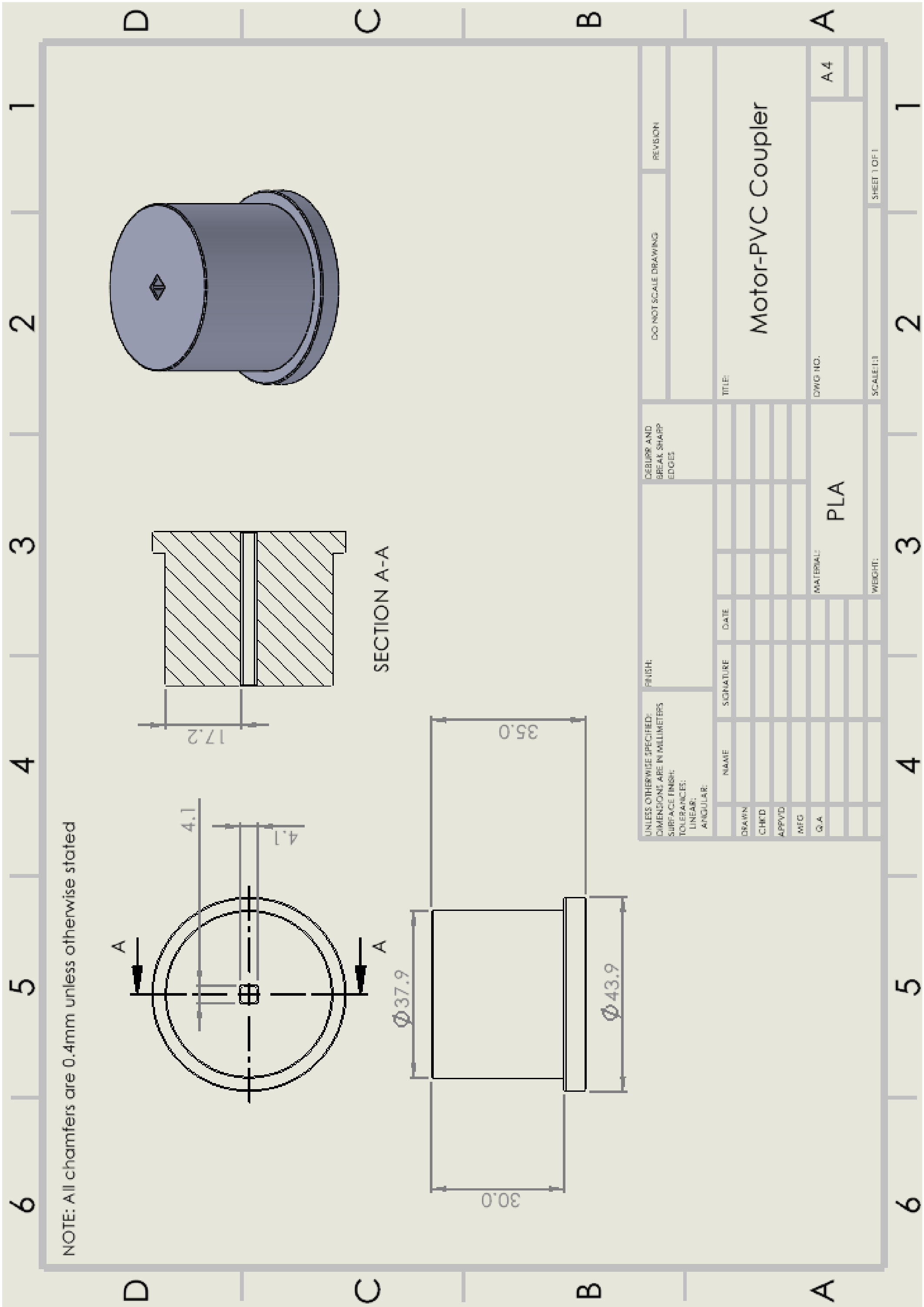


Figure 33: Motor-PVC coupler drawing

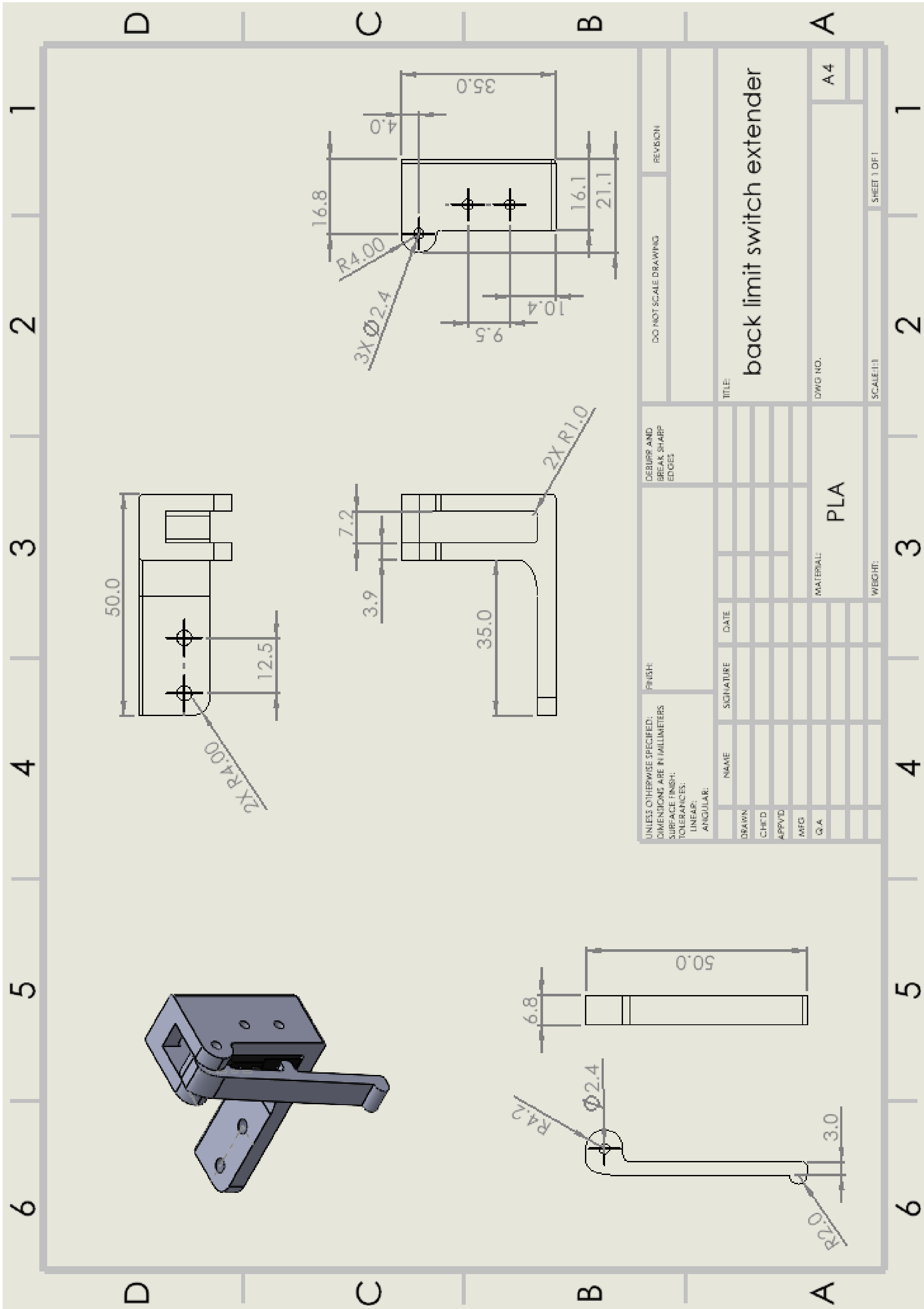


Figure 34: Limit switch extender drawing

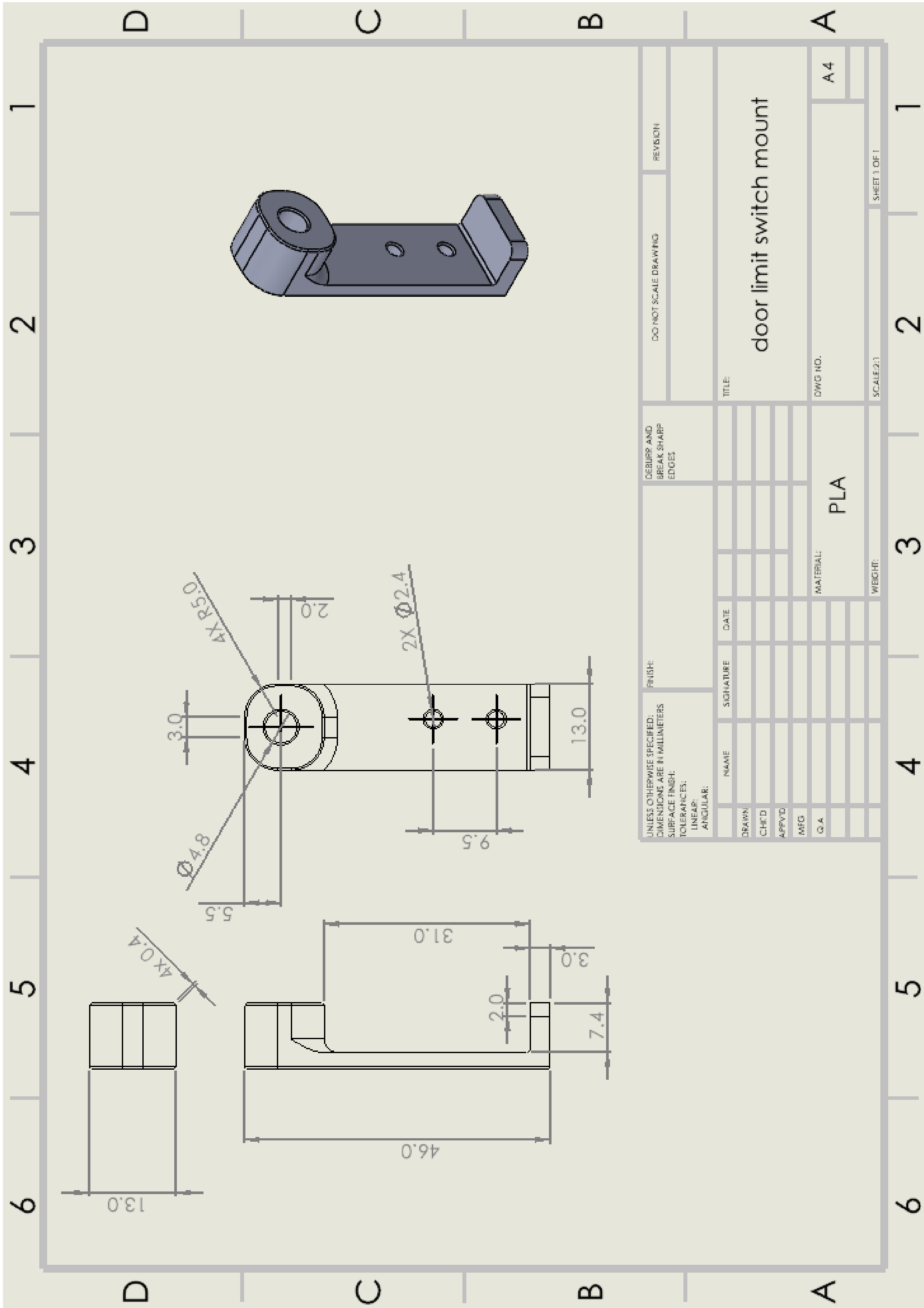


Figure 35: Back door limit switch mount drawing

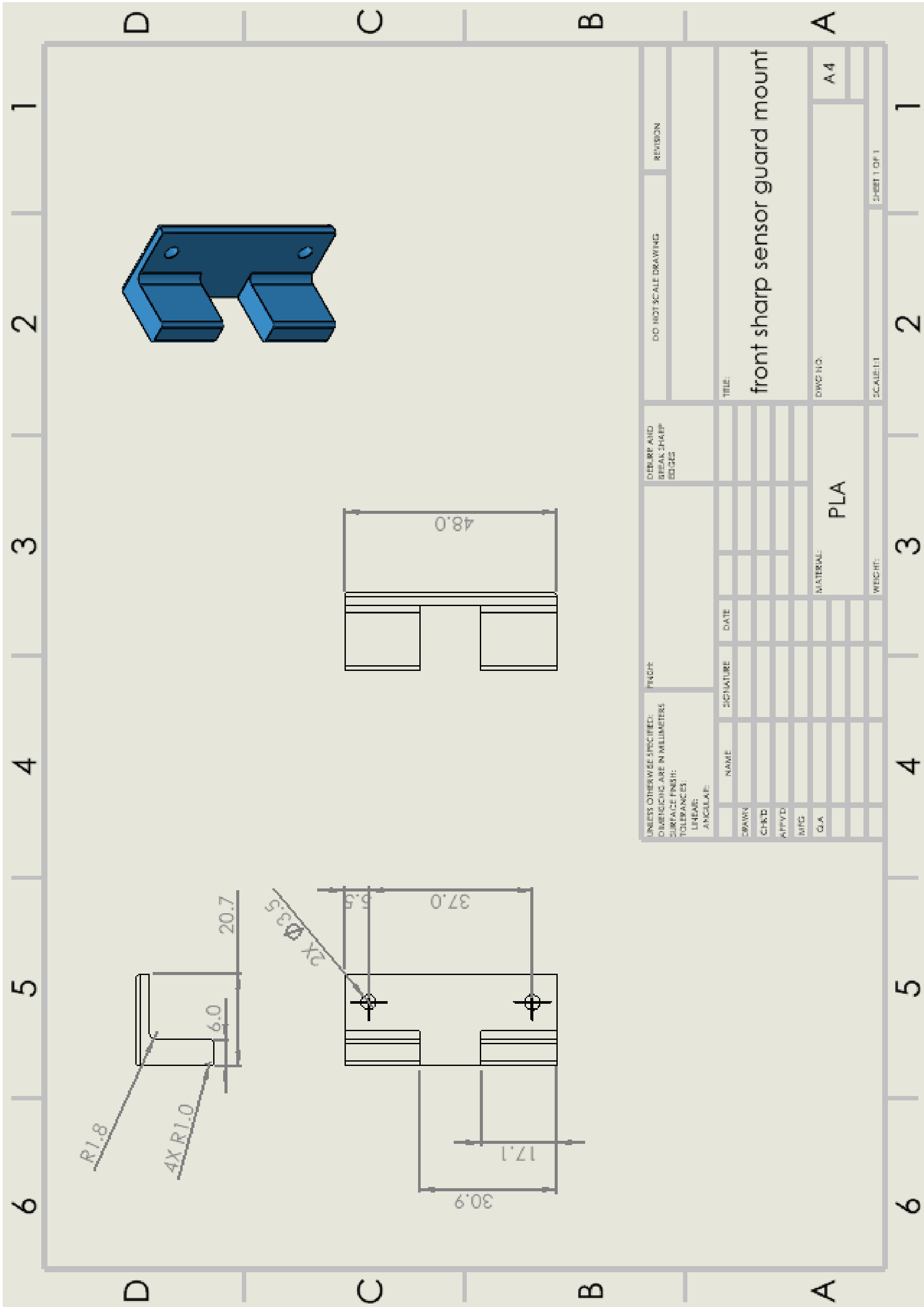


Figure 36: Sharp sensor front guard mount drawing

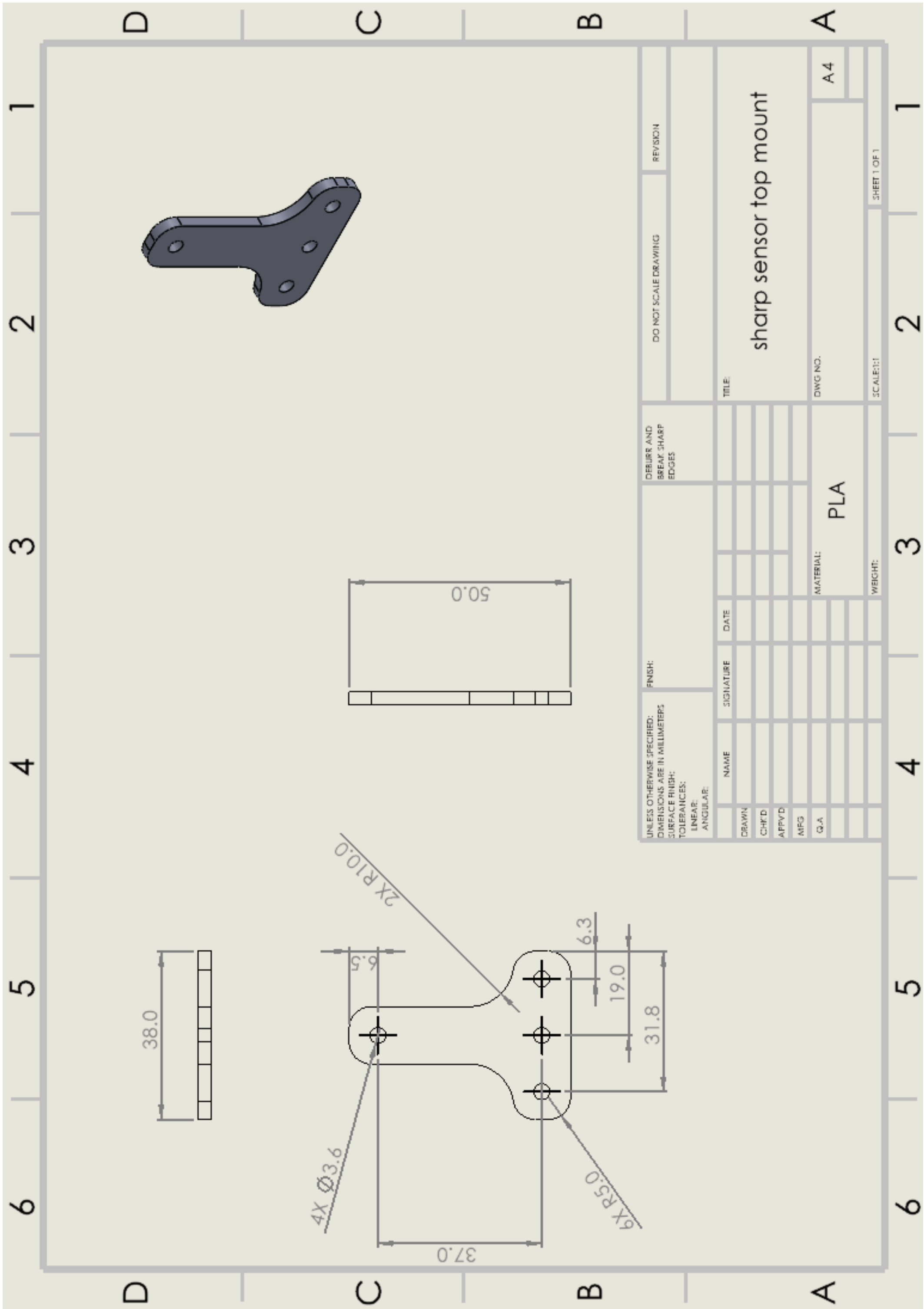


Figure 37: Sharp sensor top mount drawing

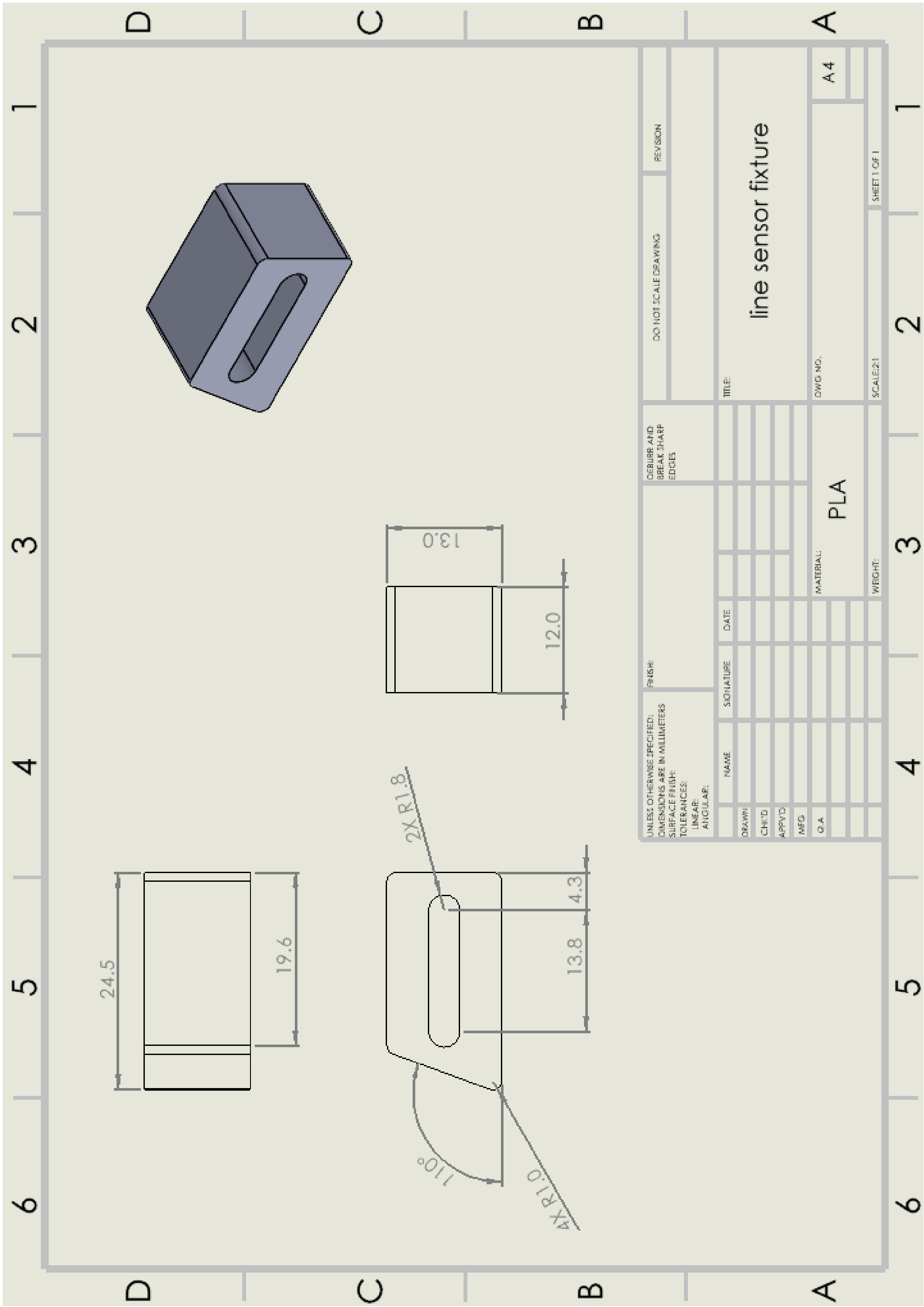


Figure 38: Line sensor fixture drawing

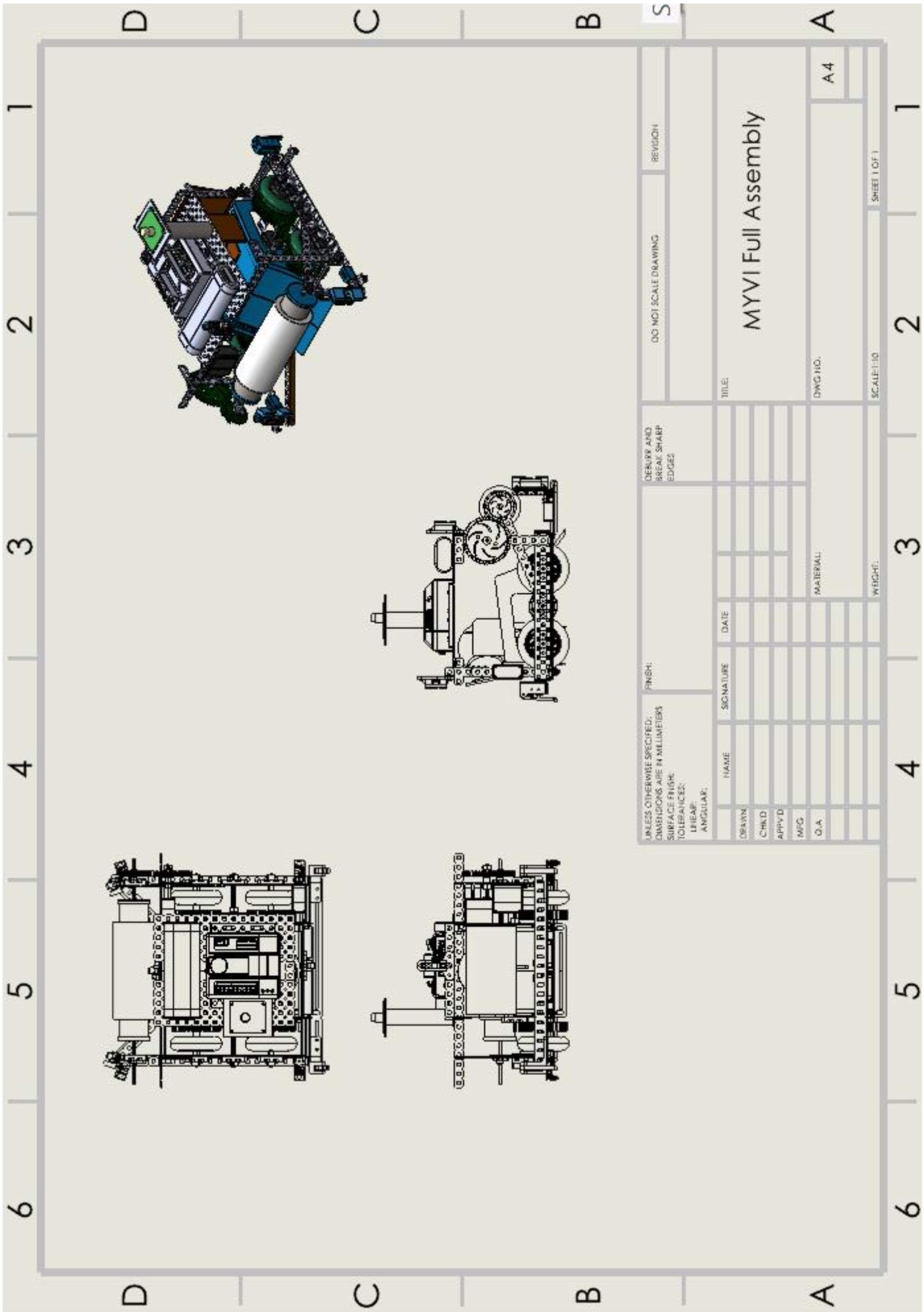


Figure 39: Full assembly drawing of MYVI

6 Electrical System

6.1 Motor selection and motor placement

Our team is given 2 types of motors for the competition, namely, 393 motors (2-wire motor) and 276-2163 (3-wire motor). By referring to the motor datasheet attached below in Appendix C3 & C4, we have extracted relevant and important motor characteristics as shown in the table below.

Quantity	Motor Type	Stall torque (Nm)	Free Speed (rpm)
2	393 (2-wire motor)	1.68	100
2	276-2163 (3-wire motor)	0.73	100

Table 23: Motor technical specifications

With the data points given at 2 different motor operating conditions, our team decided to plot the data out. By doing so, we are able to visualise the data better and aid in our motor selection process.

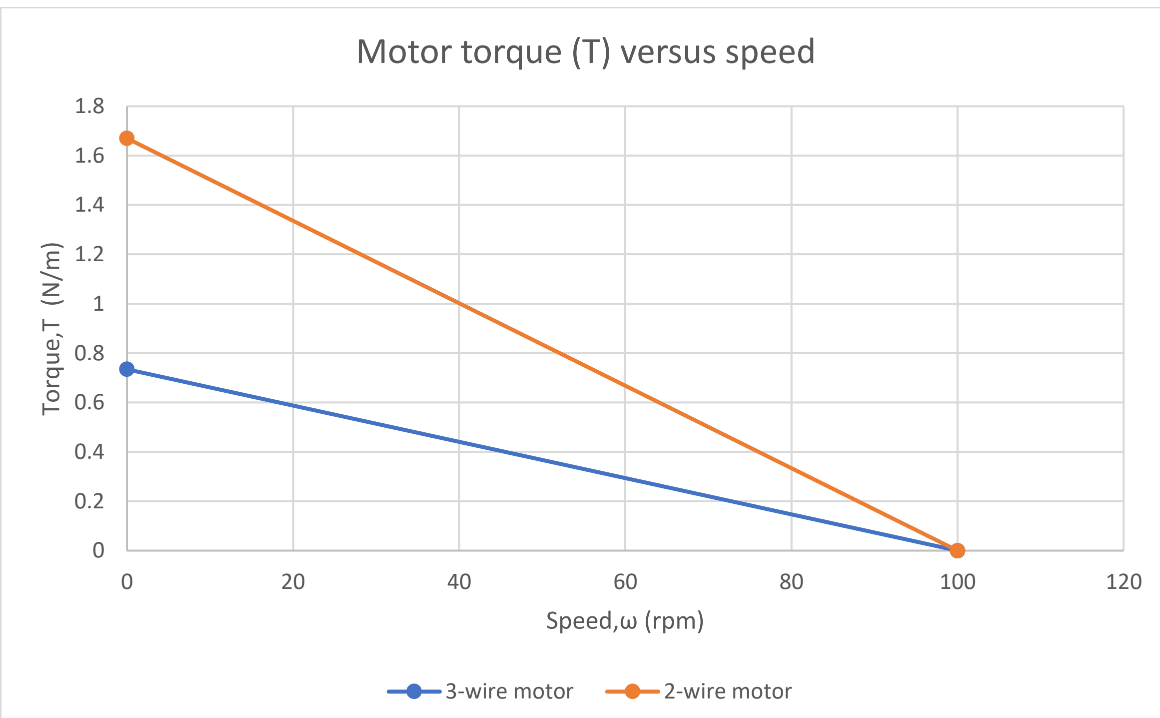


Figure 40: VEX motor torque speed characteristics

From the graph shown, 2-wire motor have more than twice the stall torque of 3-wire motor. With higher torque, the robot is able to overcome the breakaway torque when stationary. Besides, our robot will have higher acceleration at all speed, as the 2-wire motor have higher torque at all speed compared to 3-wire motor.

Thus, 3-wire motors are selected to drive the robot's left and right wheels respectively and 2-wire motor are used to drive the collector and dispenser mechanism respectively. This is because the collector and dispenser does not require high torque when it is operated.

6.2 Sharp Sensor

6.2.1 Sharp sensor calibration

In order for the robot to collect the tennis ball autonomously, the robot must be able to sense its surrounding and detect the presence of the ball before it is able to collect the tennis ball effectively. This is analogous to the sensory organ in human beings such as our eyes and ears.

For this project, we used distance sensor and Infrared line sensor to detect our surroundings. As the robot will only operate in the arena, the surrounding is relatively easier to navigate where the Sharp distance sensor and line sensor will be sufficient to differentiate between the boundary line, tennis ball and opponent vehicle. In contrast, if we were to use camera system for this task, an edge computing device with significantly higher compute power, such as jetson nano or jetson orin nano will be required to deploy deep learning object detection models to identify the boundaries, opponent and tennis ball.

Sharp sensor

For this project, we used 3 long range Infrared (IR) sensor and 1 short range IR sensor. By referring to the Sharp sensor datasheet attached below in Appendix C1 & C2, we have summarised the relevant technical specification as shown in the table below.

Quantity	Type of sensor	Optimal measuring distance (cm)
3	Short range IR sensor (GP2Y0A21SK0F)	10 - 80
1	Short range IR sensor (GP2Y0A41SK0F)	4 - 30

Figure 41: Sharp sensor specifications

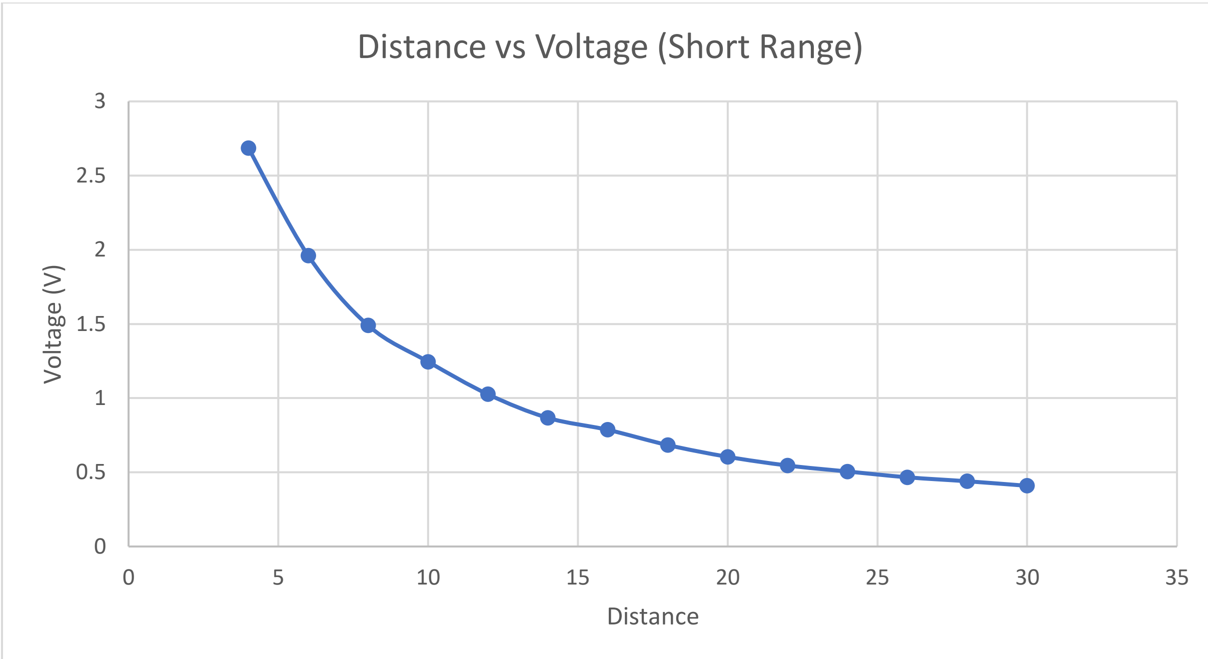


Figure 42: Short distance sharp sensor calibration

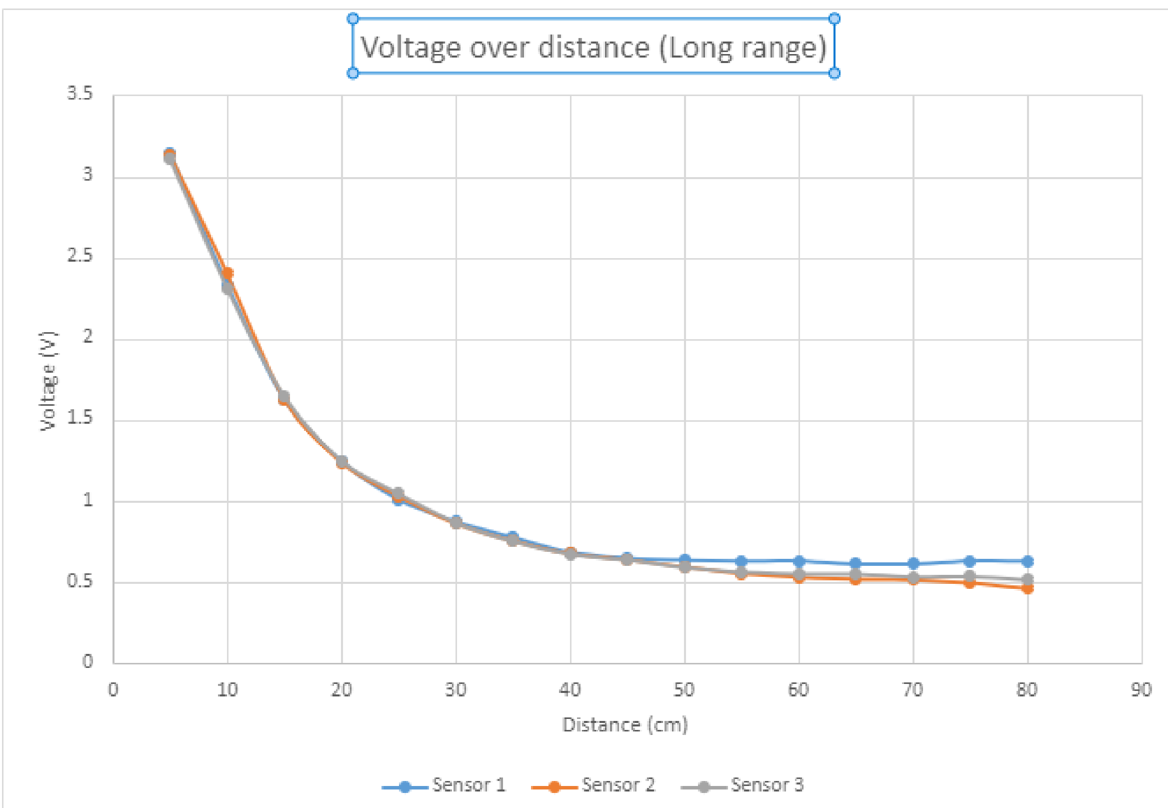


Figure 43: Long distance sharp sensor calibration

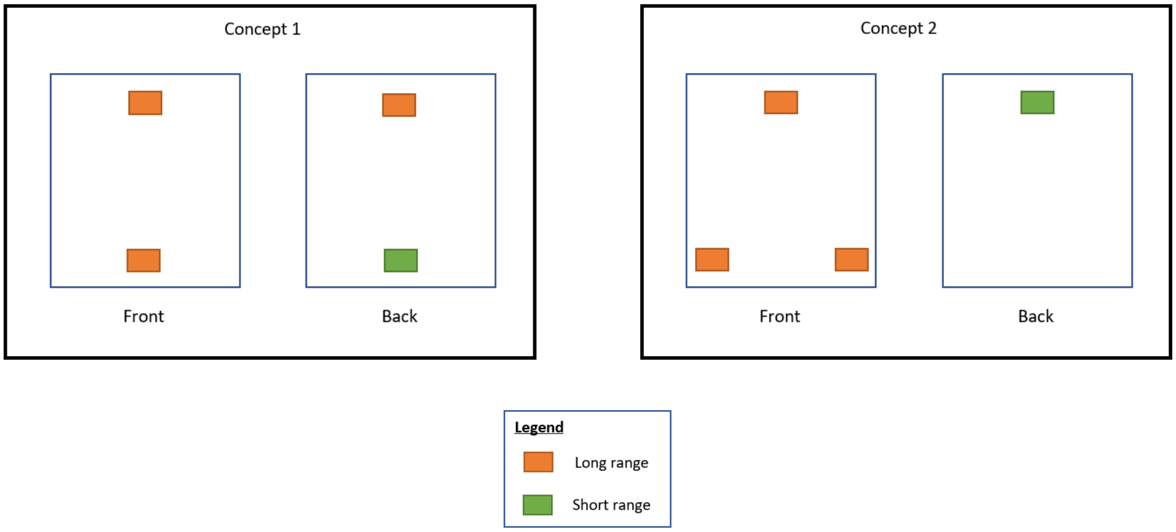
We have calibrated both the long range and short range and it matches the values given by the manufacturer in the datasheet in Appendix C1 and C2.

Initially, we would like to vary the time of the robot moving forward based on the distance of the tennis ball. However, we found that the sensor output values vary every time when we tested it using a tennis ball, as opposed to the values only calibrated using a piece of flat white A4 paper for the graph above. Therefore, we only set an arbitrary value for the positive detection of the tennis ball.

After testing in the arena, we found that the long-range sensor is more accurate and responsive when the distance is < 50cm when tested using the tennis ball. This corresponds to the sensor output value of around 800.

For opponent detection, we found that the sensor keeps fluctuating around 700 and 800 in the arena even without the presence of an opponent. Thus, we choose a value of 1100 for positive opponent detection after testing the sensor using opponent's vehicle.

6.2.2 Sharp Sensor Placement



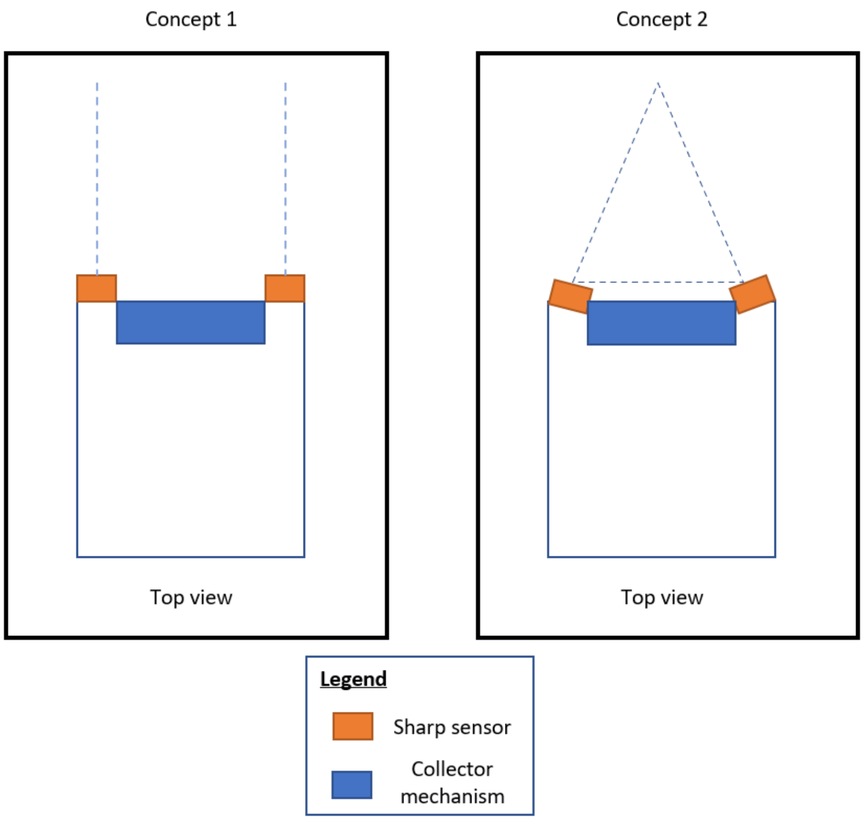
Concept 1	
Pros	Cons
Equal number of sensors in front and back	Front bottom sensor blocking ball collection mechanism
Able to detect tennis ball both in front and back	Back bottom sensor blocking the ball dispensing mechanism
Front bottom sensor is aligned with the ball collection mechanism	

Concept 2	
Pros	Cons
Larger searching area in the front of robot	Unable to search for balls at the back
Will not block the ball collection mechanism	Sensor not aligned with the ball collection mechanism
Will not block the ball dispensing mechanism	
Front bottom sensor is aligned with the ball collection mechanism	

Table 24: Sharp sensor placement analysis

After considering the physical limits where the sensor will interfere with the pathway of the tennis ball, our team eliminated the first concept and selected **Concept 2**. Although the sensor is not aligned with the ball collection mechanism, this problem can still be solved as detailed in the section below.

6.2.3 Sharp Sensor Angle

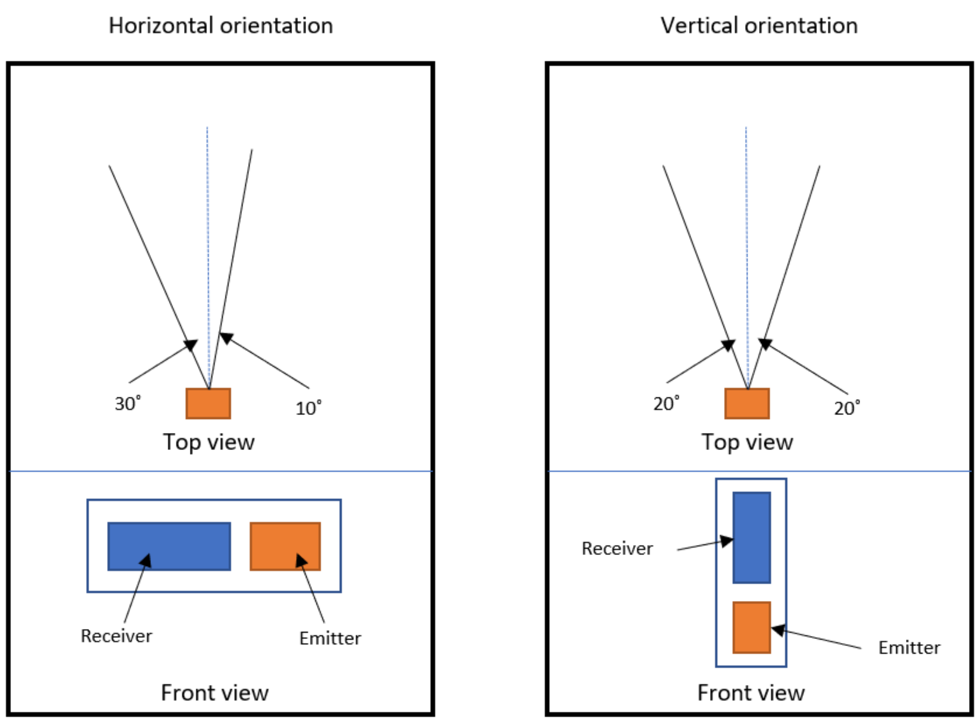


Concept 1 (Straight)	Concept 2 (Angled)
Bigger scanning area at given distance (rectangular)	Smaller scanning area at given distance (triangular shape)
Ball detection region not aligned with collector	Optimal ball detection region (any point on the line), directly in front of the collector
Robot need rotate, or else the sensor will collide with the ball	Very high redundancy
Need to rotate before the robot is able to collect the ball	
Require high accuracy of measured distance, for rotation angle calculation	
Require high accuracy of wheel encoder, for robot rotation control	

Table 25: Sharp sensor angle analysis

Thus, **Concept 2** is selected for the front Sharp sensor placement as the implementation is more flexible and have higher redundancy in collecting the tennis ball.

6.2.4 Sharp Sensor Orientation



Horizontal Orientation	Vertical Orientation
Can be attached firmly to the horizontal bar	Attached to the frame via a 90˚ aluminium bar
More resistant to collision	Less resistant to collision
Skewed detection angle, more on the receiver side	Equal detection angle from the top view
Harder to change the angle of orientation (z-axis)	Easier to change the angle of orientation (z-axis)

Table 26: Sharp sensor orientation analysis

The **vertical orientation** is selected as suggested in the data sheet in Appendix C2, where moving direction and the sharp sensor should be perpendicular. As it is less resistant to collision from opponent, we have designed a sharp sensor casing for it where the opponent will crash into the casing instead of the sensor. This can help to maintain the accuracy of the sharp sensor throughout the competition.

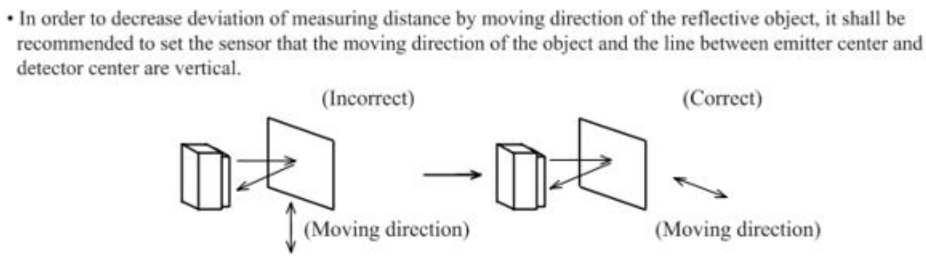


Figure 44: Sharp sensor orientation datasheet guide

<u>Final design of the sharp distance sensor</u>

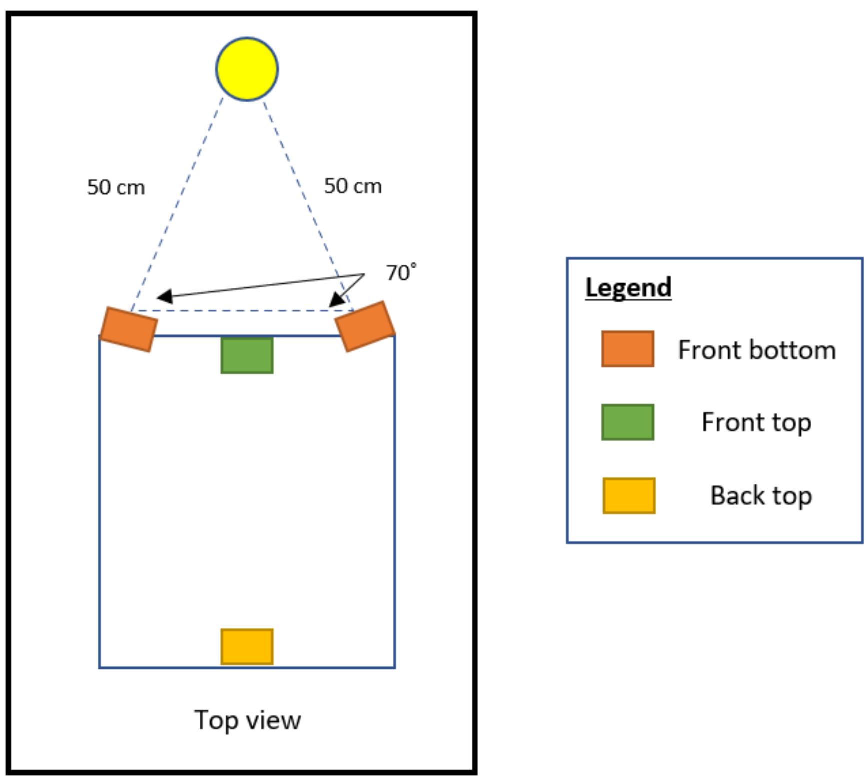


Figure 45: Sharp sensor final design

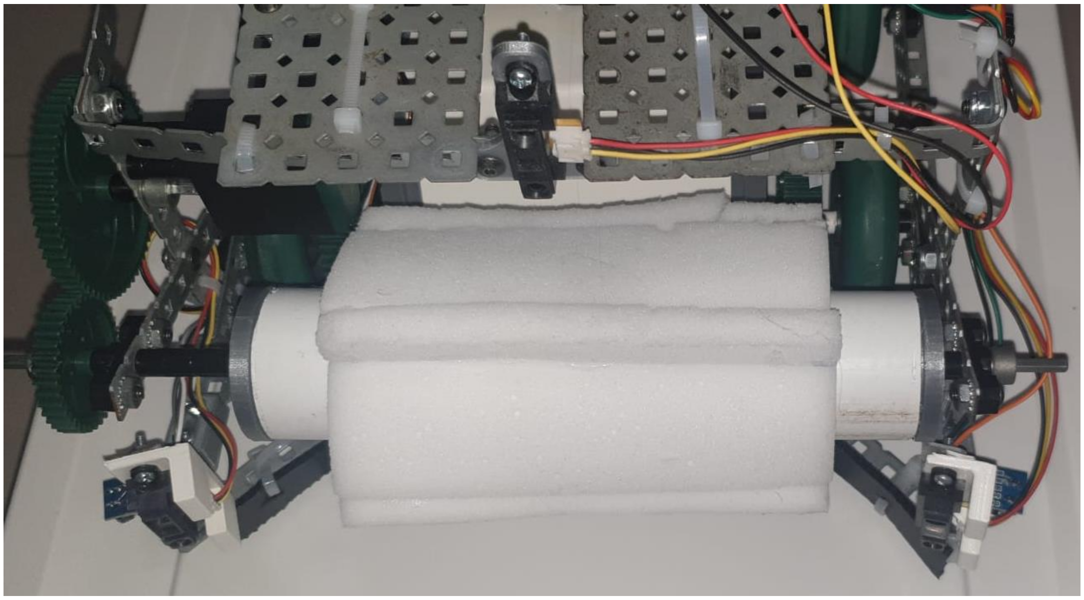


Figure 46: Front Sharp Sensor placement

Initially our team used just a screw to fix the orientation of the sharp sensor. However, after testing in the arena, we found out that the orientation of the sharp sensor will be altered after collision. Besides, resetting the orientation of the sharp sensor is very troublesome as well. Therefore, our team 3d printed the sharp sensor casing and a block to fixed the rotation of the sharp sensor of exactly 70˚.

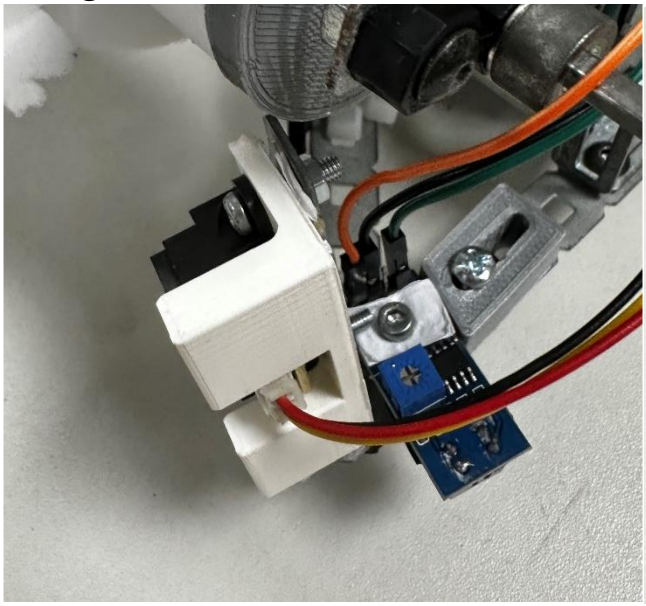


Figure 47: Sharp and line sensor side view

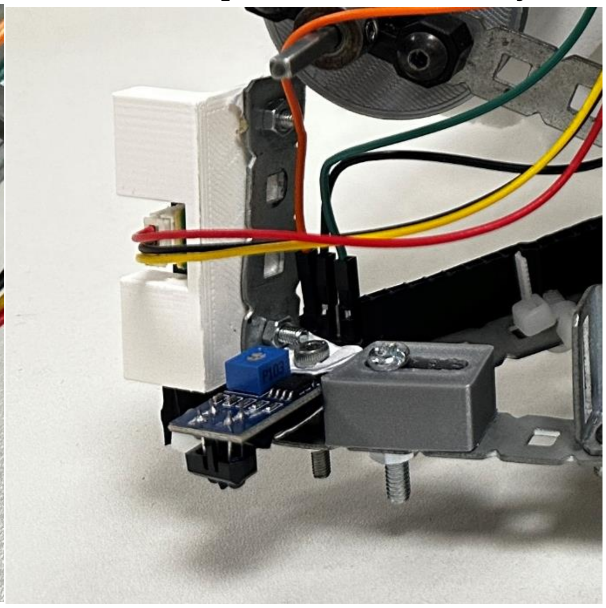


Figure 48: Sharp and line sensor top view

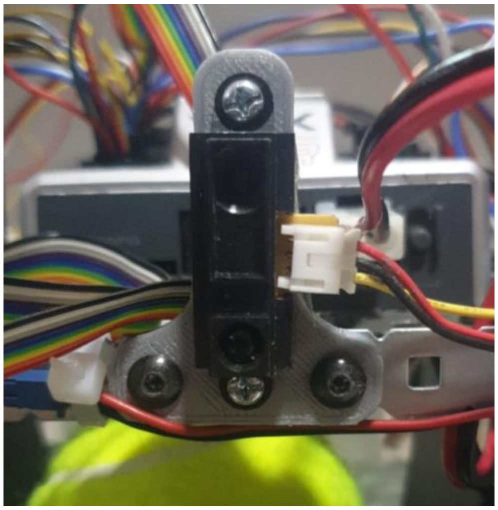


Figure 49: Back top sharp sensor mounting

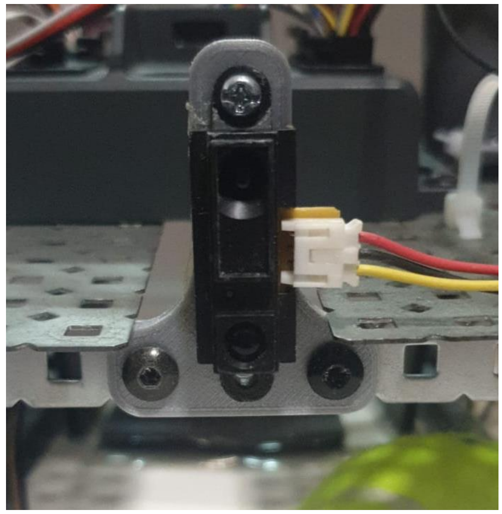


Figure 50: Front top sharp sensor mounting

Our team just 3d printed a mounting for the front and back top sharp sensors instead of a casing as it is very unlikely for the sensor to be affected or distorted in the event of a collision.

6.3 Limit switches

6.3.1 Tennis ball detector

Figure 51: Tennis ball detector in temporary storage compartment

A limit switch is placed in the middle of the ball collection compartment to detect the presence of the tennis ball in order to stop the ball collecting process and the robot will start to go back to the origin.

6.3.2 Home Detector

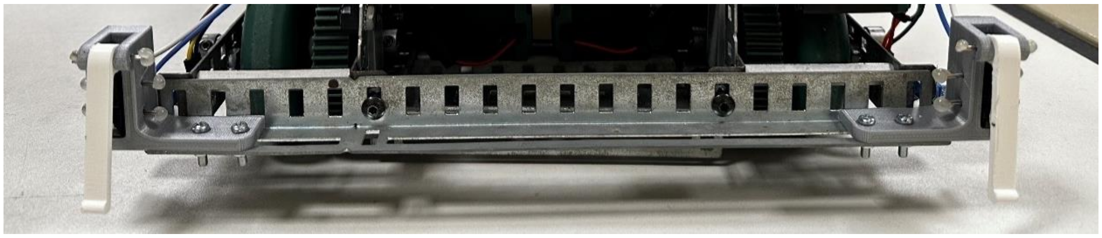
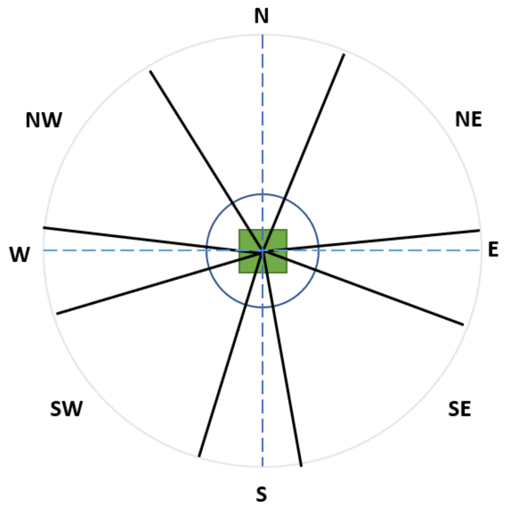


Figure 52: Ball collection area limit switch

Two limit switches are placed on the back left and right respectively to ensure that the back of the robot is parallel to the delivery area before the ball is dispensed. The limit switch is also extended to increase the sensitivity of the limit switch.

6.4 Compass



Direction	Angle
N	338 – 16
NE	16 – 82
E	82 – 108
SE	108 – 179
S	179 – 208
SW	208 – 256
W	256 – 275
NW	275 – 338

Figure 53: Digital compass calibration

After calibration, our team find out that the directional out put from the compass is quite accurate, and therefore can be used to implement some critical functions such as spin search and back to origin.

To ensure the accuracy of the compass, we lifted it as high as possible while conforming to the height restriction of 30 cm in order so the compass to stay as far as possible from other metallic parts. Therefore, we avoid using metallic components by using PVC pipe and corrugated board as the holder.

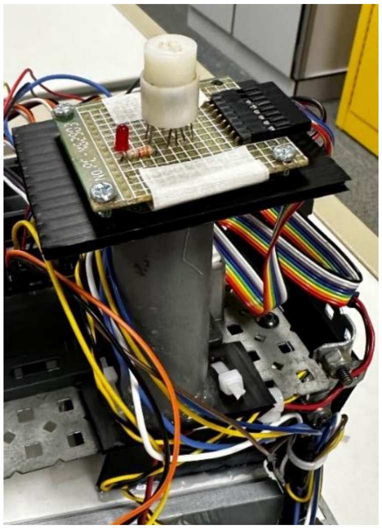


Figure 54: Digital compass placement

6.5 Line Sensor

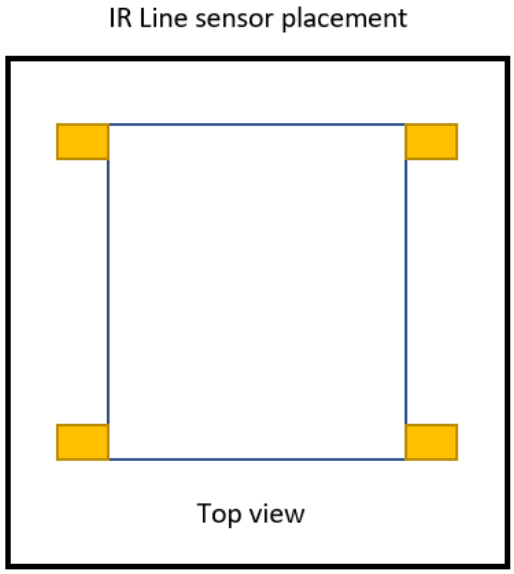


Figure 55: Line sensor placement design

In order to prevent collision of the main body and especially the sharp sensor with the boundary and the barrier at the ball delivery area, line sensors are placed at the 4 corners of the robot.

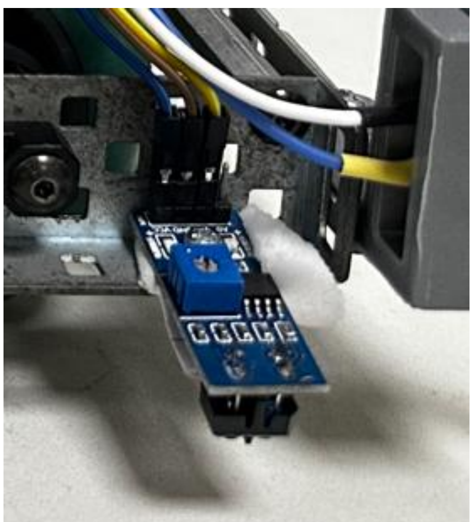
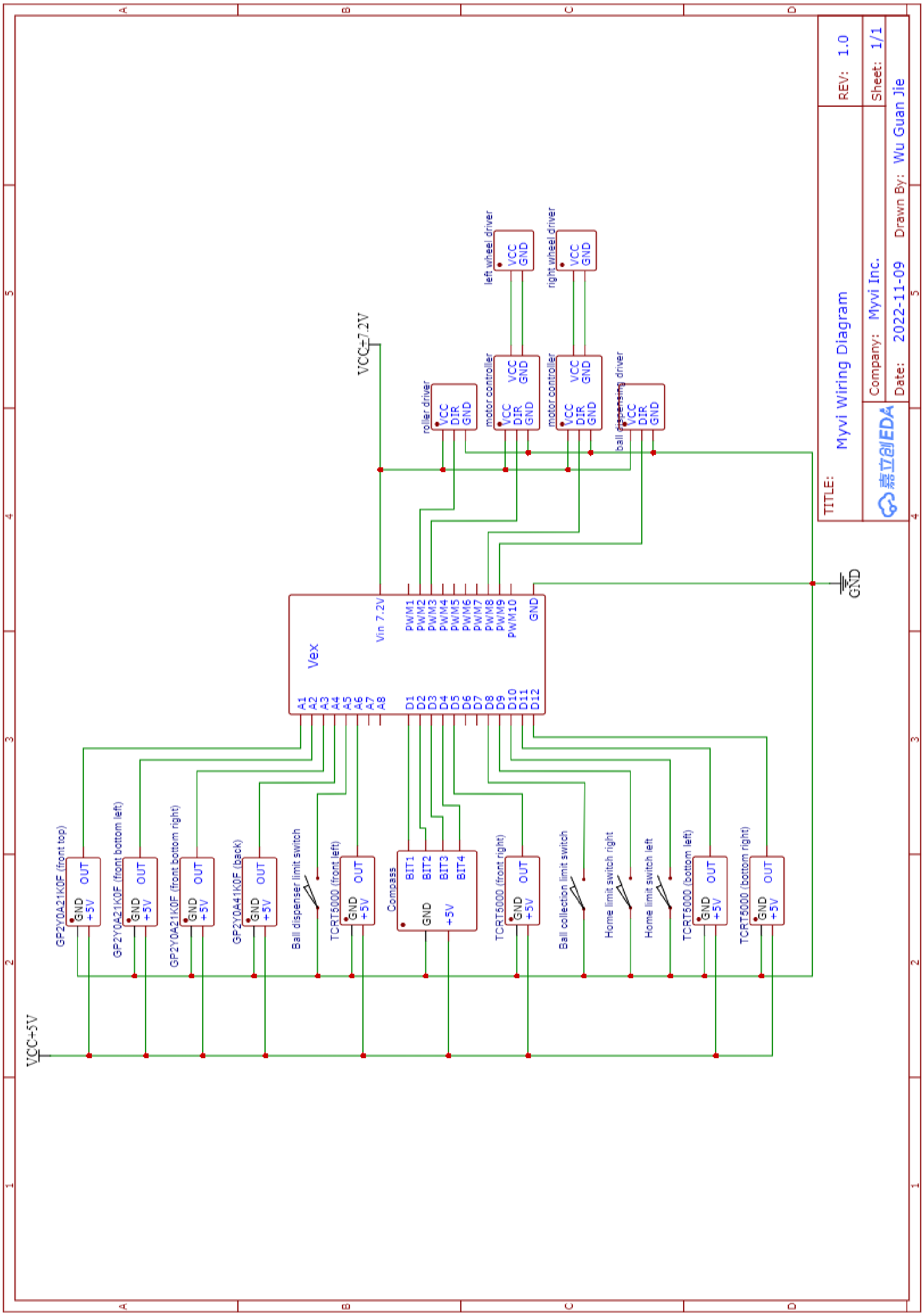


Figure 56: Line sensor mounted on spacer

As we are unable to find its datasheet, we find out that the line sensor will work the best in the range of 1cm to 10 cm after calibration. Thus, we have to increase the height of the line sensor by placing a spacer between the line sensor and the frame.

6.6 Wiring Diagram



7 Programming

7.1 Structure

In our program, we broke down our strategy and identify key individual functions needed to accomplish the task given. These functions are:

a. starting_procedure()
b. spin_search()
c. move_forward()
d. collect_ball()
e. dispense_ball()
f. avoid_boundaries(), and
g. avoid_opponent()

These functions take care of the high-level logic and does not interact directly with the hardware. Instead, we defined a separate set of hardware interfaces to control the motors and read the sensors. This approach of separation between control logic and hardware interfaces increases the readability and reusability of our code base.

The main function controls the entire logic flow of the program. It uses the functions mentioned earlier to perform actions according to a pre-determined logic sequence. After execution, each of the functions returns a value which indicates whether the operation is a success. Based on this return value, the main function will then decide the next logic function to execute.

7.2 Strategy Planning

1) Initial Strategy

From the past competition video, the tennis ball will be placed at the top 1/3 area of the arena, far from the starting procedure of our robot. Regardless if our robot is placed on the right side or the left side of the arena, our robot must first go forward as fast as possible in order to collect the ball before our opponent.

By doing so, the tennis ball will be in the detection range of the front sharp sensor as soon as possible. If the robot detects any tennis ball while moving forward in the starting procedure, the robot will activate the roller to collect the ball. Else, the robot would activate its spin search function

2) Search Strategy

After the robot reach the targeted area of the arena, the robot will spin 360˚ to search for the tennis ball. If the ball is not found within a 50cm radius in the vicinity of the robot, it will move forward and continue searching for tennis ball in another different area where it did not search before this.

3) Opponent strategy

i) Active defence

Our team decided that it is best to avoid the opponent when encountering one instead of just colliding them out of the arena. This is because there is a high chance that our sensors might malfunction after collision and the sensor output will deviate from our calibrated values.

ii) Passive defence

However, we can only avoid opponents colliding from the front and the back of the robot. Thus, after discussing with the hardware team, we decide to lower our center of gravity and choose rubber tires at all 4 sides to increase the stability of the robot when side collision is unavoidable.

4) Mapping strategy

After observing from the past competition, we realised that it is almost impossible to map the localise the robot and the opponent in the arena given the limitation of our sensors. This is because, even if we were to track the rotation of each wheel by using the encoder, a collision with the opponent, which have a high probability of occurrence will cause the wheels to slip in any direction. Besides, the starting position of the robot cannot be predetermined or determined when the robot is in the arena. This will certainly add another layer of complexity in the mapping and localization process of the robot.

Lastly, we are not convinced of the benefits of mapping using the encoder provided due to the uncertainty of events and the uncertainty of sensor value. Thus, we employ a rule-based strategy for the motion planning of our robot.

7.3 Pseudocode

```
void main()
{
    starting_procedure();
    while (1)
    {
        if (ball_found)
        {
            if (ball_collected)
            {
                back_to_origin();
                dispense_ball();
            }
            else if (!ball_collected){
                collect_ball();
            }
        }
        else if (!ball_found)
        {
            spin_search();
            if (!ball_found)
            {
                moving_forward();
            }
        }
    }
}
```

Figure 57: Pseudocode of the Main Logic

7.4 Behaviours

In this section, we will be talking about the behaviour of our robot in the competition. It will be explained by discussing how the basic functions and modules works together in our program.

7.4.1 Basic Functions

The basic functions provide another layer of abstraction when interfacing with the hardware. While some of these functions are one-liners and do not necessarily simplify our code, it improves the context and readability of the code for easy maintenance in the future.

```
void control_motor(int left_speed, int right_speed)
```

Controls the speed of the left and right driving train of our vehicle.

Input:

left_speed: Speed of the left driving train, accepts value from -127 to 127

right_speed: Speed of the left driving train, accepts value from -127 to 127

```
void read_compass()
```

Sets the compass status to the current reading from Digital compass, uses the Orientation enum below.

```
void scan_boundary()
```

Updates the current state of line sensor reading to check if the vehicle is over the boundary in any directions, uses the BoundarySide enum below.

```
void scan_ball()
```

Takes in input from the ultrasonic distance sensors and determines whether a ball has been detected according to our sensor configuration.

7.4.2 Enumerations

Enumerations are used in our program to represent variables that has multiple non-linear states. It helps to enhance the readability of our program.

```
enum Orientation
{
    NORTH,              // 0
    NORTH_EAST,         // 1
    EAST,               // 2
    SOUTH_EAST,         // 3
    SOUTH,              // 4
    SOUTH_WEST,         // 5
    WEST,               // 6
    NORTH_WEST,         // 7
    INVALID_COMBINATION // 8
};
```

Orientation represents the 9 possible states (including invalid combination) of the compass reading.

```
enum BoundarySide
{
    FRONT_LEFT,
    FRONT_RIGHT,
    BACK_LEFT,
    BACK_RIGHT,
    NO_BOUNDARY_DETECTED
};
```

BoundarySide represents the possible states of detected boundaries (either no boundary detected, or the direction of boundary detected)

7.4.3 Modules

Modules are high level functions that encompasses logic of individual strategies. They use the basic functions above to interact with hardware.

```
void spin_search()
```

This module is responsible for searching the ball by turning 360 degrees and calling spin_ball() continuously. If a ball is detected, the vehicle will stop spinning and update the ball_found variable. But if nothing is found, the spinning will stop after 360 degrees.

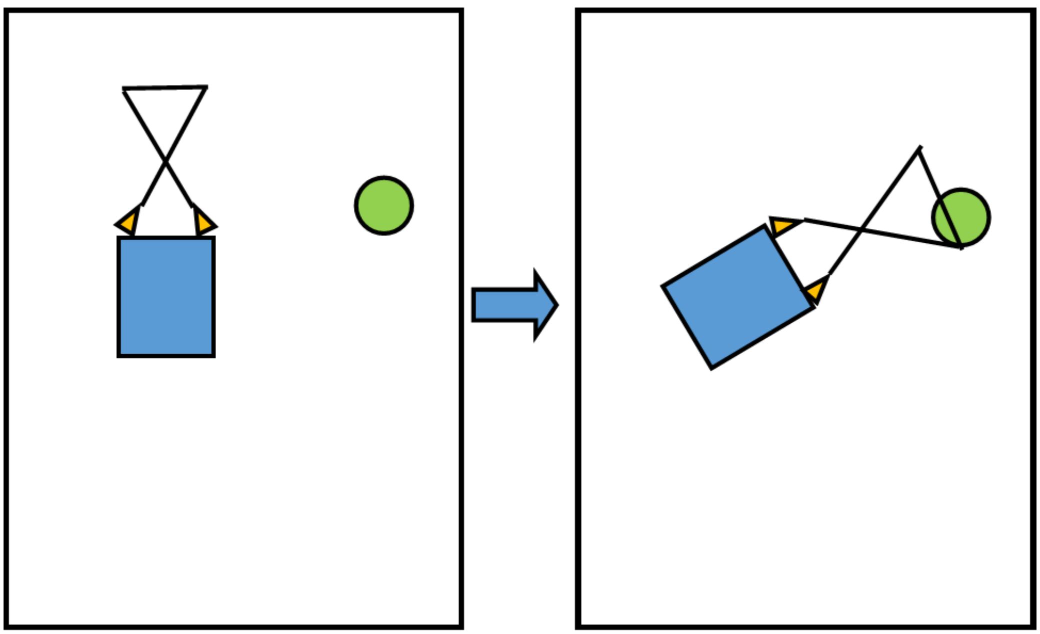


Figure 58: The vehicle spins and stop when a ball is detected.

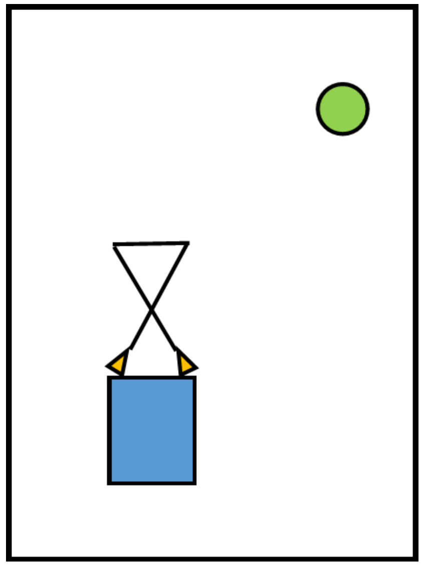


Figure 59: If the ball is too far away, the vehicle stops after spinning 360 degrees as nothing is detected.

```
void back_to_origin()
```

This module guides the vehicle back to the origin by using the on-board compass. It constantly polls the sensor and check if the current direction is correct and if the line sensors has detected any boundaries.

```
void avoid_boundaries(BoundarySide line_sensor_number)
```

Input:

line_sensor_number: Indicates which line sensor has detected a boundary.

This module is activated when one of the line sensors has detected a boundary. According to the position of the line sensor, a corresponding avoidance sequence is executed such that the vehicle distant itself and points away from the boundary.

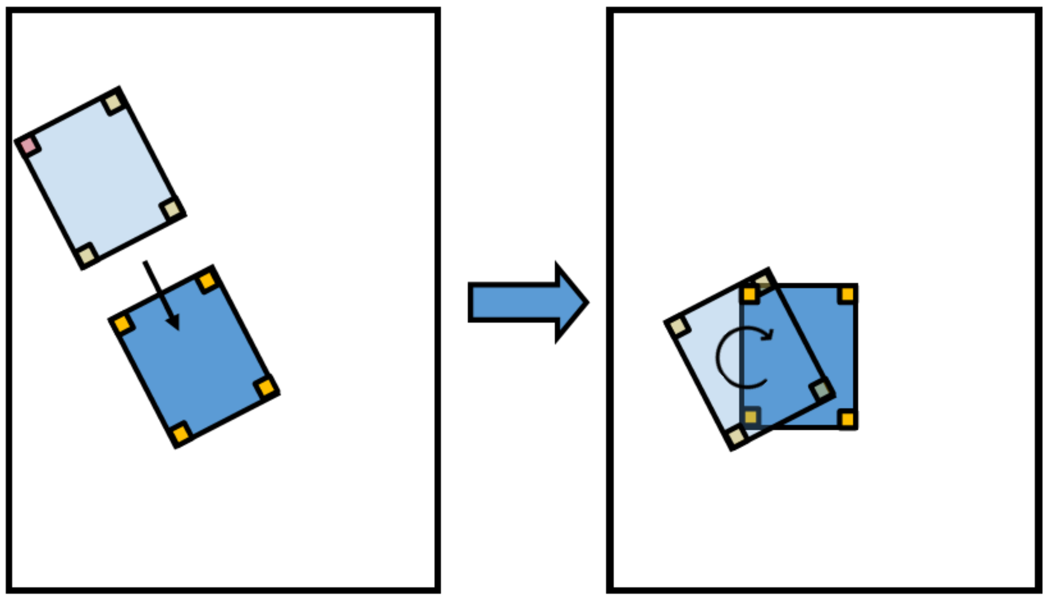


Figure 60: When boundary is detected by the front right line sensor, the vehicle first reverse then turns in the clockwise direction to point away from boundary.

```
void collect_ball()
```

This module activates after a ball has been found. It will activate the roller and move the vehicle forward for a total of 4 seconds in attempt to collect the ball. After 4 seconds, the attempt will be aborted, and the roller will be turned off to conserve electricity.

```
void avoid_opponent()
```

This module checks if another vehicle is too close in front or at the back of our vehicle. If the calculated distance is within a certain threshold, it moves our vehicle away from the direction of opponent vehicle to provide collision. Once a safe distance has been reached, an avoiding sequence will be executed.

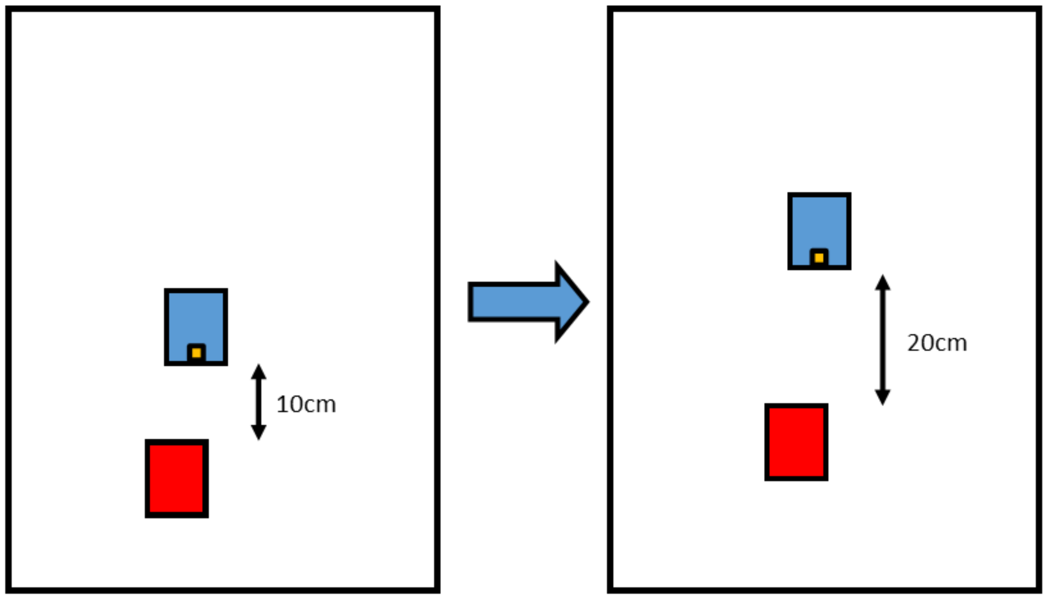


Figure 61: For Example, if the opponent vehicle is detected at behind, our vehicle will accelerate forward to pull out a safe distance and avoid collision.

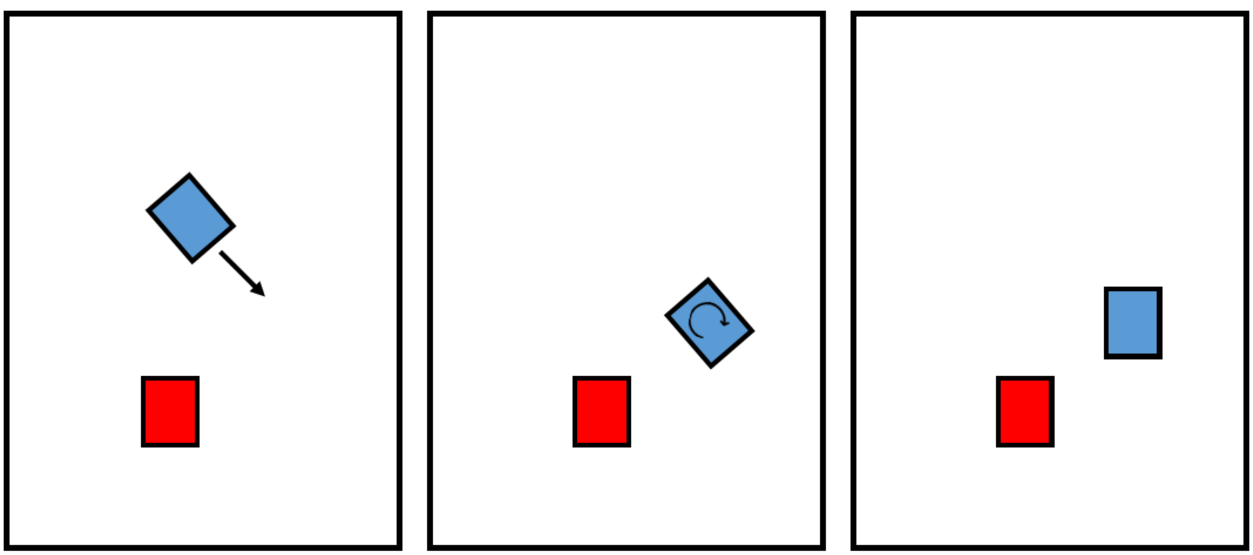


Figure 62: After a safe distance is reach, a sequence of maneuvers will be executed to avoid running into the opponent vehicle in the same direction.

```
void starting_procedure()
```

This module executes every time when our vehicle starts from the initial zone, which includes both upon start up and every time after a ball is dispensed successfully. It programs our vehicle to move for a longer distance as our analysis determines that the ball will likely land at the back part of the arena.

7.5 Flowchart

7.5.1 Main Function

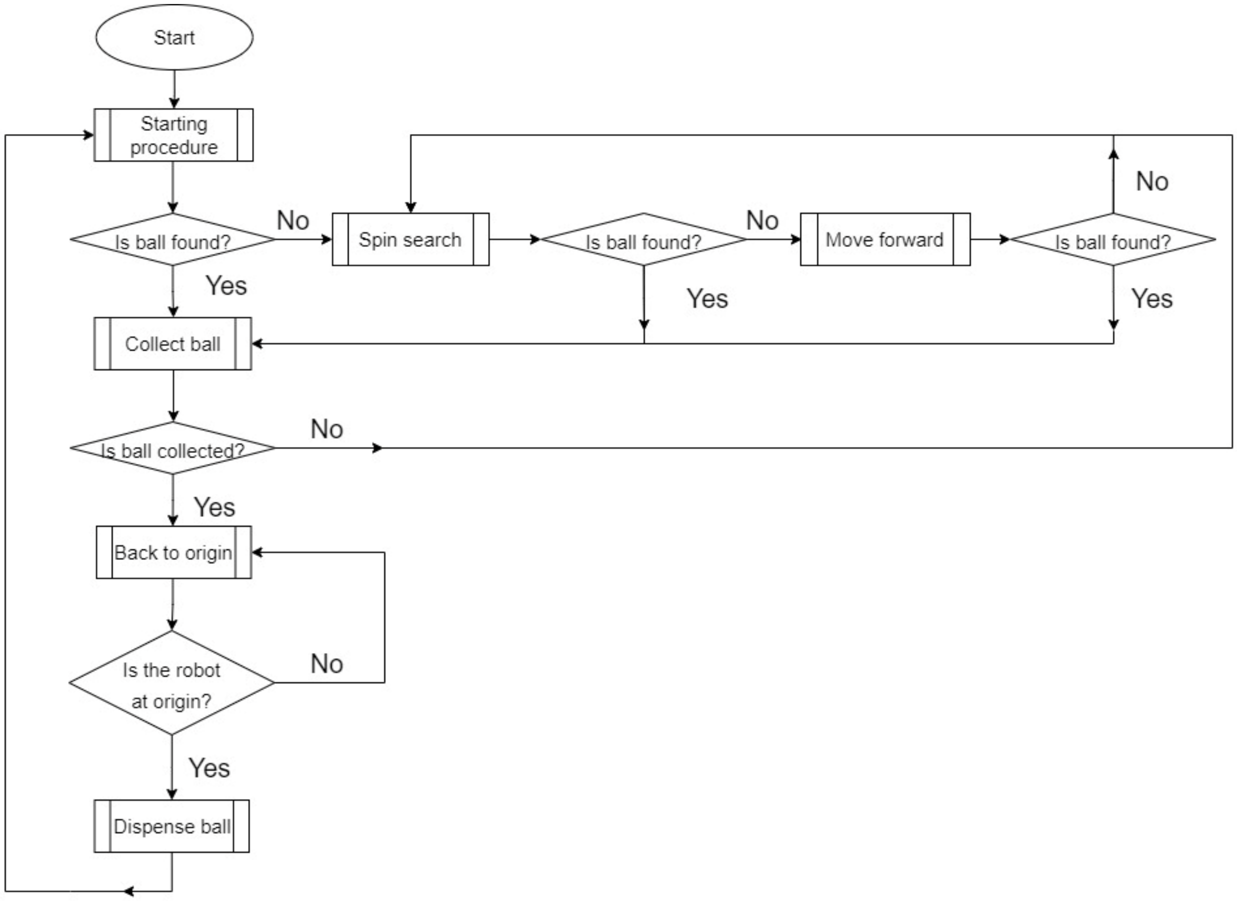


Figure 63:Main function flowchart

7.5.2 Starting Procedure

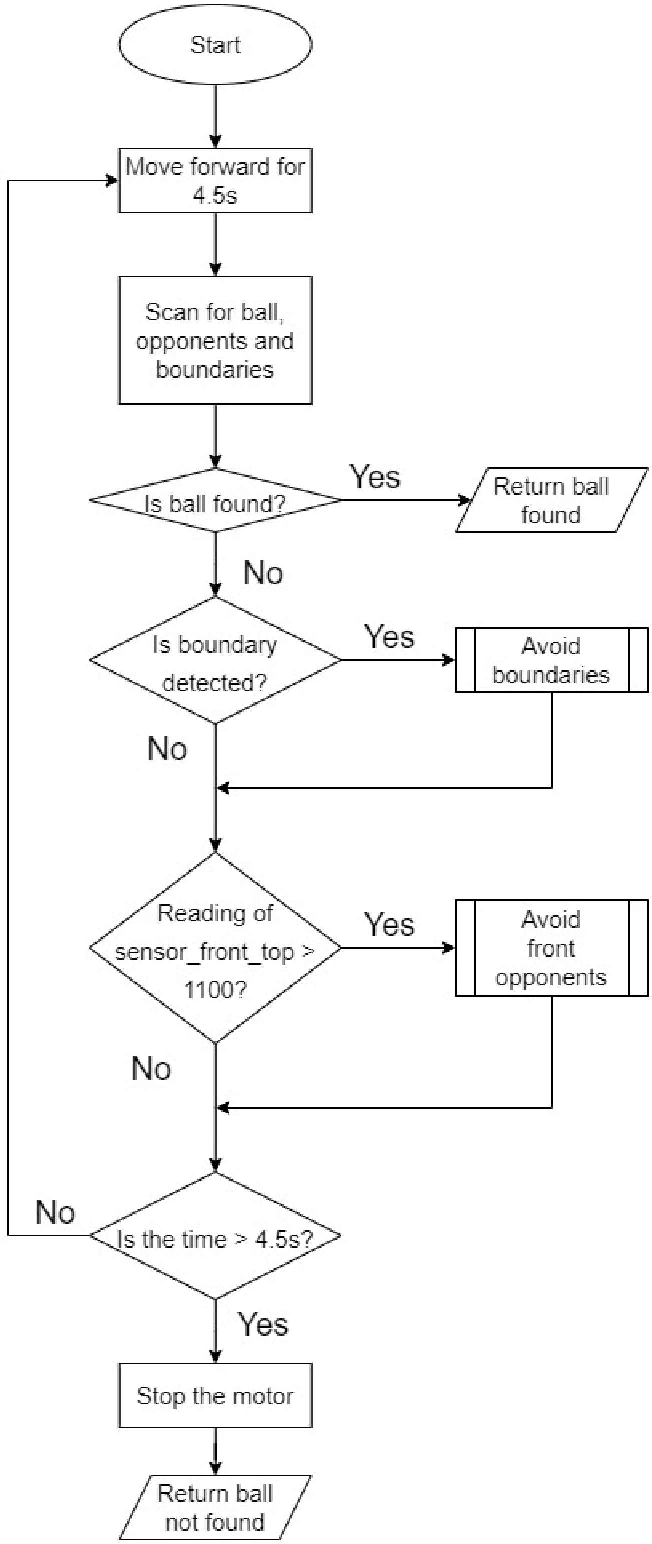


Figure 64: Starting procedure flowchart

7.5.3 Spin Search

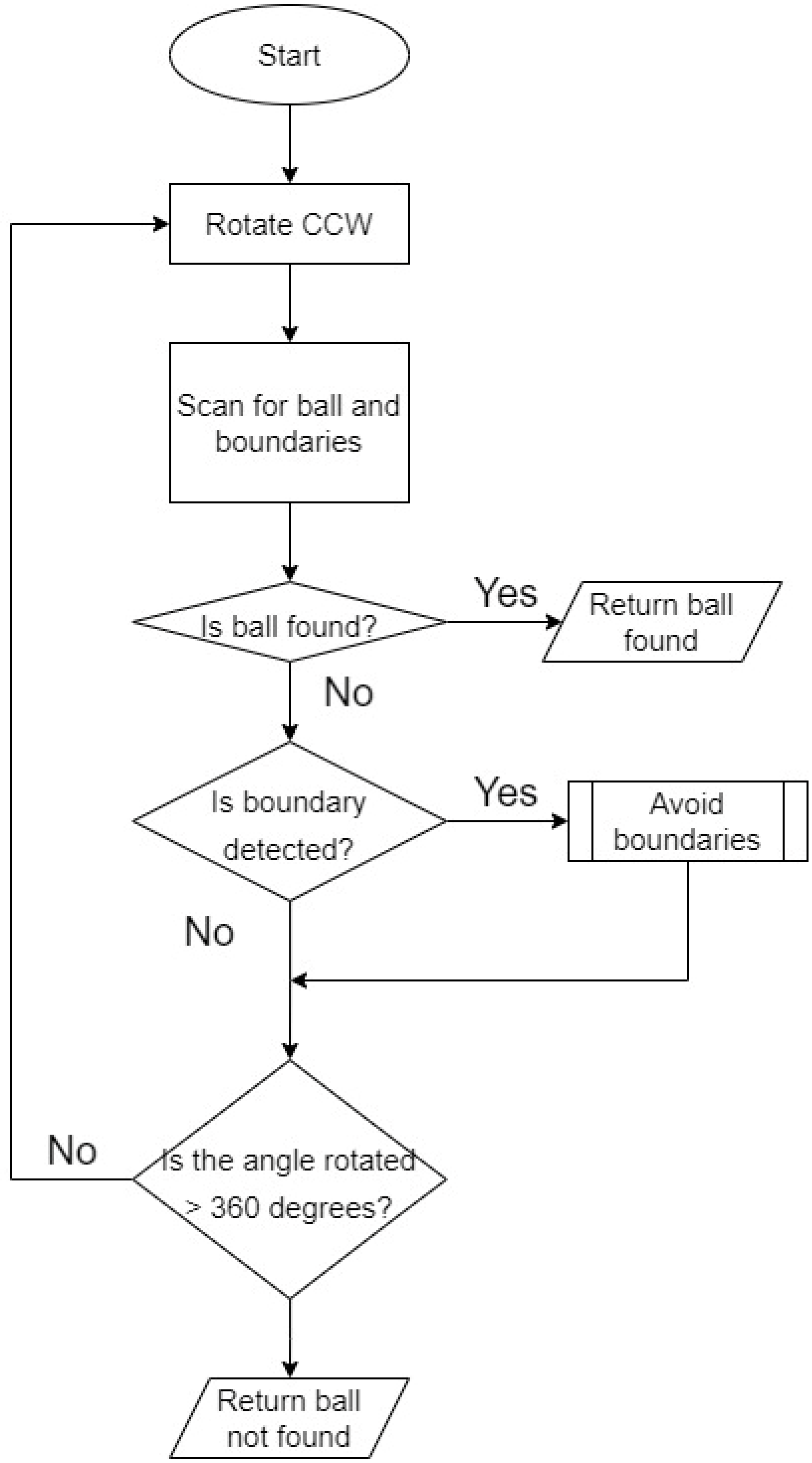


Figure 65: Spin search flowchart

7.5.4 Move Forward

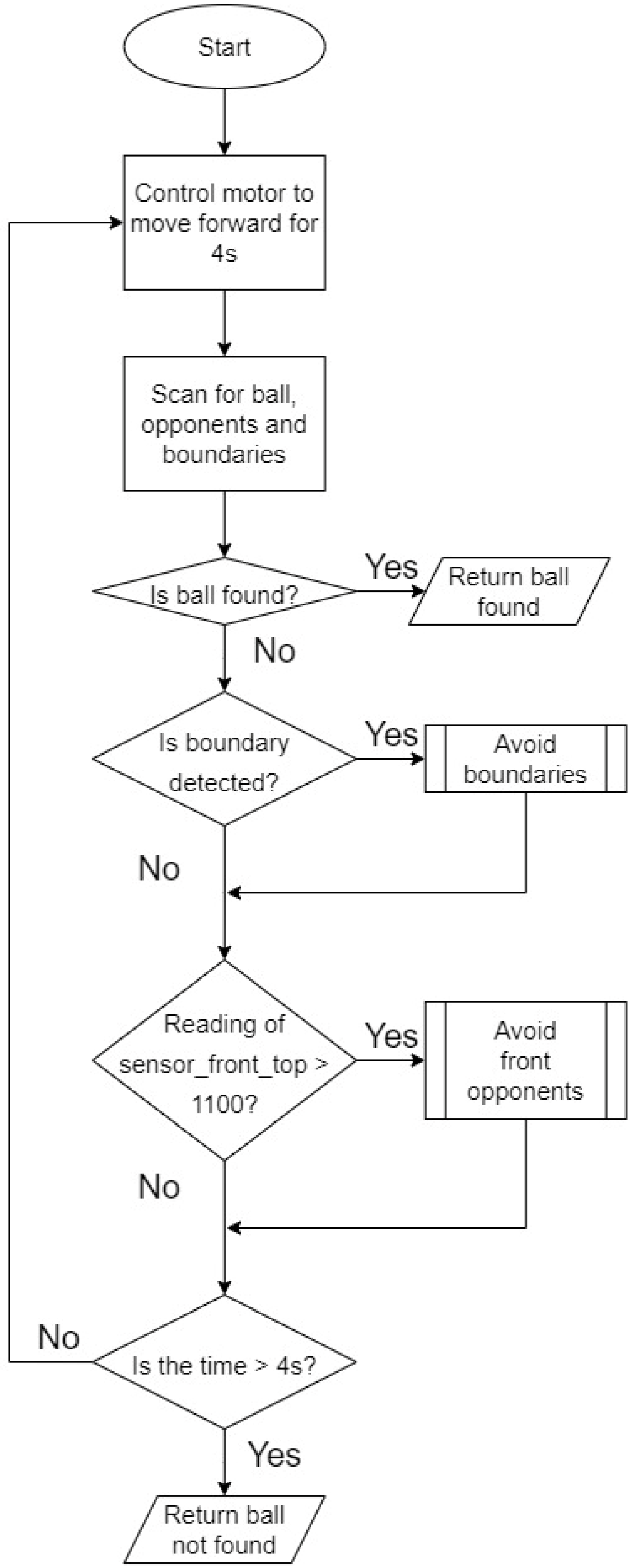


Figure 66: Move forward flowchart

7.5.5 Collect Ball

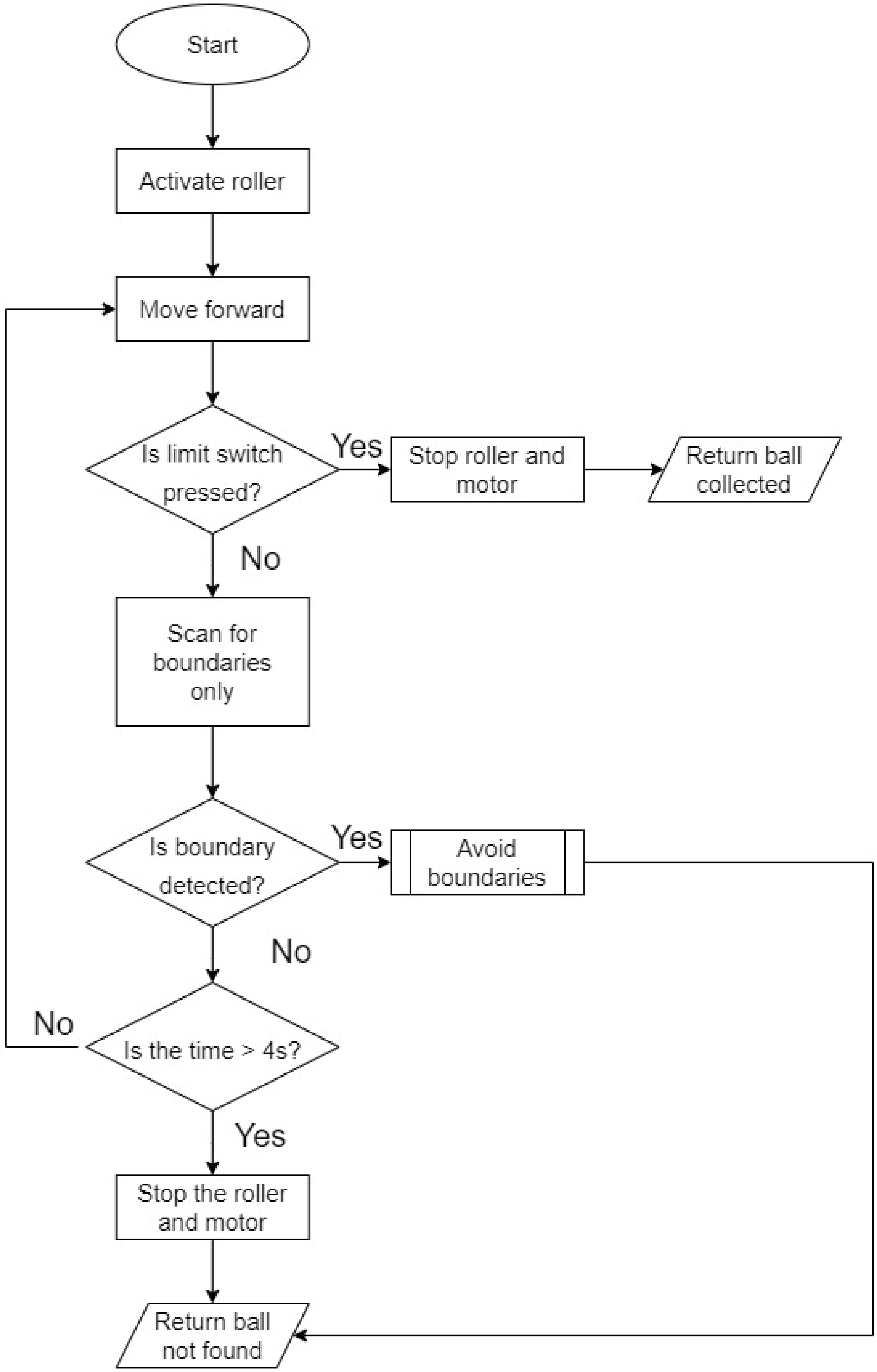


Figure 67: Collect ball flowchart

7.5.6 Dispense Ball

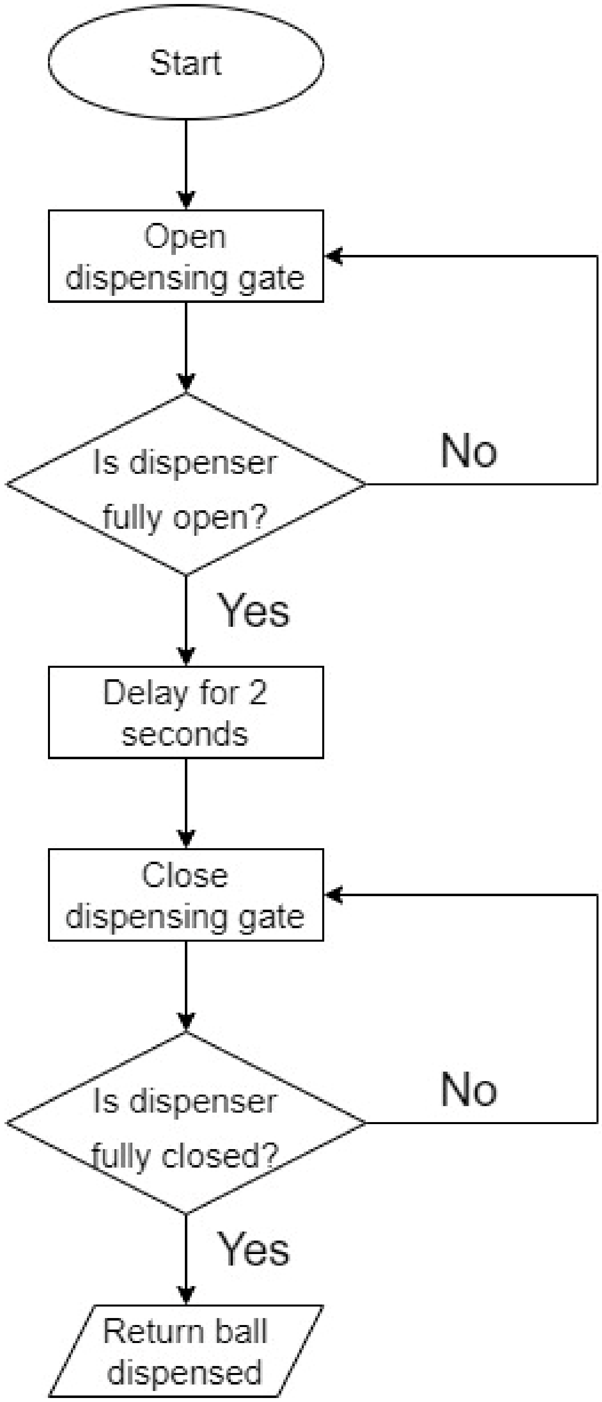


Figure 68: Dispense Ball flowchart

7.5.7 Avoid Boundaries

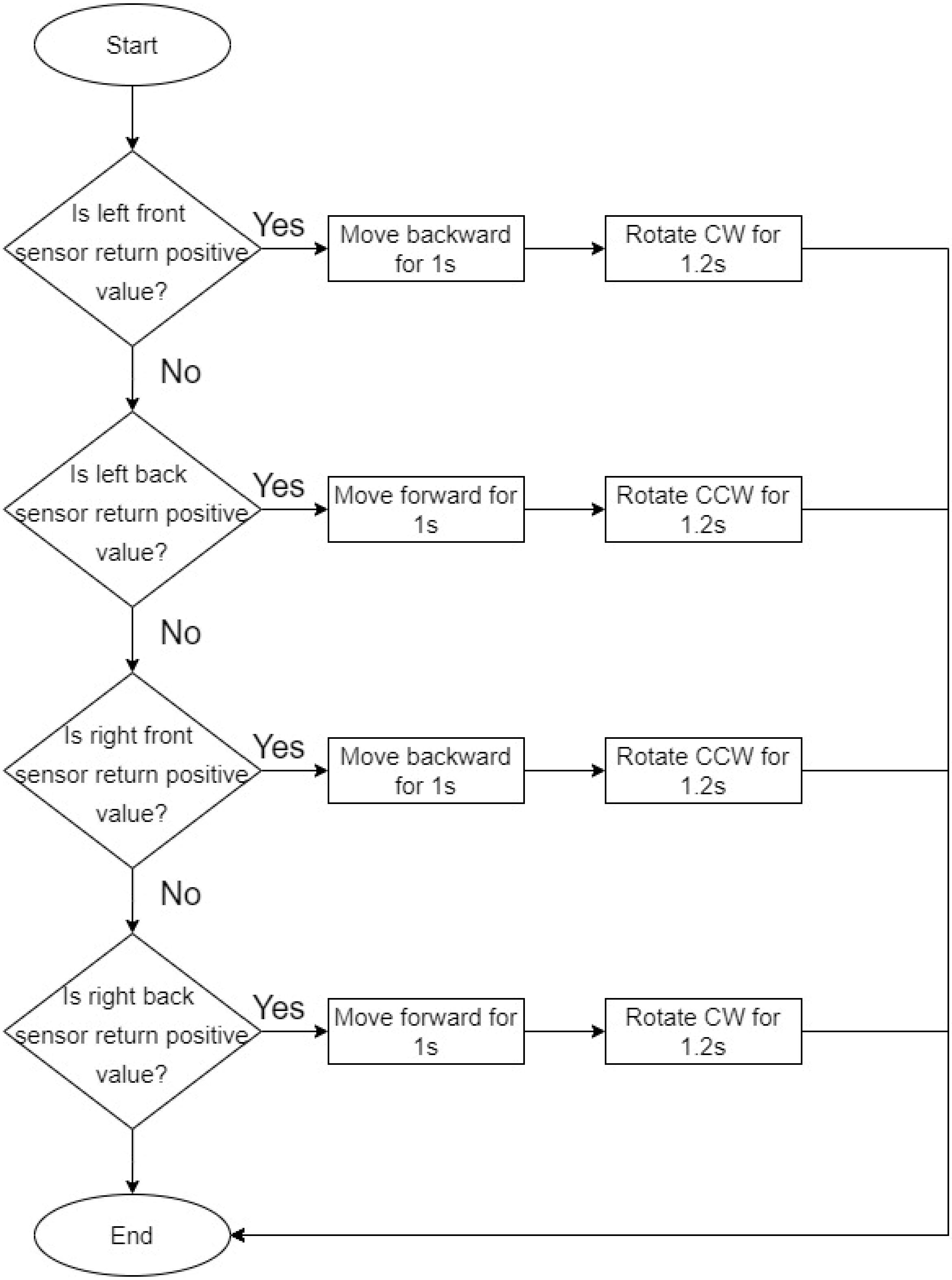


Figure 69: Avoid boundaries flowchart

7.5.8 Back to Origin

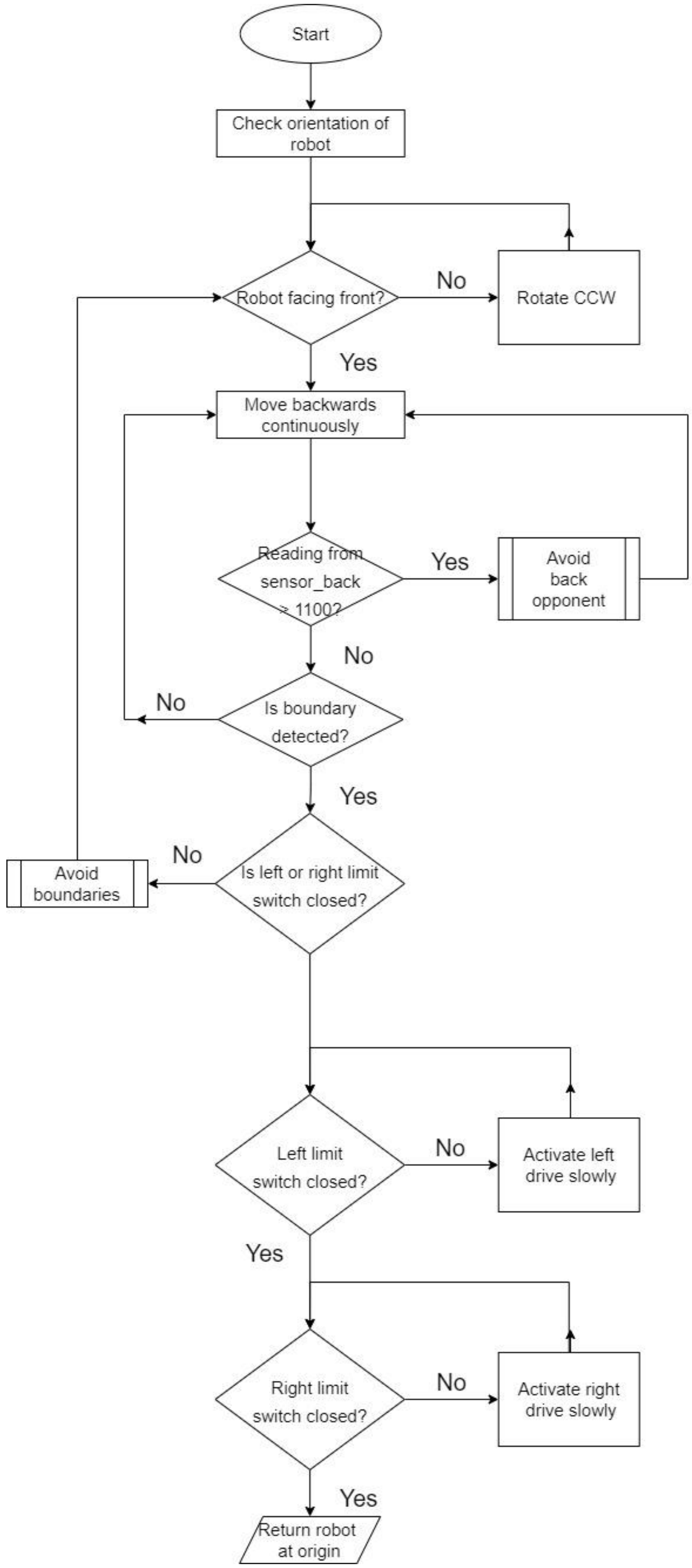


Figure 70: Back to origin flowchart

7.5.9 Avoid Front Opponent

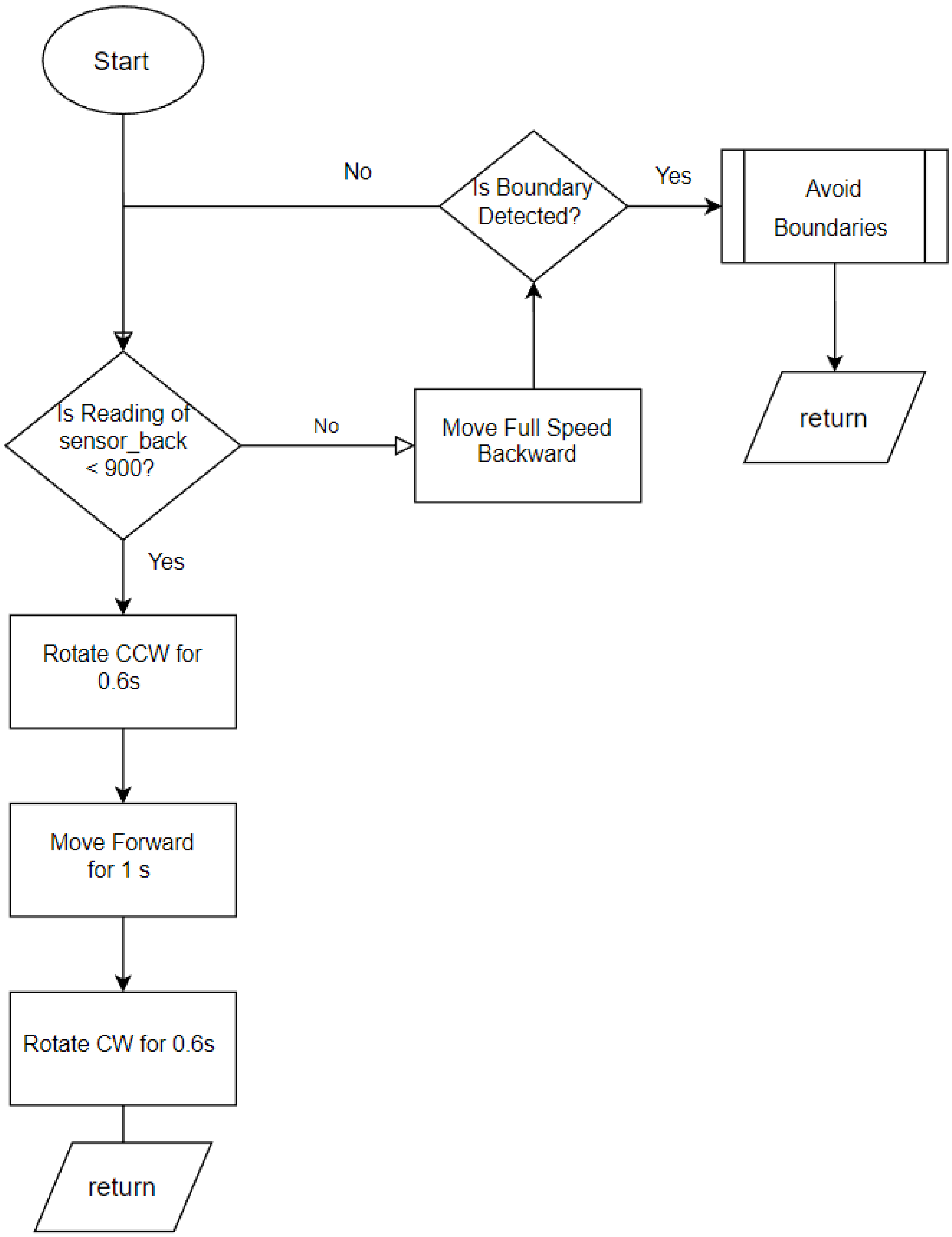


Figure 71: Avoid front opponent flowchart

7.5.10 Avoid Back Opponent

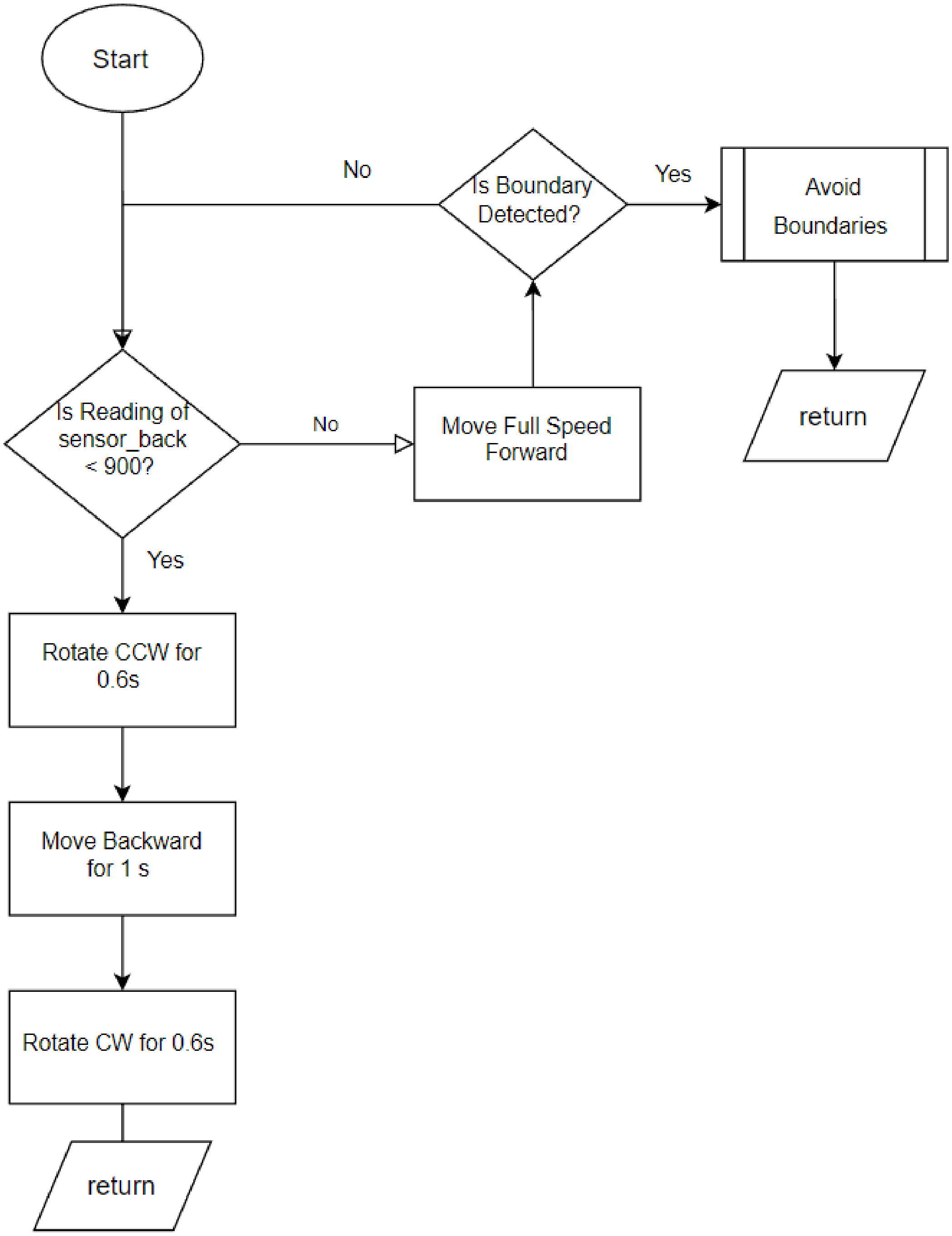


Figure 72: Avoid back opponent flowchart

8 Lesson Learnt

8.1 Hardware Aspect

One of the challenges that the hardware team faced is to make the robot as sturdy as possible with the limited number of materials given by the laboratory. Some materials such as bars and plates that were given to us are essential in creating our base structure of the robot but are lacking in quantity. The extra bars and plates that were given to us are either in the inappropriate shape or of different lengths. This is remedied using 3D- printed parts designed in the CAD by the hardware team but the team does not have constant access to the 3D printer and are limited to the availability of the mechatronic laboratory and the time that can be given to us to do the printing. If the 3D part design concept is deemed too time consuming or insufficient time is available for the part to print, the final resort would be to modify various given parts using machining under the permission of the mechatronic laboratory technicians.

Another challenge faced by the hardware team would be the frequent revision of robot design due to various circumstances. The robot design that we envisioned when the whole group brainstormed together may not be practical or suitable for implementation due to some challenges faced when building up the robot like the robot wheel’s gear being too close to the ground, lacking sufficient spacer to properly strengthen the structure or even major challenges like the overall design exceeding the size limits. Other similar challenges would be the failure to consider possible wiring placement from the motors and sensors to the vex controller. This have led to additional modifications needed to ensure the wires have enough slack to prevent it from being detached in a case of it being caught as well as for cable management for easy identification and aesthetics.

From these challenges faced, most of the modifications would require the machine to be torn down and reassembled. Some parts that the team felt that would be essential to the robot would only be realised in the very last minute during the testing phase and thus, many improvised parts and quick modifications is done to the various parts of the robot. The team have ultimately improved on our quick thinking and improvisation skills.

8.2 Software Aspect

8.2.1 Importance of Early Testing in the Competition Arena

Before we tested our vehicle in the arena, the distance sensor has been providing a consistent reading throughout the initial setup. But we later found out that the physical barriers surrounding the arena interferes a lot with the sensors. Being around the same height, the barrier is constantly misidentified as a tennis ball which created a lot of false positives. As a temporary solution, we increased the detection threshold which helped with the false positives but compromised the detection range. Our robot is basically half-blinded with half of the search range as we planned initially.

Also, another problem that arose during the competition is the return to home algorithm. We assumed that when one of the 2 backward facing limit switch is pressed, it means that the vehicle is touching the dispense area with a slanted angle. The logical step will then be turning the vehicle to align with the wall. However, during the competition, the limit switch is constantly triggered by the opponent vehicle due to accidental collisions. This puts our vehicle in constant loops of turning until something else triggers the other limit switch.

These problems are hard to thought of before an actual trial run. It highlights the importance of early testing in the actual environment and showed how some of our basic presumptions could be wrong. An early testing would have given us more time to tackle and solve the issues.

8.3 Individual Aspect

Member	Lesson Learnt
Chua Jun Yang Chalmers	During the project, there were many valuable lessons the I learnt that I felt that could be applied in real-life situation One of the important lessons learnt was to not be afraid of starting out. Initially the group have multiple ideas during the brainstorming session held during the weekend of the same week a certain component was discussed in the tutorial class. Everyone has multiple ideas of a certain part or system of the robot, and each have valid reasons on why their concept would be the better solution. This have led to some discussion on which concept is better which led to some delay in the creation of the robot. However, some concepts may look good but is not feasible and this is realised only during the building stage of the robot. Only during the building stage would the team be able to realize which concept is better in getting to our goal. Once a good concept has been identified, this idea is able to be finetuned it further to improve its functionality and design. Another lesson learnt would be to always have an alternative solution as the circumstances and situation is unpredictable. One instance would be the creation of the ramp system and the battery placement of the robot. The team have identified the placement of the battery to be at the bottom of the robot for a more stable design under the ramp and designed cut-outs and slots for it to fit nicely around it. However, upon 3D printing the ramp out fitting around the battery, we realised that the battery has not enough space to allow charging, this have led to quick modification and the eventual design to install the battery above the vehicle on the platform. Although this design may make the vehicle less stable as discussed with the team, it was found that

	the base of the robot is wide and heavy enough to offset the potential instability the battery placed on top may create. Overall, this has been a very memorable project with my teammates, I have gained invaluable experiences through the course of this project.
Khoo Tiong Hee	Being the software team, this is my first time of building a whole program with others. Through this experience, I learnt that it is important to lay down the basic software structure and adapt common practice when it comes to writing code. This ensures a consistency throughout our program and makes it easier for one to understand code written by other programmers. Also, we defined clearly the input, output, and desired outcome for each module, which allowed us to work in parallel without blocking each other's progress. Also, I have learnt that working as a team, it is important to define clearly the expectations for each member and the responsibilities they bear. Giving a clear goal motivates a member to deliver and be accountable for their work. But of course, defining targets isn't the job of the leader alone. Members should be proactive in the discussions to break down the tasks. Overall, this has been a fruitful project of which I have learnt good practices in software and things to note when working in a team. A big thank you to Prof Gerald and all the lab assistant for facilitating and assisting us throughout the competition.
Ooi Jia Min	I am a hardware team member. One of my biggest takeaways from this project is to always plan ahead and think through the idea before building it. When we were designing the base frame of the robot (the

	wheel gear train system), we started by trying to recreate a base frame design we found on the Vex website. We used up most of the chassis bumper and we have only noticed that we did not have enough chassis bumper to build up other parts of the robot. We had to tear down, modify and rebuild the robot. Situations like this happened quite often initially, and it is very time consuming to rebuild the robot repeatedly. Hence, we came out with the solution of using SolidWorks to help us to better visualise our idea during the planning phase. This project has helped me to develop the skills of foreseeing potential problems and coming out with contingency solutions. Besides, I have also learnt to be more flexible and creative when designing the robot structure and the ramp. For example, the ramp that our team has designed was quite big, and it takes around 10hrs to 3D print out the ramp. Due to the limited access to 3D printing in the lab (maximum of 3hrs per print), our team has decided to break down the ramp into 4 separate parts to be printed. We came out with a design that can be assembled like puzzle pieces while trying not to compromise the strength of the ramp. I believe that the ability to adapt is one of the most important soft skills that I have picked up from this project. Although our MYVI was not perfect, but I am glad that we were able to accomplish challenge and to further enhance not only our engineering skillsets and knowledge, but also soft skills in this project. This project has been one of the most memorable and fruitful experiences I have had in MAE.

Tang Chi Onn	I had a very interesting learning experience from the project. On the hardware side, I learnt the importance of rapid prototyping. Although we fortunately had the option of 3d printing through the lab, it was definitely not as accessible compared to if we had a 3d printer. This led to some inconveniences and inflexibility. However, I was grateful to get the chance to learn how to adapt to this challenge. For example, we were not allowed to print large components which took longer than 3 hours as there are other teams which may need to use the 3d printer as well. As a result, we had to split our parts into multiple smaller parts and ensure that we're doing it in a way where we can assemble it in a way which was easy, simple and cheap. Besides that, as we had limited time to print, we also had to be very careful with our designs as to not waste time and resources. We also had to learn to adapt and accept that sometimes, we might not be able to make the perfect design due to time constraints, but we should adapt quickly to find alternative solutions. Ultimately, I believe that I have learnt a lot from this project and have improved not only my design skills but also myself as a whole.
Teoh Chee Sern	I learned how to communicate effectively. We are not saints. All people, including myself, think they contribute more than others. This can easily create a mental imbalance. This is where the conflict begins. It's worse when there is one or many "alpha" people in a group. Things can get out of hand and difficult to deal with. When there is a person who refuses to communicate or a person who is pre-emptive, things can go to a bad or even unmanageable state of affairs. We are lucky that we don't have this type of incident, at least not on the surface. Whenever things are going to go that way, someone will step up and calm it down. It's also fortunate that no splits have

	occurred and that no one has chosen to ostracize a particular person or group. Besides, time management is utterly important. As what Prof John Heng said after our competition, a team will only perform well if they tested their robot a lot. Our team is slightly behind schedule and testing was not done until the very end. Even though our robot ultimately succeeded and was able to operate as expected, I was still a little anxious and worried about it during the competition. Therefore, it's still essential for me to comprehend the importance of time management. Finally, this project was successful. I grow my social network, fortify my ties, and learn a lot.
Wong Shang Yi	I have learnt that project management skills such as breaking down a complex project into multiple concise and clear tasks is crucial in order to complete a task smoothly and perfectly. By doing so, everyone will be able to work on their area of interest instead of everyone doing the same thing at the same time. This will greatly increase the productivity and the project is able to push through smoothly and sequentially as the time progresses. Besides, a clear objective, clearly defined task and self-discipline is the key to guide us through times where we are stuck in a particular issue. Secondly, I have also learnt more about a mechatronics project especially on the software implementation. For example, our codes and the calibrated values might not work as how we imagined it to be under the ideal situation. This is truer than ever in mechatronics compared to pure software project. Therefore, I also learnt important skills such as debugging a robot where we must be able to understand the root cause of a failure from its symptom itself.

	Lastly, I really enjoyed this project where there is a sense of competitiveness between different teams to perfect our robot although we only have one opponent for this sem.
Wu Guan Jie	It is a very interesting learning experience and I have learnt a lot from this project. There are a lot of ups and downs throughout the journey and I am glad that we made it in the end. In doing the debugging and testing, it is like a 3-days hackathon and I would say it is a very tiring yet fruitful experience. I have a strong sense of satisfaction when I put the robot onto the arena and it works out as I expected. Definitely, it is not perfect due to the short testing time and I should have done better in terms of the project time management. Also, in terms of project management, I should have done better. Our meeting time was set on Saturday due to members' packed working and learning schedule. This is the biggest mistake I made because we would not have access to the materials and tools in the lab. Also, this brings the headache of booking PDR of the Arc at 12 am weekly. Perhaps, in retrospect, I should have insisted our meeting time to be on Wednesday. This let me understand thoroughly on when to compromise and when not to compromise in project management. Overall, it is a good learning experience to pick up C programming skill, brainstorming, trying to be creative and learn to manage a team effectively.

Appendix

Appendix A: Github repository

https://github.com/Khoo395/Tennis_Ball_Collector

Appendix B: Source Code

Main Function (main.c)

```
#pragma config(Sensor, in1, sharp_front_top, sensorAnalog)
#pragma config(Sensor, in2, sharp_front_bottom_l, sensorAnalog)
#pragma config(Sensor, in3, sharp_front_bottom_r, sensorAnalog)
#pragma config(Sensor, in4, sharp_short, sensorAnalog)
#pragma config(Sensor, in5, dispense_limit_switch, sensorAnalog)
#pragma config(Sensor, in6, front_l_line, sensorAnalog)
#pragma config(Sensor, in8, compass_power, sensorNone)
#pragma config(Sensor, dgtl1, compass1, sensorDigitalIn)
#pragma config(Sensor, dgtl2, compass2, sensorDigitalIn)
#pragma config(Sensor, dgtl3, compass3, sensorDigitalIn)
#pragma config(Sensor, dgtl4, compass4, sensorDigitalIn)
#pragma config(Sensor, dgtl5, front_r_line, sensorDigitalIn)
#pragma config(Sensor, dgtl8, ball_collection_limit, sensorDigitalIn)
#pragma config(Sensor, dgtl9, home_limit_r, sensorDigitalIn)
#pragma config(Sensor, dgtl10, home_limit_l, sensorDigitalIn)
#pragma config(Sensor, dgtl11, back_l_line, sensorDigitalIn)
#pragma config(Sensor, dgtl12, back_r_line, sensorDigitalIn)
#pragma config(Motor, port2, roller_driver, tmotorServoContinuousRotation, openLoop,
reversed)
#pragma config(Motor, port3, left_driver, tmotorVex393_MC29, openLoop, reversed)
#pragma config(Motor, port8, right_driver, tmotorVex393_MC29, openLoop)
#pragma config(Motor, port9, ball_dispense_driver, tmotorServoStandard, openLoop)
//*!!Code automatically generated by 'ROBOTC' configuration
wizard               !!*//
//!!Code automatically generated by 'ROBOTC' configuration wizard               !!//

#include "sensor_output.h"
#include "motor_control.h"
#include "avoid_boundaries.c"
#include "spin_search.c"
#include "move_forward.c"
#include "starting_procedure.c"
#include "collect_ball.c"
#include "dispense_ball.c"
#include "back_to_origin.c"

task main()
{
    starting_procedure();
    while (1)
    {
```

```
        if (ball_found)
        {
            if (ball_collected)
            {
                back_to_origin();
                dispense_ball();
            }
            else if (!ball_collected)
            {
                collect_ball();
            }
        }
        else if (!ball_found)
        {
            spin_search();
            if (!ball_found)
            {
                moving_forward();
            }
        }
    }
}
```

Motor Control Interface (motor_control.c)

```
#pragma config(Sensor, in1, sharp_front_top, sensorAnalog)
#pragma config(Sensor, in2, sharp_front_bottom_l, sensorAnalog)
#pragma config(Sensor, in3, sharp_front_bottom_r, sensorAnalog)
#pragma config(Sensor, in4, sharp_short, sensorAnalog)
#pragma config(Sensor, in5, dispense_limit_switch, sensorAnalog)
#pragma config(Sensor, in6, front_l_line, sensorAnalog)
#pragma config(Sensor, in8, compass_power, sensorNone)
#pragma config(Sensor, dgtl1, compass1, sensorDigitalIn)
#pragma config(Sensor, dgtl2, compass2, sensorDigitalIn)
#pragma config(Sensor, dgtl3, compass3, sensorDigitalIn)
#pragma config(Sensor, dgtl4, compass4, sensorDigitalIn)
#pragma config(Sensor, dgtl5, front_r_line, sensorDigitalIn)
#pragma config(Sensor, dgtl8, ball_collection_limit, sensorDigitalIn)
#pragma config(Sensor, dgtl9, home_limit_r, sensorDigitalIn)
#pragma config(Sensor, dgtl10, home_limit_l, sensorDigitalIn)
#pragma config(Sensor, dgtl11, back_l_line, sensorDigitalIn)
#pragma config(Sensor, dgtl12, back_r_line, sensorDigitalIn)
#pragma config(Motor, port2, roller_driver, tmotorServoContinuousRotation, openLoop,
reversed)
#pragma config(Motor, port3, left_driver, tmotorVex393_MC29, openLoop, reversed)
#pragma config(Motor, port8, right_driver, tmotorVex393_MC29, openLoop)
#pragma config(Motor, port9, ball_dispense_driver, tmotorServoStandard, openLoop)
//*!!Code automatically generated by 'ROBOTC' configuration
wizard               !!*//

//!!Code automatically generated by 'ROBOTC' configuration wizard               !!//
```

```
#include "sensor_output.h"
#include "motor_control.h"
#include "avoid_boundaries.c"
#include "spin_search.c"
#include "move_forward.c"
#include "starting_procedure.c"
#include "collect_ball.c"
#include "dispense_ball.c"
#include "back_to_origin.c"

task main()
{
    starting_procedure();
    while (1)
    {
        if (ball_found)
        {
            if (ball_collected)
            {
                back_to_origin();
                dispense_ball();
            }
            else if (!ball_collected)
            {
                collect_ball();
            }
        }
        else if (!ball_found)
        {
            spin_search();
            if (!ball_found)
            {
                moving_forward();
            }
        }
    }
}
```

Sensor output Interface (sensor_output.c)

```
// enum
enum Orientation
{
    NORTH,              // 0
    NORTH_EAST,         // 1
    EAST,               // 2
    SOUTH_EAST,         // 3
    SOUTH,              // 4
    SOUTH_WEST,         // 5
    WEST,               // 6
    NORTH_WEST,         // 7
    INVALID_COMBINATION // 8
```

```
};

enum BoundarySide
{
    FRONT_LEFT,
    FRONT_RIGHT,
    BACK_LEFT,
    BACK_RIGHT,
    NO_BOUNDARY_DETECTED
};

// sharp sensor parameter
int top_detection_value = 1100;
int bottom_detection_value = 800;

// distance
float dist_ft;
float dist_bl;
float dist_br;
float dist_back;

// ball status
int ball_found = 0;
int ball_collected = 0;

// compass status
int compass_status;
int goal_compass_status;

// limit_switch_status
int dispense_limit_status;
int dispense_limit_switch_voltage; // analog

// line_sensor_status
BoundarySide line_sensor_status;

// spin_search alternation
int spin_CCW = 0;

void read_sharp_front_top()
{
    dist_ft = SensorValue(sharp_front_top);
    return;
}

void read_sharp_front_bottom_l()
{
    dist_bl = SensorValue(sharp_front_bottom_l);
    return;
}

void read_sharp_front_bottom_r()
{
    dist_br = SensorValue(sharp_front_bottom_r);
    return;
```

```
}

void read_short_sharp()
{
    dist_back = SensorValue(sharp_short);
    return;
}

void read_dispense_limit_switch()
{
    // analog limit switch
    dispense_limit_switch_voltage = SensorValue(dispense_limit_switch);
    if (dispense_limit_switch_voltage != 0)
    {
        dispense_limit_status = 1;
        return;
    }
    else
    {
        dispense_limit_status = 0;
        return;
    }
}

void is_ball_on_vehicle()
{
    if (SensorValue(ball_collection_limit) == 0)
    {
        ball_collected = 1;
        return;
    }
    else
    {
        ball_collected = 0;
        return;
    }
}

void scan_boundary()
{
    int frontLeft = SensorValue(front_l_line);
    int frontRight = SensorValue(front_r_line);
    int backLeft = SensorValue(back_l_line);
    int backRight = SensorValue(back_r_line);

    if (frontLeft < 800)
    {
        line_sensor_status = FRONT_LEFT;
        return;
    }
    else if (frontRight == 0)
    {
        line_sensor_status = FRONT_RIGHT;
        return;
    }
```

```
    else if (backLeft == 0)
    {
        line_sensor_status = BACK_LEFT;
        return;
    }
    else if (backRight == 0)
    {
        line_sensor_status = BACK_RIGHT;
        return;
    }
    else
    {
        line_sensor_status = NO_BOUNDARY_DETECTED;
        return;
    }
}

void read_compass()
{
    int pin1 = SensorValue(compass1);
    int pin2 = SensorValue(compass2);
    int pin3 = SensorValue(compass3);
    int pin4 = SensorValue(compass4);
    int combination = pin1 * 1000 + pin2 * 100 + pin3 * 10 + pin4;

    switch (combination)
    {
    case 1110:
        compass_status = NORTH;
        return;
    case 1100:
        compass_status = NORTH_EAST;
        return;
    case 1101:
        compass_status = EAST;
        return;
    case 1001:
        compass_status = SOUTH_EAST;
        return;
    case 1011:
        compass_status = SOUTH;
        return;
    case 0011:
        compass_status = SOUTH_WEST;
        return;
    case 0111:
        compass_status = WEST;
        return;
    case 0110:
        compass_status = NORTH_WEST;
        return;
    default:
        compass_status = INVALID_COMBINATION;
        return;
    }
```

```
}

void scan_ball()
{
    if (SensorValue(sharp_front_bottom_l) > bottom_detection_value ||
SensorValue(sharp_front_bottom_r) > bottom_detection_value)
    {
        if (SensorValue(sharp_front_top) < top_detection_value)
        {
            ball_found = 1;
            return;
        }
        else
        {
            ball_found = 0;
            return;
        }
    }
    else
    {
        ball_found = 0;
        return;
    }
}
```

Avoid Boundaries (avoid_boundaries.c)

```
void avoid_boundaries(BoundarySide line_sensor_number)
{
    if (line_sensor_number == FRONT_LEFT)
    {
        writeDebugStreamLine("%s", "front left line sensor activated");
        clearTimer(T2);
        while (time1(T2) < 1000)
        {
            // move backward
            control_motor(-45, -45);
        }
        clearTimer(T2);
        while (time1(T2) < 1200)
        {
            // rotate CW
            control_motor(60, -60);
        }
        stop_motor();
        return;
    }
    else if (line_sensor_number == BACK_LEFT)
    {
        writeDebugStreamLine("%s", "back left line sensor activated");
        clearTimer(T2);
        while (time1(T2) < 1000)
        {
```

```
            // move forward
            control_motor(60, 60);
        }
        clearTimer(T2);
        while (time1(T2) < 1200)
        {
            // rotate CCW
            control_motor(-60, 60);
        }
        stop_motor();
        return;
    }
    else if (line_sensor_number == FRONT_RIGHT)
    {
        writeDebugStreamLine("%s", "front right line sensor activated");
        clearTimer(T2);
        while (time1(T2) < 1000)
        {
            // move backward
            control_motor(-45, -45);
        }
        clearTimer(T2);
        while (time1(T2) < 1200)
        {
            // rotate CCW
            control_motor(-60, 60);
        }
        stop_motor();
        return;
    }
    else if (line_sensor_number == BACK_RIGHT)
    {
        writeDebugStreamLine("%s", "back right line sensor activated");
        clearTimer(T2);
        while (time1(T2) < 1000)
        {
            // move forward
            control_motor(60, 60);
        }
        clearTimer(T2);
        while (time1(T2) < 1200)
        {
            // rotate CW
            control_motor(60, -60);
        }
        stop_motor();
        return;
    }
}
```

Avoid Opponents (avoid_opponent.c)

```
void avoid_opponent(){
```

```
    if (SensorValue(sharp_front_top) > 1100){
        while(SensorValue(sharp_front_top) > 900){
            control_motor(-127,-127);
        }
        stop_motor();
        clearTimer(T1);
        while (time1(T1) < 600)
        {
            control_motor(-60, 60);
        }
        clearTimer(T1);
        while (time1(T1) < 1000)
        {
            control_motor(60, 60);
        }
        clearTimer(T1);
        while (time1(T1) < 600)
        {
            control_motor(60, -60);
        }
        stop_motor();
    }

    else if(SensorValue(sharp_short) > 1100){
        while(SensorValue(sharp_front_top) > 900){
            control_motor(127,127);
        }
        stop_motor();
        clearTimer(T1);
        while (time1(T1) < 600)
        {
            control_motor(-60, 60);
        }
        clearTimer(T1);
        while (time1(T1) < 1000)
        {
            control_motor(-60, -60);
        }
        clearTimer(T1);
        while (time1(T1) < 600)
        {
            control_motor(60, -60);
        }
        stop_motor();
    }
}
```

Back to Origin (back_to_origin.c)

```
// int sharp_short_threshold = 1100;
void back_to_origin()
{
    writeDebugStreamLine("%s", "activate back to origin");
    while (1)
```

```
    {
        turn_to_south_position();
        for (int i = 0; i < 10; i++)
        {
            control_motor(-127, -82);
            // scan boundaries
            scan_boundary();
            if (line_sensor_status != NO_BOUNDARY_DETECTED)
            {
                avoid_boundaries(line_sensor_status);
                clearTimer(T2);
                while (time1(T2) < 1000)
                {
                    // move backward
                    control_motor(-127, -82);
                }
            }
             avoid_back_opponent();
            clearTimer(T2);
            while (time1(T2) < 5000)
            {
                if (SensorValue(home_limit_r) == 0 || SensorValue(home_limit_l) == 0)
                {
                    stop_motor();
                    while (SensorValue(home_limit_r) != 0)
                    {
                        motor[right_driver] = -40;
                    }
                    motor[right_driver] = 0;
                    while (SensorValue(home_limit_l) != 0)
                    {
                        motor[left_driver] = -40;
                    }
                    motor[left_driver] = 0;
                    writeDebugStreamLine("%s", "i am at the origin");
                    return;
                }
            }
        }
    }
}
```

Collect Ball (collect_ball.c)

```
void collect_ball()
{
    writeDebugStreamLine("%s", "collecting ball...");
    // activate roller
    activate_roller(127);

    // move forward
    // max speed will go out of boundaries
```

```
    clearTimer(T1);
    while (time1(T1) < 4000)
    {
        control_motor(80, 80);
        is_ball_on_vehicle();
        if (ball_collected == 1)
        {
            writeDebugStreamLine("%s", "ball collected");
            stop_roller();
            stop_motor();
            return;
        }

        // scan boundaries
        scan_boundary();
        if (line_sensor_status != NO_BOUNDARY_DETECTED)
        {
            avoid_boundaries(line_sensor_status);
        }
    }
    stop_roller();
    stop_motor();
    ball_found = 0;
    ball_collected = 0;
    writeDebugStreamLine("%s", "ball not collected");
    return;
}
```

Dispense Ball (dispense_ball.c)

```
void dispense_ball()
{
    writeDebugStreamLine("%s", "calling dispensing function");
    open_dispense_gate();
    close_dispense_gate();
    ball_found = 0;
    ball_collected = 0;
    starting_procedure();
}
```

Move Forward (move_forward.c)

```
void moving_forward()
{
    writeDebugStreamLine("%s", "moving forward");
    clearTimer(T3);
    // move forward
    control_motor(100, 100);
    while (time1(T3) < 4000)
```

```
    {
        // scan ball
        if (ball_found == 1)
        {
            stop_motor();
            writeDebugStreamLine("%s", "ball found in moving forward");
            return;
        }

        // scan boundaries
        scan_boundary();
        if (line_sensor_status != NO_BOUNDARY_DETECTED)
        {
            avoid_boundaries(line_sensor_status);
        }
        avoid_front_opponent();
    }
    stop_motor();
    ball_found = 0;
    return;
}
```

Spin Search (spin_search.c)

```
void spin_search()
{
    // update compass status & set goal compass status
    // +2 is used because of the shaking compass
    read_compass();
    goal_compass_status = compass_status + 2;
    if (goal_compass_status == 6)
    {
        goal_compass_status = 1;
    }
    if (goal_compass_status == 7)
    {
        goal_compass_status = 2;
    }

    // Rotate CW
    control_motor(60, -60);
    while (1)
    {
        // Scan for ball
        scan_ball();
        if (ball_found == 1)
        {
            writeDebugStreamLine("%s", "ball found... from spin search");
            return;
        }
        else
        {
```

```
            // Scan for boundaries
            scan_boundary();
            if (line_sensor_status != NO_BOUNDARY_DETECTED)
            {
                avoid_boundaries(line_sensor_status);
            }

            avoid_opponent();

            // turning for 360
            read_compass();
            if (compass_status == goal_compass_status)
            {
                stop_motor();
                ball_found = 0;
                return;
            }
        }
    }
}
```

Starting Procedure (starting_procedure.c)

```
// parameters for starting
float sp_time = 4500; // how long should the robot move forward when starting
float sp_speed = 127; // max speed in competition

void starting_procedure()
{
    float starting_sp_time = nSysTime;
    float sp_time_elapsed = 0;
    float total_boundary_time = 0;
    float current_time = 0;

    // loop
    while (1)
    {
        writeDebugStreamLine("%s", "starting procedure...");
        // move forward
        control_motor(sp_speed, sp_speed - 45);
        scan_ball();
        if (ball_found == 1)
        {
            writeDebugStreamLine("%s", "ball found from starting procedure");
            stop_motor();
            return;
        }

        // Scan for boundaries
        scan_boundary();
        if (line_sensor_status != NO_BOUNDARY_DETECTED)
        {
```

```
            avoid_boundaries(line_sensor_status);
            total_boundary_time += 1500;
        }
        avoid_front_opponent();

        current_time = nSysTime;
        sp_time_elapsed = current_time - starting_sp_time - total_boundary_time;
        writeDebugStreamLine("%d", sp_time_elapsed);
        if (sp_time_elapsed > sp_time)
        {
            stop_motor();
            ball_found = 0;
            return;
        }
    }
}
```

Appendix C: Data Sheet

1. Short range Sharp Sensor (GP2Y0A41SK0F)

GP2Y0A41SK0F

Distance Measuring Sensor Unit
Measuring distance : 4 to 30 cm
Analog output type

■Description

GP2Y0A41SK0F is a distance measuring sensor unit, composed of an integrated combination of PSD (position sensitive detector) , IR-LED (infrared emitting diode) and signal processing circuit.
The variety of the reflectivity of the object, the environmental temperature and the operating duration are not influenced easily to the distance detection because of adopting the triangulation method.
This device outputs the voltage corresponding to the detection distance. So this sensor can also be used as a proximity sensor.

■Agency approvals/Compliance

1. Compliant with RoHS directive (2011/65/EU)

■Applications

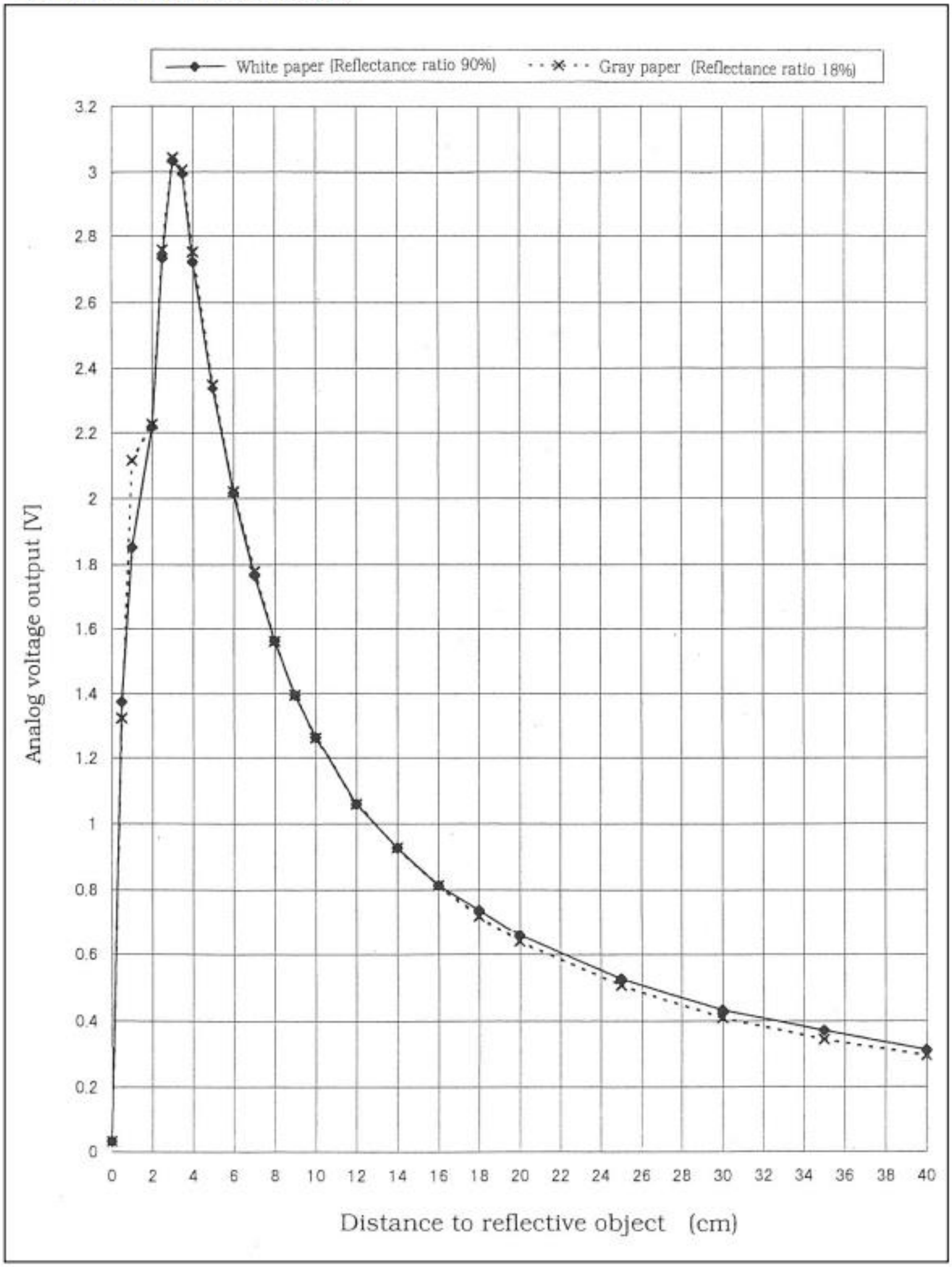
1. Cleaning robot
2. Personal robot
3. Sanitary

■Features

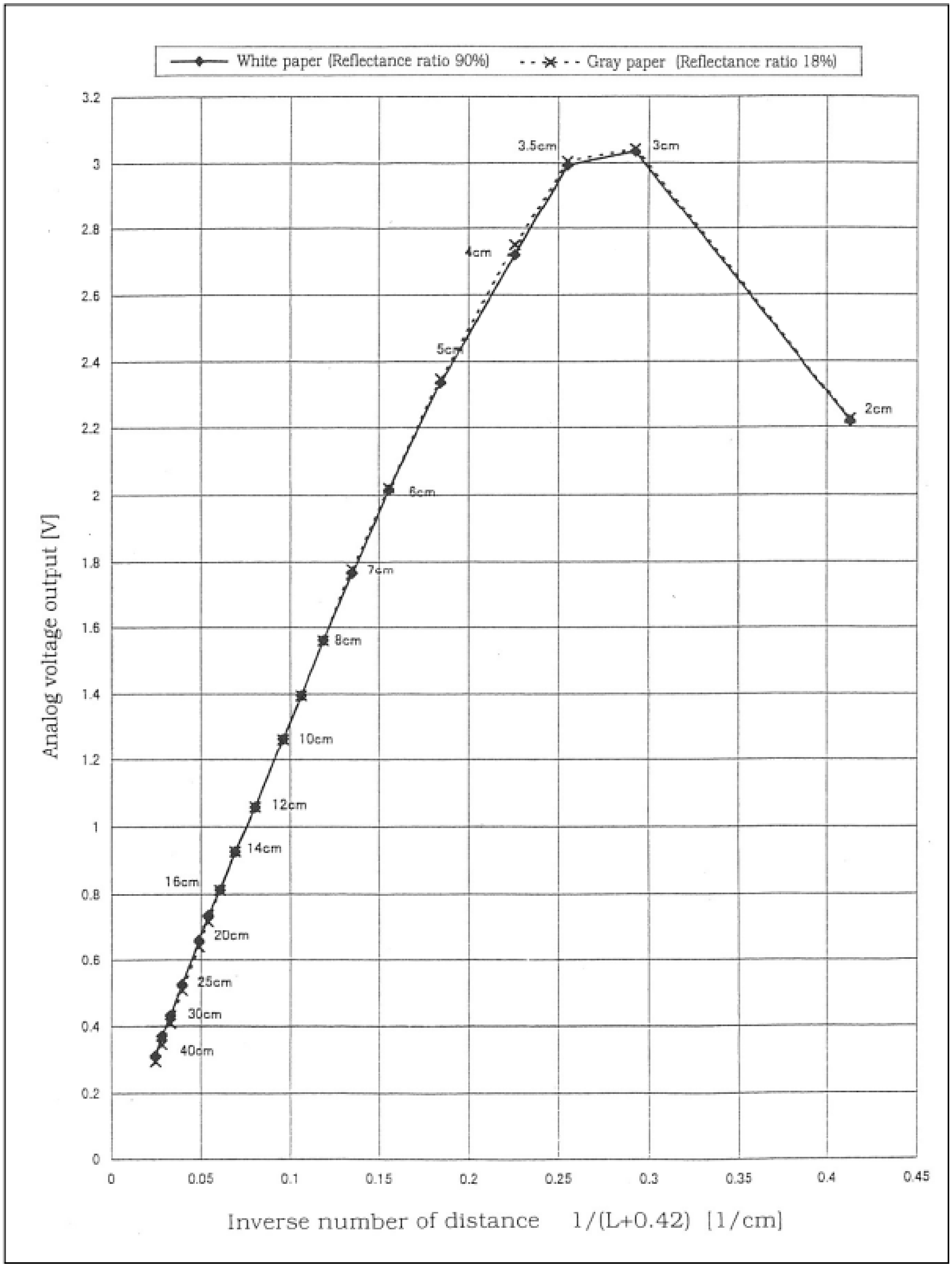
1. Distance measuring sensor is united with PSD, infrared LED and signal processing circuit
2. Short measuring cycle (16.5ms)
3. Distance measuring range : 4 to 30 cm
4. Package size (29.5 × 13.0 × 13.5mm)
5. Analog output type

■Supplements

•Example of output distance characteristics



●Example of output distance characteristics with the inverse of distance



2. Long range Sharp Sensor(GP2Y0A21SK0F)

GP2Y0A21YK0F

Distance Measuring Sensor Unit
Measuring distance: 10 to 80 cm
Analog output type

■Description

GP2Y0A21YK0F is a distance measuring sensor unit, composed of an integrated combination of PSD (position sensitive detector) , IRED (infrared emitting diode) and signal processing circuit.
The variety of the reflectivity of the object, the environmental temperature and the operating duration are not influenced easily to the distance detection because of adopting the triangulation method.
This device outputs the voltage corresponding to the detection distance. So this sensor can also be used as a proximity sensor.

■Features

1. Distance measuring range : 10 to 80 cm
2. Analog output type
3. Package size : 29.5×13×13.5 mm
4. Consumption current : Typ. 30 mA
5. Supply voltage : 4.5 to 5.5 V

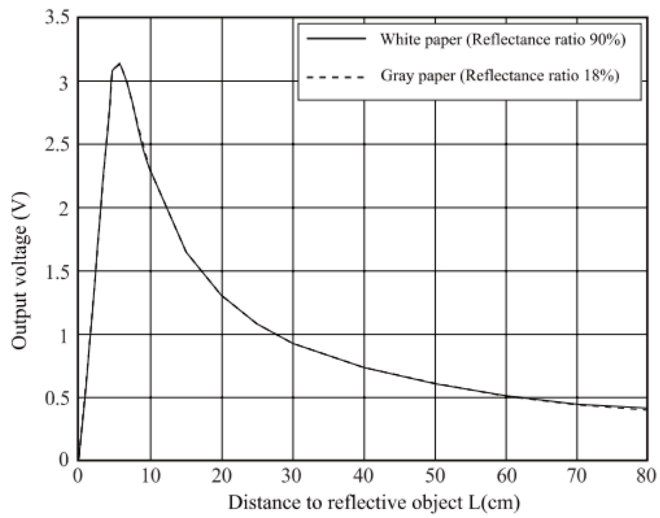
■Agency approvals/Compliance

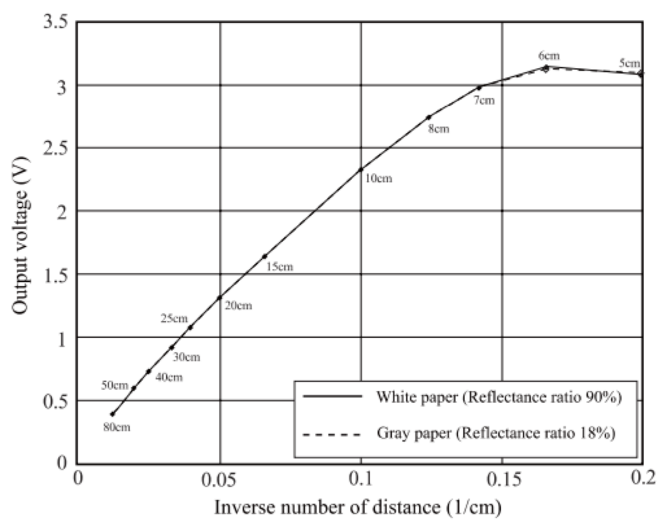
1. Compliant with RoHS directive (2011/65/EU)

■Applications

1. Touch-less switch
   (Sanitary equipment, Control of illumination, etc.)
2. Robot cleaner
3. Sensor for energy saving
   (ATM, Copier, Vending machine)
4. Amusement equipment
   (Robot, Arcade game machine)

Fig. 2 Example of distance measuring characteristics(output)





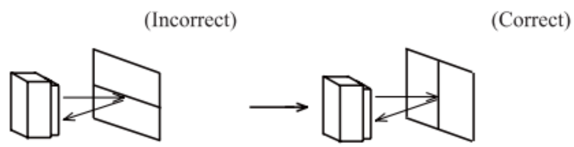
■Notes

●Advice for the optics

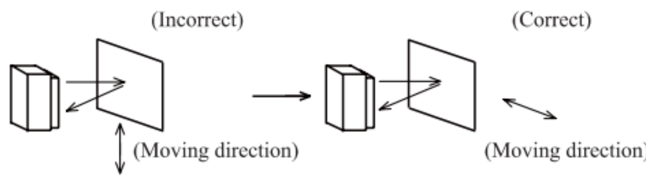
- The lens of this device needs to be kept clean. There are cases that dust, water or oil and so on deteriorate the characteristics of this device. Please consider in actual application.
- Please don't do washing. Washing may deteriorate the characteristics of optical system and so on. Please confirm resistance to chemicals under the actual usage since this product has not been designed against washing.

●Advice for the characteristics

- In case that an optical filter is set in front of the emitter and detector portion, the optical filter which has the most efficient transmittance at the emitting wavelength range of LED for this product ($\lambda = 870 \pm 70$nm), shall be recommended to use. Both faces of the filter should be mirror polishing. Also, as there are cases that the characteristics may not be satisfied according to the distance between the protection cover and this product or the thickness of the protection cover, please use this product after confirming the operation sufficiently in actual application.
- In case that there is an object near to emitter side of the sensor between sensor and a detecting object, please use this device after confirming sufficiently that the characteristics of this sensor do not change by the object.
- When the detector is exposed to the direct light from the sun, tungsten lamp and so on, there are cases that it can not measure the distance exactly. Please consider the design that the detector is not exposed to the direct light from such light source.
- Distance to a mirror reflector can not be sometimes measured exactly. In case of changing the mounting angle of this product, it may measure the distance exactly.
- In case that reflective object has boundary line which material or color etc. are excessively different, in order to decrease deviation of measuring distance, it shall be recommended to set the sensor that the direction of boundary line and the line between emitter center and detector center are in parallel.



- In order to decrease deviation of measuring distance by moving direction of the reflective object, it shall be recommended to set the sensor that the moving direction of the object and the line between emitter center and detector center are vertical.



●Advice for the power supply

- In order to stabilize power supply line, we recommend to insert a by-pass capacitor of 10μF or more between Vcc and GND near this product.

●Notes on handling

- There are some possibilities that the internal components in the sensor may be exposed to the excessive mechanical stress. Please be careful not to cause any excessive pressure on the sensor package and also on the PCB while assembling this product.

3. 393 2-pin motor

Motion Accessories

2 Wire Motor 393

The 2 Wire Motor 393 provides up to 60% more torque than the standard motor, which will allow more powerful mechanisms and drive bases. All of the internal gears are made from a steel alloy, which means that clutches and replacement gears are no longer required. The 2 wire motor can be directly connected to the Cortex and ARM 9 microcontrollers' internal motor controllers. An external motor control module is required to connect the 2 wire motor to the PIC Microcontroller V0.5. External motor control modules can also be used with the Cortex and ARM 9 microcontrollers.

INSERT THIS PAGE at the **back of the Motion Chapter** in your VEX Inventor's Guide.

High Speed Option

Want to go faster than the standard motor but still have the same output torque as the standard motor? No problem! The 2 Wire Motor 393 kit can be configured into a "high speed" version. Simply follow the "Gear Change Procedure" step-by-step instructions to increase the output speed by 60%.

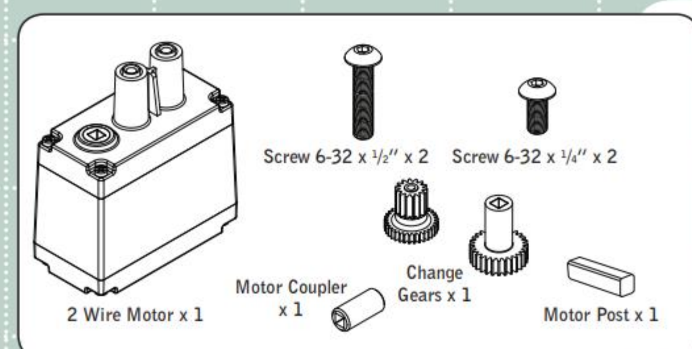
Motor Coupler

The 2 Wire Motor 393 Kit includes the new shaft coupler which can be used in place of the clutch to connect the motor to VEX shafts. The coupler can also be used to connect VEX shafts together.

Motor Specifications

All motor specifications are at 7.2 volts. Actual motor specifications are within 20% of the values below.

Description	As Shipped	High Speed Option
Stall Torque	13.5 in-lb [1.68 N-m]	8.4 in-lb [1.05 N-m]
Free Speed	100 RPM	160 RPM
Stall Current	3.6 Amps	
Free Current	0.15 Amps	



Limited 90-day Warranty
This product is warranted by Innovation First against manufacturing defects in material and workmanship under normal use for ninety (90) days from the date of purchase from authorized Innovation First dealers. For complete warranty details and exclusions, check with your dealer.

Innovation First, Inc.
1519 IH 30 W
Greenville, TX 75402

12/10

For More Information, and additional Parts & Pieces refer to:
www.VEXrobotics.com

accessories

Inventor's Guide insert

2 Wire Motor 393 • 1

4. 276-2163 3-pin motor

Robot Kits

VEX Motion

Click photo to enlarge.

VEX Robotics Continuous Rotation Motor Kit

Part# VEX-276-2163

(average customer rating)

$19.99

Stock Status: Removed

This Product Has Been Discontinued

General Product Information | Alternate Photos | Other Documentation | Reviews & Comments

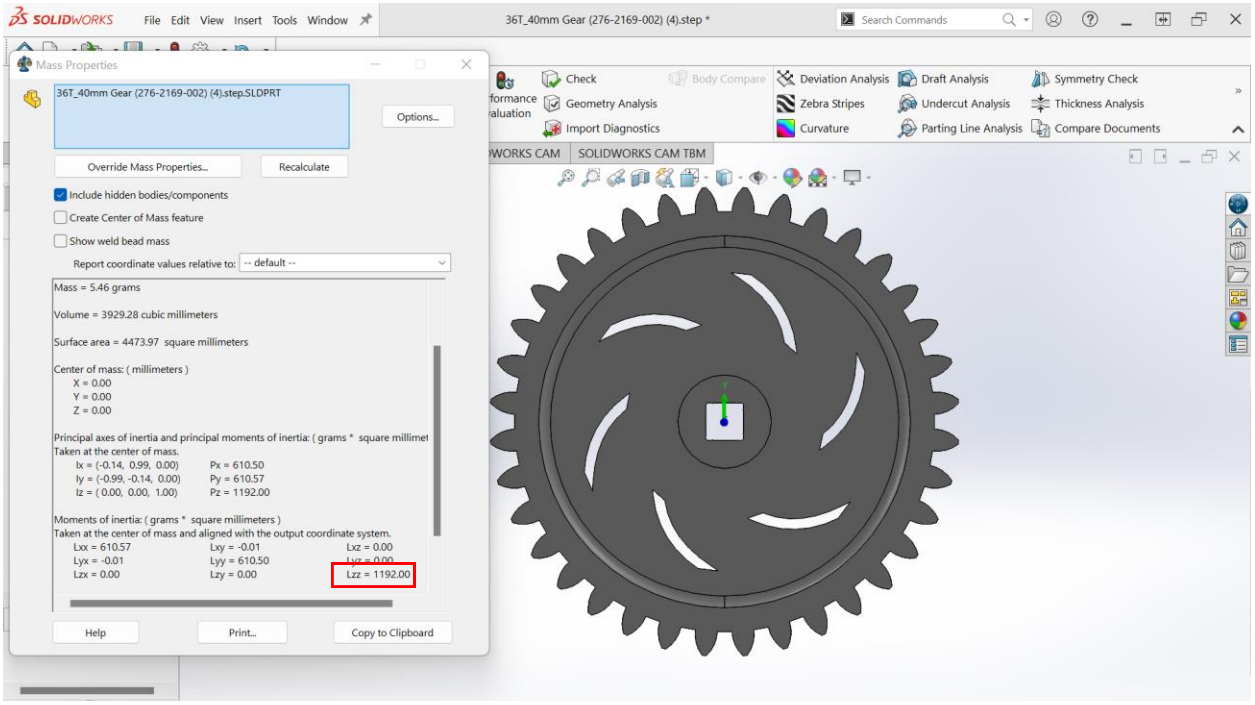
Expand your robot's navigation with the continuous-rotation motor. This motor is made to mate perfectly with the drive shafts of the Vex Robotics System.

Includes everything you need for assembly: 1 motor, 4 mounting screws and replacement gear set
For use with Robotic Starter Kit #276-2151

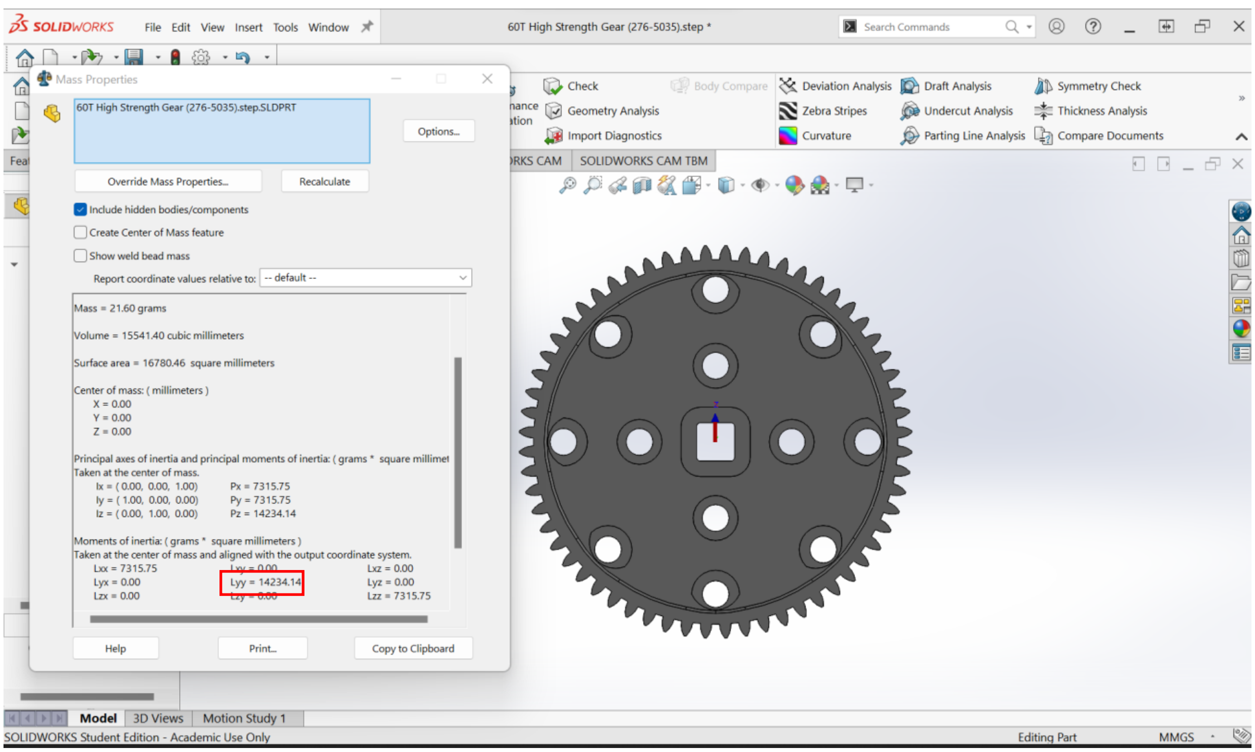
Technical Info	
Free Speed	100 rmp at 7.5V
Stall Torque	6.5 in-lbs.
Min/Max Voltage	4.4 - 15V (Motor life will be reduced operating outside the VEX Controller range of 5.5 - 9.0V)
PWM Input	1ms - 2ms will give full reverse to full forward; 1.5ms is neutral
Nominal Dead Band	1.47ms - 1.55ms
Weight	0.21 lbs.
Wiring	Black - ground; Orange - (+) power; White - PWM motor control signal
Current Draw	5mA to about 1 Amp. at stall per Motor
List of Contents	
1	VEX Continuous Rotation Motor
1	VEX motor gear set (4 total gears, 1 replacement for each gear inside)
2	6-32 x 1/4in. Screws
2	6-32 x 1/2in. Screws
1	Clutch Post
1	Inventor's Guide Insert

Appendix D: Moment of Inertia

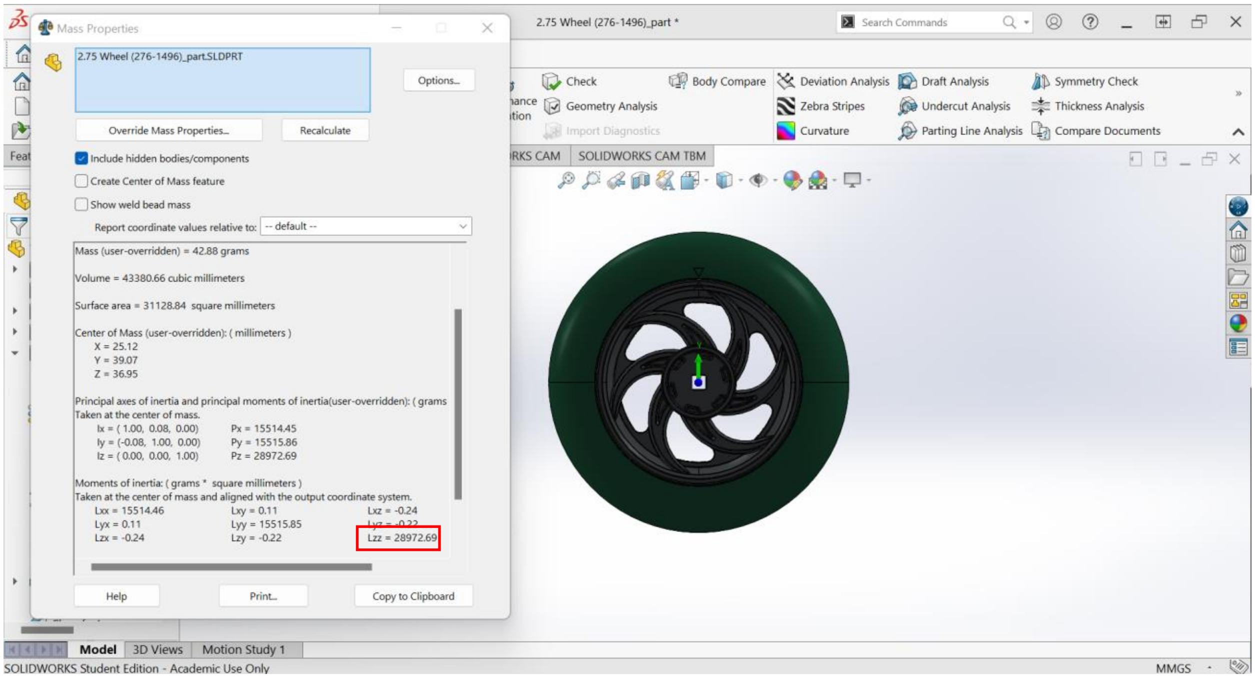
1. 36T Gear



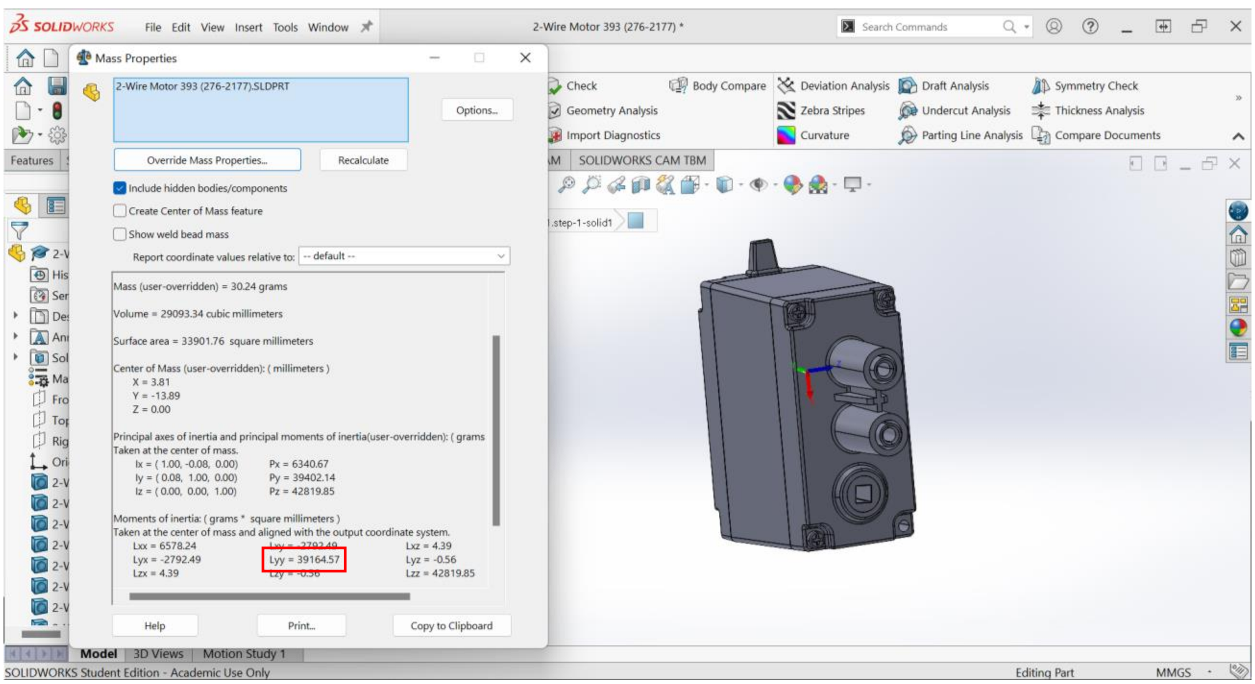
2. 60T High Strength Gear



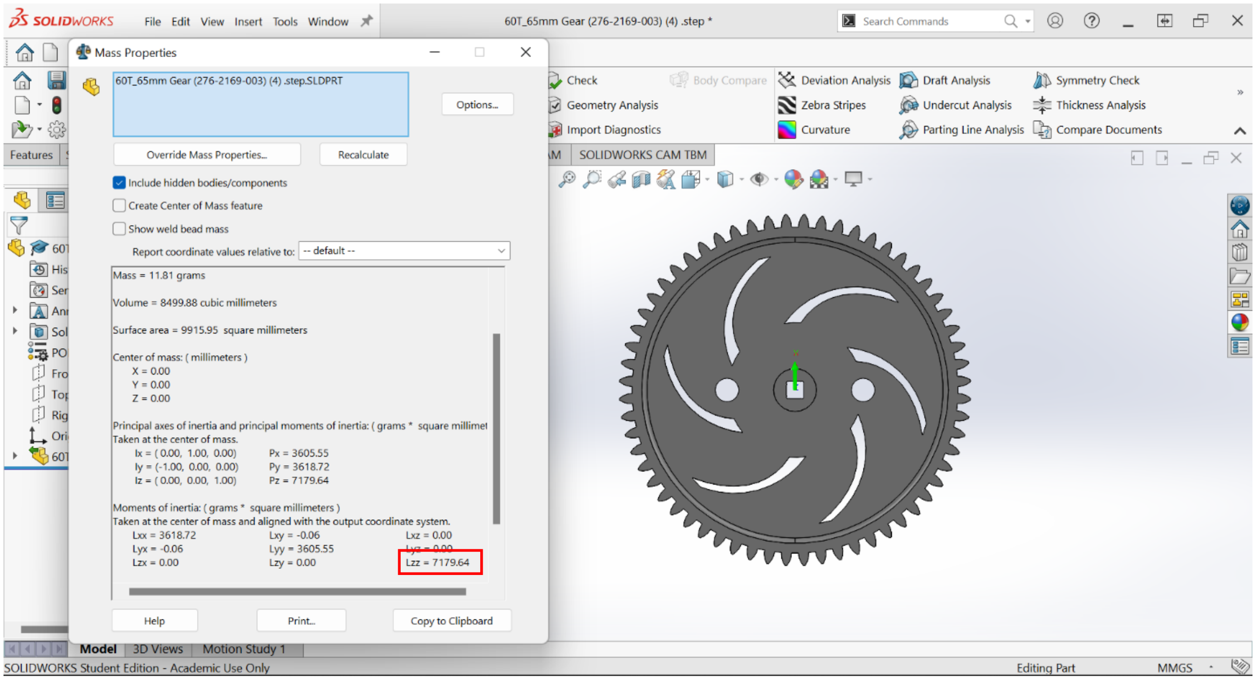
3. Wheel



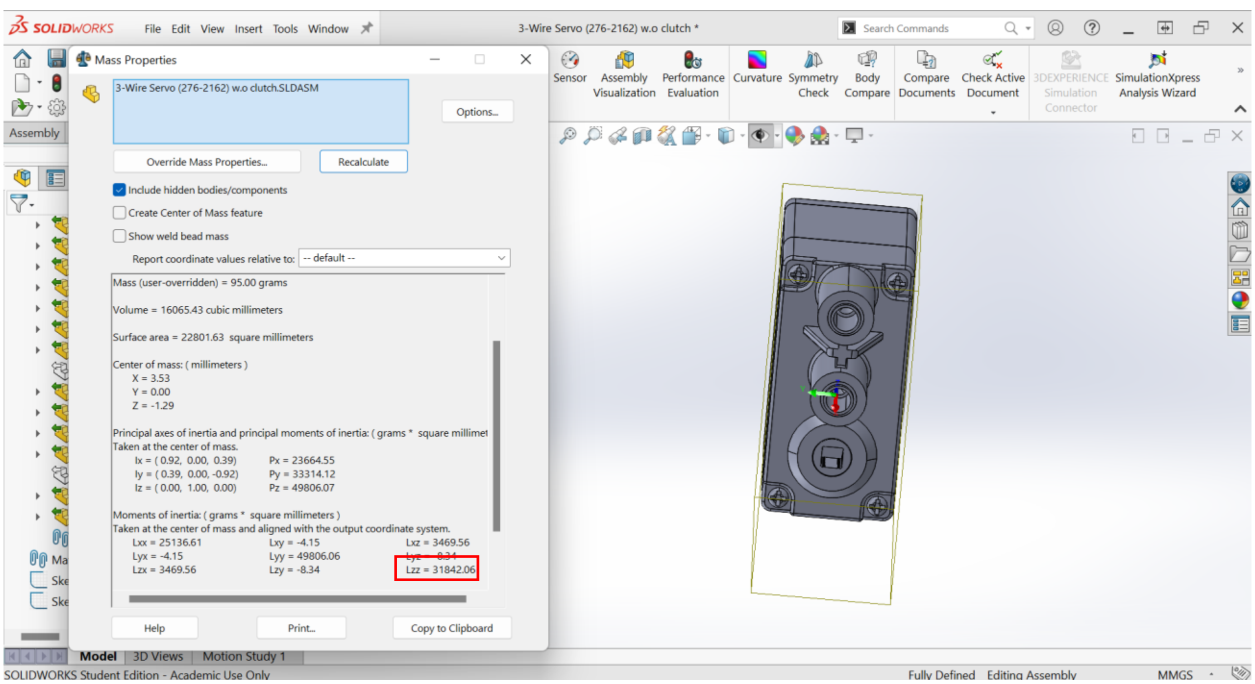
4. 2-Wire Motor



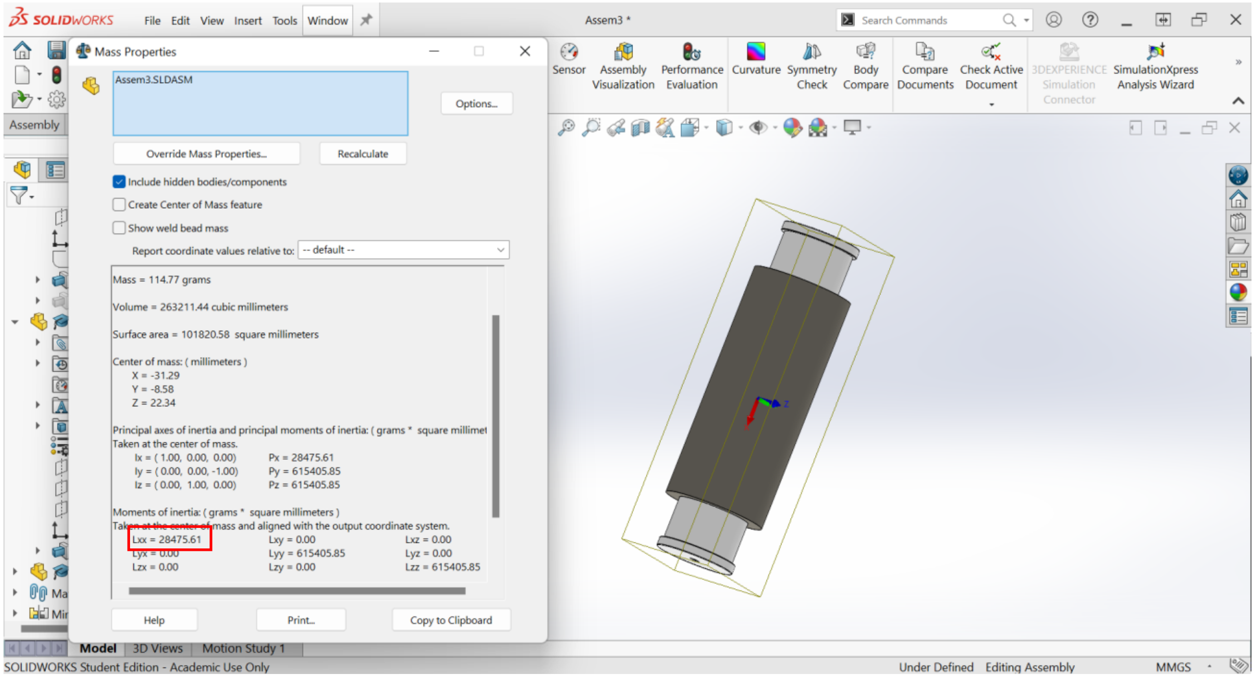
5. 60T Gear



6. 3-Wire Motor



7. Roller



Mass Properties
Assem3.SLDASM
Options...
Override Mass Properties...
Recalculate
Include hidden bodies/components
Create Center of Mass feature
Show weld bead mass
Report coordinate values relative to: -- default --
Mass = 114.77 grams
Volume = 263211.44 cubic millimeters
Surface area = 101820.58 square millimeters
Center of mass: (millimeters)
X = -31.29
Y = -8.58
Z = 22.34
Principal axes of inertia and principal moments of inertia: (grams * square millimet
Taken at the center of mass.
Ix = (1.00, 0.00, 0.00) Px = 28475.61
Iy = (0.00, 0.00, -1.00) Py = 615405.85
Iz = (0.00, 1.00, 0.00) Pz = 615405.85
Moments of inertia: (grams * square millimeters)
Taken at the center of mass and aligned with the output coordinate system.
Lxx = 28475.61 Lxy = 0.00 Lxz = 0.00
Lyx = 0.00 Lyy = 615405.85 Lyz = 0.00
Lzx = 0.00 Lzy = 0.00 Lzz = 615405.85
Help
Print...
Copy to Clipboard
SOLIDWORKS Student Edition - Academic Use Only
Under Defined Editing Assembly
MMGS