Design and Control of Manipulators

Introduction

The Design and Control of Manipulators is one section the roco224 coursework, alongside 'Programming Industrial Robots' It consists of a student-driven group design project, where the goal is to fulfil a chosen task related to the field of robotic manipulators. Our group is named 'Rossumovi Roboti' and consists of four members:

Jack GELL	Student Number: 10518789
Spencer PERDOMO-DAVIES	Student Number: 10535055
Faisal FAZAL-UR-REHMAN	Student Number: 10538828
Robert GRAINGER	Student Number: 10541139

There are 8 project challenges to choose from, though the group could design a new challenge if desired. After some discussion, our group has settled on challenge 4 – The Redundant Freedom Arm. However, due to our large group size we have also decided to incorporate elements of 2 – Pick and Place. This will allow us to allocate a suitable quantity of work amongst ourselves.

Description of hardware

Regarding hardware, we have used 8 RX-28 Dynamixels for the redundant part of the arm, two Mega 2560 R3 Controller boards and one 9-gram hobby motor for the end effector.

RX-28 Dynamixel

The RX-28 Dynamixel Robot Servo Actuator has the ability to track its speed, temperature, shaft position, voltage, and load. The control algorithm used to maintain its shaft position can also be adjusted individually for each servo, allowing you to control the speed and strength of the motor's response. All the sensor management and position control is handled by the servo's built-in microcontroller. This distributed approach leaves your main controller free to perform other functions. This is a servo specifically designed for applications in robotics. However, we will not be needing all the functions of this component for this specific task.

The specifications of this actuator are as follows...

As we will be using an operating voltage of 12v we have only included the relevant information on specification for this. However, this actuator can also perform at an operating voyage of 16V.

Operating Voltage 12V

Stall Torque* 28.3 kg·cm/ 393 oz·in (Note that that Stall torque and weight will be an important factor for our task)

No-load Speed 0.167 sec/60°

Weight 72g

Size 50.6 x 35.6 x 35.5 mm

Resolution 0.29°

Reduction Ratio 1/193

Operating Angle 300° or Continuous Turn

Max Current 1200mA

Standby Current 50 mA

Operating Temp -5°C ~ 85°C

Protocol RS485 Asynchronous Serial Module Limit 254 valid addresses

Com Speed 7343bps ~ 1Mbps

Position Feedback Yes

Temp Feedback Yes

Load Voltage Feedback Yes

Input Voltage Feedback Yes

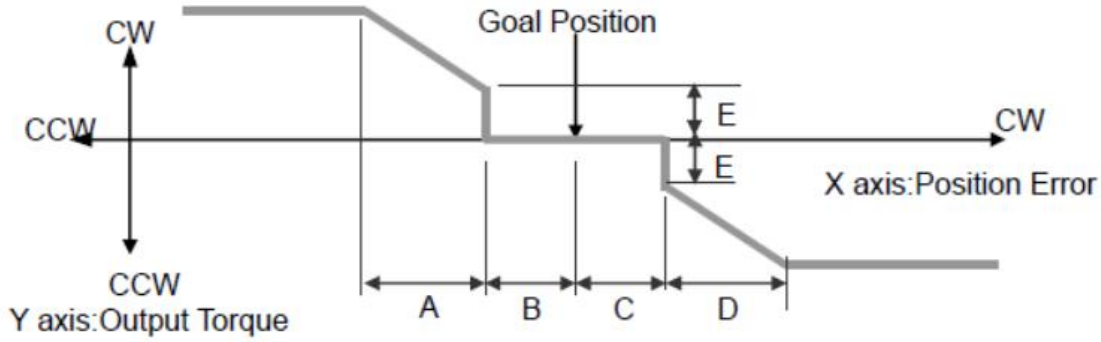
Compliance/PID Yes

Material Metal Gears & Engineering Plastic Body Motor Maxon RE-MAX

Compliance

Compliance is to set the control flexibility of the motor.

The following diagram shows the relationship between output torque and position of the motor.



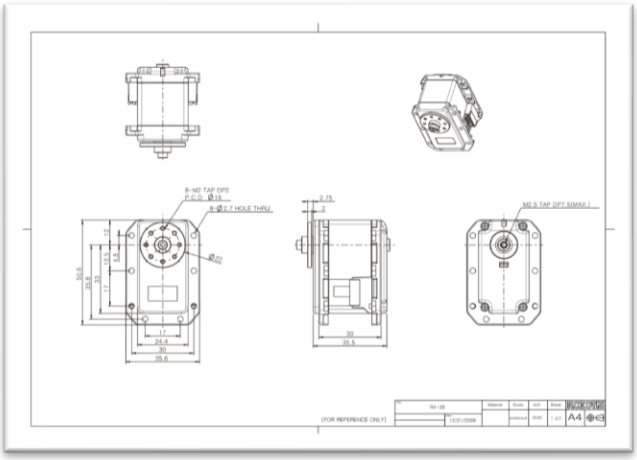
A : CCW Compliance Slope(Address0x1D)
B : CCW Compliance Margin(Address0x1B)
C : CW Compliance Margin(Address0x1A)
D : CW Compliance Slope (Address0x1C)
E : Punch(Address0x30,31)

Compliance Margin

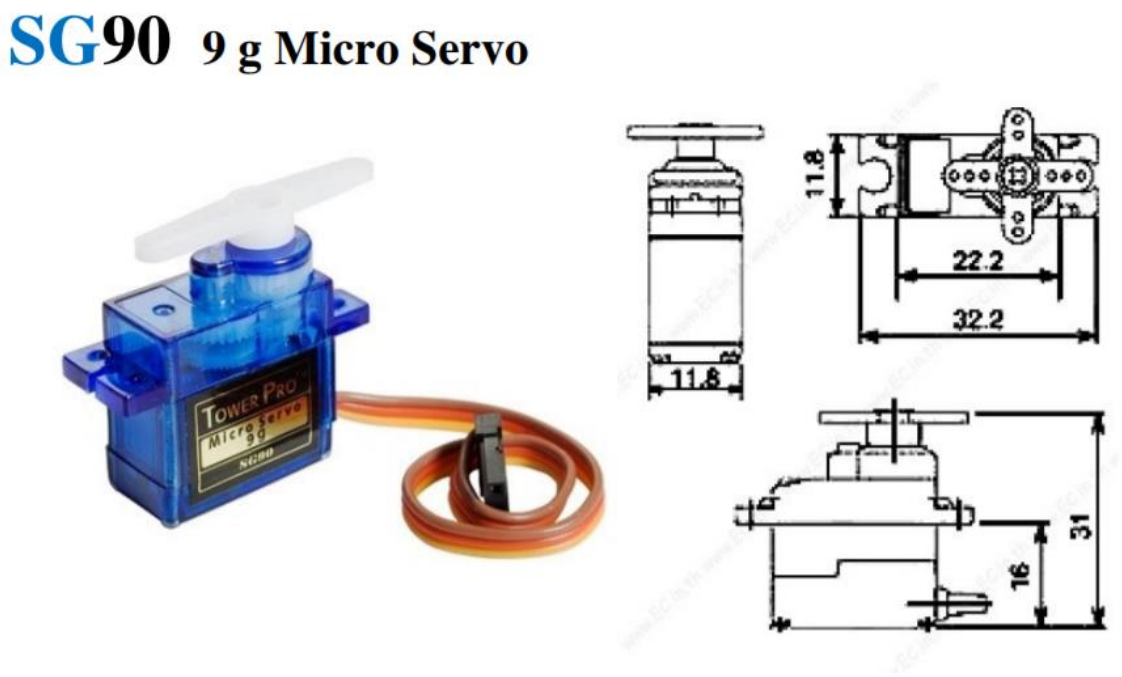
It exists in each direction of CW/CCW and means the error between goal position and present position.

The range of the value is 0~255, and the unit is the same as Goal Position.(Address 30,31)

The greater the value, the more difference occurs.



Tower Pro SG90 9g Micro Servos hobby motor



We are using this motor for the end effector as it is extremely light weight, robust and can be easily implemented into our design.

Specifications Modulation:

Analog Torque: 4.8V: 25.00 oz-in (1.80 kg-cm)

Speed: 4.8V: 0.12 sec/60° Weight: 0.32 oz (9.0 g)

Dimensions: Length: 0.91 in (23.0 mm)

Width: 0.48 in (12.2 mm)

Height: 1.14 in (29.0 mm)

Motor Type: 3-pole

Gear Type: Plastic

Rotation/Support: Bushing

Rotational Range: (add)

Pulse Cycle: (add) Pulse

Width: 500-2400 µs

Connector Type: JR

Elegoo MEGA 2560 R3 Controller Board ATmega2560 ATMEGA16U2 with USB Cable Blue Version Compatible with Arduino Mega Kit

Elegoo MEGA 2560 R3 Controller Board ATmega2560 ATMEGA16U2 with USB Cable Blue Version Compatible with Arduino Mega Kit. We will be using this primarily for the end effector control.

Item Weight 36.3 g

Product Dimensions 10.5 x 5.5 x 1 cm

Elegoo Mega 2560 board is 100% compatible with Arduino IDE and designed for Arduino Mega Kit Project Elegoo Mega 2560 R3 board use Atmega2560-16au and Atmega16u2 as the controller chip which is the same with the official Arduino Mega 2560 board Improved and expert version: 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes. Elegoo Mega 2560 uses stronger RESET circuit as to offer better function experience This 2560 board contains everything needed to support the microcontroller and has an easy set up process.

All of which will be linked together with 3d printed brackets made of a Polylactide polymer more details of this can be seen in the design process.

Description of task

Our manipulator will consist of two segments.

Section A – the end effector, a claw to grip and position objects.

Section B – the redundant arm, a series of RX-28 servos to showcase redundancy and position the end effector.

We therefore have two main features to showcase – the redundancy and the ability to pick and object up. The idea is to have the arm manoeuvre into an enclosed area or around an obstacle. The claw will pick up an object, and then section B of the arm will move whilst keeping the end effector in place, demonstrating redundancy

An overview of your approach to the problem

The first thing we did was discuss various ideas such as:

- Where to place the base of the arm
- Consideration of weight and servo torque
- Number of servos that need to be implemented (A redundant arm needs a minimum of 7 servos).
- Implementation of forward and inverse kinematics.
- How we can demonstrate that our final product has achieved - the task we have set and the time scale we had for what could be achieved realistically.

Once we had all agreed on the requirements, we made a Gantt chart with the delegated tasks. Then created a GitHub repository for us all to collaborate on. Then we began on the design process.

Link to our GitHub repository is below...

https://github.com/slperdomo-davies/ROCO_224

Design Process

Our design process began with sketching and discussing ideas for ‘section B’ of the arm. We decided early on to have the servos attached together with short brackets rather than limb-like links. Our reasoning was to reduce weight, length and overall complexity. With a shorter arm weight and length, the servos will experience less force and can operate more efficiently.

Once we had a general idea of section B, we split the group in two – a pair working on section A and another to continue section B.

For section A, the end effector, we investigated different claw designs, sketching these ideas on paper. Once we had a design we felt was feasible, we created parts in fusion 360 and made an assembly of these parts to model the claw as one unit. The end effector was to include a 9g servo and the claw would be 3D printed as separate parts then assembled.

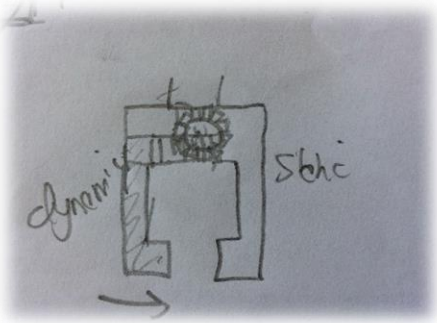
Since the arm has 8 servos, the idea behind the design was to keep it as light as possible. We decided to directly attach each servo on to the previous one without any links between them except for one link that goes after the shoulder joint. After we had a rough idea of our design, the first thing we did was to make sure that the RX-28 servos (especially the servo that will need maximum amount of torque) will be able to deliver the required torque for the design, for this we derived the moment formulas for our design.

What design iterations have you gone through?

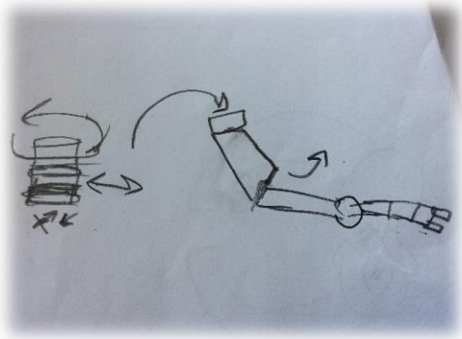
Part A - End Effector Design

End Effector Design Iterations (very rough ideas):

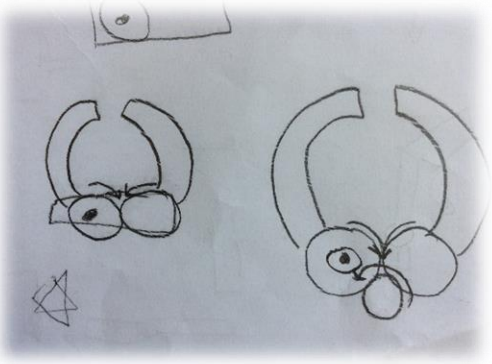
Sliding gripper model –



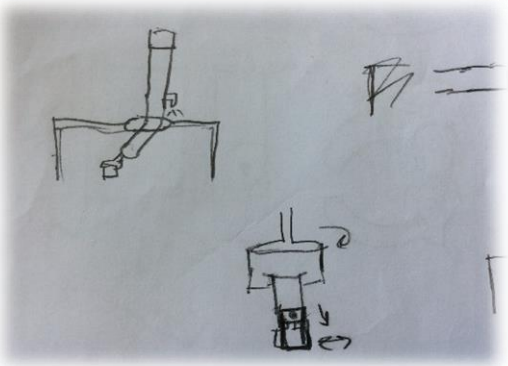
Early Arm design -



Spur gear design - The right-hand drawing is the basic principle we decided to take forward to the real design. A 9g servo will rotate one cog and thus operate one side of the claw. Another cog has interlocking teeth with the first and so will also spin, rotating the second claw part.



Top left - an idea on how to test redundancy by having the arm investigate through a hole. Right - An idea for a way to have 3 degrees of freedom in one segment



One idea was to use the end effector to scan a area from left to right, then move down one space and keep on repeating to build an image of the area below, made of different colour characters. Each colour pixel would represent the distance of the surface detected. Essentially this would capture an image and send it to a window in the terminal. By utilising a serial input you could let the end-effector know which pixel to move towards. This did manage to successfully work but it took a lengthy time to capture a image. It was also bottle necking processing speed and we decided it was far too taxing to keep it included within the code.

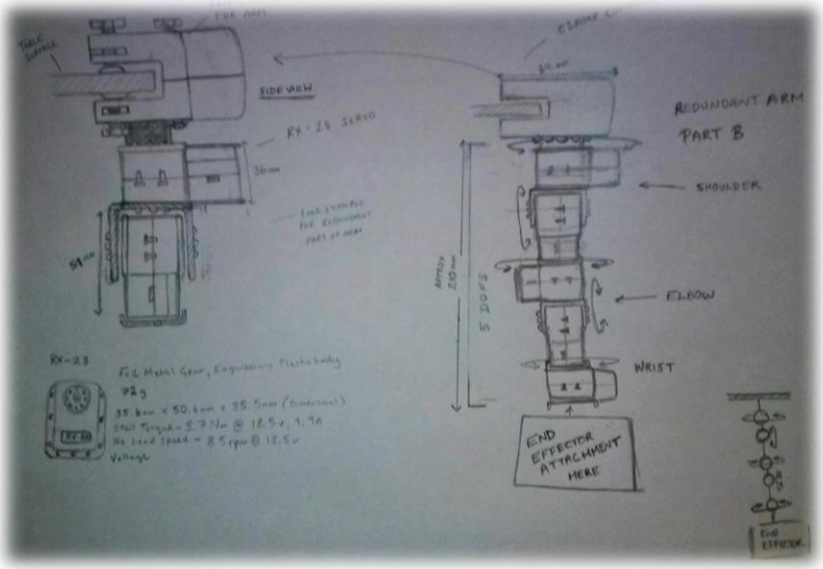
Part B - Redundant arm Design

We first made a few thumb nail sketches to get a feel for the design that we wanted, as seen below...

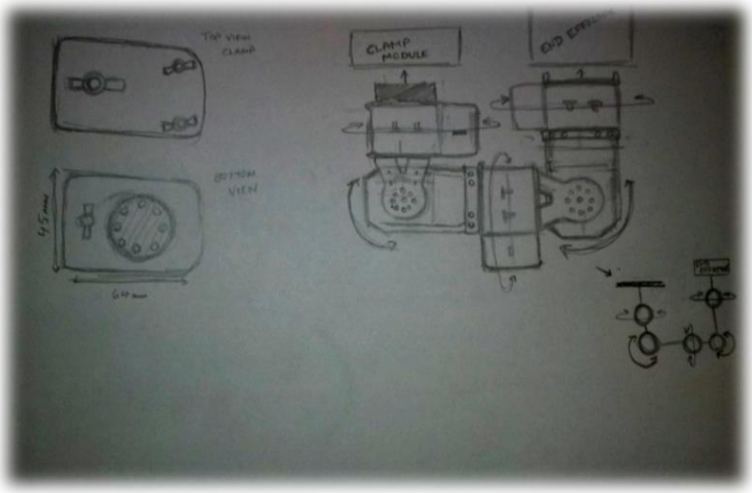
As we were essentially linking the servos together our part of the design consisted of brackets that would be light in weight and robust but could still carry the load of the

8 servos and the end effector. We also needed the brackets to be compatible with M2 screws to match the servos we are using and make it easier to attach them.

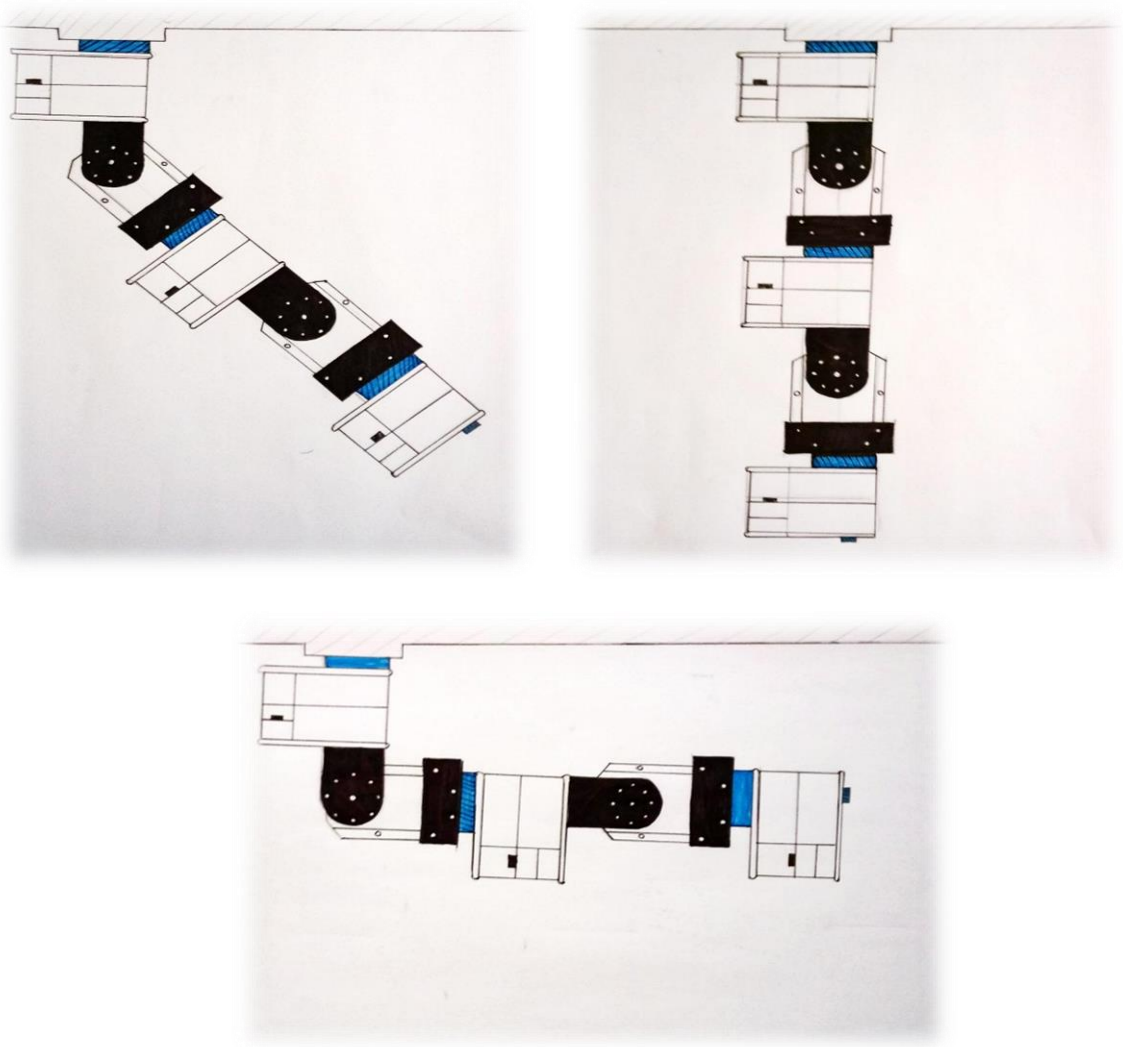
Another factor we needed to consider was the servo torque. This is the main reason that we had decided to configure the arm to hang vertically down. This would take a lot of stress from off the servos when performing the intended task and stop it from stalling at higher angles.



Thumbnail sketch 1



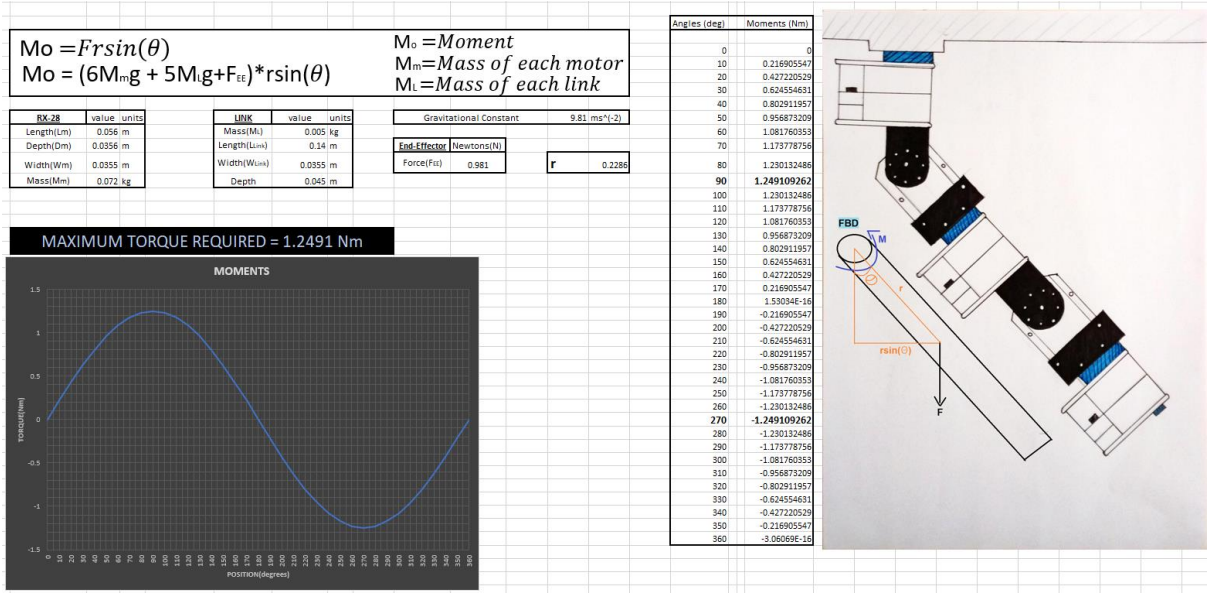
Thumbnail sketch 2



Final Sketches

When we were reasonably happy with the initial designs we drew another set of sketches in various positions and implemented some structural analysis. Then we finally started the design process in Fusion 360. the reason we are using Fusion 360 is purely because it is a very powerful CAD tool that makes it easy to realise our ideas in a short time period.

Final calculations for moments: *(click to open picture)*



Structural Analysis

After discussing which materials we should use for 3d printing we thought our best option was to use PLA as stated earlier. The reasons for this is that we have used PLA for similar projects previously and have found it to be very reliable and works very well with the Flash Forge Finder that we are using to print out the components.

Properties of the Polylactide (PLA)

The PLA is a Polyester Bio Plastic which is environmentally friendly, derived from renewable resources, non-aromatic, cost efficient and insoluble in water. It has the highest It is non-toxic to the extent that it is being used now in the medical industry to help reconstruct bone fractures and is used for other similar medical applications.

Apart from the above factors, it has a higher 3d print speed and is more aesthetically pleasing than ABS. Although there was a possibility of using the more durable ABS which does have a higher ductile strength. PLA is useful in a broad range of printing applications, has the advantage of being both odourless and low-warp, and does not require a heated bed which is important as the 3d printer that we are using does not have one, but we have 24-hour access to it. Which is important given our time frame.

Material properties

Density- 1.210-1.430 g.cm^3

Melting point- 150 to 160 degrees

Elastic (Youngs, Tensile)

Modulus- 3.5 GPa

Flexural Modulus- 4.0 GPa

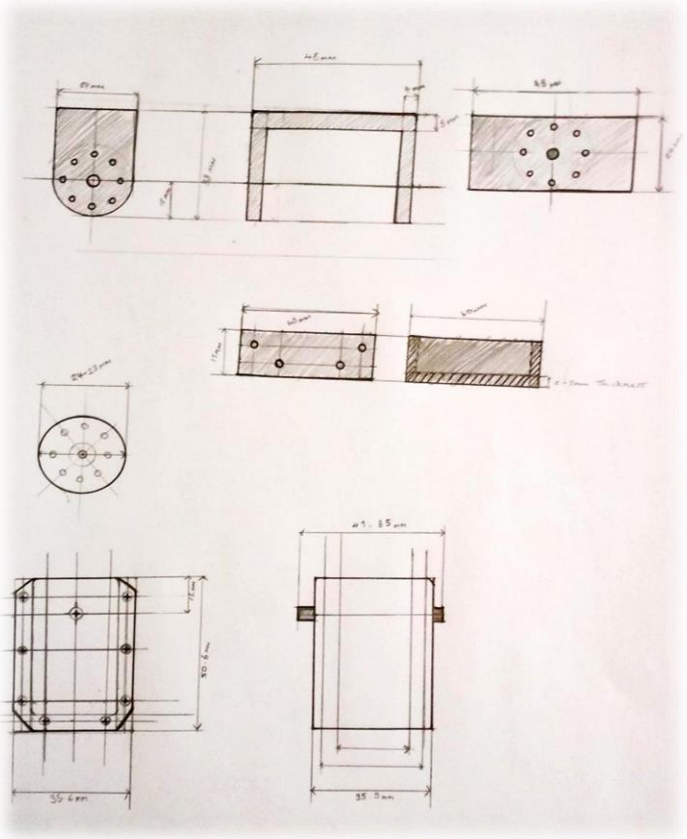
Shear Strength- 2.4 GPa

Tensile Strength- 50 GPa

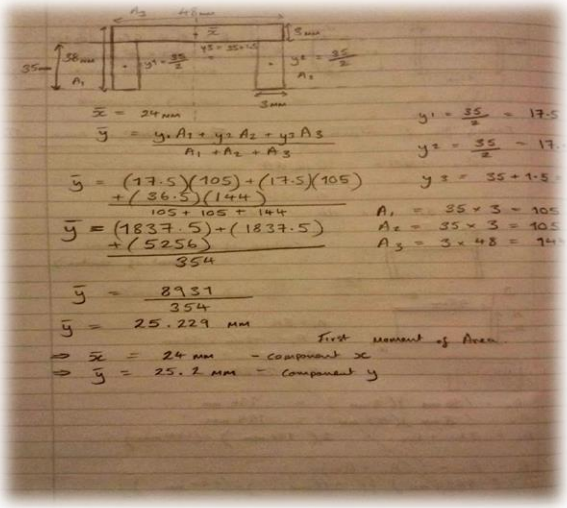
The following rough calculations are for the basic shapes of the overall geometry of the brackets we need to design, as these designs may change. We just need to have a general idea and consider these limitations of the mechanical properties such as bending stress etc.

Libre writer file for torque equations

https://github.com/Faisal-f-rehman/ROCO_224/blob/master/ManipulatorDesignProject/maths/JackGell_torque%20equation_%2090degres.docx



Moment of area



$\bar{x} = 24\ mm$

$\bar{y} = \frac{y_1 A_1 + y_2 A_2 + y_3 A_3}{A_1 + A_2 + A_3}$

$\bar{y} = \frac{(17.5)(105) + (17.5)(105) + (36.5)(144)}{105 + 105 + 144}$

$\bar{y} = \frac{(1837.5) + (1837.5) + (5256)}{354}$

$\bar{y} = \frac{8931}{354}$

$\bar{y} = 25.229\ mm$

First Moment of Area

$\Rightarrow \bar{x} = 24\ mm$ - Component x

$\Rightarrow \bar{y} = 25.2\ mm$ - Component y

$y_1 = \frac{35}{2} = 17.5$

$y_2 = \frac{35}{2} = 17.5$

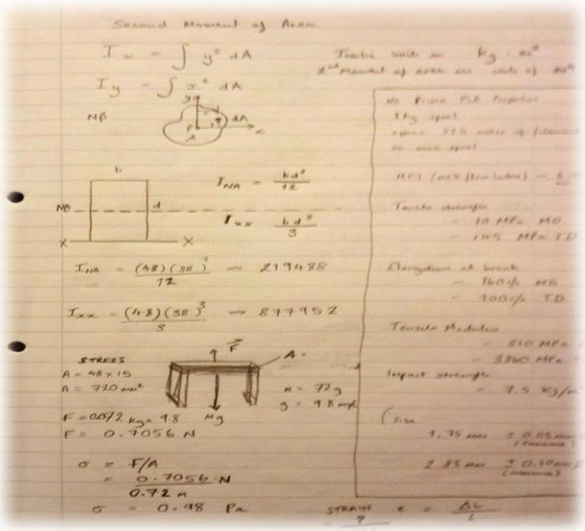
$y_3 = 35 + 1.5$

$A_1 = 35 \times 3 = 105$

$A_2 = 35 \times 3 = 105$

$A_3 = 3 \times 48 = 144$

Second moment of Area



Second Moment of Area

$I_x = \int y^2\, dA$

$I_y = \int x^2\, dA$

NA

$I_{NA} = \frac{bd^3}{12}$

$I_{xx} = \frac{bd^3}{3}$

$I_{NA} = \frac{(48)(38)^3}{12} = 219488$

$I_{xx} = \frac{(48)(38)^3}{3} = 877952$

STRESS

$A = 48 \times 15$

$A = 720\ mm^2$

$F = 0.072\ kg \times 98\ m/s$

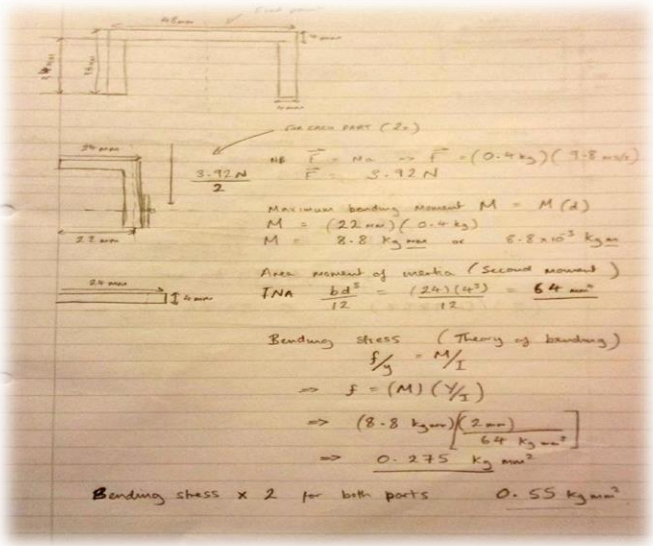
$F = 0.7056\ N$

$\sigma = F/A$

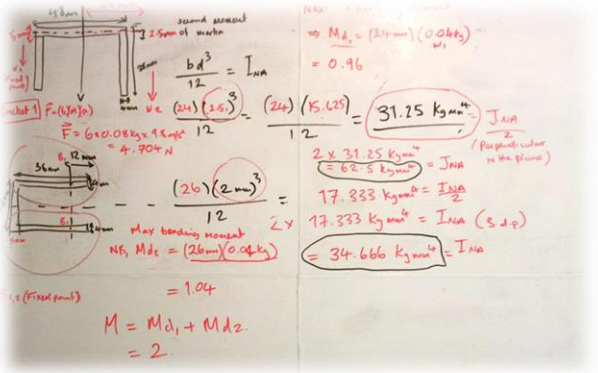
$= \frac{0.7056\ N}{0.72\ m}$

$\sigma = 0.98\ Pa$

Rough Calculations for Bending Stress



And maximum bending moment



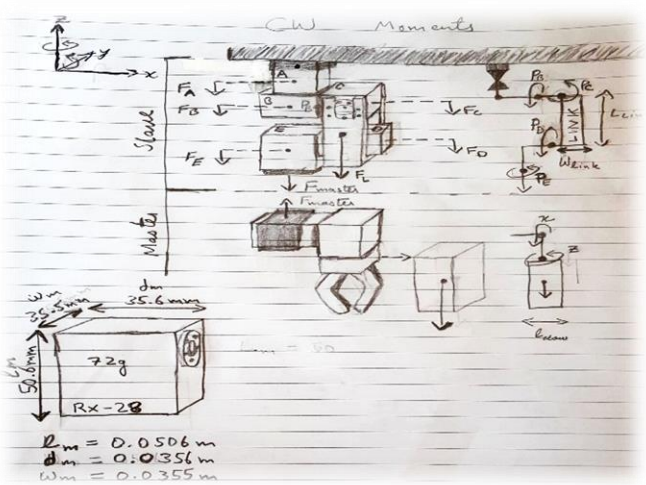
We found from structural analysis, that for the brackets to be robust, light weight and not too big of a structure, we needed to thicken the shell of each bracket.

As the designs were based on the brackets that come with the Dynamixels, we found that the dimensions of the originally designed brackets from Trossen Robotics, were too weak for the structural integrity of the PLA material we were using. The obvious reason for this is that the Trossen Robotics brackets were made from steel which has a high tensile strength. This made them very versatile regarding carrying loads. However, we still had to consider stall torque of the servo, especially at the base of our arm.

We discovered from the analysis that to increase the integrity of our PLA 3d printed parts we simply needed to increase the thickness of the corners by 2-3 mm. Making them robust enough to take the load at the base and the shoulder of the rest of the arm, which weighs approximately 430g (including all brackets) exerting a vertical force of 3.97 Nm, before they fractured, warped or broke.

We also designed the brackets to encase the servos and have them fit as tight as we possibly could as you will see later in our CAD designs.

Rough workings for moments:



Moments of Slave (Redundant)

About motor B

$$\sum P_B = 0$$
$$= -F_C\left(\frac{W_m}{2}\right) - F_L\left(\frac{W_{link}}{2}\right) - F_D\left(\frac{d_m}{2}\right) + F_E\left(\frac{W_m}{2}\right) + F_{master}\left(\frac{W_m}{2}\right) \quad \text{..... (1)}$$

About motor C

$$\sum P_C = 0$$
$$= +F_L(0) + F_D(0) + F_E\left(\frac{W_m}{2} + \frac{W_m}{2}\right) + F_{master}\left(\frac{W_m}{2} + \frac{W_m}{2}\right)$$
$$= F_E(W_m) + F_{master}(W_m)$$

About Motor D

$$\sum P_D = 0$$
$$= F_E(W_m) + F_{master}(W_m)$$

About Motor E

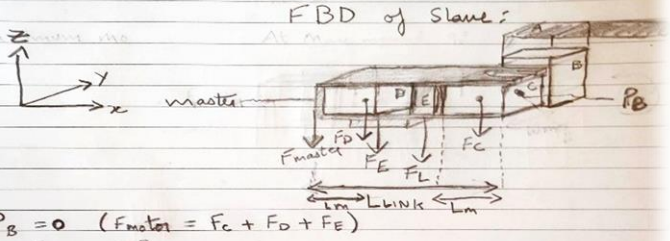
$$\sum P_E = 0$$
$$= F_{master}(0) = 0$$

From (1): since $F_C = F_D = F_E = F_{servo}$

$$\sum P_B = -F_{servo}\left(\frac{W_m}{2}\right) - F_L\left(\frac{W_{link}}{2}\right) - F_{servo}\left(\frac{d_m}{2}\right) + F_{servo}\left(\frac{W_m}{2}\right) + F_{master}\left(\frac{W_m}{2}\right)$$
$$= -F_{servo}\left(\frac{W_m}{2} + \frac{d_m}{2} - \frac{W_m}{2}\right) - F_L\left(\frac{W_{link}}{2}\right) + F_{master}\left(\frac{W_m}{2}\right)$$
$$= -F_{servo}\left(\frac{d_m}{2}\right) - F_L\left(\frac{W_{link}}{2}\right) + F_{master}\left(\frac{W_m}{2}\right)$$

Moment about P_B, maximum at 90°

FBD of Slave:



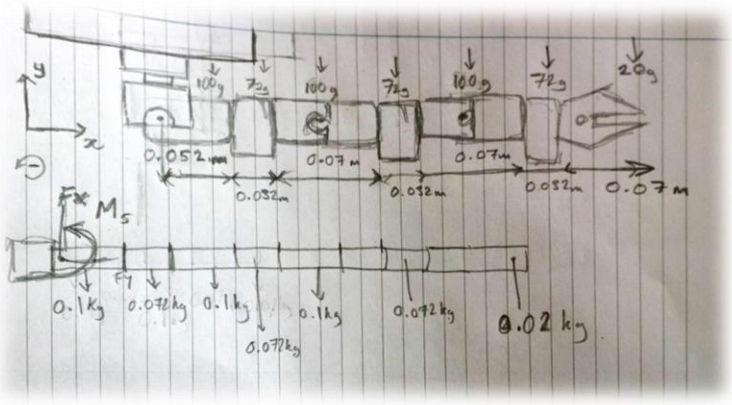
$$\sum P_B = 0 \quad (F_{master} = F_C + F_D + F_E)$$
$$= F_C(0) + F_L\left(\frac{L_{link}}{2}\right) + F_D\left(L_{LINK} - L_m\right) + F_E\left(L_{LINK} - L_m\right) + F_{master}\left(L_{LINK}\right)$$

Substitute F_{motor}:

$$= F_L\left(\frac{L_{LINK}}{2}\right) + F_{motor}\left(L_{LINK} - L_m\right) + F_{motor}\left(L_{LINK} - L_m\right) + F_{master}\left(L_{LINK}\right)$$

$$\sum P_B = F_L\left(L_{LINK}\right) + 2F_{motor}\left(L_{LINK} - L_m\right) + F_{master}\left(L_{LIN}\right)$$

Calculations for Torque around the shoulder joint



$$\uparrow \quad \Sigma F_y = 0$$

$$F_y - 0.1\,kg(9.81) - 0.072(9.81) - 0.1\,kg(9.81) - 0.072(9.81) - 0.1\,kg(9.81) - 0.072(9.81) - 0.02(9.81) = 0$$

$$\Rightarrow \quad F_y = (0.981) + (0.71) + (0.981) + (0.71) + (0.981) + (0.71) + 0.1962$$

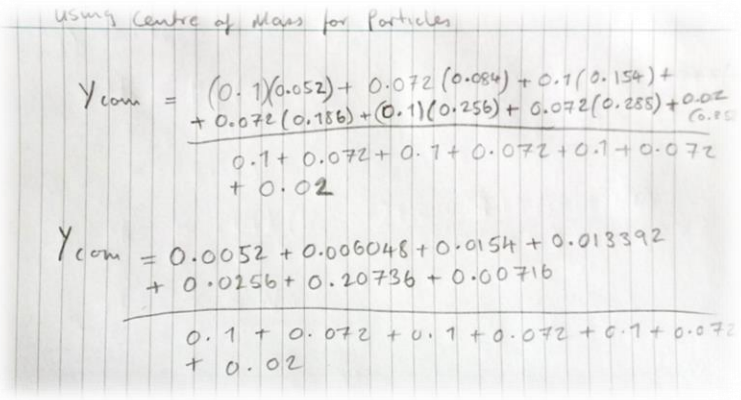
$$F_y = 5.2692\ N$$

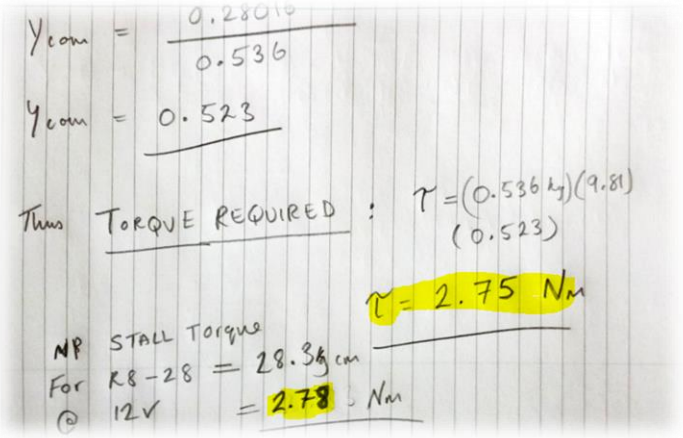
$$+\circlearrowleft \quad \Sigma M_A = 0$$

$$M_s - F_y(0.052) - (0.71)(0.032) - (0.981)(0.07) - (0.71)(0.032) - (0.981)(0.07) - (0.71)(0.032) - (0.1962)(0.07) = 0$$

$$M_s = 5.2692(0.052) + (0.023) + (0.06867) + (0.023) + (0.06867) + (0.023) + 0.022772$$

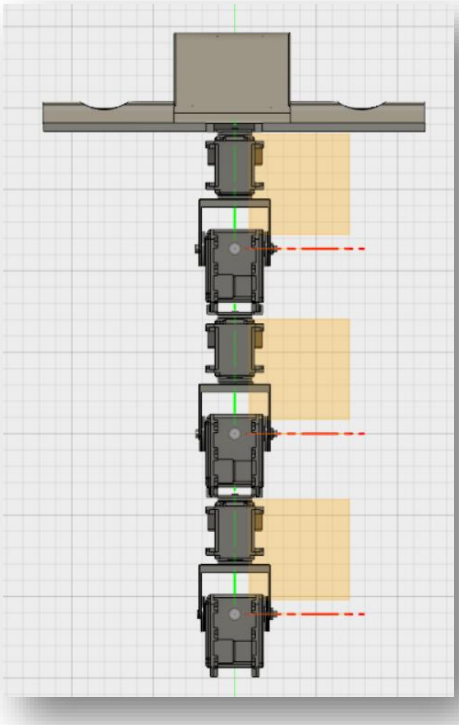
$$M_s = 0.5\ Nm$$





Nb – Torque required is approximately the same as the as the maximum Torque at operational voltage 12v for the RX-28. If any more weight is added this could be problematic. We added another servo to the shoulder joint to pre-empt tis problem.

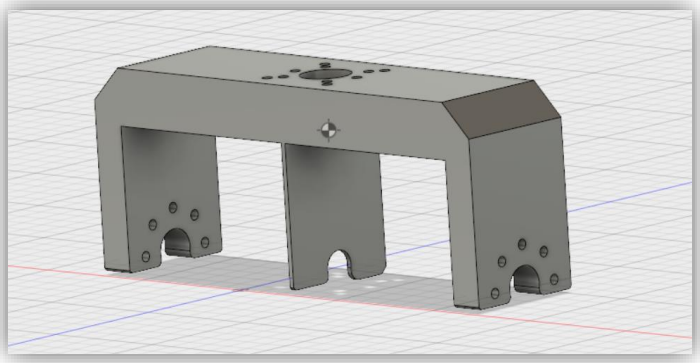
CAD DESIGNS



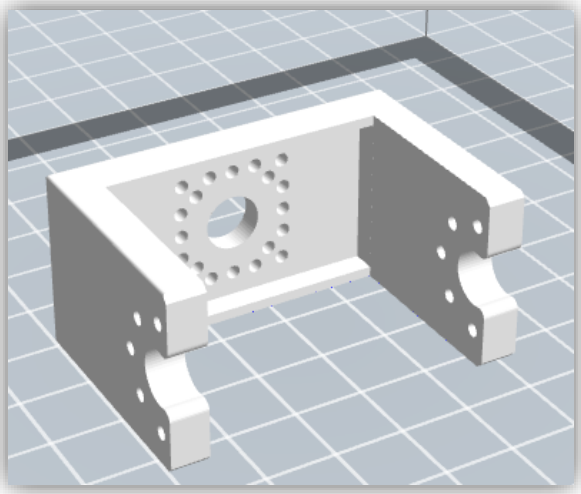
Fusion 360 Design v_ 1

Final CAD Designs

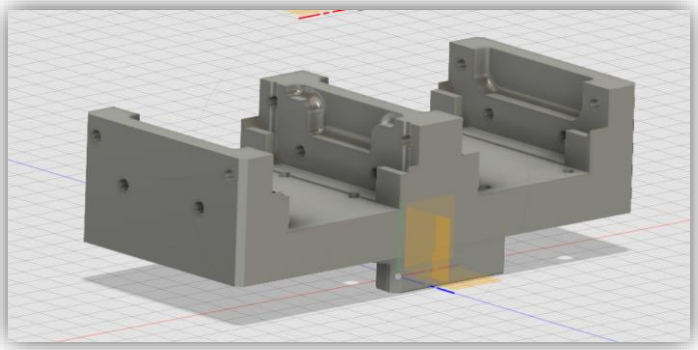
Top shoulder Bracket



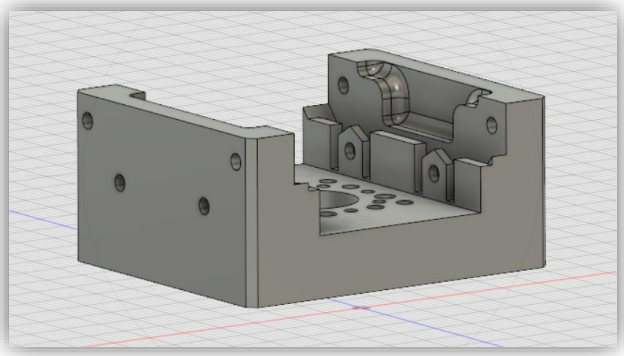
Top Bracket



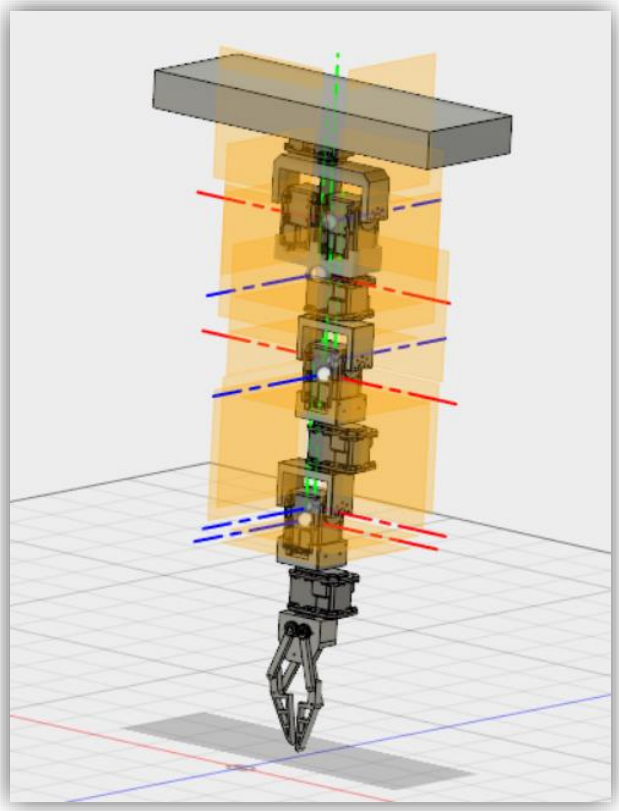
Bottom Shoulder Bracket



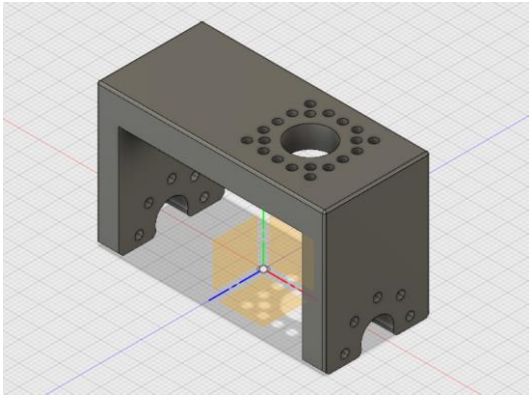
Bottom Universal bracket



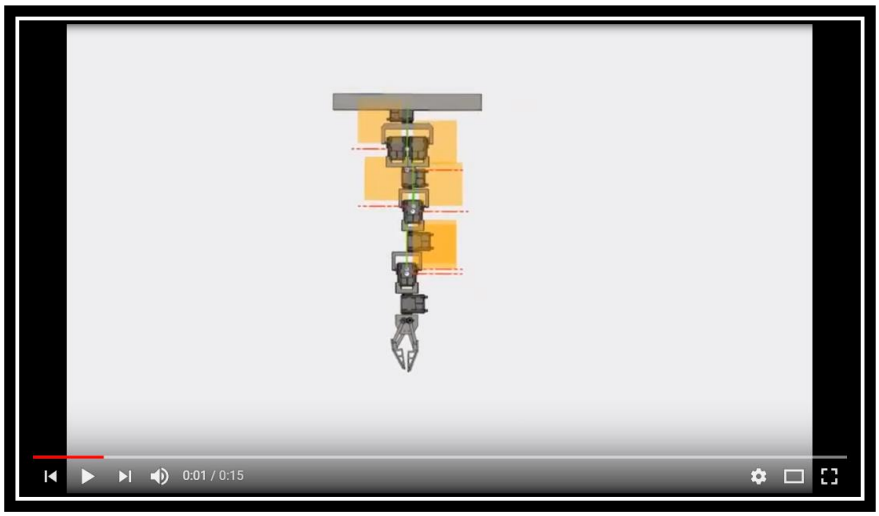
Full Assembly including end effector



Below is the redesigned bracket ready for offset of axis.



Click on picture below for 'Exploded View Video'



Who worked on what and when?

Initially Robert and Jack were to design and implement the Arduino code, Elegoo controller and sensors of Part A (End effector) whereas Faisal and Spencer were to design and 3d print the necessary components, implement the forward /inverse kinematics using ROS and MATLAB for part B the (Arm redundancy). However, due to time constraints there was inevitable cross over on the tasks and everyone worked on what was priority to achieve the task at hand.

What problems did you encounter, and how did you solve them?

Our original design for the redundant arm had 7 Dynamixel servos in line. We realised an issue may arise with the second servo from the base, which would have to lift the weight of all the subsequent servos and the end effector. To account for this, we decided to attach an additional servo in parallel to help spread the load of the rest of the arm. The first servo is not an issue because it is rotational as will not have to act against the weight of the others. To further combat weight concerns, we streamlined he design to include very few printed parts.

The end effector claw is operated by a 9-gram servo. This must be connected to the Arduino at the base of the arm with wire. The issue - these wires need to be slack enough to allow the arm joints to bend, but we also wanted to avoid loose wires that could become tangled when the arm moves. Our solution to this was to join attach these wires closely to the servos, but at joints the wire would loop slightly to account for the joint angle changing.

Another problem was the actual physical configuration of the joints, as they were all aligned on the Z axis which caused singularities when implementing the Inverse kinematics when simulating the arm. The arm movement did work with Forward kinematics, but there was no other way other than redesigning and printing some of the brackets to offset the joints to correct this problem. This will be demonstrated later when we discuss testing.

Implementation

Please click on the links below for details on implementation by ROS and Matlab, Implementation by Arduino and Kinematics

Implementation (ROS & MATLAB)

Implementation (Arduino)

End-effector with Arduino

Kinematics

Part A - End Effector

The end-effector for the redundant arm is a flat - edged gripper which is to be operated by a servo. The user will be able to control this servo from the serial terminal (created using program PuTTY, interfaced with the Arduino) and information will be fed back on the angle and the state of the gripper.

Printing

We are using a Flash Forge Finder for all printed parts. The benefits of PLA have been listed earlier in the report.

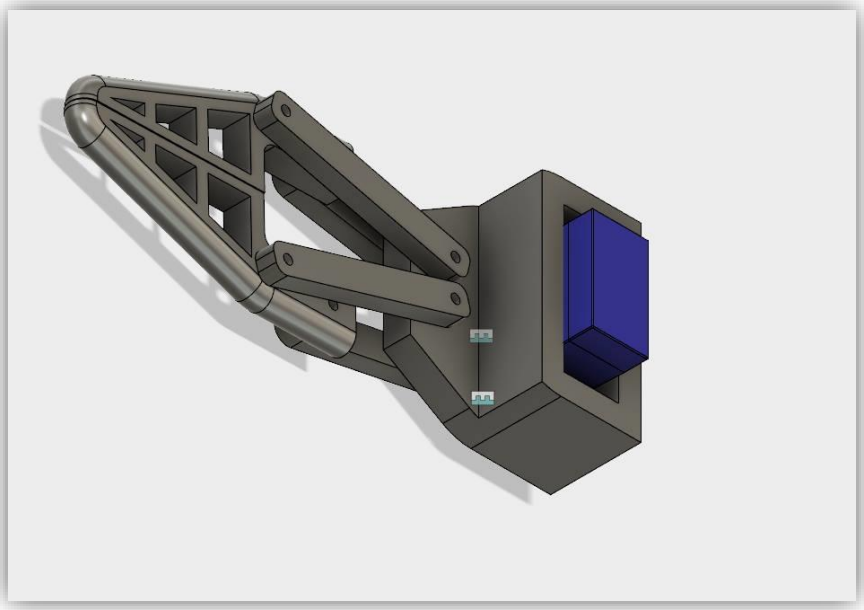
Control

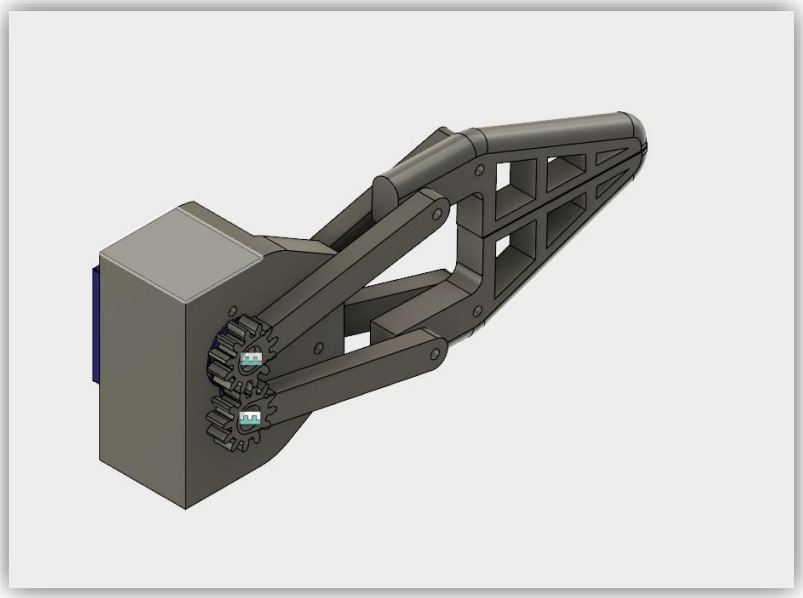
As mentioned above the end-effector will be controlled with the serial input to putty. We will be displaying a user interface using an Arduino mega which will take data from the keyboard to control the end-effector.

This is the End-effector portion of the terminal is shown below and is kept towards the bottom left of the terminal. The object distance relates to the ultra-sonic sensor. The gripper can be opened and closed by using the A and D keys. There are Q and E keys to allow the gripper to rotate using a second 9g servo, however we removed this servo from our design.

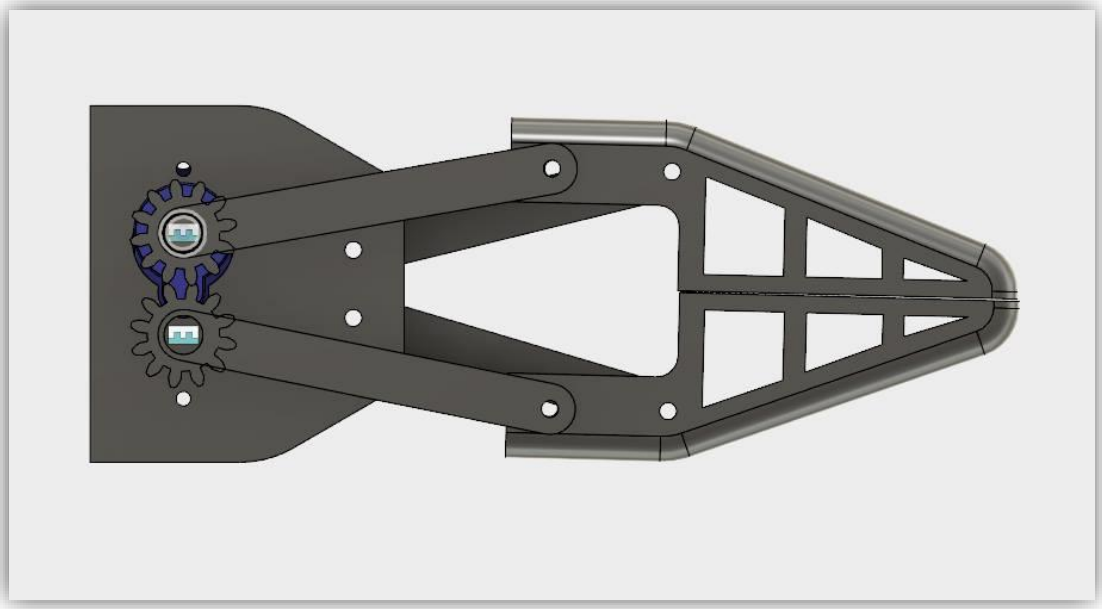
```
##################################################################################
#                                End Effector                                    #
##################################################################################
#        Rotation @ 000  degrees       ##       object distance 0  cm           #
##################################################################################
#   Rotate claw     ####  User control ## H open/close   ####  user control    #
##################################################################################
#    Clockwise      ==>>        Q       ##    Open        ==>>        A          #
#Counter clockwise  ==>>        E       ##    Close       ==>>        D          #
##################################################################################
```

Part of the gripper is hollowed out to save on weight. The grippers have a long flat edge to allow the two to close and not miss the object between.



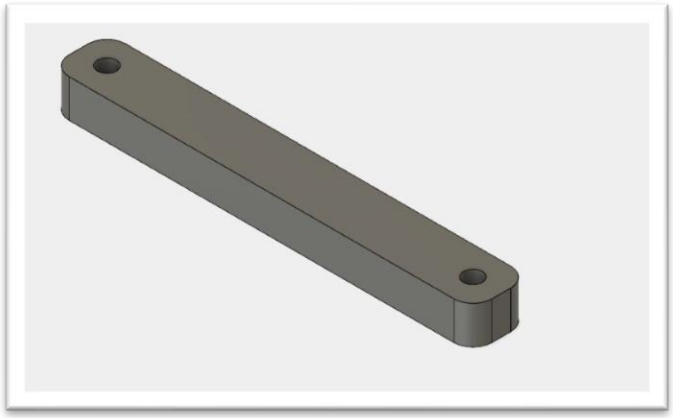


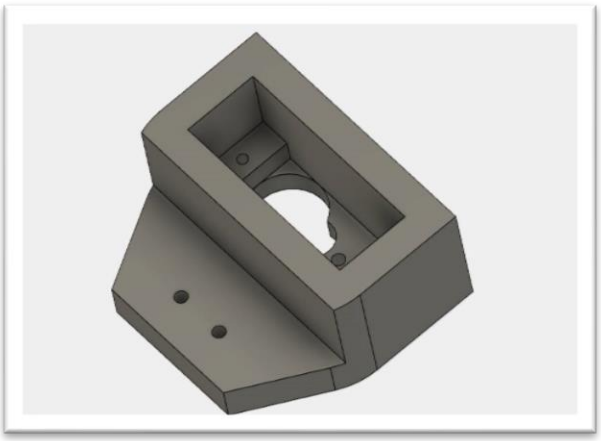
As seen here one servo operate both of the grippers by using cogs. This ensures that both the gripper segments are in mirror image with each other and saves weight of not have an extra servo, and the weight required from PLA to house another servo.

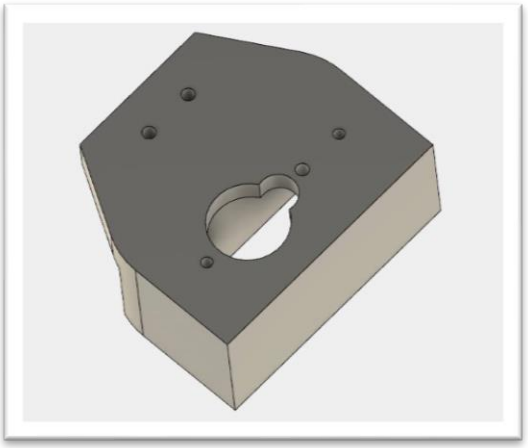


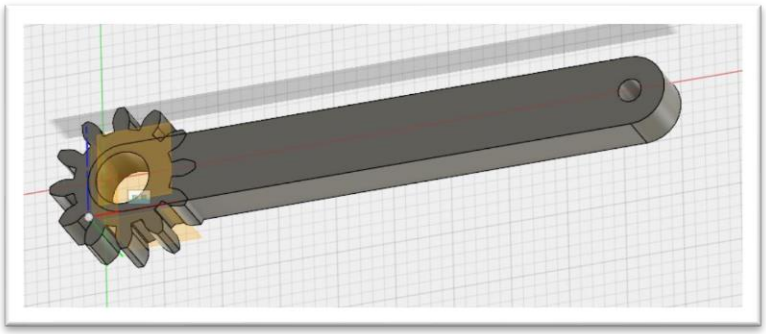
By looking closely there are holes so that the servo can be screwed into place.

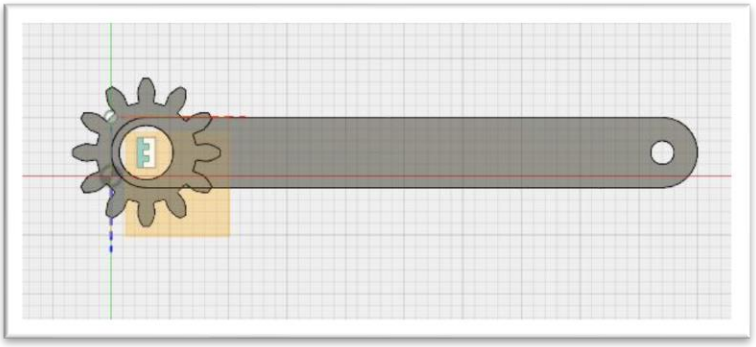
Individual components



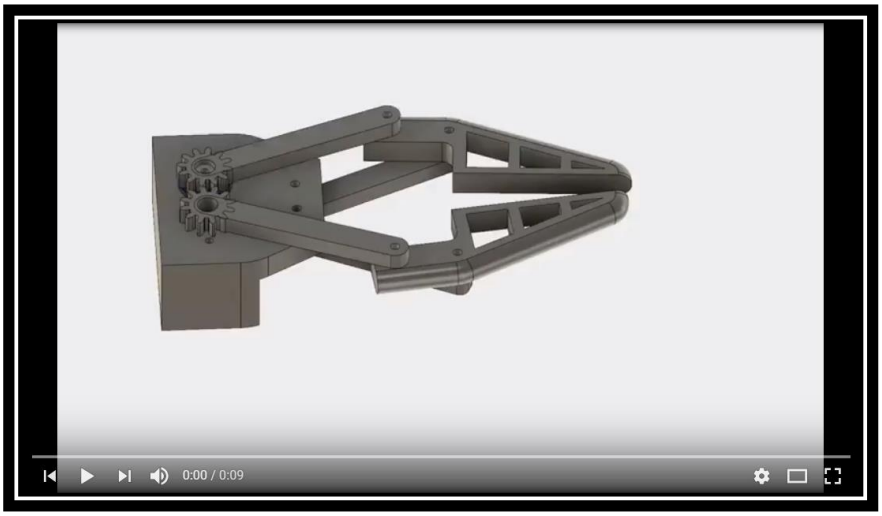








Click on Picture below for 'exploded view video'



Part B – Redundant Arm

Below is the final assembly of our arm including the end effector.

As you can see the arm brackets work very well with no sign of breakage or notable strain and the servos are taking the load of the weight very well.

Kinematics:

Motion Planning:

Arduino:

For this project we are using two Arduino Megas (a master and a slave) to display information to putty, export joint positions from Matlab, control the end effector and communicate between each other using I2C. I will break down the task of each Arduino board below. The reason for using the two boards is that each board only has one com port and we require 2 one for putty and another for Matlab.

A full in-depth explanation is explained on Github.

https://github.com/slperdomo-davies/ROCO_224/blob/master/ManipulatorDesignProject/Arduino_Code.md

Arduino(Master)

I2C

This board is the I2C master board. I2C will be used to fetch the positions in space for each individual Dynamixel and print their position to putty. The method to know which variable to get will be by using a flag. The flag will be raised by sending an interrupt request to the slave device.

Putty

This section uses the vt100 escape sequences to change the text colour and cursor position, write over old data and to setup a user-friendly terminal display. This display allow each piece of information and optional commands to be given to the user. By using a keyboard, we can control the 9g servo on the end effector to open and close the gripper.

The terminal is broken into sections. There is a portion dedicated to end effector for controlling and displaying information and there is another section that will display all the coordinates of the Dynamixels. The final section is to display the distance of the ultra-sonic sensors on redundant arm. These sensors have been limits to detect up to 99cm but can be used to detect as far out as 3 meters.

As we are using Arduino Megas we have been able increase the baud rate to 115200, necessary as there is such a large amount of data being sent to putty.

Arduino(Slave)

This board is linked with MATLAB using ROS and constantly reading and updating the positions of the Dynamixels, waiting for a request from the master device. Once the master sends a request the slave will then be interrupted and start checking flags (Which GPIO pin is high). When a flag has been read as high the slave will know which integer to send over the line and then return to updating joint positions until the next interrupt occurs

Experiments

What testing have you performed of the robot (or subsystems)

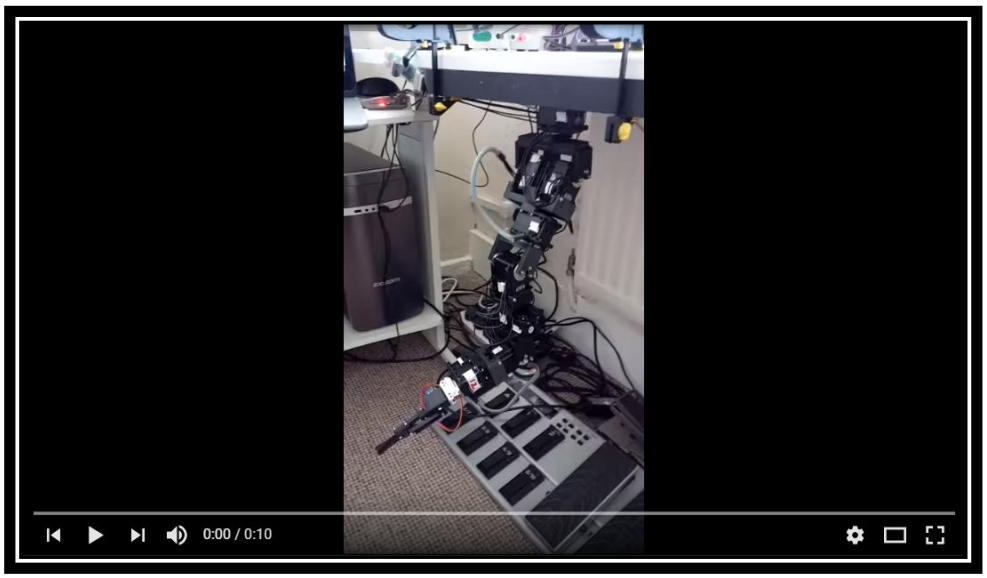
Experiment 1 - Testing Arm movement with forward kinematics

Method

This is our first test with the arm full assembled. We initially used Forward Kinematics using ROS and MATLAB to make sure that the arm was in working order.

Results

Click on picture below for video



Discussion

As seen in the video, the servos are rotating to allow the arm to move to a pre-configured location. This is using forward kinematics. This experiment does not showcase redundancy because the end effector is not kept in place.

Final Experiment – Implementation by motion planning with forward kinematics

Method

For demonstration purposes, as our inverse kinematics could not be resolved we decided to use motion planning with forward kinematics instead.

Result

Discussion

Although in its most basic form we were however able to achieve visual observation of redundancy in the arm, since the end-effector remains at a fixed point while the rest of the arm could move.

Conclusion

Our primary aim for this project was to showcase redundancy. By using more than 6 servos and through the implementation of MATLAB, we have met this aim.

Kinematics for the arm have been calculated to allow us to control its behaviour. Arduino embedded programming has enabled us a way to show information and allow some degree of control over the manipulator. The use of Fusion 360 and 3D printing gave us a custom design that we could fine-tune according to structural analysis on the system.

We have added our own elements to this project in the form of a 'gripper' end effector and sensors to provide the manipulator further automation.

References:

https://www.makeitfrom.com

https://www.trossenrobotics.com

https://servodatabase.com

http://akizukidenshi.com