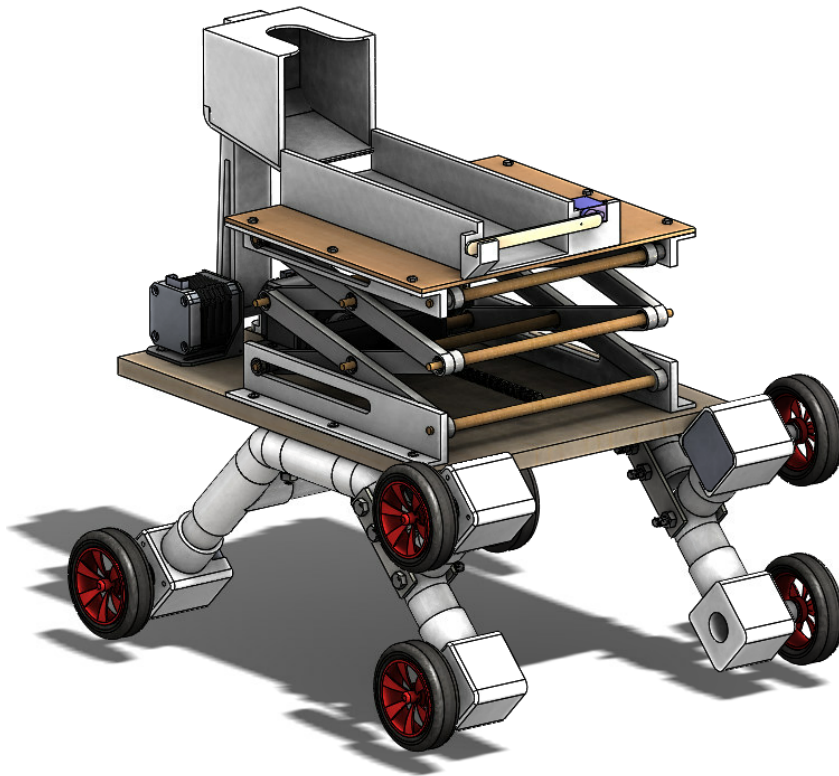


Engineering Portfolio

Warman Design Challenge 2023

Ryan Kembrey



Fuel Vessel Delivering Robot Built In
Mechanical Design Studio 1



Mechanical Engineering
University Of Technology Sydney
Australia

September 12, 2023

Contents

I	Personal Profile	5
1	About Self	6
1.1	Reflection	6
II	Team Overview	7
2	Summary of the Project	8
2.1	Background	8
2.2	The Team & Roles	9
2.3	Final Design Overview	10
2.3.1	Chassis	11
2.3.2	Scissor Lift	11
2.3.3	Collection Arm	12
2.3.4	Deposit System	12
2.4	Design Requirements	13
2.4.1	Course Requirements	13
III	Individual Documentation	14
3	Ideation	15
3.1	Ideation	15
3.1.1	Crazy 8s	15
3.1.2	Developed Solution	16
3.2	Compare Subsystems	17
3.2.1	Storage - Solutions	17
3.2.2	Storage - Score Matrix	18
3.2.3	Elevation - Solutions	18
3.2.4	Elevation - Score Matrix	18
4	Prototyping	19
4.1	Low Fidelity Prototypes	19
4.1.1	Linear Actuator & Box	19
4.1.2	Scissor Lift & Topless Container	20
4.1.3	Archimedes' Screw & Silo	21
4.1.4	Prototypes Score Matrix	22
5	Scissor Lift Design (Artefact 1)	23
5.1	First Design	23
5.1.1	Leg Length	23
5.1.2	Force	25
5.1.3	Torque	26
5.1.4	Stepper Motor Chosen	27
5.1.5	CAD	28
5.1.6	Design Flaws	28

5.2	Second Design	29
5.2.1	Design Improvements	29
5.2.2	CAD	29
5.2.3	Calculations	31
5.2.4	Procurement Plan	32
5.2.5	Verdict	32
6	Collection Arm Redesign (Artefact 2)	33
6.0.1	Goal	33
6.0.2	Design	33
6.0.3	Calculations	36
6.0.4	Verdict	37
7	Storage Design (Artefact 3)	38
7.0.1	Goal	38
7.0.2	Design	38
7.0.3	Calculations	38
7.0.4	Verdict	39
8	Mechatronics Development	40
8.1	Pixy Camera (Artifact 4)	40
8.1.1	Goal	40
8.1.2	Pseudo Code	40
8.1.3	Developed Code	40
8.1.4	Testing And Verdict	43
8.2	Acknowledgements	44
A	Ideation of the Group	45

List of Figures

1.1	A photo of myself	6
2.1	The track in which the competition takes place on (taken from the Warman Competition Rules)	8
2.2	Photo of the final design of the robot	10
2.3	The CAD of the chassis, designed by Bashaar	11
2.4	The CAD of the chassis, designed by Ryan (myself)	11
2.5	Collection arm dimetric view, designed by Ryan (myself)	12
2.6	Collection arm frontal view, designed by Ryan (myself)	12
2.7	The CAD of the deposit system, designed by Ryan (myself)	12
3.1	Pt.1 Crazy 8's 16 Designs	15
3.2	Pt.2 Crazy 8's 16 Designs	15
3.3	Pt.1 Crazy 8's 8 Designs	15
3.4	Pt.2 Crazy 8's 8 Designs	15
3.5	Pt.1 Crazy 8 Final 4	15
3.6	Pt.2 Crazy 8 Final 4	15
3.7	Pivoting Triangular	15
3.8	Rough Developed Solution	16
3.9	Detailed Developed Solution	16
3.10	Side View Of Velcro Solution	17
3.11	Front View Of Box Solution	17
3.12	Top View Of Topless Container Solution	17
3.13	Side View Of Rail Solution	17
3.14	Side View Of Silo Solution	17
3.15	Side View Of Vacuum Solution	17
3.16	Side View Of Scissor Lift Solution	18
3.17	Side View Of Archimedes' Screw Solution	18
3.18	Side View Of Linear Actuator Solution	18
4.1	Linear actuated prototype	19
4.2	Linear actuated prototype	19
4.3	Scissor Lift & Topless Container Phase 1	20
4.4	Scissor Lift & Topless Container Phase 2	20
4.5	Scissor Lift & Topless Container Phase 3	20
4.6	Archimedes' screw prototype	21
4.7	Version 2 of the Archimedes' Screw	21
5.1	Lowest height of scissor lift	23
5.2	Smallest x (100mm) of scissor lift	23
5.3	The angle θ of the scissor lift.	24
5.4	The angle θ for every position x on the scissor lift	24
5.5	FBD of scissor lift	25
5.6	The force required for every angle on the scissor lift	25
5.7	The torque required for the force of every position of the scissor lift	26
5.8	The torque required for every angle in the scissor lifts motion	27
5.9	Nema 17 stepper motor	27
5.10	First scissor lift in compressed form	28

5.11	First scissor lift in extended form.	28
5.12	Second scissor lift in compressed form	29
5.13	Second scissor lift in extended form.	30
5.14	Force needed to raise second scissor lift at each angle	31
5.15	Torque needed to raise second scissor lift at each angle	31
5.16	Scissor lift on top of the chassis	32
6.1	Inspiration for the collection arm (Credit: Malte Ahlers)	33
6.2	Scooping box for collection arm	34
6.3	Arm piece for collection arm	34
6.4	Axle support for collection arm	34
6.5	Collection arm assembly	35
6.6	Angles from 0 to 180°in arm motion	36
6.7	Free body diagram of the arm at 90°	36
6.8	Torque required to lift arm	37
7.1	Deposit system design	38
7.2	FBD of deposit system forces	38
8.1	Pixy Camera	40
8.2	Build of the pixy robot	43

List of Tables

2.1	Team members and their roles	9
2.2	Requirements for the Warman Challenge	13
3.1	Scoring Matrix for Solutions	18
3.2	Elevation scoring matrix for solutions	18
4.1	Prototypes scoring matrix for solutions	22
5.1	Lead screw equation values	26
5.2	Scissor Lift Parts Procurement Table (Bought Parts)	32
5.3	Scissor Lift Parts Procurement Table (Printed Parts)	32

List of Listings

1	Calculating the required torque for arm at all positions	36
2	<i>acquireBlock()</i> function listing	40
3	<i>trackBlock()</i> function listing	41
4	<i>rotateToObject()</i> function listing	41
5	<i>driveToObject()</i> function listing	42
6	<i>void loop()</i> function listing	42

Part I

Personal Profile

1 | About Self

First and foremost, I consider myself a creative thinker. Challenging myself to design new things in all areas of life, such as composing songs for a jazz big band, drawing and creating new ideas to solve real world problems. It is this strength that I carry through into all areas of life, especially engineering. My love of creation and design is what drew me to pursuing a career as an engineer, as it would help me to harness my strengths and combine them with knowledge to design a better world.

I am a current 2nd year student studying a bachelor of engineering, majoring in mechanical at the University of Technology Sydney. After now being in the degree for 1.5 years, I can say with certainty I have found a path that gives me drive and passion. Every day I look forward to bettering my capabilities as an engineer. This mentality can be seen throughout this portfolio.



Figure 1.1: A photo of myself

1.1 Reflection

Prior to taking this class, I would have said CAD was a weakness of mine. However, I challenged myself to be a part of the CAD team. This decision has proved to be a strong one, as I ended up designing and CADing three out of the four systems on our robot. I believe this project showed me what my true capabilities are, and now I feel equipped more equipped than ever to tackle any engineering task, and apply the engineering mentality.

Retrospectively, the way I handled the studio was successful. From week 1, I put countless hours into the project. After hearing what we would be tasked with in this studio, I was immediately filled with excitement, which turned into the drive to do well. I aimed to document my journey thoroughly in my portfolio, so that it would be of a quality that a future employer could read it as an example of previous work. In the pursuit from professionalism, I chose to use [L^AT_EX](#) to typeset my portfolio. Using this software made it easier to make the document look professional, and also included my hobby for software in my workflow.

Of course, there are many design alterations I would make to the robot if I had the chance to retake the studio, one being deciding to instead not climb the ledge. The reason for these things I would change are of course, because I have improved as an engineer. I hope to take these lessons learned into the future studios, classes and eventually workplace, and am excited for the next challenge that comes my way.

Part II

Team Overview

2 | Summary of the Project

2.1 Background

The goal of this Warman challenge is to create a small-scale prototype of a transport system that can collect and deliver six spherical vessels to their respective silos. This must be done within a 120 second time limit otherwise the robot must stop wherever it is currently at in the process.

Our approach to the Warman Challenge began by analysing the requirements and limitations, providing us with a framework to base our ideas on. This includes having the robot fit within required mass (6kg) and dimension (400x400x400mm cube) constraints, alongside other requirements such as the robot having to stay on the track (no flying). The outcome of our design should be a robot that can deposit the vessels into the correct silos within the given timeframe of 120 seconds.

Below in figure 2.1, the warman track is shown. The image communicates where the robot must start, where the cells must lie, and the positions of the silos. The two ledges are also notable features in the track.

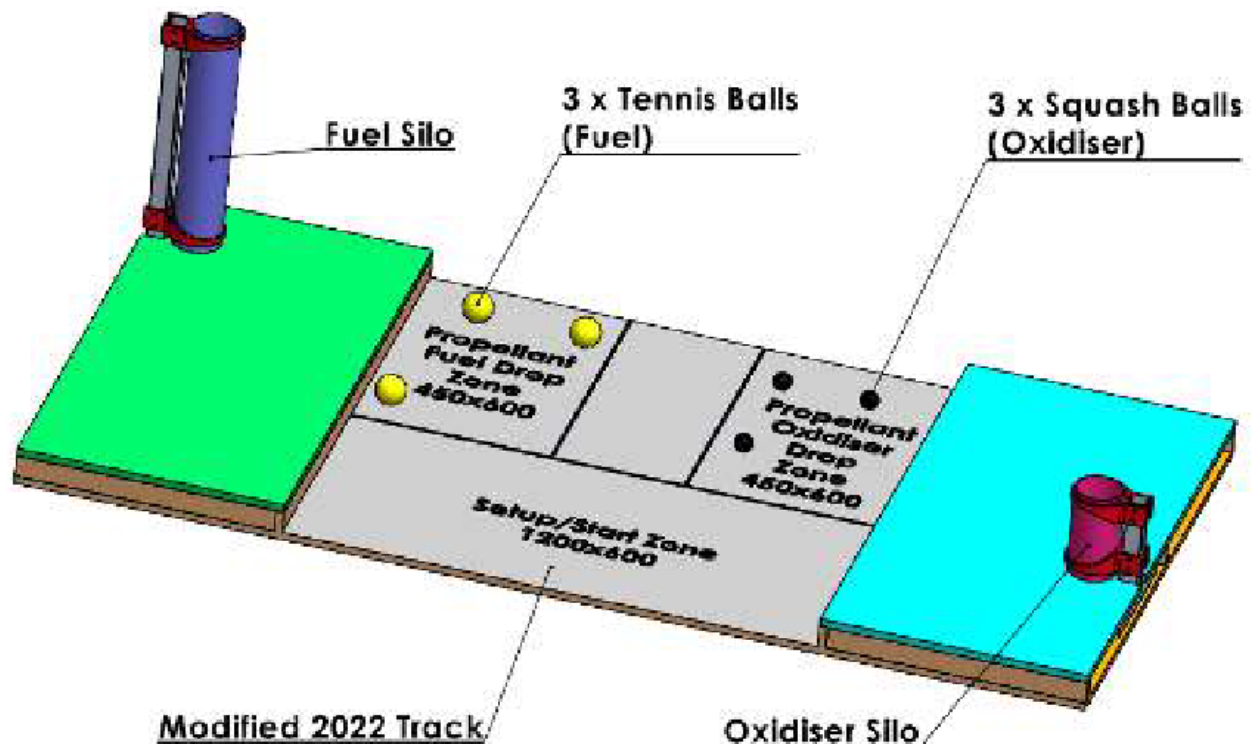



Figure 2.1: The track in which the competition takes place on (taken from the Warman Competition Rules)

2.2 The Team & Roles

Table 2.1: Team members and their roles

Name	Role	Requirement of Role	Photo
Chris Finos	Team leader	In charge of organizing team meetings and planning deadlines for tasks. Will also be aiding in the programming of the project. Programming the collection arm system.	
Ryan Kembrey	Lead Designer	The lead designer of the robot. Responsible for designing the overall functionality of the robot, developing subsystems in CAD & assisting in construction.	
Jack Gruber	Lead Programming	In charge of the programming side of the assignment, being responsible for the coding of the motors. Programming the movement for the robot. In charge of 3D printing custom parts.	
Bashar Yaacoub Agha	Lead Construction	Role focused on creating the physical body of the machine both in person and through CAD. In charge of providing the materials to build the body. Assembling the robot's subsystems. Assembling the robot in Solidworks.	
Seongkyu Choi (Kevin)	CAD Team	Role focused on creating the physical body of the machine both in person and through CAD. In charge of providing the materials to build the body.	

2.3 Final Design Overview

Below (Figure 2.2) is a photo of the final design, which is about 90% finished in construction.

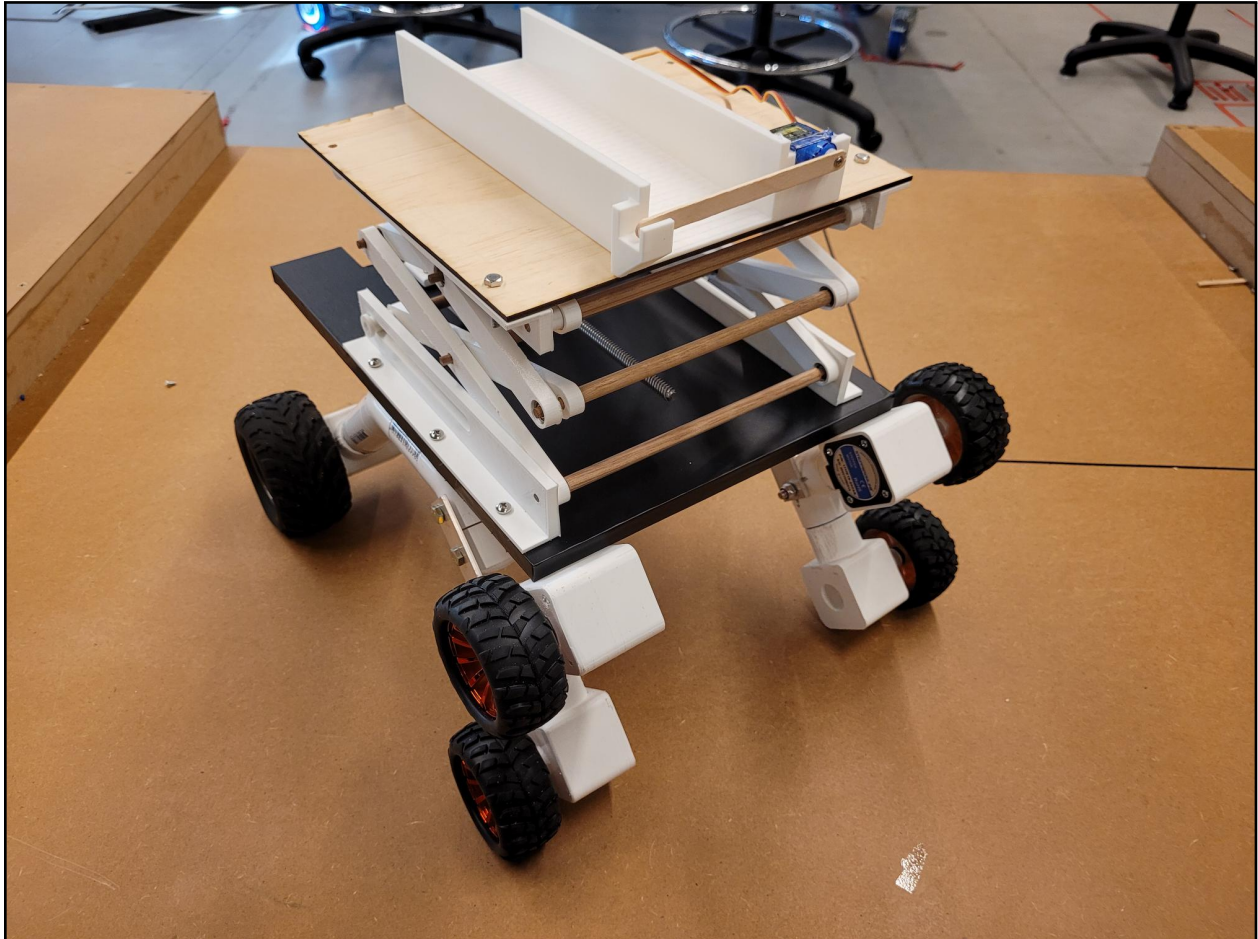


Figure 2.2: Photo of the final design of the robot

The main idea behind the robot is to drive to the fuel cells, collect them, drive to the silo, deposit them. This is then repeated for the oxidiser cells. To accomplish this, the robot was divided into multiple subsystems; one for each of the following tasks.

1. Driving around the track
2. Collecting the balls
3. Elevating the balls
4. Depositing the balls

The members of the group each took charge of a subsystem, with two members doubling up on ball collection. After some ideation, prototyping and testing, the final designs for each subsystem were created.

2.3.1 Chassis

The chassis is the main subsystem where it takes charge in the maneuverability around the track as well as the base platform where all other subsystems including the scissor lift, deposit system and collection system are directly attached onto.

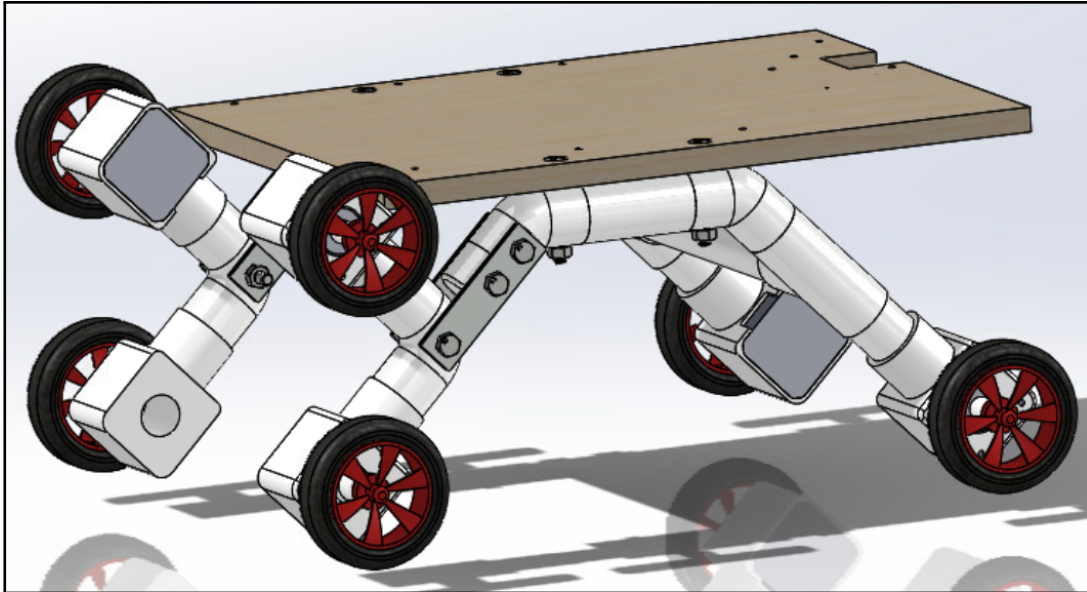


Figure 2.3: The CAD of the chassis, designed by Bashaar

2.3.2 Scissor Lift

The scissor lift raises the balls to the right height to be deposited at their respective silo. It is powered by a Nema 17 turning a lead screw.

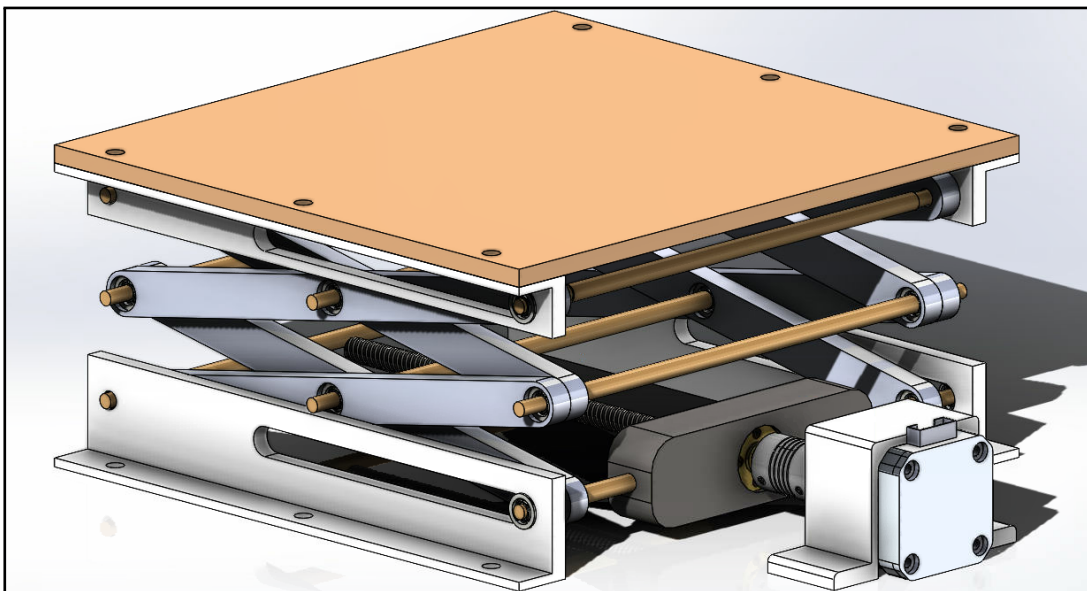


Figure 2.4: The CAD of the chassis, designed by Ryan (myself)

2.3.3 Collection Arm

The collection arm as the name suggests is the component responsible for picking up the tennis balls and squash balls. The balls collected by the arm will need to be sent to the deposit system and as such, the arm will rotate up around its axis to the collection system. As such, there will be a motor attached to the arm that will be responsible for lifting it from the track to the deposit system.

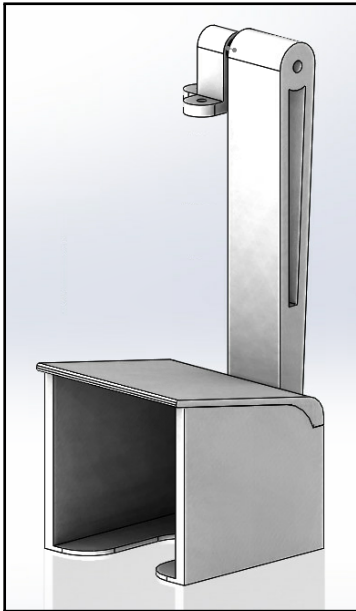


Figure 2.5: Collection arm dimetric view, designed by Ryan (myself)

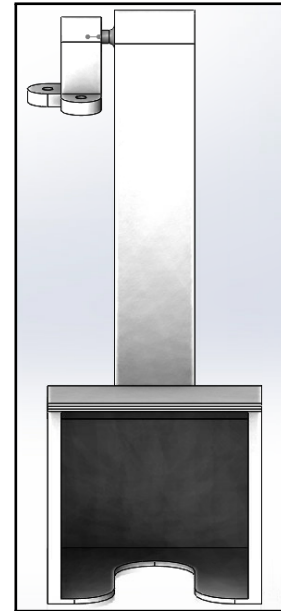


Figure 2.6: Collection arm frontal view, designed by Ryan (myself)

2.3.4 Deposit System

The depositing system is rectangular in shape and the main platform is slightly slanted so when the vessels are deposited, they will roll towards the paddle pop stick attached to the servo as seen in the figure below.

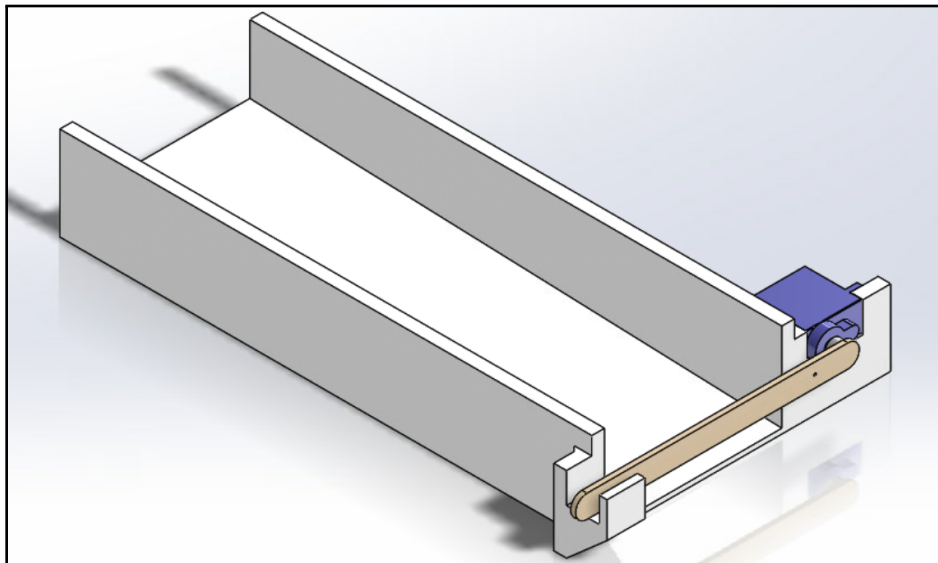


Figure 2.7: The CAD of the deposit system, designed by Ryan (myself)

2.4 Design Requirements

Derived from the [competition rules](#).

2.4.1 Course Requirements

Requirement	Value	Counter Measure	Reference
Function			
Time limit	120 seconds	Take multiple balls at once	R 28.
Attempts	2	Test reliability	R 8.
Track, Equipment and Environment			
Length of track	1200 mm	Robot can reach/drive	Dwg 1-3
Width of track	2400 mm	Robot can reach/drive	Dwg 3
Height of ledge	68 mm	Robot can climb ledge or reach over ledge	Dwg 5
Number of tennis balls	3	Storage space for 3 balls	G 21.
Number of squash balls	3	Storage space for 3 balls	G 22.
Robot			
Flight	No	Ensure robot contacts track	G 29.
Throwing	Allowed	Consider as a design option	G 32.
Remote control	No	Hardcode or feedback sensor loop	FAQ 3
Device is autonomous	Yes	Hardcode or feedback sensor loop	FAQ 3
Fit inside 400 mm cube	Yes	Ensure designs don't exceed 400 mm	G 30.
Weight	6 kg	Lightweight materials. Keep robot simple	R 10.
Compressed gas	Allowed	Consider as a design option	G 6.
Separate subsystems	No	Design robot to function as a single body	G 33.
Arduino Microcontroller	Allowed	Consider using an arduino to control motors	R 5.
Possesion of oxidiser & fuel	No	Code not hold balls at same time	G 33.
Safety			
Fuse	Yes	Implement a fuse	G 5.
Protective clothing	Yes	Saftey glasses, enclosed shoes	G 3.

Table 2.2: Requirements for the Warman Challenge

Part III

Individual Documentation

3 | Ideation

3.1 Ideation

3.1.1 Crazy 8s

During class, an activity called "Crazy 8's" was used to kick-start the engineering ideation process. In this activity, each member of the team created possible design solutions under a time limit.

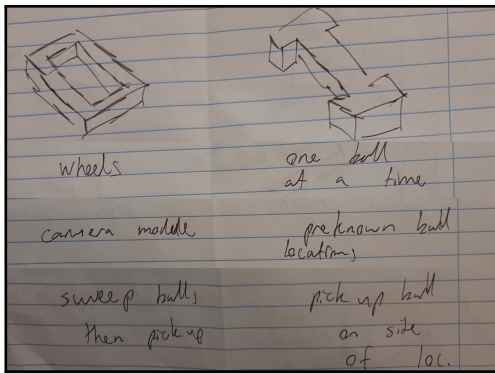


Figure 3.1: Pt.1 Crazy 8's 16 Designs

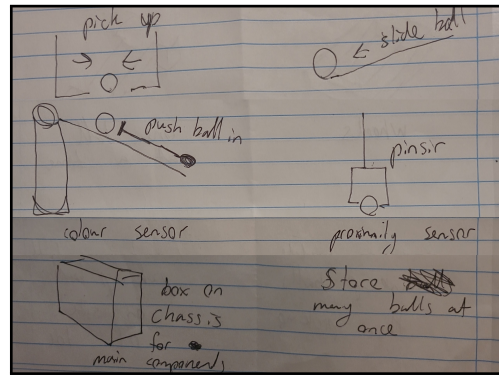


Figure 3.2: Pt.2 Crazy 8's 16 Designs

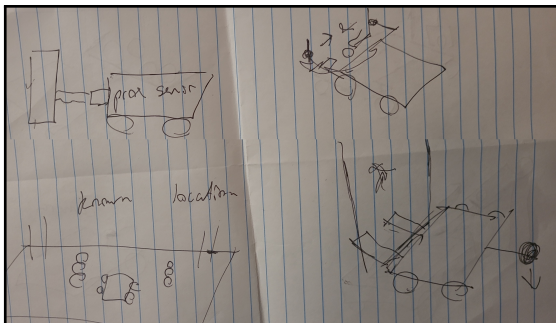


Figure 3.3: Pt.1 Crazy 8's 8 Designs

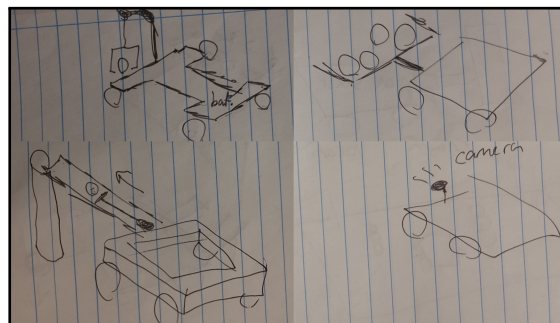


Figure 3.4: Pt.2 Crazy 8's 8 Designs

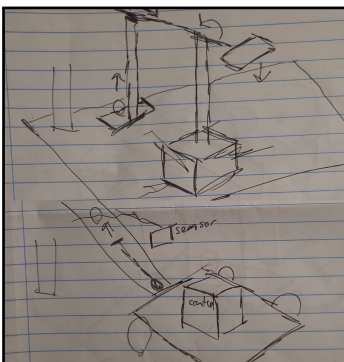


Figure 3.5: Pt.1 Crazy 8 Final 4

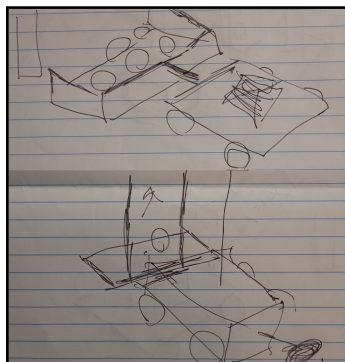


Figure 3.6: Pt.2 Crazy 8 Final 4

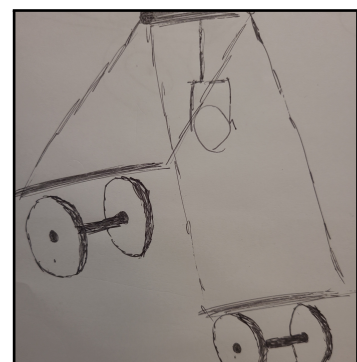


Figure 3.7: Pivoting Triangular

3.1.2 Developed Solution

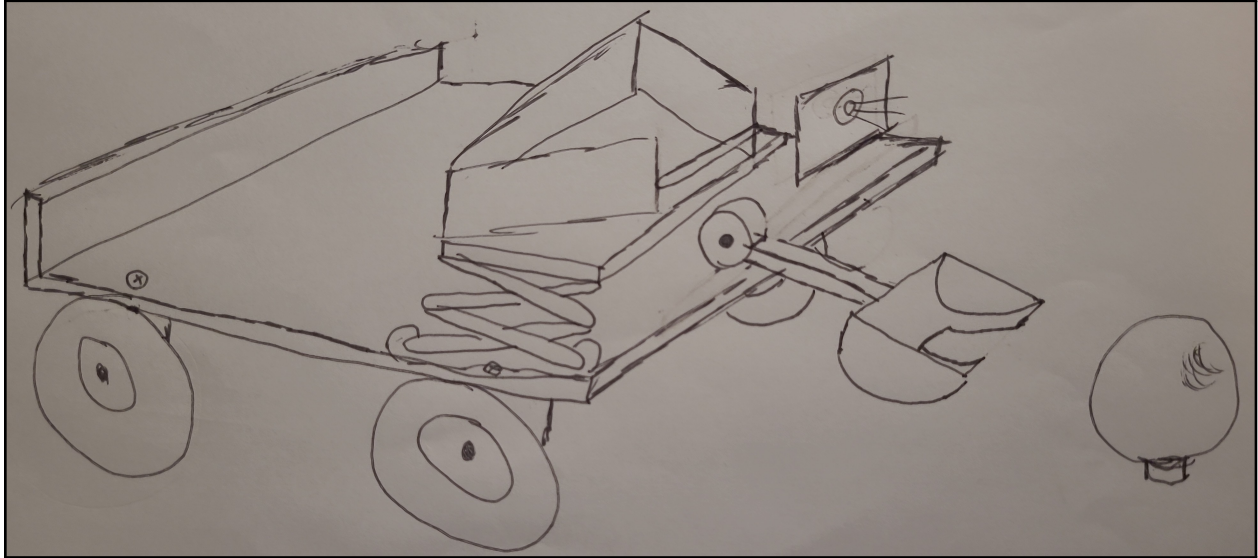


Figure 3.8: Rough Developed Solution

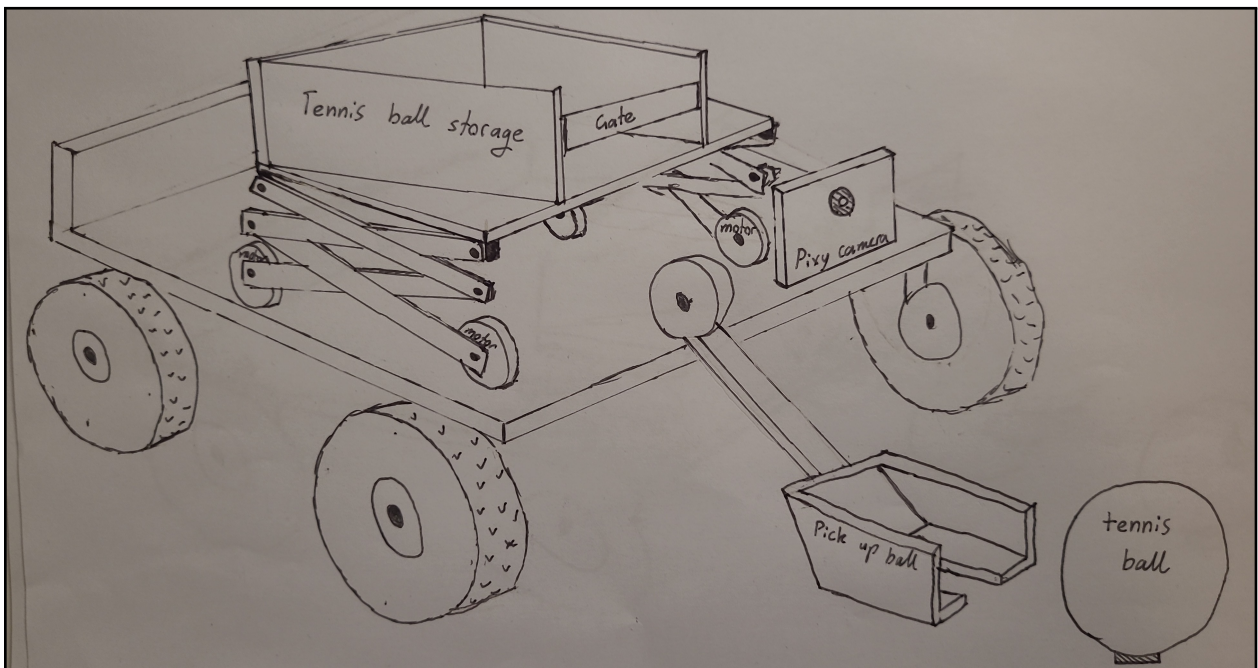


Figure 3.9: Detailed Developed Solution

3.2 Compare Subsystems

My allocated subsystems to explore were storage and elevation. Given the complexity of the final chosen elevation design, and how crucial it was to the success of the robot, it was decided that the elevation component would just be focused on.

The following contains ideas for storage and elevation solutions.

3.2.1 Storage - Solutions

The purpose of the storage system is to provide a secure way to hold the balls between the motions of collecting and depositing. Six preliminary solutions were created for the storage subsystem.

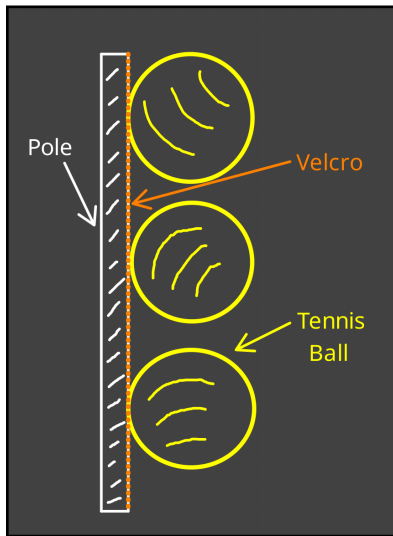


Figure 3.10: Side View Of Velcro Solution

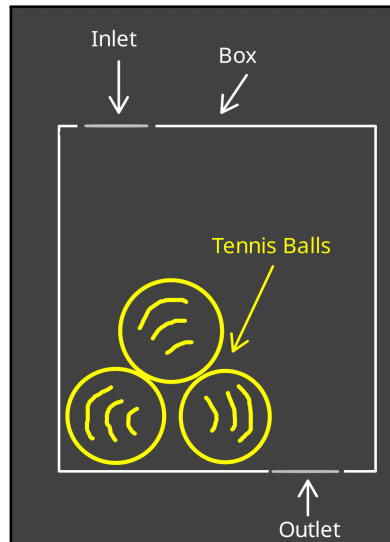


Figure 3.11: Front View Of Box Solution

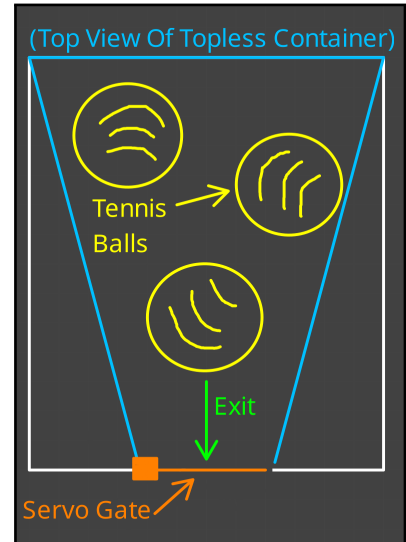


Figure 3.12: Top View Of Topless Container Solution

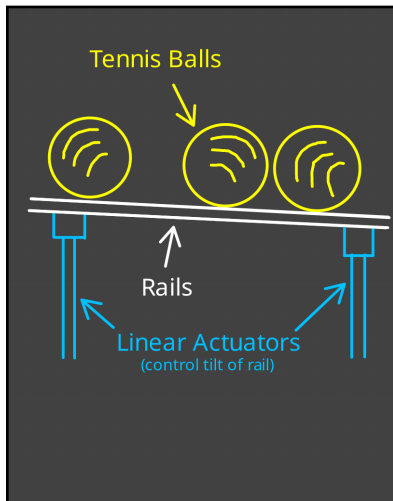


Figure 3.13: Side View Of Rail Solution

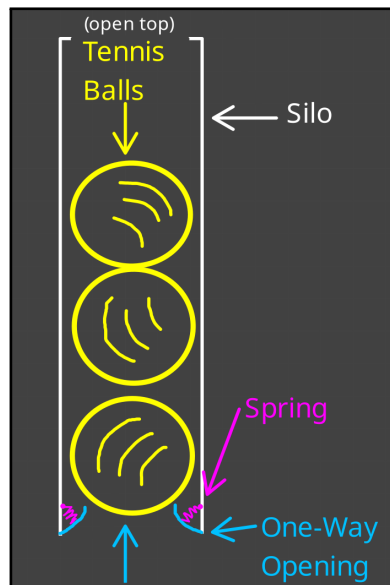


Figure 3.14: Side View Of Silo Solution

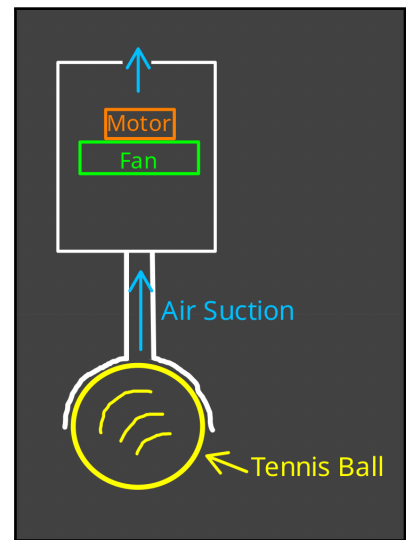


Figure 3.15: Side View Of Vacuum Solution

3.2.2 Storage - Score Matrix

The storage solutions were compared in a score matrix. The top three scorers were velcro, box and topless container. The velcro scored quite high in all categories except for versatility, as it would not be able to pick up squash balls, only tennis balls. The topless container had all the advantages of the box design, but with less weight and more versatility. As the highest scoring design, the topless container was chosen for incorporation into the robot.

Table 3.1: Scoring Matrix for Solutions

Solution	Buildability	Cost	Power Use	Versatile	No. Parts	Weight	Total Score
Velcro	5	5	5	0	5	5	25
Box	4	5	5	4	5	3	26
Topless Container	4	5	5	5	4	4	27
Rail	2	2	3	4	3	3	16
Silo	4	5	5	4	4	4	26
Vacuum	1	1	1	3	1	2	9

3.2.3 Elevation - Solutions

The purpose of the elevation system is to provide a way for the balls to be risen to the required height so that they may be deposited into the silo. Three designs were created.

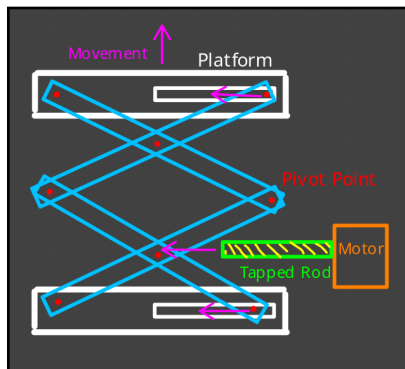


Figure 3.16: Side View Of Scissor Lift Solution

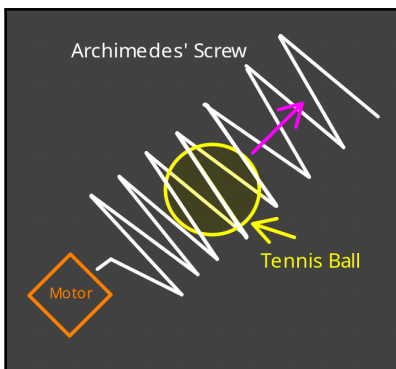


Figure 3.17: Side View Of Archimedes' Screw Solution

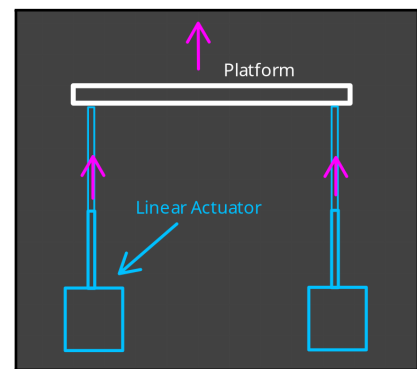


Figure 3.18: Side View Of Linear Actuator Solution

3.2.4 Elevation - Score Matrix

In the scoring matrix (Table 3.2), the highest scorer was the Archimedes' screw, mainly due to its simplicity and few number of parts.

The elevation subsystem was chosen to be prototyped, where the designs will be scrutinised under testing and maths. The score matrix will then be re-evaluated to determine which design should be chosen to include in the robot.

Table 3.2: Elevation scoring matrix for solutions

Solution	Buildability	Cost	Power Use	Versatile	No. Parts	Weight	Total Score
Scissor Lift	3	3	3	4	3	3	19
Archimedes' Screw	4	4	4	3	4	5	24
Linear Actuator	3	3	2	3	3	3	17

4 | Prototyping

4.1 Low Fidelity Prototypes

4.1.1 Linear Actuator & Box

The linear actuated design consists of 4 actuators that thrust the platform vertically upwards. Assuming a load (W) of 0.5kg, the force required from each actuator is as follows, where n is the number of actuators.

$$\begin{aligned} F &= \frac{W}{n} \\ &= \frac{0.5}{4} \\ &= 0.125\text{N} \end{aligned} \tag{4.1}$$

Alternatively, three actuators could be used with would yield a required force of 0.167N.



Figure 4.1: Linear actuated prototype



Figure 4.2: Linear actuated prototype

Linear actuators are expensive, with the lowest price examples starting at about \$100 AUD. Due to the high price, to fit within a reasonable budget only a single linear actuator could be used. The minimum required thrust (N) for a single actuator is equal to that of the load on the platform, which means the actuator must be able to produce a thrust of 4.905 N, which is well below what most budget linear actuators can produce.

If only one actuator were to be used, a system would have to be designed to provide support for the platform, so it does not tilt when the load is not at the center.

4.1.2 Scissor Lift & Topless Container

The scissor lift prototype aims at creating a basic design for how the parts will interact to create to upward motion of the platform holding tennis balls. Top and bottom joints of the scissor lift will each have one pivoting joint and a sliding mechanism, so that the Δx between the two joints may increase and decrease with their respective motions. The lift would be powered by a stepper motor turning a lead screw. The force



Figure 4.3: Scissor Lift & Topless Container Phase 1

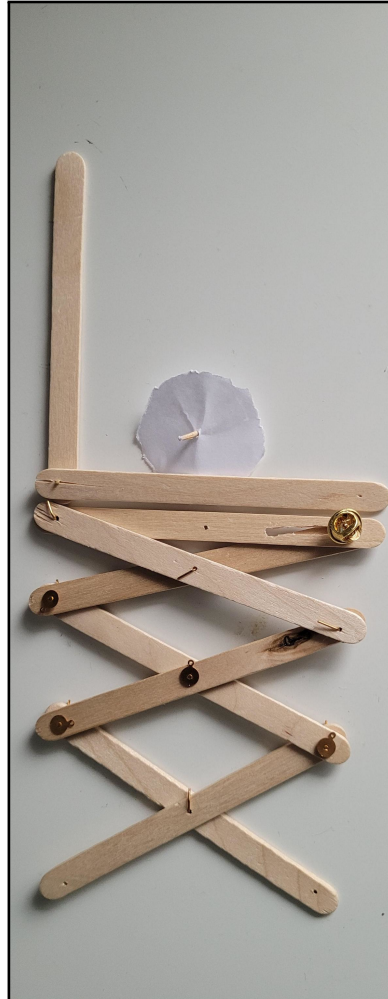


Figure 4.4: Scissor Lift & Topless Container Phase 2

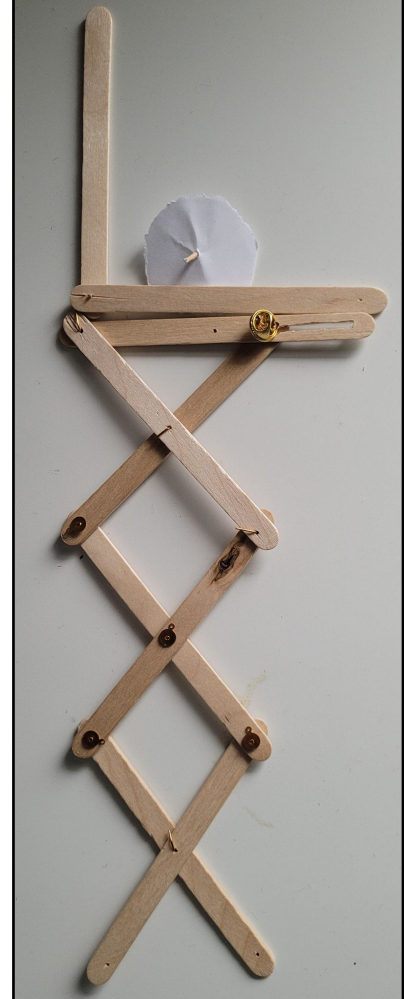


Figure 4.5: Scissor Lift & Topless Container Phase 3

that the lead screw must overcome is as follows, when the assumed mass is 2kg and the angle of the lift is 10 degrees.

$$F = \frac{W}{\tan(\theta)}$$

$$F = \frac{2 \times 9.81}{\tan(10)}$$

$$F = 111.27\text{N}$$

The force required at the leg to raise the lift is 111.27N, which is a high value. Design considerations should be taken into account to minimise the weight on the scissor lift, and the dimensions of the scissor lift to ensure a high enough force can be produced by the lead screw.

4.1.3 Archimedes' Screw & Silo

The Archimedes' screw and silo designs were combined into a single prototype. The subsystem works but inputting the balls at the bottom of the screw. The screwing motion will then push the balls upwards.

However, the only way this mechanism could work is if the tennis ball was fit extremely snugly into the screw. Even if we were able to achieve this, the squash balls could still not be lifted by the Archimedes' screw. Based on this major downside, the Archimedes' screw will have to be redesigned to work properly.

Below is a proposed new design that is based off how water is lifted by an Archimedes' screw.



Figure 4.6: Archimedes' screw prototype

To calculate the required torque for the motor, a height risen per revolution must be chosen. For this example, the load is 0.5kg (4.905N), the height per revolution (lead) is 70mm, the screw will be perpendicular to the ground, the mean diameter (d_m) is 120mm and the coefficient of friction (μ) is 0.2.

$$\tau = F \cdot \frac{d_m}{2} \cdot \left(\frac{L + \pi \cdot \mu \cdot d_m}{\pi \cdot d_m - \mu \cdot L} \right) \quad (4.2)$$

$$\tau = 4.905 \cdot \frac{0.12}{2} \cdot \left(\frac{0.07 + \pi \cdot 0.2 \cdot 0.12}{\pi \cdot 0.12 - 0.2 \cdot 0.07} \right)$$

$$\tau = 11.8 \text{ N-cm}$$

The required torque of the stepper motor is 11.8 N-cm, which is an acceptable torque to produce by a single motor at high RPM. Based on these calculations, a NEMA 17 would be a suitable stepper motor, with at least a torque of 20 N-cm at around 300 RPM.

Based on the nature of the Archimedes' screw, the height the balls must be lifted is the minimum height of the Archimedes screw. The design is unable to reduce its size and expand when needed to meet the 400x400x400mm size design requirement, mentioned in the design requirements table (2.2). This design flaw will be reflected in the Archimedes' screw size score in the table on the following page (4.1).

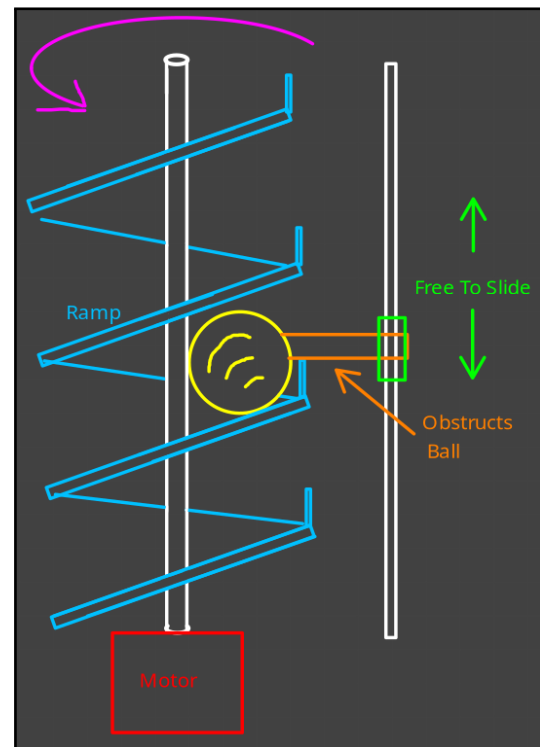


Figure 4.7: Version 2 of the Archimedes' Screw

4.1.4 Prototypes Score Matrix

After the prototyping phase and some calculations, the score matrix for the elevation designs has been re-evaluated based on objective testing and math. A score for size has also been introduced.

Table 4.1: Prototypes scoring matrix for solutions

Solution	Buildability	Cost	Power	Versatility	Parts	Weight	Size	Score
Scissor Lift	3	3	3	4	3	3	4	22
Archimedes' Screw	3	3	3	1	4	4	2	20
Linear Actuator	3	1	3	3	3	3	5	21

Based off the results, it was chosen to design a scissor lift to carry the balls to the right height to deposit.

5 | Scissor Lift Design (Artefact 1)

5.1 First Design

5.1.1 Leg Length

At its lowest point, the chosen height for a single set of legs (see figure 5.2 for what a leg set is) is 40mm. This is because the flatter the legs are of the scissor lift at its lowest position, the more torque required to lift it. Since there is two sets of legs per side of the scissor lift, the total lowest height will be 80mm, seen in figure 5.1.

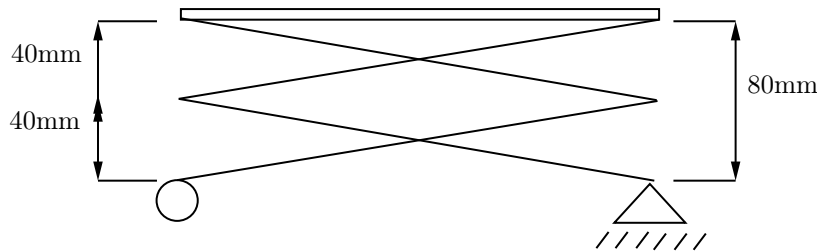


Figure 5.1: Lowest height of scissor lift

At the scissor lifts max height, the distance between the pin joint and the roller joint is 100mm (depicted in figure 5.2). It is important to not make this value too small, or the scissor lift will likely becomes too unstable at its highest point.

The total height the scissor lift must lift the balls given by the following equation.

$$\begin{aligned}
 \text{total} &= \text{silo} - (\text{chassis} + \text{minimum scissor}) \\
 &= 530 - (120 + 80) \\
 &= 330\text{mm}
 \end{aligned}
 \tag{5.1}$$

This assumes the chassis height is 120mm, but it is likely the chassis will end up being closer to 150mm. By overshooting how much height the scissor lift needs to gain, this allows flexibility in the chassis design should it fail to climb the ledge. The smallest silo is 200mm high, and the lowest dropping point possible by the scissor lift is 80mm above the chassis. This means that the squash balls will likely have to fall from a distance into their silo.

To lift the balls 330mm, each set of legs needs to raise in 165mm. Since the lowest height of a set is 40mm, the maximum height for a set must be 205mm. In summary, the max height of a set must be 205mm and its width at max height is 100mm.

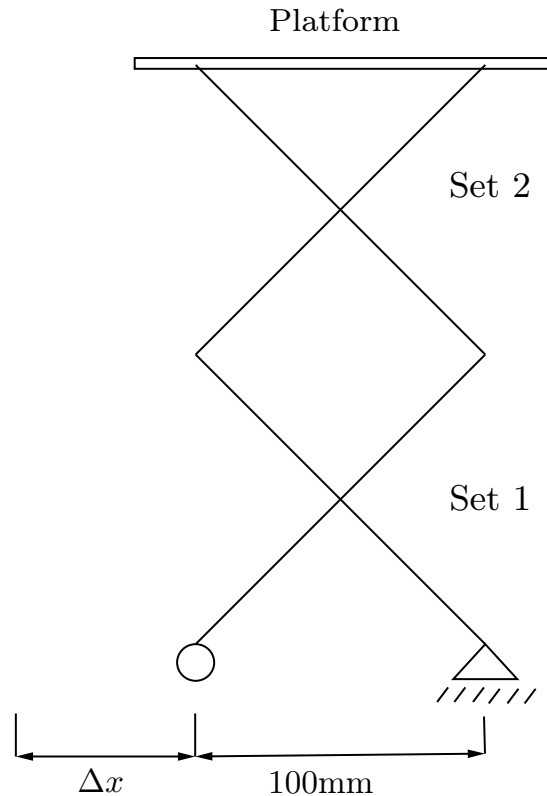


Figure 5.2: Smallest x (100mm) of scissor lift

To find the required leg length, these values can be placed into Pythagoras' Theorem.

$$c^2 = a^2 + b^2$$

$$c = \sqrt{a^2 + b^2}$$

Substituting the scissor lift max height values in gives the leg length (L).

$$L = \sqrt{205^2 + 100^2}$$

$$= 228.09\text{mm} \quad (5.2)$$

Additionally, the length Δx can be found, where h_{low} is the lowest height and x_{small} is the smallest width.

$$\Delta x = \sqrt{L^2 - h_{low}^2} - x_{small}$$

$$= \sqrt{228.09^2 - 40^2} - 100$$

$$= 124.55\text{mm} \quad (5.3)$$

Interestingly, the angle for every point x the scissor lift moves represents a non-linear relationship. This gives some insight as to why a scissor lift with a lowest height less than 40mm is not viable for this robot, as the motor would have to overcome the steeper part curve (explored further in section 5.1.3).

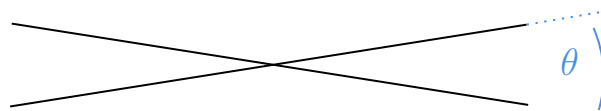


Figure 5.3: The angle theta of the scissor lift.

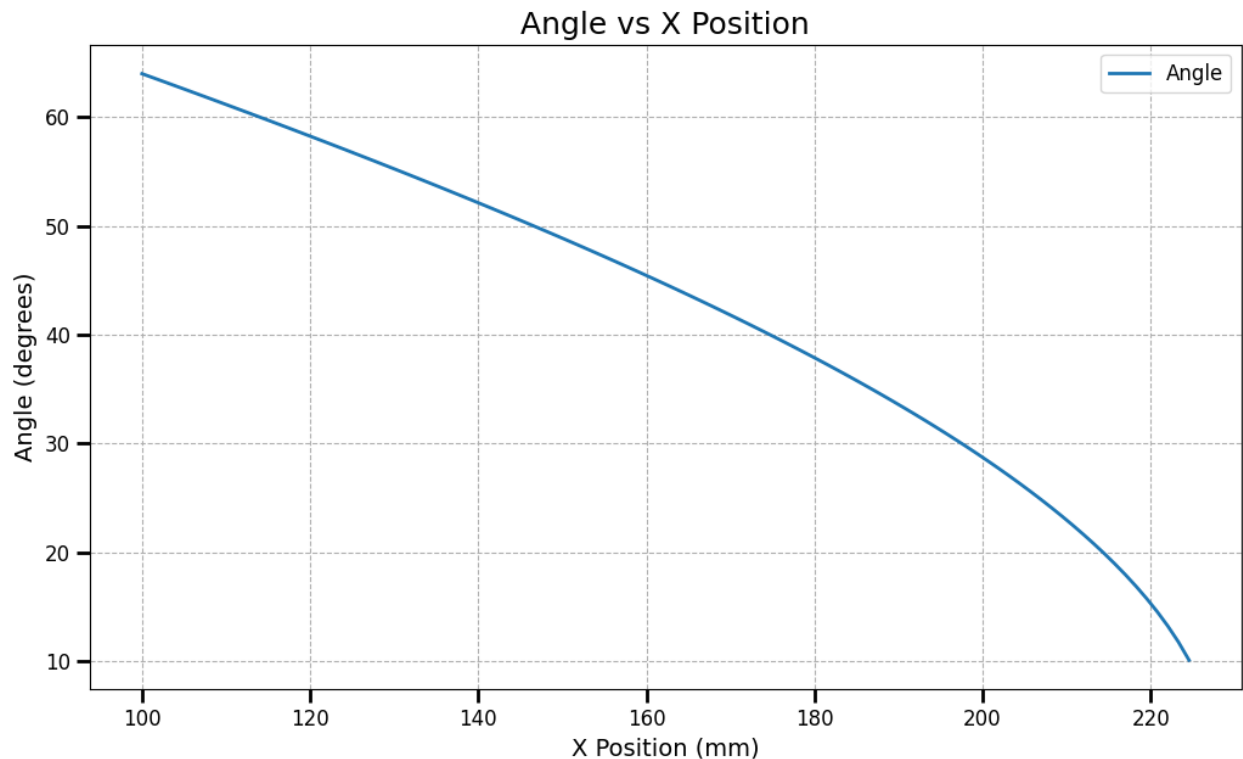


Figure 5.4: The angle θ for every position x on the scissor lift

5.1.2 Force

To find the force required (F_{req}) to hold the scissor lift at angle θ , we can use the equation below, where W is the total load at the center of the platform.

$$F = \frac{W}{\tan(\theta)} \quad (5.4)$$

The maximum weight for a tennis ball is 59.4 grams. The heaviest load the scissor lift will carry will be three tennis balls plus the weight of the scissor lift itself, which will be over-estimated to be 2kg. Assuming its center of mass is at the point W , the total load can be found.

$$\begin{aligned} W &= g(m_{balls} + m_{lift}) \\ &= 9.81(3 \cdot 0.0594 + 2) \\ &= 21.37\text{N} \end{aligned} \quad (5.5)$$

Now substituting in the value for W into equation 5.4 gives:

$$F = \frac{21.37}{\theta} \quad (5.6)$$

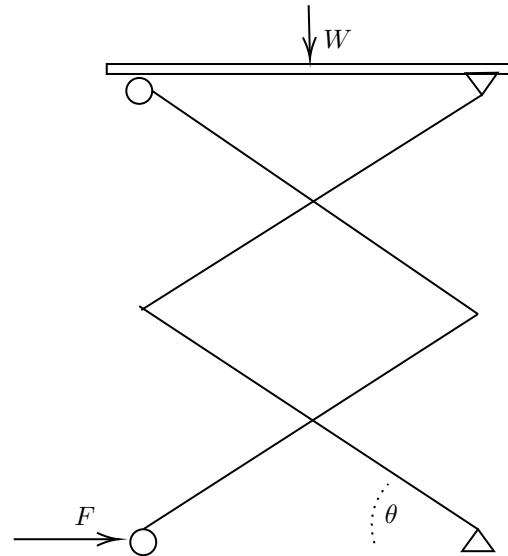


Figure 5.5: FBD of scissor lift

The force for every angle of the scissor lifts legs was calculated using python.

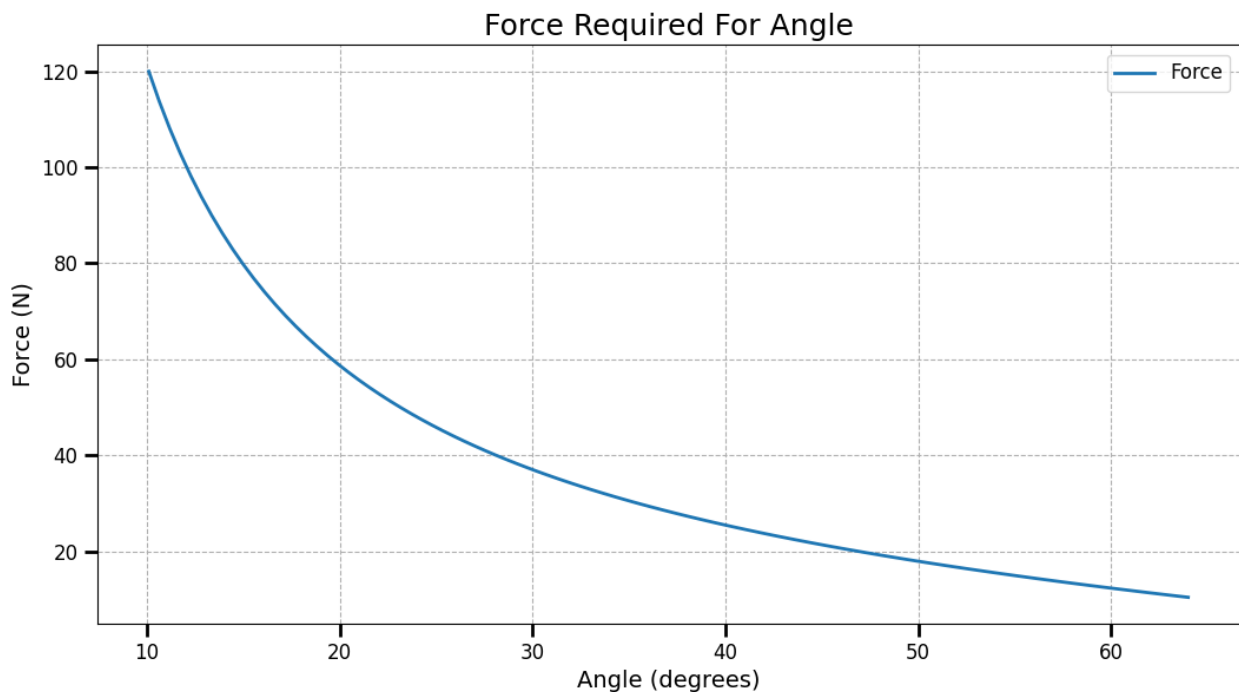


Figure 5.6: The force required for every angle on the scissor lift

5.1.3 Torque

To calculate the torque (τ) required by the stepper motor for every force, the lead screw formula can be used.

$$\tau = \frac{Fd_m}{2} \left(\frac{l + \pi f d_m \sec(\alpha)}{\pi d_m - f l \sec(\alpha)} \right) \tag{5.7}$$

I have chosen an [8mm lead screw](#). The following table shows what each symbol in the equation is, plus its corresponding value with respect to the chosen lead screw.

Table 5.1: Lead screw equation values

Symbol	Description	Value
F	Force	-
l	Lead	8mm
d	Diameter	8mm
d_m	Mean Diameter	6mm
p	Pitch	2mm
α	Thread Angle	14.5°
f	Coefficient of Friction	0.15

To find the mean diameter, the following equation was used, where the diameter is (d) and the pitch is (p).

$$\begin{aligned} d_m &= d - 2 \left(\frac{1}{p} \right) \\ &= 8 - 2 \left(\frac{1}{2} \right) \\ &= 6\text{mm} \end{aligned} \tag{5.8}$$

For two lubricated steel surfaces, the coefficient of friction can be assumed to be 0.15.

Subbing in the values into the equation gives the following graphs; torque plotted against force, and torque plotted against the angle θ .

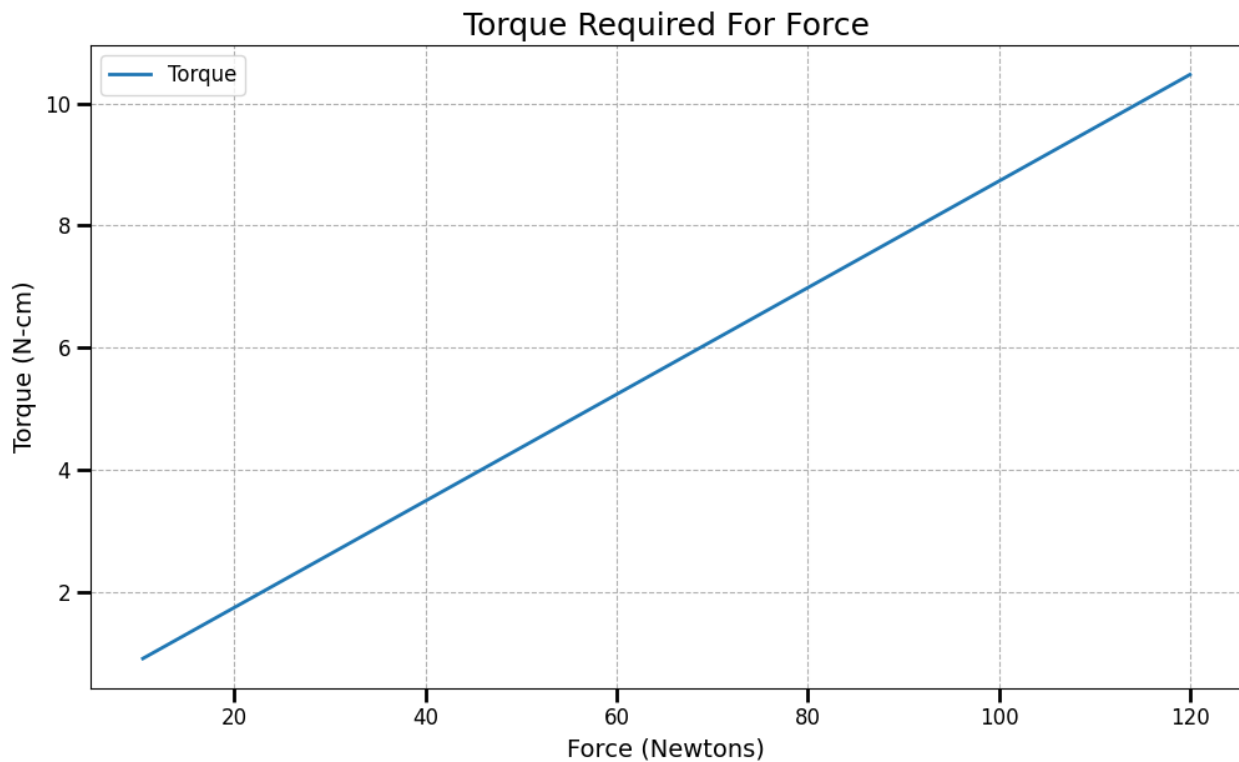


Figure 5.7: The torque required for the force of every position of the scissor lift

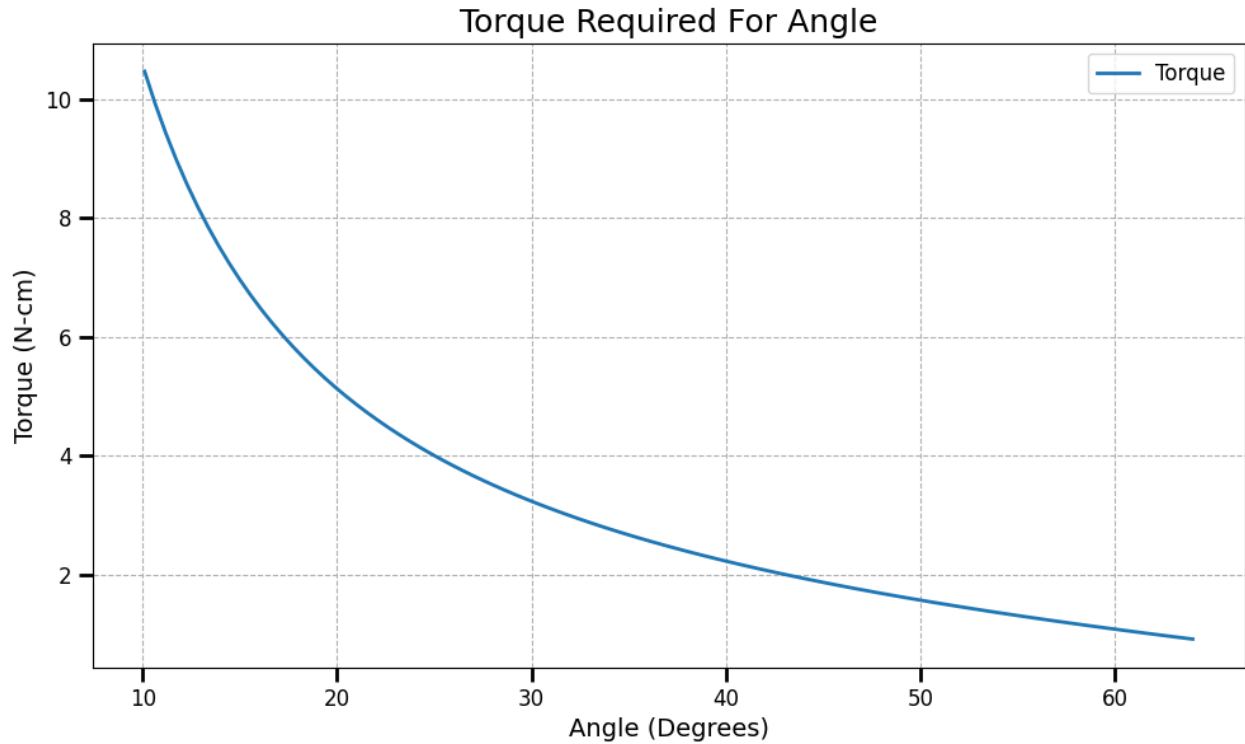


Figure 5.8: The torque required for every angle in the scissor lifts motion

5.1.4 Stepper Motor Chosen

The Nema 17 has a recommended operating RPM of 200-600 RPM, making it a suitable motor for the driver of the scissor lift. The motor will be required to output $> 10\text{N-cm}$ of torque

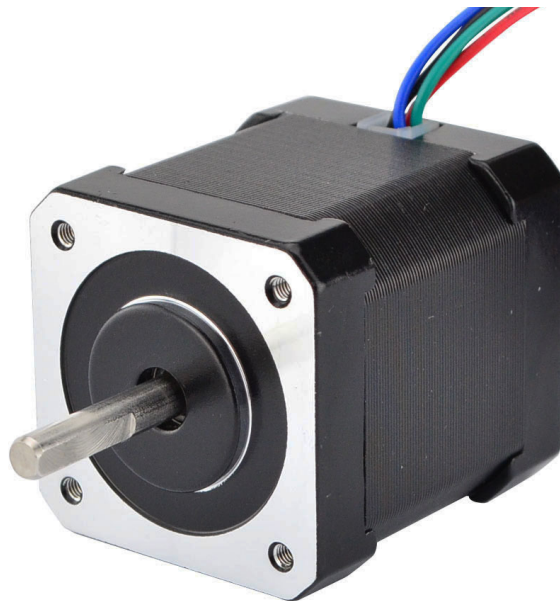


Figure 5.9: Nema 17 stepper motor

5.1.5 CAD

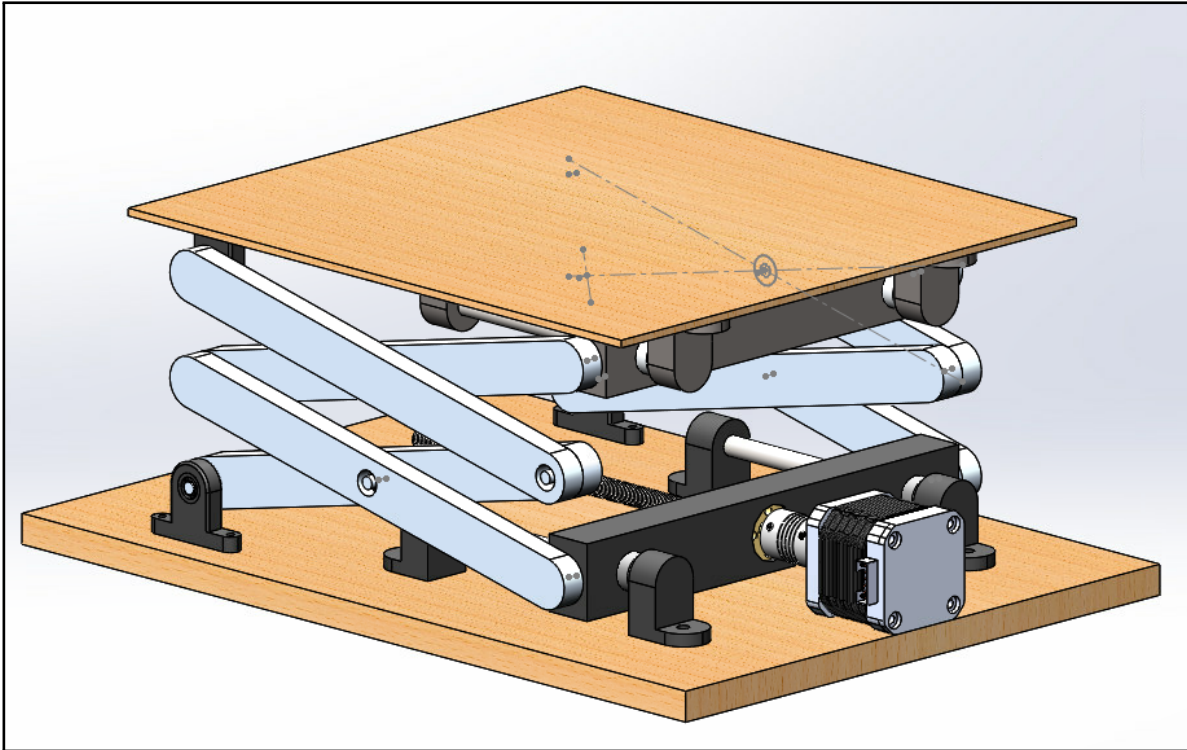


Figure 5.10: First scissor lift in compressed form

5.1.6 Design Flaws

The scissor lift hosted a number of design flaws. Here is a list.

- Too large for the platform to fit the scissor lift and the arm.
- Too heavy (eg four smooth rods)
- Flexible in ways no intended (eg twisting)
- Complexity of build (eg six different configurations for legs)

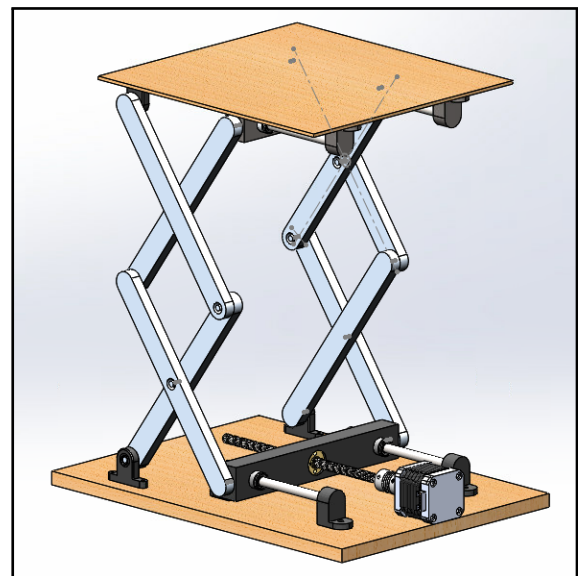


Figure 5.11: First scissor lift in extended form.

5.2 Second Design

The last design process was based on doing calculations first, then designing a CAD that fits those calculations. In the second design process, the CAD will be made first because most of the design issues arose from dimensions being too large. The dimensions will be checked during the CAD making process to make sure no extra length is being used to make the scissor lift as light as possible to fit the weight requirement. The python scripts will be re-run to find the new force, torque and RPM values.

5.2.1 Design Improvements

The following design aims to improve the following:

- Weight reduction
- Reduce complexity of build
- Increase rigidity of design
- Decrease the size of the scissor lift

5.2.2 CAD

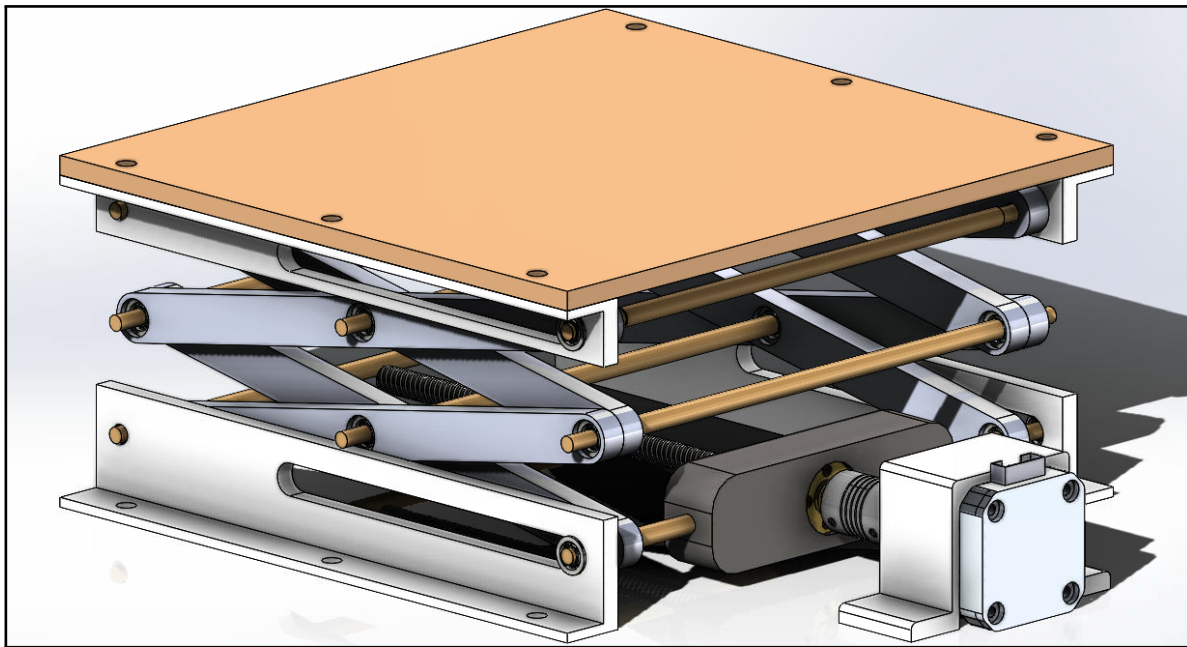


Figure 5.12: Second scissor lift in compressed form

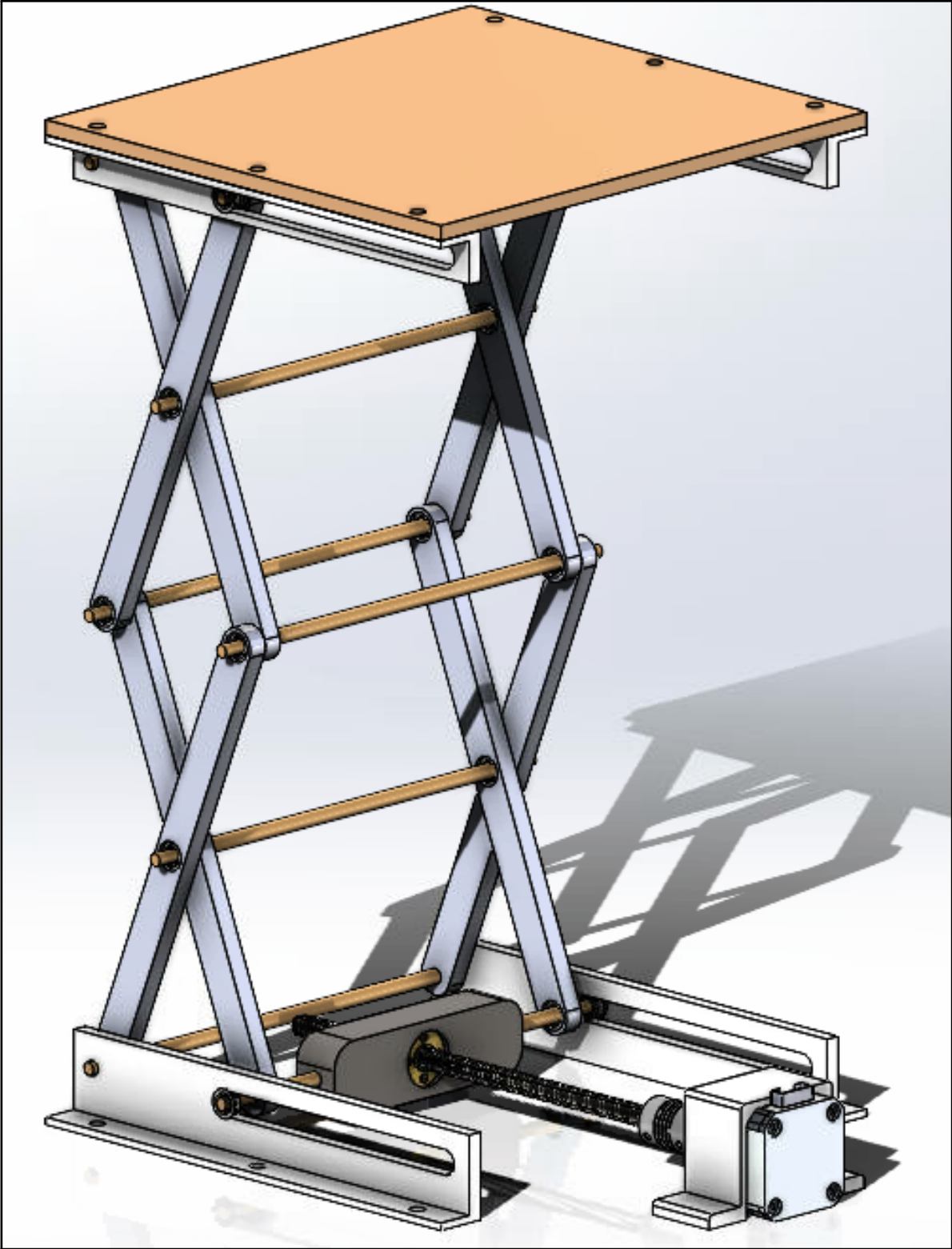


Figure 5.13: Second scissor lift in extended form.

5.2.3 Calculations

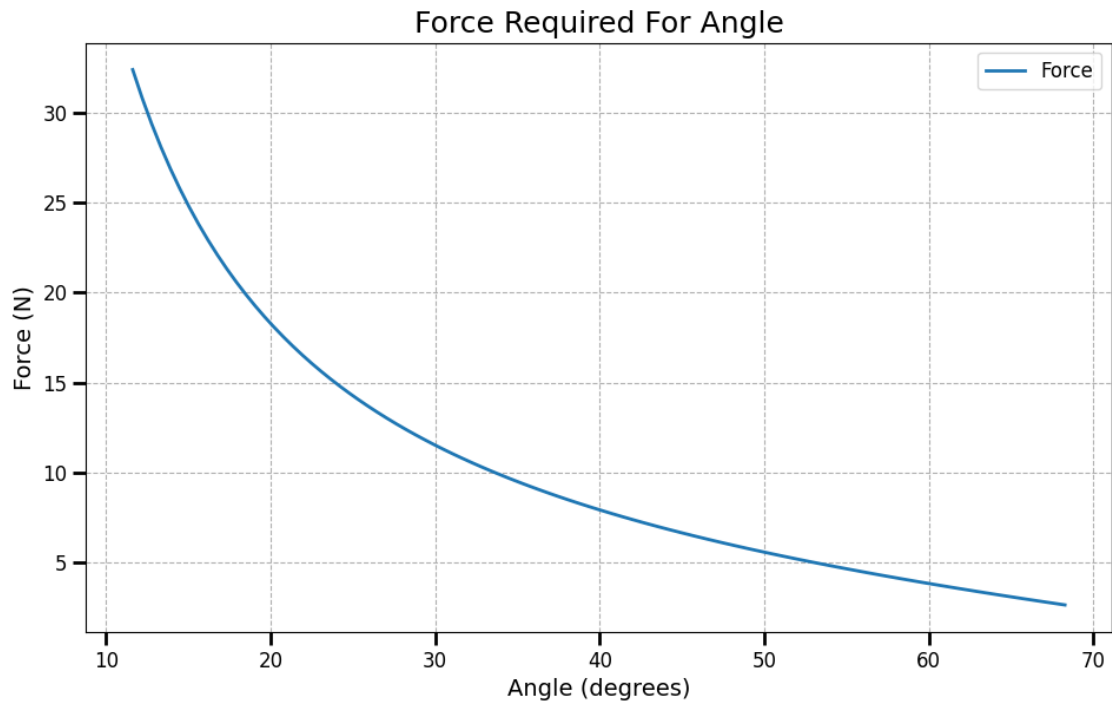


Figure 5.14: Force needed to raise second scissor lift at each angle

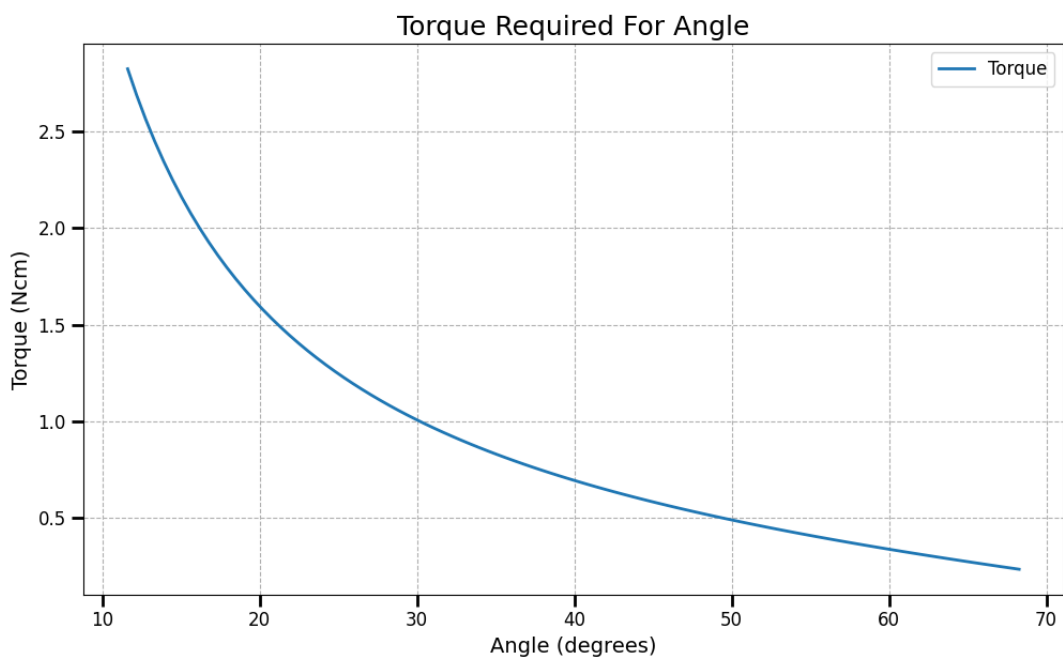


Figure 5.15: Torque needed to raise second scissor lift at each angle

5.2.4 Procurement Plan

The following items will be bought. The following items will be 3D printed.

Table 5.2: Scissor Lift Parts Procurement Table (Bought Parts)

Bought Items				
Item	Amount	Specifications	Purchase Source	Cost (\$)
Nema 17	1	55 N-cm	Core Electronics	20.00
A4988 Driver	1	-	MakerStore	10.00
Lead Screw	1	8mm, 4 start	MakerStore	8.00
Lead Nut	1	8mm	MakerStore	3.00
Flanged Bearings	24	MR105-RS	MakerStore	22.00
Dowel Rod	1	2m	Bunnings	5.00
3mm Plywood	1	3mm	UTS	0.00

Table 5.3: Scissor Lift Parts Procurement Table (Printed Parts)

Self-Manufactured Items				
Item	Amount	Manufacture Method	Source	Cost (\$)
Transfer Bar	1	3D Printed	UTS Protospace	0.00
Legs	8	3D Printed	UTS Protospace	0.00
Rollers	4	3D Printed	UTS Protospace	0.00
Spacers	2	3D Printed	UTS Protospace	0.00
Brackets	4	3D Printed	UTS Protospace	0.00
Nema Bracket	1	3D Printed	UTS Protospace	0.00

5.2.5 Verdict

The second scissor lift design was built, and works as intended. A video of the scissor lift working can be found [here](#).

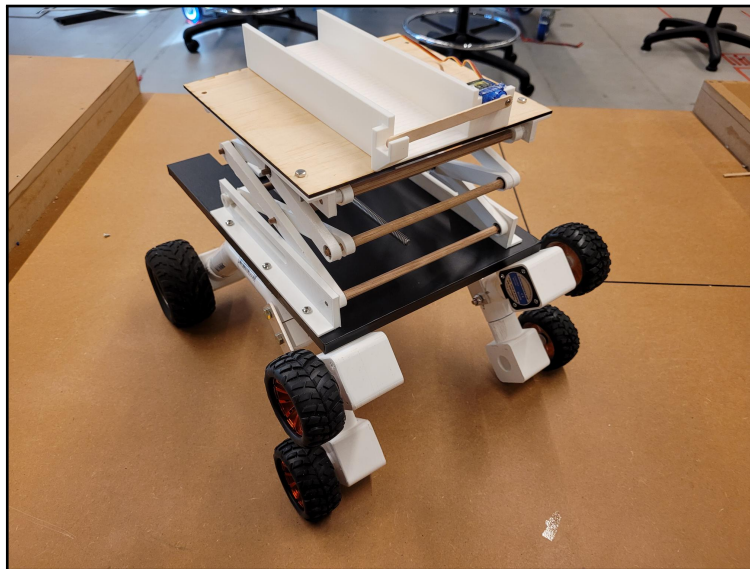


Figure 5.16: Scissor lift on top of the chassis

6 | Collection Arm Redesign (Artifact 2)

6.0.1 Goal

Following the failed design of the collection arm by another team member, I decided to design a backup system. The new design should fix the following design flaws from the previous design.

1. Weight (too heavy for two nema 17s to lift the arm up, even without a ball)
2. Size (did not fit onto the robot and meet the 400x400x400mm cube design requirement)
3. Need for two motors

The arm should also be able to operate off a single nema 17 stepper motor.

6.0.2 Design

The basic design for the new arm was inspired by the following YouTube video, which can be watched [here](#). Below (6.1) is a photo of the collection arm on the robot in the video.

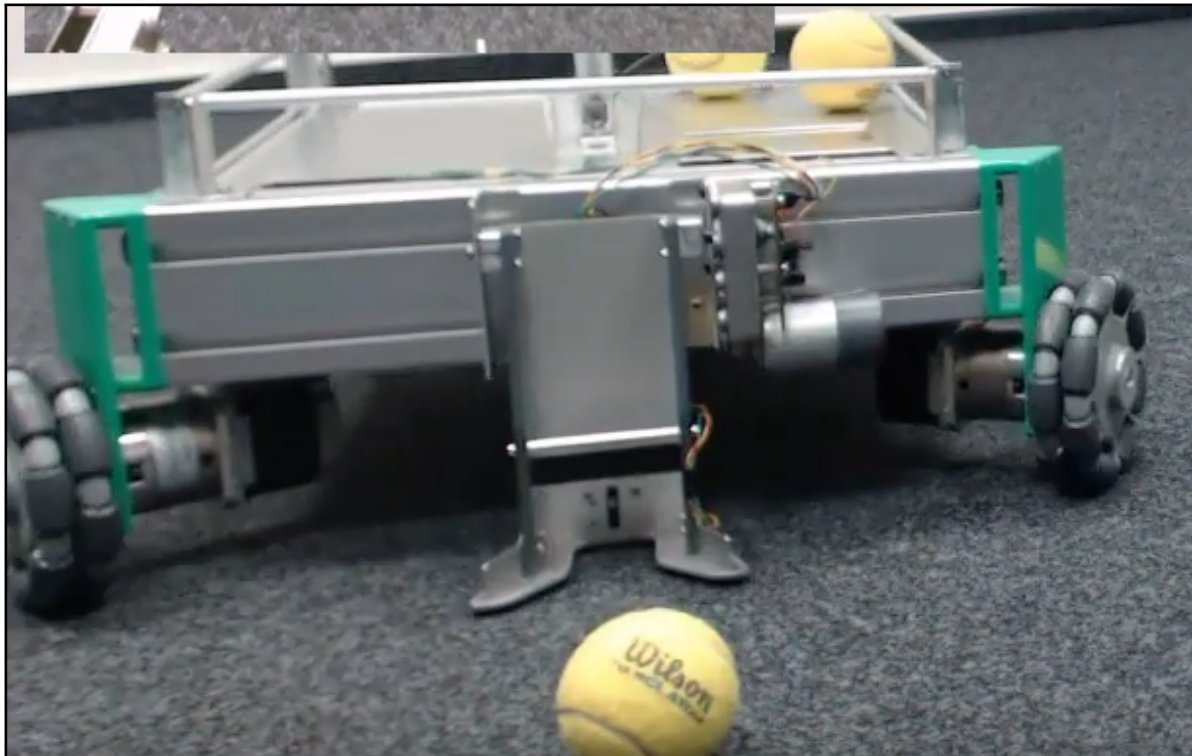


Figure 6.1: Inspiration for the collection arm (Credit: Malte Ahlers)

Firstly, I designed the scooping box. The aim of this box is to be able to slide underneath the tennis or squash ball, then rotate 180° so the ball may roll off the slanted roof into the depositing system. Secondly, I

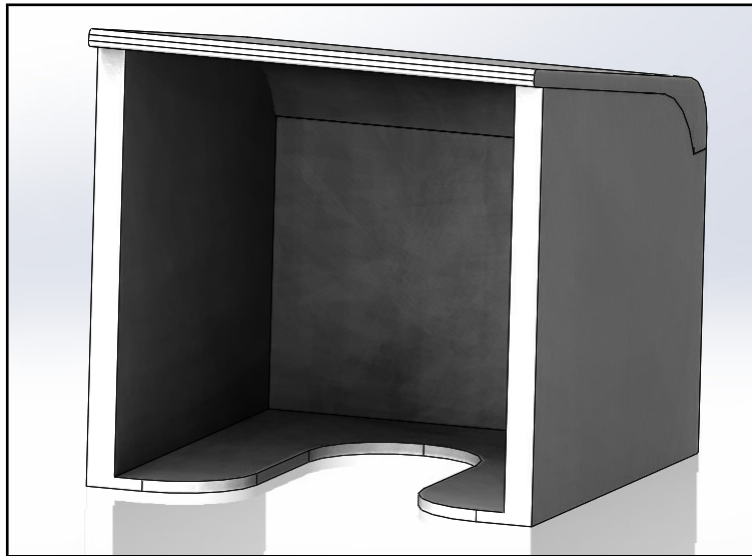


Figure 6.2: Scooping box for collection arm

designed the arm. The nema motor shaft will fit into the hole snugly, so that when the shaft turns, the arm will rotate. This will attach to the scooper box with strong double sided tape.

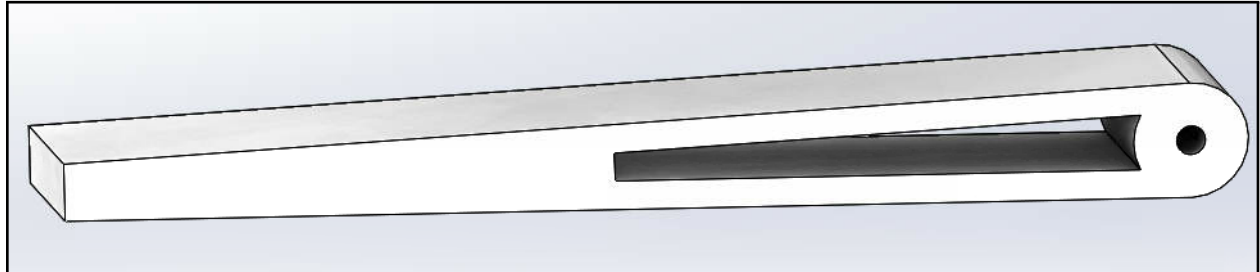


Figure 6.3: Arm piece for collection arm

The arm needed some way to stabilise on the side where the motor was not attached, so I designed an axle support part.

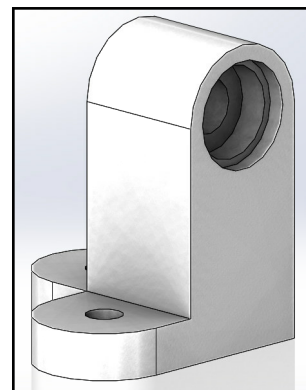


Figure 6.4: Axle support for collection arm

The final design is as follows. A MR105-RS flanged bearing sits inside the axle support to minimise friction.

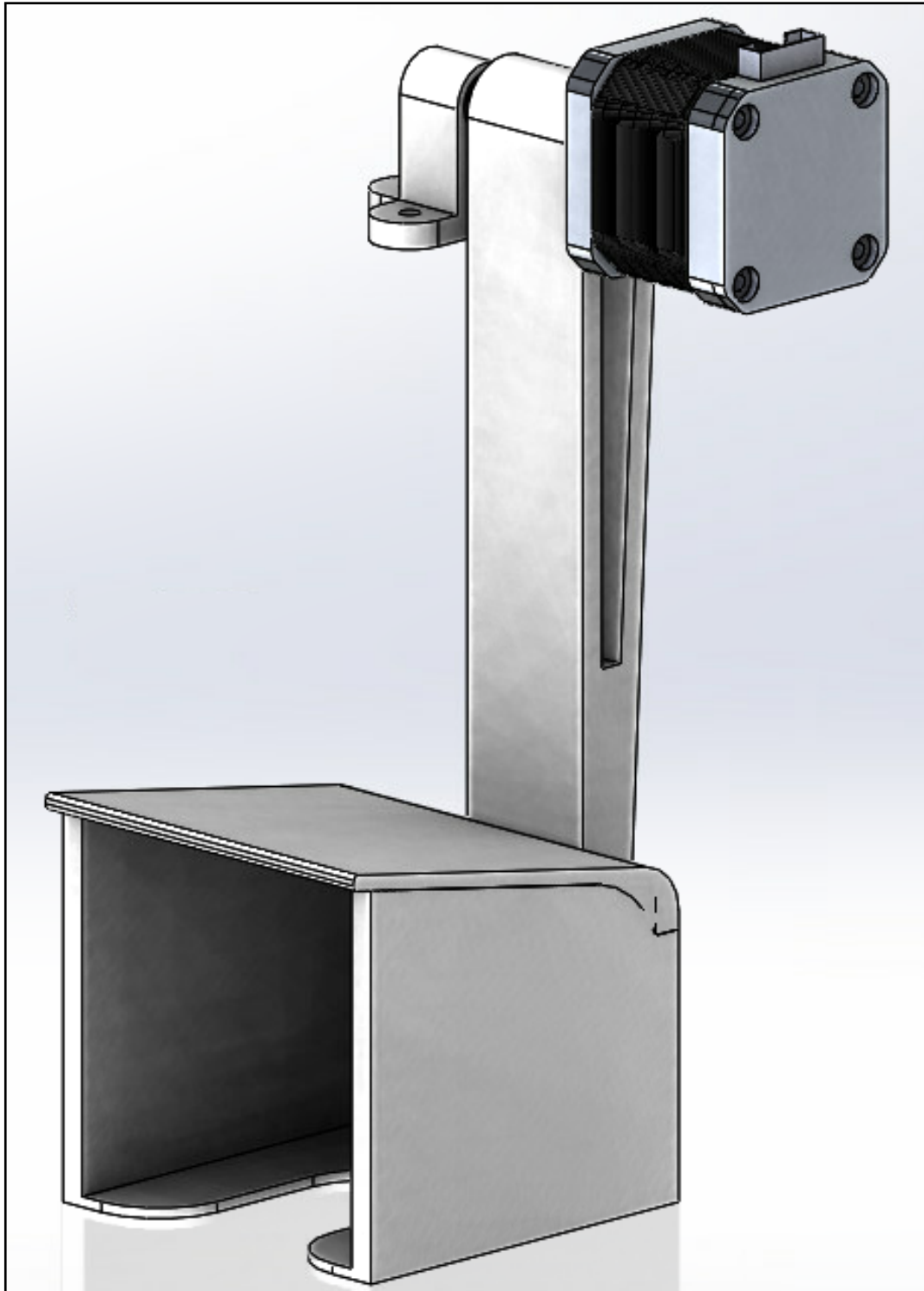


Figure 6.5: Collection arm assembly

6.0.3 Calculations

The goal of these calculations is to determine the torque of the arm at every point in its 180° rotation.

$$\tau = F \cdot d \quad (6.1)$$

To find the total torque required, the masses of each component and their distance from the motor shaft must be calculated. This was evaluated in Solidworks, and the data was transferred into a python script. The script uses the following equation to determine the torque (τ) at position θ .

$$\tau = \Sigma mgd \cdot \sin(\theta)$$

$$\tau = \sin(\theta) \cdot (m_{ball} \times d_{ball} \times g + m_{box} \times d_{box} \times g + m_{arm} \times d_{arm} \times g)$$

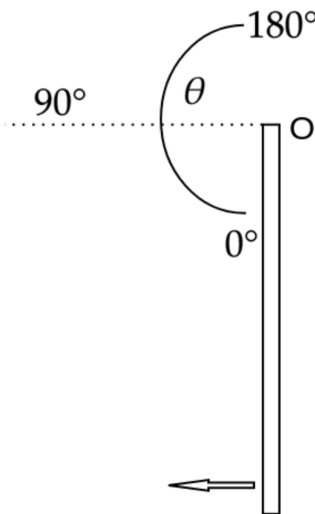


Figure 6.6: Angles from 0 to 180° in arm motion

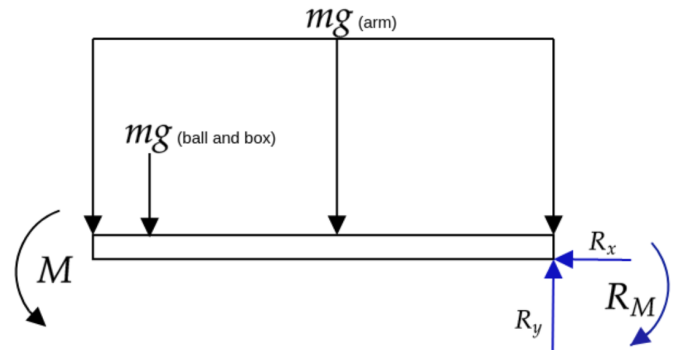


Figure 6.7: Free body diagram of the arm at 90°

The load of the arm can be simplified into a point load at the center of its mass. R_M is the torque τ supplied by the stepper motor.

Listing 1 Calculating the required torque for arm at all positions

```

1  import numpy as np
2  from matplotlib import pyplot as plt
3
4  ballM = 0.056 #kg
5  ballD = 15.748 #cm
6  boxM = 0.085 #kg
7  boxD = 15.748 #cm
8  armM = 0.048 #kg
9  armD = 9.0 #cm
10
11  g = 9.81
12
13  angles = np.linspace(0,180,360)
14  torque = np.sin(angles*np.pi/180) * (ballM*g*ballD + boxM*g*boxD + armM*g*armD)

```

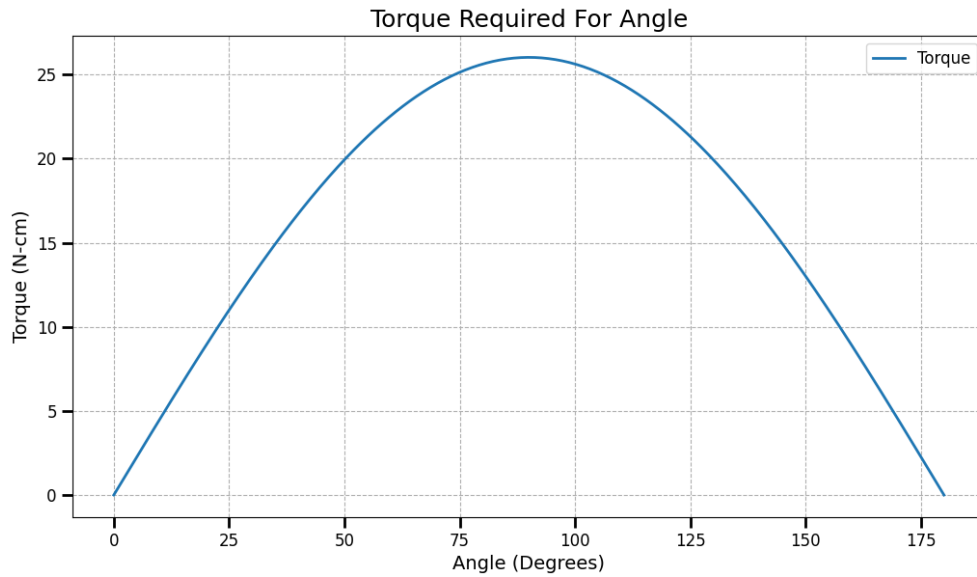


Figure 6.8: Torque required to lift arm

6.0.4 Verdict

The motor needs to be able to supply 26N-cm of torque to the arm to lift the ball. After some preliminary testing, the motor struggled to lift the arm past 60°. The motor used has a holding torque of 56N-cm, and as such the lack of success of the arm can be attributed to an incorrect electrical setup (suspected incorrect potentiometer position on the A4988 driver). Further testing will be carried out next week.

Without the redesign of the collection arm (which the group later agreed to use as a replacement), the robot would not have been allowed to participate in the Warman Demonstration due to exceeding the weight and size limit.

7 | Storage Design (Artefact 3)

7.0.1 Goal

The goal of this artefact is to design a system that can store and deposit the tennis and squash balls into the silos. It should fit onto the top of the scissor lift, and be as light as possible. Anything heavier than 300g is too heavy, otherwise the scissor lift will struggle.

7.0.2 Design

The following (7.1) is the CAD made in Solidworks for the deposit system.

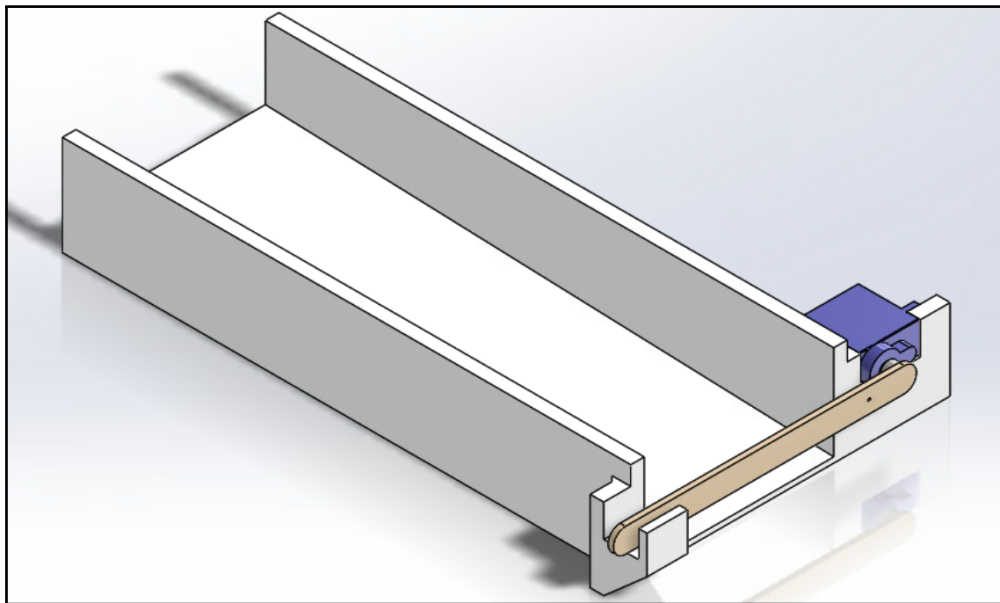


Figure 7.1: Deposit system design

7.0.3 Calculations

The gate needs a support on the opposite side to the servo, so that any force applied by the balls can have a reaction force supplied by the deposit body. If the support was not there, the balls would create a torque on the gate, would stress the joint between the gate and servo motor. The torque on the gate with no support is as follows.

$$F_y = m_{balls} \cdot g$$
$$F_y = 3 \times 0.056 \times 9.81$$

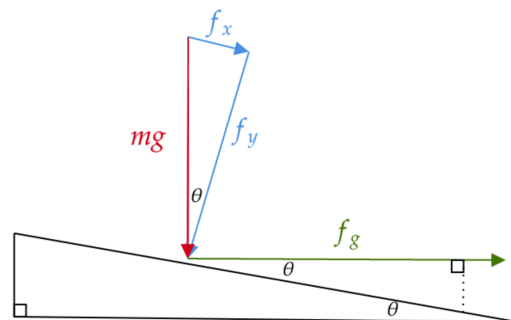


Figure 7.2: FBD of deposit system forces

$$F_y = 1.65\text{N}$$

Finding the force in the x-direction (direction of slope).

$$F_x = F_y \tan(\theta)$$

$$F_x = 1.65 \times \tan(10)$$

$$F_x = 0.29\text{N}$$

Finding the force on the gate (F_g)

$$F_g = F_x \cdot \cos(\theta)$$

$$F_g = 0.29 \cdot \cos(10)$$

$$F_g = 0.28\text{N}$$

Using the force on the gate (F_g) to find the torque on the gate. Force is assumed to be in the middle of the gate (0.035m).

$$\tau = F_g \cdot d$$

$$\tau = 0.28 \times 0.035$$

$$\tau = 0.0097\text{N-m}$$

The torque on the gate would be 0.97N-cm. However, this is only the torque when the balls are stationary. When a ball is dropped onto the deposit system, it will roll into the gate. The torque at impact will now be calculated for a single ball.

First, the impact velocity must be calculated. The force in the direction of the slope has already been calculated (F_x).

$$F_x = ma_x$$

$$a_x = \frac{F_x}{m}$$

$$a_x = \frac{0.29}{0.056}$$

$$a_x = 5.18\text{m/s}^2$$

$$v_x^2 = u^2 + 2a_x s$$

The initial velocity will be assumed to be 0m/s.

$$v_x = \sqrt{2a_x s}$$

$$v_x = \sqrt{2 \times 5.18 \times 0.21}$$

$$v_x = 1.47\text{m/s}$$

Finding the impact force (F_i).

$$F_i = \frac{mv}{t}$$

$$F_i = \frac{0.056 \times 1.47}{0.1}$$

$$F_i = 0.82\text{N}$$

The force at impact is 0.82N. Finding the torque at impact.

$$\tau_i = F_i \cdot d$$

$$\tau_i = 0.82 \times 0.035$$

$$\tau_i = 0.0288\text{N-m}$$

7.0.4 Verdict

The torque at impact is 2.88N-cm. This is enough for a justification to build a support for the gate onto the deposit mechanism. If no support was added, the gate would be more likely to come off the servo with repeated testing of the system.

8 | Mechatronics Development

8.1 Pixy Camera (Artifact 4)

8.1.1 Goal

The goal of this artefact is to explore the capabilities of the pixy camera as a way of navigation the track. By the end of this artefact it should be known the feasibility using the pixy camera as the primary source of navigation. Testing should include:

1. Object recognition of the silos, tennis balls and squash balls using colour recognition
2. Reliability of results
3. Create functions that make for an easy to understand void loop for other team members



Figure 8.1: Pixy Camera

8.1.2 Pseudo Code

Algorithm 1 Pixy Camera Pseudo Code

```
while Balls Picked Up < 3 do  
    Find Largest Ball  
    Rotate To Ball  
    Drive To Ball  
    Flash LED  
end while
```

8.1.3 Developed Code

The pseudo code was developed into C++ code for an Arduino Uno. The following code creates a function *acquireBlock()*, which returns the index of the largest detected object if it has lasted more than 30 frames. The additional criteria of the minimum amount of frames before the block is considered an object helps filter out any noise blocks. The function returns the index of a detected object (0 – n), and returns -1 if no object is found.

Listing 2 *acquireBlock()* function listing

```
1         int16_t acquireBlock() {  
2             pixy.changeProg("color_connected_components");  
3             if (pixy.ccc.numBlocks && pixy.ccc.blocks[0].m_age>30)  
4                 return pixy.ccc.blocks[0].m_index;  
5  
6             return -1;  
7         }
```

The `trackBlock()` function takes the index of the block locked onto in `acquireBlock()` and returns a pointer to it if it is found. Otherwise the function returns NULL.

Listing 3 `trackBlock()` function listing

```

1      Block *trackBlock(uint8_t index) {
2          uint8_t i;
3
4          for (i=0; i<pixy.ccc.numBlocks; i++) {
5              if (index==pixy.ccc.blocks[i].m_index)
6                  return &pixy.ccc.blocks[i];
7          }
8          return NULL;
9      }

```

The `rotateToObject()` function reads the x-coordinate in the centre of the object being tracked. using this value the objects x position relative to the robot can be determined. If $x < 150$, the robot rotates right until x value reads a value of 160, within a tolerance of ± 10 . Likewise, if the object reads $x > 170$, the robot will rotate left until within the 10 pixel tolerance of the centre of the field of view.

Listing 4 `rotateToObject()` function listing

```

1      void rotateToObject() {
2          if (block && drivingStart == false) {
3              objectXPosition = block->m_x; // fixed line
4              analogWrite(enA, rotateSpeed);
5              analogWrite(enB, rotateSpeed);
6              while (abs(objectXPosition - 160) > 10) {
7                  pixy.ccc.getBlocks();
8                  if (objectXPosition < 150) {
9                      driveRobot("right");
10                 } else if (objectXPosition > 170) {
11                     driveRobot("left");
12                 }
13                 block = trackBlock(block->m_index);
14                 objectXPosition = block->m_x;
15             }
16             drivingStart = true;
17         }
18     }

```

After the ball has rotated itself so that the object is in front of it, it drives forward with the `driveToObject` function. First the function reads the y-coordinate of the object. Then the robot either drives forward, or rotates to fix any x-alignment issues if they arise. The x-alignment correction is important for the chassis used because one of the axels was misaligned, meaning it can't drive in a straight line whilst power to each wheel is equal. Fortunately, the feedback system with the pixy camera allows for constant error correction.

The robot will drive forward until the object disappears from view due to the robot being close to the object. The pixy camera is tilted in such a fasion that when the object disappears from view, the object is in the right position to be picked up. A green led is then flashed twice to indicate the robot is in the picking up position and the integer `tennisBallsCollected` is incremented by 1.

Listing 5 *driveToObject()* function listing

```

1         void driveToObject() {
2             if (drivingStart == true) {
3                 int objectYPosition = block->m_y;
4                 driveRobot("forward");
5                 while (pixy.ccc.numBlocks) {
6                     if (objectXPosition < 150){
7                         analogWrite(enA, turnSpeed);
8                         analogWrite(enB, driveSpeed);
9                     } else if (objectXPosition > 170) {
10                        analogWrite(enA, driveSpeed);
11                        analogWrite(enB, turnSpeed);
12                    } else {
13                        analogWrite(enA, driveSpeed);
14                        analogWrite(enB, driveSpeed);
15                    }
16                    pixy.ccc.getBlocks();
17                    block = trackBlock(block->m_index);
18                    objectYPosition = block->m_y;
19                    objectXPosition = block->m_x;
20                }
21                driveRobot("stop");
22                tennisBallsCollected ++;
23                digitalWrite(greenled, high);
24                delay(100);
25                digitalWrite(greenled, low);
26                delay(100);
27                digitalWrite(greenled, high);
28                delay(100);
29                digitalWrite(greenled, low);
30                drivingStart = false;
31            }
32        }

```

The *loop()* function calls the previously defined functions, and loops through each object until **tennisBallsCollected** = 0.

Listing 6 *void loop()* function listing

```

1         void loop() {
2             // Find and collect tennis balls
3             while (tennisBallsCollected < 3) {
4                 pixy.ccc.getBlocks();
5                 findClosestObject();
6                 rotateToObject();
7                 driveToObject();
8                 findClosestObject();
9                 if(index == -1) {
10                    driveRobot("stop");
11                }
12            }
13        }

```

8.1.4 Testing And Verdict

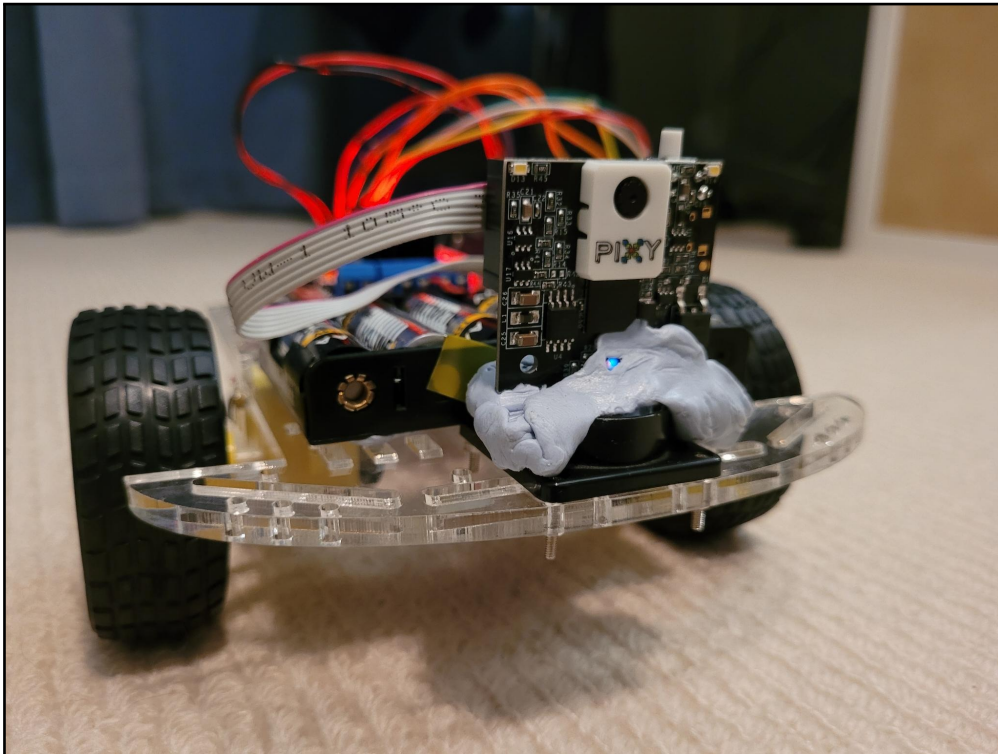


Figure 8.2: Build of the pixy robot

Multiple tests were carried out with the pixy camera. The first test involved detecting a bright red battery bank on a yellow/white background. The pixy camera was successfully able to track this object, and the robot reliably drove towards it from distances up to 5m away. The robot was also tested driving to a tennis ball, which was successful also. [Here is a linked YouTube video](#) of the robot following a tennis ball when rolled in front of it in a random direction.

The main issue arising from these tests were inconsistent lighting. Each day, the camera's parameters would have to be readjusted based on the conditions. Even lighting between facing the window in my room and facing away from the window required more adjusting. Since the lighting of the room on the test day can not be controlled by the group, it is likely a function that adjusts the camera's exposure based on the average intensity of the light in the direction it is facing. At this point, the complexity of the execution of this sensor is increasing, and with increased complexity comes a higher chance for something to go wrong.

The camera was also tested at university on the track. It was at this point that it was realised the camera cannot detect black or white objects, because neither of these colours have a colour hue, which is required by the camera to detect an object. The black squash ball could not even be detected on a light brown background. At this point, it was also apparent how much background noise would be present in the field of view of the camera. This could be somewhat mitigated by pointing the camera downwards slightly, so only the track was visible most of the time, but this comes at the cost of limited vision across the table, which is required for navigation.

The silos were also unable to be detected as they are white, which is a significant setback in terms of navigating the track. The only features of the track that could reliably be detected were the tennis balls.

8.2 Acknowledgements

Fellow Team Members:

- Jack Gruber
- Chris Finos
- Bashaar Yaacoub Agha
- Seongkyu Choi (Kevin)

Subject Coordinator:

- Anna Lidfors Lindqvist

Session Mentor:

- Rowan Smith

The lab staff

The University of Technology Sydney

A | Ideation of the Group