University of Kalamoon

Faculty of engineering

Department of Mechatronics

Implementation of a robotic arm and it's controlling, using motion capturing

A project submitted in partial fulfillment of the requirement of the degree B.Sc. in the Department of Mechatronics

Done by:

Baraa Akbik **Ahmad Malas**

Supervised by:

Dr.Abd Alrazzak Dabbour Dr.Sami Sharbak

2024-2025

Abstract

This project presents the design and implementation of a robotic arm and neck controlled through motion capturing technology, utilizing BNO055 sensors to capture hand movements, MPU-6500 to capture head movements and potentiometers to monitor finger movements.

The robotic arm, designed with seven degrees of freedom (7-DOF), the head and neck are designed with three degrees of freedom (3-DOF), mirrors the intricate movements of a human hand and head providing a detailed and responsive control system. The arm and neck structure are modeled using SolidWorks and fabricated with a resin printing method, allowing for precise articulation and flexibility.

To facilitate accurate motion replication, the project explores advanced sensor fusion techniques to integrate data from the BNO055 sensors, MPU-6500 and potentiometers effectively. Furthermore, it includes the application of Field-Oriented Control (FOC) algorithms for the precise management of Brushless DC motors, ensuring smooth and efficient motion of the robotic hand.

This will outcome a sophisticated robotic system capable of performing tasks with human-like dexterity, contributing to the fields of robotics and human-computer interaction. This work demonstrates the potential for enhanced robotic control systems through the integration of motion-sensing technologies and advanced control methodologies.

The aim of this study is to develop and implement a sophisticated robotic arm that can accurately mimic human hand movements through the utilization of an advanced motion capturing system.

Key Words:

Kinematics, Motion Capture, Split ring compound planet epicyclic gearboxes, Brushless DC Motor, Field-Oriented Control, IMU, Sensor Futions, SolidWorks, Degrees of Freedom.

Table of Contents

CHAPTER 1 ..... **1**

THE GENERAL FRAMEWORK OF THE PROJECT **1**

1.1 Introduction 1
1.2 Purpose of Study 3
1.3 Study challenges 4
1.4 Literature Review 5
1.4.1 Introduction *5*
1.4.2 MIT's Cheeta Robot *6*
2.8.2 NOMAD Quadrupedal Robot *7*

CHAPTER 2 ..... **8**

THEORETICAL FRAME **8**

2.1 Human hand movement analysis 8
2.1.1 Carpometacarpal (CMC) Kinematics: *9*
2.1.2 Biaxial Joints: *10*
2.1.3 CMC Joints in the Palm: *11*
2.1.4 Functional Importance of CMC Joints: *13*
2.2 Human Spine analysis: 14
2.2.2 Balance Mechanism: *15*
2.3 Study tools: 16
2.3.1 Hardware: *16*
2.3.2 Actuators: *18*
2.3.3 3D printer: *20*
2.3.4 Other Tools: *20*
2.3.5 Software: *21*
2.3.6 Controller: *22*
2.4 Sensors Implementation: 23
2.4.1 Sensor Fusion: *24*
2.5 Selection of The Actuator: 28
2.5.1 A2212 motor: *28*
2.5.2 Servo90: *28*
2.5.3 Nema 17 (single): *29*
2.5.4 MG996R servo: *29*
2.5.5 SG-90: *30*
2.6 Brushless Motor Position Control: 30
2.6.1 Field-Oriented Control (FOC) [5]: *30*
2.6.2 PID Control: *32*
2.7 Kinematics of the arm: 33
2.8 Additive production strategies: 35
2.9 Selection of Gearbox: 37
2.9.1 Type of gears: *37*
2.9.2 Types of gears: *43*

CHAPTER 3 ..... **45**

THE DESIGNING OF THE PROJECT **45**

3.1 PROJECT HYPOTHESES: 45
3.1.1 Primary Hypothesis (System-Level): *45*
3.1.2 Mechanical Design Weight Hypotheses: *45*
3.1.3 Material Optimization Hypothesis: *45*
3.1.4 Mechanical Performance Hypothesis: *46*
3.1.5 User Comfort Hypothesis: *46*
3.2 MECHANICAL DESIGN: 46
3.2.1 Force analysis: *46*
3.2.2 Cad Designs: *48*
3.2.3 Gearbox Design: *65*
3.2.3Belt Drives Design: *68*
3.3 ELECTRICAL DESIGN: 69
3.4 CAPTURING DATA FROM SENSORS: 70
3.4 RECEIVING SENSORS DATA AND MOVING THE ROBOT: 71
3.4 KINEMATICS OF THE ARM: 71

CHAPTER 4 ..... **74**

RESULTS OF THE STUDY **74**

4.1 INTRODUCTION 74
4.2 STRESS RESULTS 74
4.2.1 Shoulder Stress Analysis: *75*
4.2.2 Forearm Stress Analysis: *79*
4.2.3 Comparison between Aluminum 6060 vs Carbon Fiber: [9] *83*
4.3 THE ELECTRICAL RESULT: 84
4.4 RESULT OF SENSOR FUSION IN IMU: 85
4.4.1 Sensors data without using any filter: *85*
4.4.2 Complementary filter results: *88*
4.4.3 Kalman filter results: *91*
4.4.4 Complementary filter vs Kalman filter: *94*
4.5 GEARBOX RESULTS: 96
4.6 KINEMATICS RESULTS: 98

CHAPTER 5 ..... **99**

FUTURE PROSPECTS **99**

REFERENCES **100**

Figure index

Fig 1.1 Mit's Cheeta robot 7
Fig 1.2 NOMAD Quadrupedal 8

Fig 2.1 of Trapezium.....11
Fig 2.2 Saddle joint.....12
Fig 2.3 CMC palm.....13
Fig 2.4 human spine.....15
Fig 2.5 Human spine vertebrae.....15
Fig 2.6 The Proprioception of the spin.....16
Fig 2.7 BNO055.....17
Fig 2.8 MPU-6500.....17
Fig 2.9 rotary potentiometer.....18
Fig 2.10 Flex sensor.....18
Fig 2.11 AS5600 12-Bit Digital Magnetometer Encoder Module.....18
Fig 2.12 A2212 Brushless motor.....19
Fig 2.13 Nema 17 single.....19
Fig 2.14 Servo 50kg.....20
Fig 2.15 MG996R servo.....20
Fig 2.16 SG90 servo motor.....21
Fig 2.17 Elegoo Resin 3D Printer.....21
Fig 2.18 Solidworks.....22
Fig 2.19 MATLAB.....22
Fig 2.20 PyCharm.....22
Fig 2.21 Arduino Uno.....23
Fig 2.22 Arduino Mega.....23
Fig 2.23 Complementary filter Box Diagram.....26
Fig2.24 Kalman filter Box Diagram.....28
Fig 2.25 Quadrature and Direct Forces.....32
Fig 2.26 Field Oriented Control diagram.....32
Fig 2.27 PID Controller Diagram.....34
Fig 2.28 Euler Rotation Matrix.....35
Fig 2.29 Elegoo Resin 3D Printer.....37
Fig 2.30 Harmonic gearbox.....41
Fig 2.32 Planetary gearbox.....43
Fig 2.33 Split ring compound planet epicyclic gearbox.....44

Fig3.1 Finger motion capturing design.....49
Fig 3.2 The Base That the POTs are position at.....49
Fig 3.3 The Base that the IMU will be Mounted on.....50
Fig 3.4 Isometric view of first design shoulder.....51
Fig 3.5 Frontal view of first design shoulder.....51
Fig 3.6 Side view of shoulder.....51
Fig 3.7 Isometric View of second design shoulder.....52

Fig 3.8 Frontal view of second design shoulder.....52
Fig 3.9 side view of second design shoulder.....52
Fig 3.10 Explosion view of shoulder.....53
Fig 3.11 First design of Elbow.....54
Fig 3.12 second design of elbow.....54
Fig 3.13 Third design of elbow.....55
Fig 3.14 Third design of elbow.....55
Fig 3.15 Explosion view Elbow.....56
Fig 3.16 wrist first design.....57
Fig 3.17 wrist second design.....57
Fig3.18 Wrist third design.....58
Fig 3.19 Explosion view of the wrist.....59
Fig 3.20 First finger design.....60
Fig 3.21 Second finger design.....61
Fig 3.22 Third finger design.....61
Fig 3.23 Fourth finger design.....62
Fig 3.24 Explosion view of the entire hand.....63
Fig 3.25 Split ring gearbox exploded view.....65
Fig 3.26 isometric Gearbox rear view.....65
Fig 3.27 Gearbox isometric frontal view.....65
Fig 3.28 Block Diagram of Capturing Sensors Data and Receiving Them.....70

Fig 4.1 stress study full 3D model.....74
Fig 4.2 Stress study aluminum 1060 stress graph.....75
Fig 4.3 Stress Study aluminum 1060 displacement.....75
Fig 4.4 Stress Study aluminum 1060 strain.....76
Fig 4.5 Stress study carbon stress graph.....76
Fig 4.6 Stress Study carbon displacement.....77
Fig 4.7 Stress Study aluminum 1060 strain.....77
Fig 4.8 stress study 3D model forearm.....78
Fig 4.9 Displacement study 3D model forearm.....79
Fig 4.10 Stress study aluminum 1060 stress graph forearm.....80
Fig 4.11 Displacement study aluminum 1060 stress graph forearm.....80
Fig 4.12 Strain study aluminum 1060 stress graph forearm.....81
Fig 4.13 gyro and accelerometer pitch sensor data.....84
Fig 4.14 gyro and accelerometer roll sensor data.....85
Fig 4.15 Gyro drift.....86
Fig 4.16 roll Complementary filter results.....87
Fig 4.17 pitch Complementary filter results.....88
Fig 4.18 noise Complementary filter results.....89
Fig 4.19 pitch Kalman filtre Resuls.....90
Fig 4.20 roll Kalman filter results.....91

Fig 4.21 noise Kalman filter results.....92
Fig 4.22 Kalman vs Complementary for a stable system.....93
Fig 4.23 Kalman vs Complementary for noise/vibration.....94
Fig 4.24 Split ring gearbox.....95
Fig 4.25 split ring gearbox vs planetary gearbox.....96
Fig4.26 Quaternion to Rotation Matrix.....97

Table index

Table 2.1 comparison of motors.....29
Table 2.3 Characteristics of Elegoo ABS-like Resin and FDMs' ABS.....37
Table 2.4 Comparision between spur, helix, and bone gears.....45

Table 3.2 Finger design compression.....62
Table 3.1 Gears Parameters.....64
Table3.2 Denavit-Hartenberg parameters of the kinematic model.....71
Table3.3: DH-Table of the project.....71

Table 4.1 stress test.....74
Table 4.2 stress test.....75
Table 4.3 stress test.....75
Table 4.4 stress test.....76
Table 4.5 stress test.....76
Table 4.6 stress test.....77
Table 4.7 stress test.....77
Table 4.8 stress test.....78
Table 4.9 stress test.....79
Table 4.10 stress test.....79
Table 4.11 stress test.....80
Table 4.12 stress test.....80
Table 4.13 stress test.....81
Table 4.14 Comparison between Aluminum 6061 vs Carbon Fiber.....82

Chapter 1

The General Framework Of The Project

1.1 Introduction

Motion capture technology has transformed numerous sectors, such as computer graphics, robotics, and healthcare. Its capacity to monitor and reproduce human movement in real-time opens a wide array of applications, including animation, prosthetics, and rehabilitation.

Specifically, hand movements represent a complex and intricate facet of human motion that demands a high level of dexterity and coordination. Capturing the detailed movements of the hand, including finger articulation as well as wrist, elbow, and shoulder rotations, poses a significant challenge and has attracted considerable attention in recent years.

Traditional motion capture systems, like optical or marker-based setups, come with drawbacks regarding cost, complexity, and portability. These systems typically need a large, dedicated space and are often limited to laboratory or studio environments. In contrast, wearable motion capture systems that utilize Inertial Measurement Units (IMUs) and Potentiometers (POTs) present a more adaptable and portable alternative. These sensors can be incorporated into a wearable device, enabling real-time motion capture in various environments.

This thesis introduces an innovative method for capturing hand movements using IMUs and POTs linked to an ESP-32 microcontroller, and the communication between the ESP-32 and other micro controllers for an optimal control of the robot. The IMUs require sensor fusion algorithms to achieve the best performance from their sensors.

The goal of the system is to accurately record the intricate motions of the human hand, encompassing finger articulation as well as rotations of the wrist, elbow, shoulder, and neck, and use the data gathered by the system to move a robotic neck and arm, designed to be similar to a human neck and arm.

Some of the actuators of the robot are controlled using Field-Oriented Control (FOC) algorithms, which emerged as a pivotal advancement in the control of brushless direct current (BLDC) motors.

Which enhances the responsiveness and efficiency of robotic actuators. The implementation of FOC techniques facilitates improved torque production and dynamic performance, making it an essential component in modern automation systems. As such, the

integration of FOC in BLDC motor control not only exemplifies a significant technical breakthrough but also plays a crucial role in enhancing the capabilities and functionality of robotic systems in various applications.

1.2 Purpose of Study

The main goal of this study is to create and test a new wearable device that tracks human movements accurately. This device will control a robotic arm, and a robotic neck designed using SolidWorks software. The design focuses on improving how well the robotic arm moves and performs. By combining these technologies, we hope to find effective ways to use this system in real-time to help with assistive technologies and improve how humans and robots interact. Through thorough testing and improvements, we aim to show how movement tracking can work well with advanced robotic arms.

Specifically, the objectives of this study are:

- To design a wearable device that accurately tracks human movements.
- To build a robotic arm using SolidWorks software that can replicate these movements.
- To test how well the wearable device and robotic hand work together in real-time.
- To investigate the potential applications of the proposed system in various fields, including robotics, prosthetics.

By achieving these objectives, this study aims to contribute to the development of a novel and innovative motion capture system that can accurately track human movements in real-time and has the potential to revolutionize various fields that rely on hand movement tracking.

And to design a humanoid robot arm capable of mimicking the movement of the human operating it. the robot while operating payload capacity of 1kg.
The design must ensure efficiency precise movement and robustness while providing seven degrees of freedom (7 DOF) similar to a normal human arm and covering an average human arm workspace and designing robust fingers for the hand.

Key Requirements:

To build a robotic arm that mimics human arm movements, certain key criteria must be met. These ensure smooth, accurate, and natural motion, along with proper control and feedback. Here are the main requirements needed.

- Seven degrees of freedom movement
- Payload Capacity of 1kg
- Light weight design
- Covering an average human arm workspace
- High precision
- Robust finger movement
- Capturing the movement of the user arm joint by joint
- Simulating the movement of the arm while controlling it with suite

1.3 Study challenges

This study faces several challenges that need to be addressed to ensure the successful development and implementation the challenges are divided into two deferent category

The wearable motion capture suite of the humen arm: Some of the challenges that will be encountered in this study include:

1. **Sensor Calibration and Synchronization:** Calibrating and synchronizing the IMUs and POTs to ensure accurate and reliable data collection will be a challenge.
2. **Data Noise and Filtering:** The data collected from the IMUs, and POTs may contain noise and errors, which need to be filtered out to ensure accurate data analysis.
3. **Motion Capture System Accuracy:** The accuracy of the motion capture system in tracking hand movements will be a challenge, particularly in capturing fine finger movements.
4. **User Acceptance and Comfort:** Ensuring that the system is comfortable and acceptable to users, particularly in terms of wearability and ease of use, will be a challenge.

5. **Real-time Data Transmission:** Transmitting data from the ESP-32 microcontroller to the computer in real-time will be a challenge, particularly in terms of data transmission speed and reliability.

The design of the humanoid robot arm:

1. **The Mechanical Design of the Robot**: The design must ensure efficiency precise movement and robustness while providing seven degrees of freedom (7 DOF) like a normal human arm and (3 DOF) for the neck, covering an average human movement workspace and designing robust fingers for the hand.
2. **Motor Control and Synchronization:** Controlling and synchronizing the motors to ensure smooth and accurate movement of the 3D printed hand model will be a challenge.
3. **Power Supply and Energy Efficiency:** Ensuring a stable and efficient power supply to the system, particularly to the motors, will be a challenge.
4. **System Integration and Testing:** Integrating all the components of the system, including the IMUs, POTs, ESP-32 microcontroller, and other controllers, motors,

and power supply, and testing the system to ensure that it functions as expected will be a challenge.
5. Ensure the communication between all the micro-controllers.
6. **Cost and Affordability:** The cost of the system, particularly the cost of the IMUs, POTs, and motors, may be a challenge, particularly if the system is to be affordable for widespread use.

Addressing these challenges will be crucial to the successful development and implementation of the wearable motion capture system for hand movement.

1.4 Literature Review

1.4.1 Introduction

Robotic teleoperation systems have emerged as a transformative technology in the field of robotics, enabling the remote control of robotic devices through human input. These systems allow operators to interact with and manipulate robotic agents from a distance, effectively extending human capabilities into environments that are hazardous, inaccessible, or require high precision. By leveraging real-time feedback and control, teleoperation facilitates a more intuitive and seamless interface between humans and machines.

Among the most advanced forms of teleoperation are systems that replicate human motion with high fidelity. Human-like robotic systems, particularly those that mirror the complexity and dexterity of the human hand and head, are critical in applications where nuanced interaction and control are required. These systems are increasingly being integrated into various domains:

- **Healthcare**: In robotic surgery, rehabilitation, and prosthetics, human-like robotic limbs allow for delicate procedures and personalized patient interaction.
- **Industry**: In manufacturing and hazardous environments, teleoperated robots can perform intricate or dangerous tasks, reducing risk and increasing precision.
- **Human-Computer Interaction**: Advanced robotic systems enable more natural interfaces for virtual reality, gaming, and assistive technologies for individuals with disabilities.

The development of such robotic platforms requires the integration of multiple disciplines, including sensor technology, mechanical design, control systems, and real-time data processing. This thesis presents the design and implementation of a robotic arm and neck system that mimics human movement through motion capture using a combination of inertial and analog sensors. By integrating BNO055 and MPU-6500 sensors for tracking arm and head motion, as well as potentiometers for finger joint detection, the system aims to deliver human-like motion with high responsiveness and control accuracy.

1.4.2 MIT’s Cheeta Robot

Most of the shoulder design and overall concept were inspired by MIT's Cheetah robot. Drawing upon the biomechanical efficiency and modular construction principles demonstrated in their work, my design aims to replicate similar levels of agility and performance. A key inspiration was their integration of brushless DC (BLDC) motors coupled with planetary gearboxes—an approach that enables high torque output, compactness, and precise control, all essential for dynamic locomotion. These components are particularly well-suited for robotic applications that demand rapid, responsive movements without compromising on structural integrity.

The shoulder mechanism in my project adopts a similar configuration, leveraging the high power-to-weight ratio and reliability of BLDC motors along with the efficiency and durability of planetary gearboxes. This combination not only provides the necessary torque amplification for limb articulation but also ensures smooth motion and consistent performance under varying load conditions. By adapting and refining these design cues to suit the unique requirements of my project, I sought to balance functionality with manufacturability. The resulting mechanism supports dynamic, lifelike movements while integrating seamlessly with the rest of the robotic structure, ultimately enhancing the robot’s overall mobility and energy efficiency.

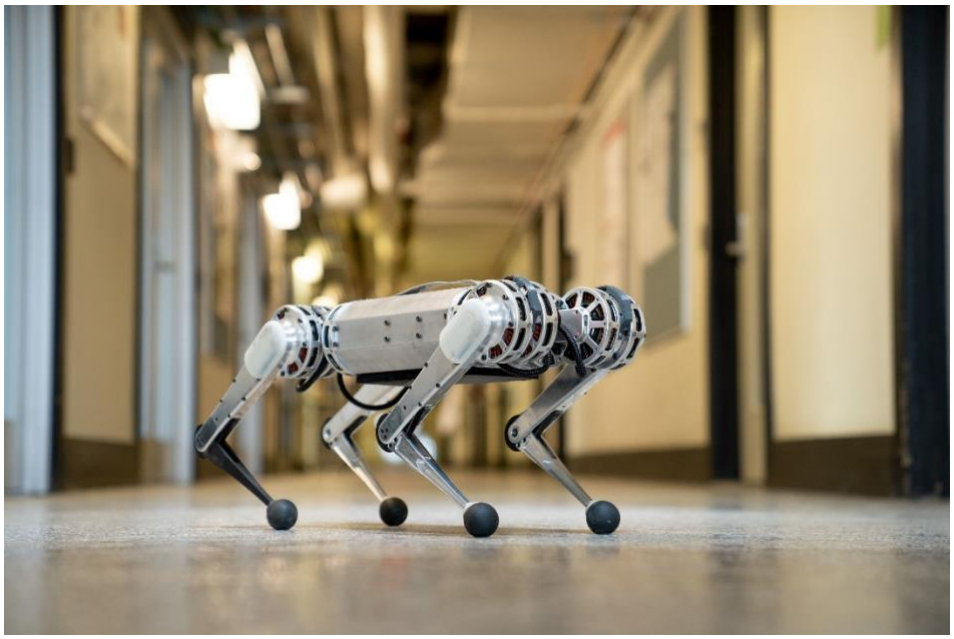


Fig 1.1 Mit’s Cheeta robot

2.8.2 NOMAD Quadrupedal Robot

Similar to MIT's Cheeta robot.

Fig 1.2 NOMAD Quadrupedal

Chapter 2

Theoretical Frame

This project, which involves the development a humanoid robot arm and neck while controlling it with wearable motion capture system, it require a multitude of components, each playing a crucial role in the overall system's accuracy and efficiency. The selection of these components was a daunting task, as it was essential to choose the most accurate and reliable ones while also considering cost constraints. Four of the most critical components in this system are Actuators and Materials are responsible for the motion and accuracy of the arm, the controller and the sensors, which are responsible for collecting and processing data captured be the human movements. The controller, which serves as the brain of the system, must be capable of processing complex data in real-time, while the sensors, which are responsible for tracking human movements, must be highly accurate and sensitive.

Given the complexity of this system, it is essential to ground it in a solid theoretical framework that takes into account the principles and concepts underlying each of its components. This framework provides a foundation for understanding the relationships between the system's components and how they work together to achieve the desired outcome. By examining the theoretical aspects of each component, this project aims to develop a robust design of the arm and controlling it with an accurate motion capture suite, while contribute to the development of more accurate and efficient systems in the future.

Understanding human movement is key to developing dexterous robotic arm. By analyzing the arm, and neck movement patterns and kinematics, we can inform the design and control of robotic hands to make them more realistic, effective, and dexterous.

2.1 Human hand movement analysis **[1]:**

The human hand is a complex and versatile anatomical structure that enables us to perform a wide range of activities, from simple grasping and manipulation to intricate tasks that require precision and dexterity. At the heart of the hand's functionality are the joints that connect the bones, allowing for movement and flexibility. One of the most important joint complexes in the hand is the Carpometacarpal (CMC) joint, which plays a crucial role in enabling the hand to perform various movements and functions. In this context, understanding the kinematics of the CMC joint is essential for appreciating the hand's overall functionality. This text provides an overview of the basic movements and functions of the CMC joint, highlighting its importance in enabling the hand to perform its many essential roles.

2.1.1 Carpometacarpal (CMC) Kinematics:

The Carpometacarpal (CMC) joints are the connection points that link the carpal bones to the metacarpal bones in the hand. These joints play a crucial role in enabling the hand to perform various and complex movements, allowing for efficient execution of daily tasks, such as grasping objects and using different tools.

2.2.1.1 Basic Movements of CMC Joints

The CMC joints allow key hand movements like flexion, extension, abduction, adduction, pronation, and supination. These motions help the hand perform various tasks with precision and flexibility.

Flexion/Extension:

- **Flexion:** The movement that decreases the angle between the metacarpal and carpal bones, resulting in the hand bending forward.
- **Extension:** The movement that increases the angle between these bones, resulting in the hand straightening backward.

Abduction/Adduction:

- **Abduction:** Moving the finger away from the midline of the hand, such as moving the little finger away from the other fingers.
- **Adduction:** Moving the finger towards the midline of the hand, such as moving the little finger towards the middle finger.

Pronation/Supination:

- **Pronation:** Rotating the hand so that the palm faces downward.
- **Supination:** Rotating the hand so that the palm faces upward.

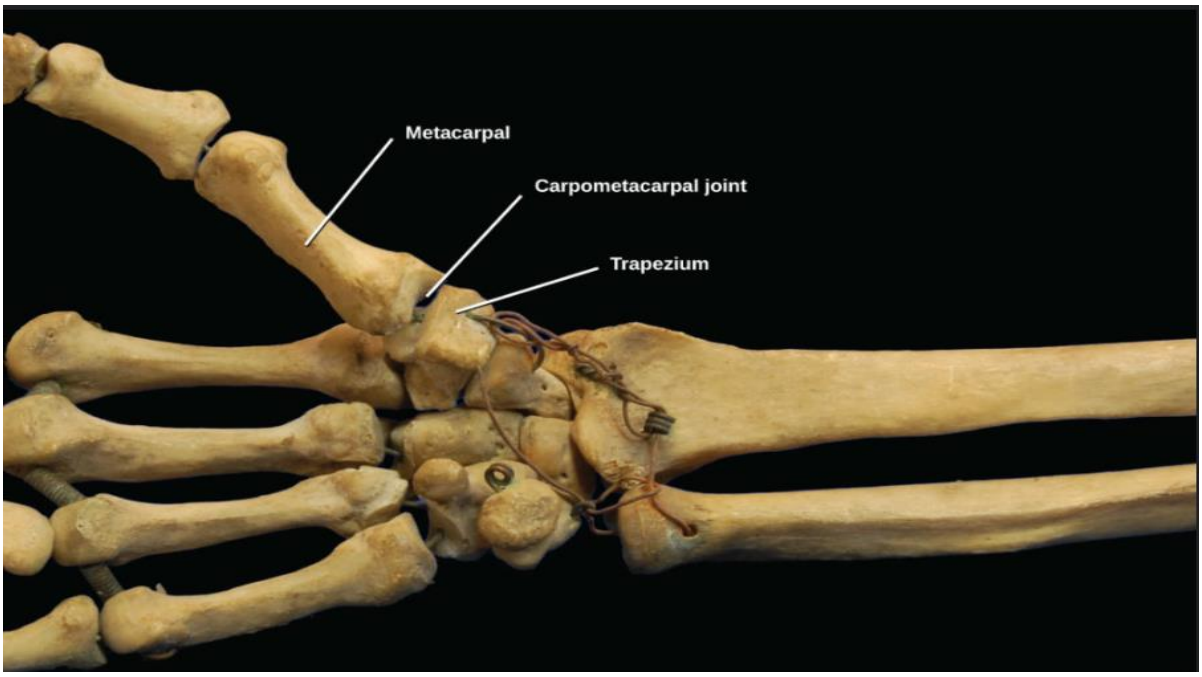


Fig2.1 of Trapezium

2.1.2 Biaxial Joints:

The saddle-shaped joints have two axes, allowing for movement in the sagittal plane and frontal plane.

The movement of the saddle-shaped joint is similar to that of the condyloid joints, allowing for movement in two planes and enabling the following movements, but without rotation around the axis:

- **Flexion** and **Extension**: 53°
- **Abduction** and **Adduction**: 42°
- **Circumduction**: 17°

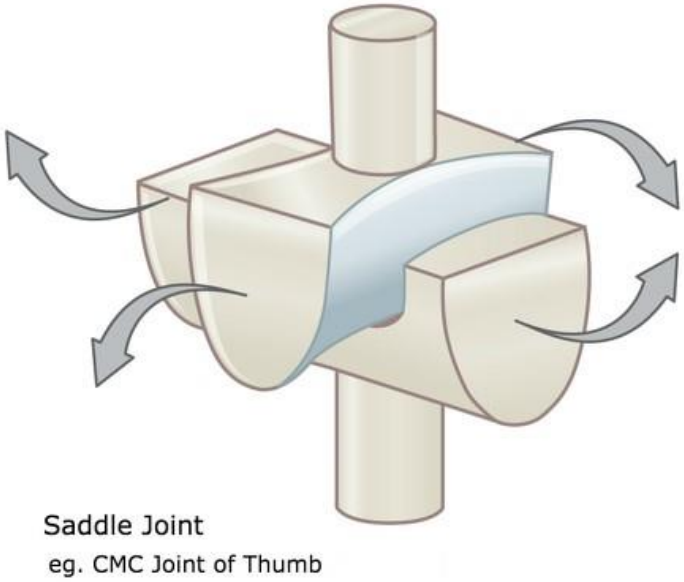


Fig2.2 Saddle joint

2.1.3 CMC Joints in the Palm:

Very important for functional tasks, especially grasping tasks.

The 2nd and 3rd CMC joints have limited movement, while the 4th CMC joint has approximately 10° of movement and the 5th CMC joint has approximately 30° of movement.

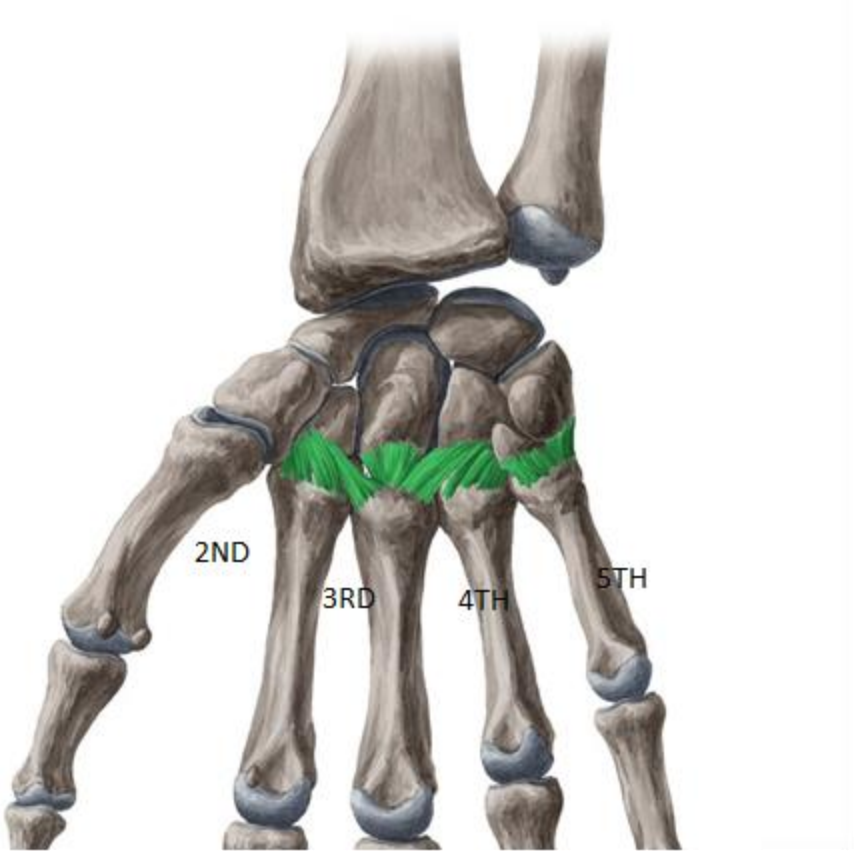


Fig2.3 CMC palm

2.1.4 Functional Importance of CMC Joints:

The CMC joints are vital for hand movement, providing both stability and flexibility for precise and strong functions.

Forming the Metacarpal Arch:

These joints contribute to forming a natural arch in the hand, which enhances grip strength and allows for even distribution of force across different parts of the hand. This arch formation is essential for performing tasks that require precision, such as writing or using small tools.

Stability and Dynamic Movement:

The CMC joints provide a balance between stability and flexibility, enabling the performance of precise movements without losing control. They allow for the execution of consecutive and smooth movements that require high coordination between muscles and joints.

Pivotal Role in Enabling Hand Function:

The Carpometacarpal joints play a pivotal role in enabling the hand to perform complex daily tasks with flexibility and precision.

Understanding the kinematics of these joints is essential for developing advanced technologies in fields such as prosthetics and robotics. Continuous progress in the study of these joints will open up new avenues for future applications, enhancing the quality of life for individuals and contributing to the advancement of biotechnology.

2.2 Human Spine analysis:

The human spine plays a crucial role in maintaining balance and posture and understanding its mechanical and structural features can provide valuable insights for developing a balancing robot [2]. Here are key aspects of the spine's evolution and its influence on balance:

2.2.1 Structure of the Spine:

- The human spine consists of 33 vertebrae, arranged in a natural curve (cervical, thoracic, lumbar, sacral, and coccygeal regions). These curves help distribute mechanical loads and maintain stability.

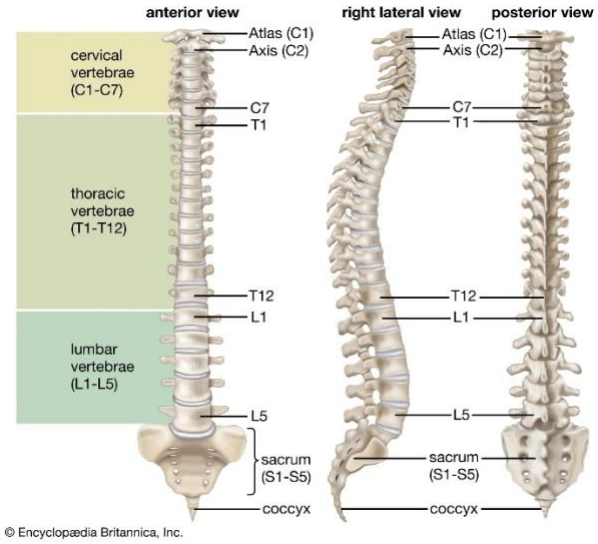


Fig 2.4 human spine

- Intervertebral discs between the vertebrae act as shock absorbers, allowing for flexibility while providing support.

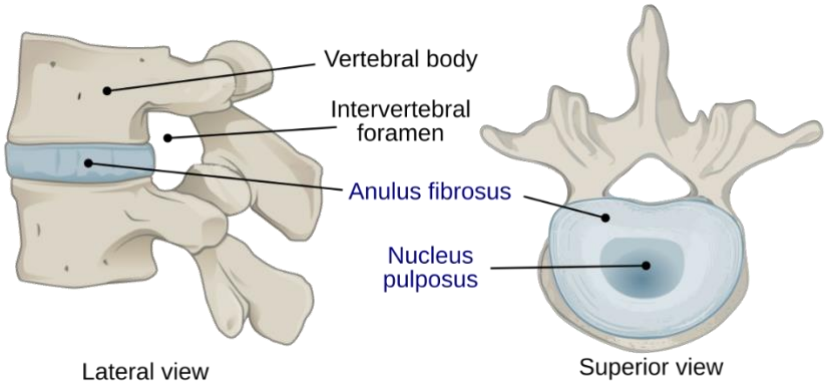


Fig 2.5 Human spine vertebrae

2.2.2 Balance Mechanism:

- The spine serves as a central support axis, allowing the body to shift its center of gravity. The vertebral column's alignment is essential for maintaining an upright posture and balance during movement.
- The presence of multiple points of articulation allows the spine to absorb and compensate for changes in position and external forces, which is crucial during dynamic activities.

- Proprioception in the Spine:
- Proprioceptors in the spine, particularly within the muscles and ligaments surrounding the vertebrae, provide feedback about body position and movement. This sensory information is vital for the central nervous system to coordinate balance.

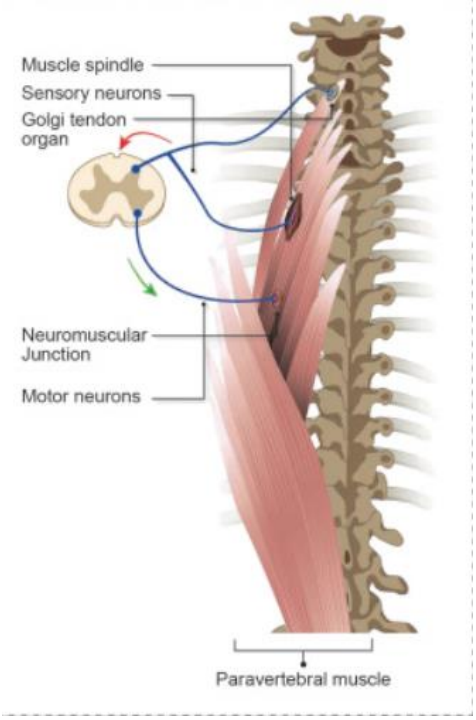


Fig 2.6 The Proprioception of the spin

- The spindles and Golgi tendon organs help the body detect changes in muscle length and tension, contributing to a refined sense of balance.

- Muscle Coordination:
- Core muscles, including the abdominal and back muscles, work in concert with the spine to maintain stability. This muscle coordination helps counteract destabilizing forces and allows for quick adjustments.
- The spine itself acts as a lever system, where muscular contraction alters body posture and aids balance during various activities, such as walking or running.

- Adaptations for Balance:
  - The evolution of the human spine involved adaptations for bipedal locomotion. The lordotic curves in the lumbar and cervical regions help stabilize an upright posture, allowing for effective balance during locomotion.
  - As humans evolved, the ability to shift the pelvis and thorax in relation to the head and lower limbs improved balance control.

2.2.3 Design Implications for Robots:

-Spinal Design: A robot's spine should have a segmented structure that mimics the flexibility and load-bearing capabilities of the human spine, allowing for dynamic posture adjustments.
- Sensory Feedback: Implementing proprioceptive sensors analogous to human proprioceptors will enable the robot to detect position and movement changes, facilitating real-time balance corrections.
- Dynamic Control: Incorporating adaptive control mechanisms that mimic human muscle coordination can enhance the robot's stability and response to external disturbances.

2.3 Study tools:

To achieve the objectives of this study, the following tools and equipment will be used:

2.3.1 Hardware:

1. **Inertial Measurement Units (IMUs):** Three BNO055 IMUs will be used to track the movement of the shoulder, wrist, and elbow, one MPU-6500 will be used to capture the head -neck- movement. The IMUs will measure the acceleration, angular velocity, and orientation of each joint.

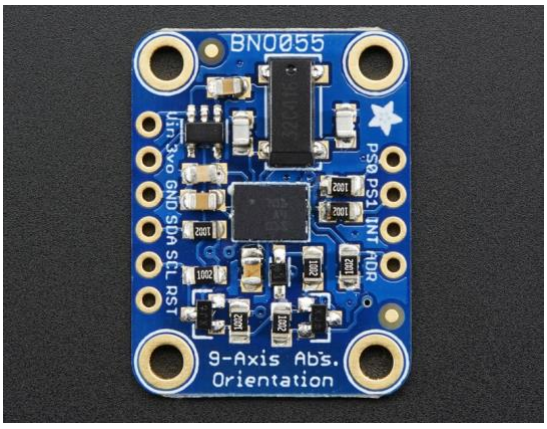


Fig2.7 BNO055

Fig 2.8 MPU-6500

2. **Potentiometers (POTs):** POTentiometers (POTs) will be employed to detect the angular position of each finger by measuring the rotation at each finger joint. These sensors can be implemented either as rotary potentiometers or flex sensors. Both types operate on a similar principle of varying resistance corresponding to the joint's movement, allowing precise quantification of finger joint rotation. This methodology provides reliable and accurate tracking of finger positions for applications requiring detailed motion analysis.

Fig2.9 rotary potentiometer

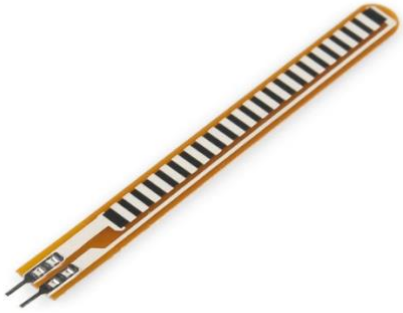


Fig2.10 Flex sensor

3. **Magnetometer encoder:** To get the position of the A2212 BLDC motor, for implementing Field-Oriented Control (FOC).

Fig2.11 AS5600 12-Bit Digital Magnetometer Encoder Module

4. **3D Printed Arm Model:** A 3D printed model of the arm will be created using a 3D printer. The model will be identical to the 3D model used in Blender.

2.3.2 Actuators:

1. A2212 Brushless motor:2 motors used for moving the shoulder-the most torque applied-on two axes.

Fig2.12 A2212 Brushless motor

2. Nema 17 stepper motor (Single): Used for rotating the elbow around the z-axis.

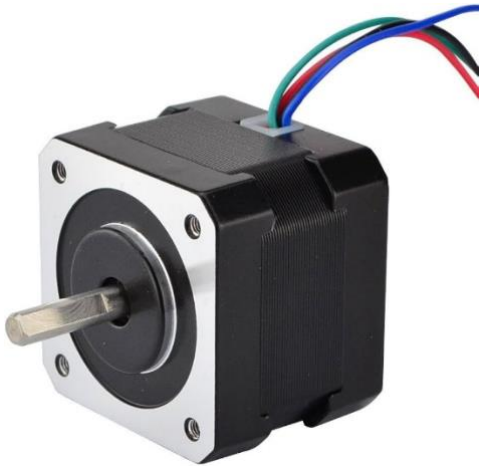


Fig2.13 Nema 17 single

3. Servo 50kg motor: 3 motors for rotating the wrist on 3-axis.

Fig2.14 Servo 50kg

4. MG996R servo motor: used for rotating the fingers.

Fig2.15 MG996R servo

5. SG90 servo motor: used for openning and closing of finger.

Fig2.16 SG90 servo motor

2.3.3 3D printer:

A resin 3D printer was essential for this project due to its high accuracy and superior resolution quality.

Fig 2.17 Elegoo Resin 3D Printer

2.3.4 Other Tools:

1. **Breadboard and Jumper Wires:** A breadboard and jumper wires will be used to connect the IMUs, POTs, and ESP-32 microcontroller.
2. **Power Supply:** A power supply will be used to power the Motors and other components.
3. **I2C Communication Software:** A communication software will be used to establish communication between the ESP-32 microcontroller and other micro-controllers.

4. **ESP-NOW wireless communication:** An ESP-NOW wireless connection is an easy way to establish a wireless connection developed by Espressif to establish a wireless connection between two ESPs.

2.3.5 Software:

1. **SolidWorks:** SolidWorks will be used to design and create a 3D model of the hand, which will be converted to STL format for 3D printing, and SolidWorks will be used for the stress analysis.
2. **MATLAB**: MATLAB will help in studying the kinematics.
3. **Pycharm**: A Python program designed to facilitate the study of kinematics by performing matrix multiplications.
4. **Arduino IDE**: For programming the micro-controller

Fig2.18 Solidworks

Fig2.19 MATLAB

2.20 PyCharm

2.3.6 Controller:

There were many types of controllers to choose from, but the most main types were Arduino Uno, ESP-32, and Arduino Mega. Each has its strengths and weaknesses, making them suitable for different applications. The Arduino Uno is easy to use and low-cost but has limited processing power and memory.

The ESP-32 is more powerful and feature-rich, with Wi-Fi and Bluetooth connectivity, which be helpful for transforming data between two controllers without worrying about cable management, and that is suitable for sensing (using one controller) and actuating (using another). The Arduino Mega has a large number of digital and analog pins, making it suitable for applications that require many sensors and actuators, but it's down size is the cost ,and the Arduino Mega does not conclude wireless connections.

After careful consideration, the ESP-32 was chosen for this project due to its unique combination of processing power, memory capacity, and wireless connectivity. The choice of sensors was also critical, with options including inertial measurement units (IMUs), potentiometers, and flex sensors. Understanding the theoretical aspects of these components is crucial for designing and developing an efficient and accurate wearable motion capture system.

Because the sensers and the actuators would be a more than the pins of the ESP-32 there mite be more than one controller moving and actuating the robot arm, this will happen by establishing eighter wired connection (ex:I2C,SPI), or by establishing a wireless connection (ex:WiFi,Bloutooth,ESP-Now).

Fig2.21 Arduino Uno

Fig2.22 Arduino Mega

2.4 Sensors Implementation:

Using variable resistors to sense finger movement is an innovative technique that enables the collection of accurate and effective data on hand performance. The first resistor measures the bending of the finger, allowing it to record the different angles at which the finger bends.

The second resistor is dedicated to sensing the distance between fingers. This data is essential for understanding hand interactions comprehensively. By providing information on the distance between fingers, the system can recognize more complex movements, such as grasping objects or forming specific shapes.

These variable resistors can work with electronic circuits, such as microcontrollers, to analyze the data and infer actions based on it.

Inertial Measurement Units (IMUs):

In this project, the BNO055 sensor was used. This advanced sensor combines acceleration, angular velocity, and magnetic field measurement technologies, making it an ideal tool for a wide range of engineering and technological applications. The device integrates a three-axis accelerometer, a three-axis gyroscope, and a three-axis magnetometer, allowing it to provide comprehensive motion tracking.

The accelerometer measures the acceleration of the body in three dimensions, the gyroscope captures changes in angle, and the magnetometer helps determine orientation relative to the Earth's magnetic field. One of the key features of the BNO055 is its built-in sensor fusion algorithm, which processes data from these three sensors to deliver highly accurate orientation information. This capability allows it to capture even the slightest changes in movement and rotation, making it exceptionally useful for applications in robotics, drones, and virtual reality.

And the MPU-6500 was used in this project to capture the movement of the head and neck, the MPU-6500 has three-axis accelerometer, and a three-axis gyroscope, allowing it to provide comprehensive motion tracking.

2.4.1 Sensor Fusion:

Even thaw the BNO055 has built-in sensor fusion, but it was very crucial to understand, and learn more about what sensor fusion is, how it works, and some algorithms of it, and implement it on the MPU-6500.

Sensor fusion is the process of combining data from multiple sensors to produce more accurate, reliable, and comprehensive information than what could be achieved with a single sensor alone. This technique leverages the strengths of different types of sensors and compensates for their weaknesses.

For example, in the context of motion tracking, an accelerometer provides information about linear motion and can detect changes in speed and orientation. However, it is sensitive to vibrations and can drift over time. A gyroscope measures angular velocity and rotation, offering precise orientation data but can also suffer from drift. A magnetometer helps provide heading information by measuring the magnetic field.

By fusing the outputs from these sensors, systems can achieve better performance in terms of accuracy and stabilizing orientation. Sensor fusion involves complex algorithms that analyze the incoming data, filter out noise, and integrate the various measurements to create a cohesive understanding of an object's movement and position. This is particularly important in applications such as robotics, autonomous vehicles, and augmented reality, where reliable motion tracking is essential for performance and safety.

Some algorithms to implement sensor fusion:

1.Using a complementary filter:

A complementary filter is a type of sensor fusion technique commonly used to combine data from different sensors, particularly in applications like motion tracking. It is especially useful for blending the outputs of accelerometers and gyroscopes.

How it Works:

1. Sensor Inputs: The complementary filter takes inputs from two sensors:
   - Accelerometer: Measures linear acceleration and can help estimate orientation but is affected by noise and can drift.
   - Gyroscope: Measures rotational rate and provides reliable orientation data over short periods but can also drift over time.

2. Weighting: The idea behind a complementary filter is to assign different weights to each sensor's data at different frequencies. The gyroscope data is trusted for rapid changes (high-frequency signals), while the accelerometer data is more reliable for providing long-term

stability (low-frequency signals).

3. Combination: The outputs from each sensor are combined using a simple formula. Typically, the orientation derived from the gyroscope is adjusted with a portion of the orientation data from the accelerometer. This helps mitigate the effects of drift from the gyroscope and the noise from the accelerometer.

Advantages:
- Simplicity: Complementary filters are relatively straightforward to implement and require minimal computational power compared to more complex algorithms like Kalman filters.
- Real-time Performance: Due to their simple nature, they are suitable for real-time applications where fast processing is essential.
- Robustness: They provide a good balance between the strengths and weaknesses of the sensors involved.
Disadvantages:

- Sensitivity to Noise: Although complementary filters reduce the effect of drift, they can still be sensitive to noise from the accelerometer. If the accelerometer readings are too noisy, it can lead to inaccurate orientation estimates.

- Fixed Tuning: The filter uses a fixed weighting factor, which might not be optimal for all applications or conditions. This can result in suboptimal performance if the dynamics of the system change.

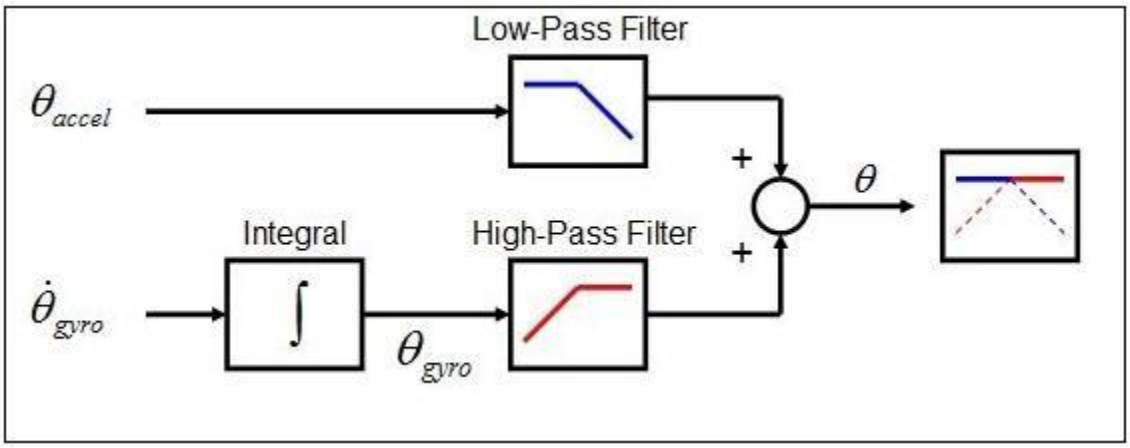


Fig 2.23 Complementary filter Box Diagram

- Low-Pass Filter for Accelerometer:

The low-pass filter is applied to the accelerometer signals to reduce the high-frequency noise while allowing the slower movements (such as changes in orientation) to pass through.

Accelerometers can be very sensitive to transient spikes (like vibrations), which can generate high-frequency noise. By using a low-pass filter, we can smooth out these spikes,

providing a cleaner signal that more accurately represents the gradual change in orientation.

- High-Pass Filter for Gyroscope:

The high-pass filter is used on the gyroscope signals to remove low-frequency drift. Gyroscopes can accumulate errors over time when they are stationary, leading to drift in the measurements.

By applying a high-pass filter, we allow the rapid changes (like quick rotations) to pass through while filtering out the slower drifts and biases. This helps maintain an accurate representation of angular velocity, which is crucial for determining orientation changes quickly.

2.Using a Kalman filter [4]:

The Kalman filter is an advanced mathematical algorithm that is widely used for estimating the state of a dynamic system from a series of noisy measurements. It is particularly valued in fields like robotics, aerospace, and navigation because it provides optimal estimates of unknown variables over time.

Key Concepts of the Kalman Filter:

1. Prediction and Update: The Kalman filter operates in two main steps:
   - Prediction: Using a mathematical model of the system, the filter predicts the next state based on the current state and known control inputs.
   - Update: Once a new measurement is available, the filter updates its estimate by combining the predicted state and the new measurement, applying a weighting based on their respective uncertainties.

2. State Estimation: The Kalman filter estimates the hidden state of a system, which can include various parameters, such as position, velocity, and acceleration, depending on the model used.

3. Handling Noise: Kalman filters are particularly effective at handling noise from sensors. The algorithm assumes that both the process (how the state changes) and measurement (how the state is observed) noises are Gaussian. This assumption allows the filter to calculate the best estimate of the state based on the available information.

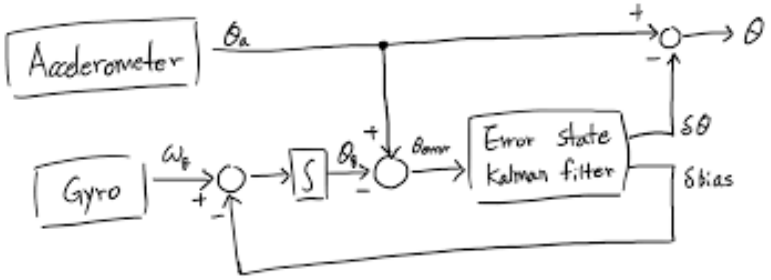


Fig2.24 Kalman filter Box Diagram

The classic Kalman filter is designed for linear systems. However, there are variations, such as the Extended Kalman Filter (EKF) or Unscented Kalman Filter (UKF), which can handle non-linear systems.

2.5 Selection of The Actuator:

It's important to choose carefully the suitable actuator for the project there were multiple options to choose the motors from below are some of the motors that were considered.

Table 2.1 comparison of motors:

	Stepper motor	Servo motor	Brushless dc motor closed loop	Dc motor closed loop
Presidion	Very high but needs feedback for detecting back drive	High	High	High
Speed	Low to moderate	Moderate	High	Moderate to high
Torque	Good while operating on a low speed	Good	High with a gearbox	Good with a gearbox
Efficiency	Moderate	high	High	Moderate

We have chosen BLDC with feedback because of it high efficiency and high torque and suitable speed while using a gearbox

The selected motors were the following

2.5.1 A2212 motor:

The 1000kv motor is a powerful three phase delta brushless dc motor compared to its size it was selected do its high power that it can provide, but in order to use it in a robotic application we need to design a special gearbox for it.

to control the motor an esc is used with a two-relay configuration because in order to control the motor it needs to be provided with a special synchronized signal in to its phases, the relays are used in order to provide a direction control.

2.5.2 Servo90:

The 50kg servo motor, an industrial-grade servo actuator, was utilized in this study for its exceptional power and precision. Weighing approximately 83g, this servo motor delivers a stall

torque of 50 kg-cm and a maximum rotational speed of 60 RPM, making it well-suited for demanding applications requiring high torque and accuracy.

There were two candidates for this motor application, the first which is Nema 17 stepper motor (double), and the other one which is Servo 50kg.

Servo Motor vs Stepper Motor: A Performance Comparison

- Servo motor (50kg) draws more current than stepper motor
- Servo motor generates more torque than stepper motor
- Servo motor is lighter than stepper motor

Implications

- Servo motor suitable for high-performance applications where torque and weight are critical
- Stepper motor suitable for applications where power efficiency and compactness are not primary concerns

Conclusion

The servo motor's high current draw is offset by its superior torque generation and reduced weight, making it an attractive option for high-performance applications.

2.5.3 Nema 17 (single):

The Nema17 single-phase stepper motor, a type of hybrid stepper motor, was utilized in this investigation to provide precise and reliable stepping motion. With a step angle of 1.8° and a holding torque of 3.5 Nm, this motor is well-suited for applications requiring high precision and moderate torque. The Nema17's compact design, featuring a 42mm frame size and a 1.8° step angle, enables precise control over motor rotation, making it an attractive option for applications such as 3D printing, CNC machining, and robotics. Furthermore, the motor's single-phase design and 2A current rating facilitate efficient operation and reduced power consumption. The Nema17's robust performance, reliability, and adaptability render it an ideal choice for a wide range of precision motion control applications.

2.5.4 MG996R servo:

The MG966R servo motor, a high-performance, digital servo motor, was employed in this study to provide precise and reliable actuation. With a stall torque of 12.5 kg-cm and a maximum rotational speed of 60 RPM, this servo motor is well-suited for applications requiring high torque and accuracy. Furthermore, the MG966R's integrated control system, featuring a 12-

bit resolution encoder, enables precise positioning and velocity control, thereby facilitating smooth and accurate motion. Additionally, the motor's compact design and low power consumption (operating at 12V and 5A) make it an attractive option for applications where space and energy efficiency are critical considerations. The MG966R's robust performance, reliability, and adaptability render it an ideal choice for a wide range of robotic and mechatronic applications.

2.5.5 SG-90:

The SG-90 servo motor, a widely utilized micro servo actuator, was selected for this research due to its exceptional combination of precision, reliability, and compactness. With a torque output of 2.5 kg-cm and a rotational speed of 60 RPM, this servo motor is particularly well-suited for applications requiring precise angular control in confined spaces. The SG-90's diminutive size, weighing only 23g, and compact dimensions (22.2mm x 12.2mm x 28.5mm) make it an ideal choice for robotic and mechatronic systems where space is a critical constraint. Furthermore, its high-resolution encoder and 180° range of motion enable precise positioning and velocity control, thereby facilitating the development of sophisticated autonomous systems.

2.6 Brushless Motor Position Control:

2.6.1 Field-Oriented Control (FOC) [5]:

DC Brush, Brushless DC (BLDC) and stepper motors are the three most commonly used motor types for advanced positioning and velocity motion control applications. Of these, Brushless DC and step motors are 'multi-phase', meaning they require some type of external coil excitation to keep the motor moving.

For Brushless DC motors, magnetic fields are generated by magnets mounted directly on the rotor and by coils in the stator. The stator windings generally come in a 3-phase configuration and are arranged to be 120 electrical degrees apart from each other. It is the sum of the force generated by these three phases that will ultimately generate useable motor rotation.

Depending on how the individual magnetic coils are driven, they can interact to create force that does not generate rotational torque, or they can create force which does generate rotation. These two different kinds of force are known as quadrature (Q) and direct (D), with the useful quadrature forces (not to be confused with quadrature encoding scheme for position feedback devices) running perpendicular to the pole axis of the rotor, and the non-torque generating direct forces running parallel to the rotor's pole axis (Figure 2.17).

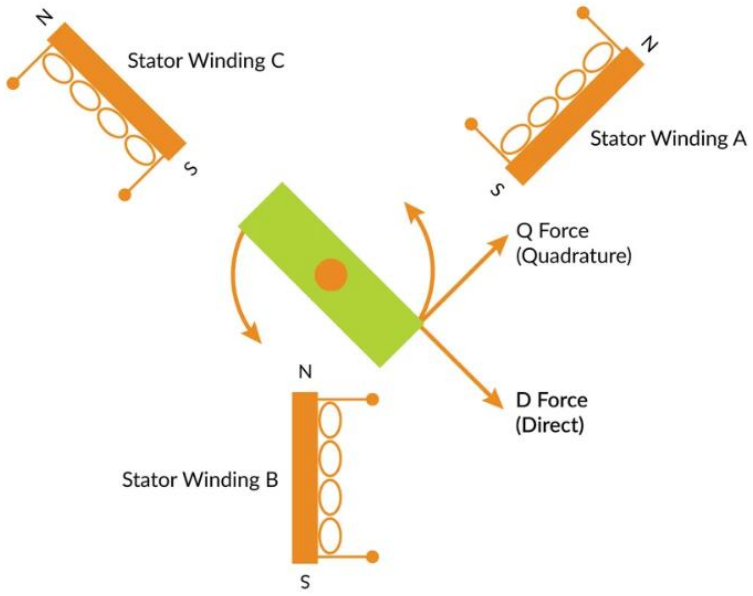


Fig 2.25 Quadrature and Direct Forces

The trick to generating rotation is to maximize Q (quadrature) while minimizing D (direct) torque generation. If the rotor angle is measured using a Hall sensor or position encoder, the direction of the magnetic field from the rotor is known.

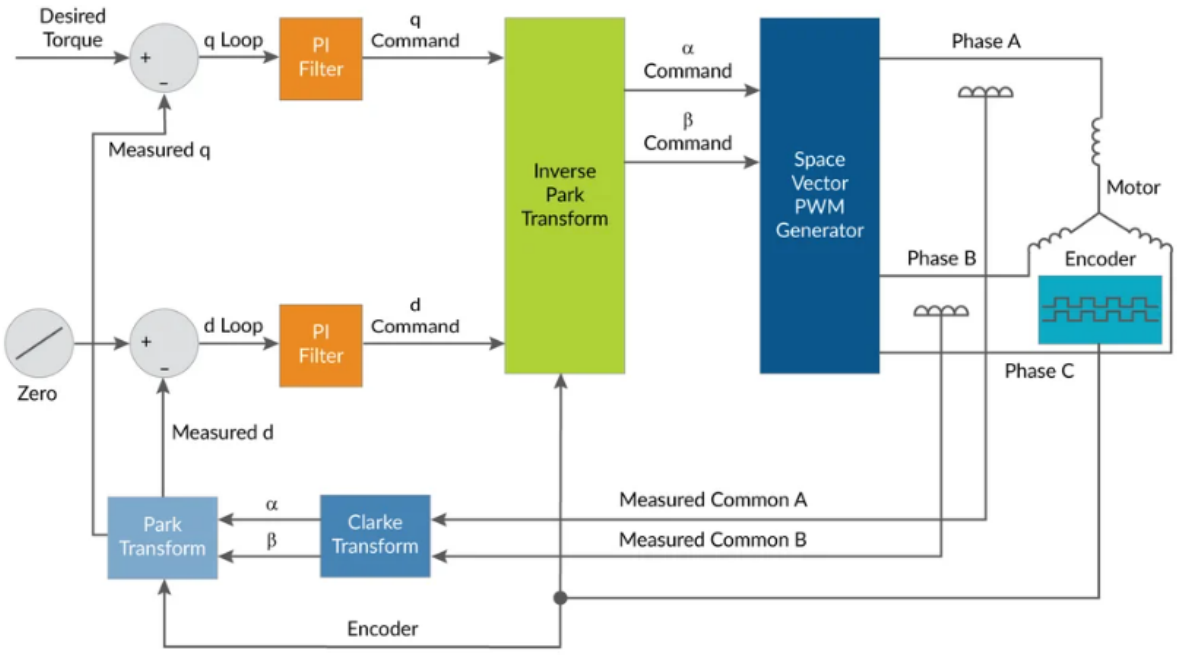


Fig 2.26 Field Oriented Control diagram

The control diagram for FOC (Fig 2.18), differs in that the current loop occurs de-referenced from the motor's rotation. That is, independent of the motor's rotation. In the FOC approach, there are two current loops, one for the Q torque and another for the D torque. The Q torque loop is driven with the user's desired torque from the servo controller. The D loop is driven with an input command of zero, so as to minimize the unwanted direct torque component.

The trick to making all of this work is math-intensive transform operations that convert the vectorized phase angle to, and from, the de-referenced D and Q reference frame. Known as Park and Clarke transforms, their practical implementation in Brushless DC drives are now commonplace due to the availability of low-cost, high-performance DSPs and microprocessors.

Motor controllers which adopt an FOC approach can drive the motor more efficiently, as high as 97 % in certain applications. This advantage is particularly pronounced at higher speeds.

2.6.2 PID Control:

Proportional-Integral-Derivative (PID) control is one of the most widely used feedback control techniques in engineering systems. It works by continuously calculating an error value as the difference between a desired setpoint and a measured process variable, and then applying a correction based on proportional, integral, and derivative terms. The proportional term addresses the present error, the integral term accounts for past errors to eliminate steady-state offset, and the derivative term predicts future error trends to improve stability. Due to its simplicity, effectiveness, and ease of implementation, PID control is commonly applied in robotics, automation, and motor control systems.

PID control combines three distinct control actions—Proportional (P), Integral (I), and Derivative (D)—to produce a control signal that adjusts a system's output toward a desired setpoint. Each component contributes uniquely to the control strategy:

1. **Proportional Control (P)**
   The proportional term produces an output that is directly proportional to the current error value. It provides immediate correction based on the magnitude of the error.

$$P_{out} = K_p \cdot e(t)$$

where:
P_{out} = proportional output
K_p = proportional gain
$e(t)$ = error at time t, defined as $e(t)=r(t)-y(t)$ (setpoint minus actual output)

2. **Integral Control (I)**
   The integral term sums the error over time and helps eliminate steady-state error. It compensates for past values of the error that were not corrected by the proportional term.

$$I_{out} = K_{i \int_0^t e(\tau)d\tau}$$

where:
Ki = integral gain.
The integral term increases the controller output as long as a persistent error exists.

3. **Derivative Control (D)**
   The derivative term predicts future error by calculating the rate of change of the error. It helps improve the system's stability and response time by damping oscillations.

$$D_{out} = K_d \cdot \frac{de(t)}{dt}$$

where:
K_d = derivative gain

4. **Total PID Output**
   The combined control signal sent to the system is the sum of the three components:

$$u(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(\tau)d\tau + K_d \cdot \frac{de(t)}{dt}$$

where:
u(t) = total control signal to the actuator.

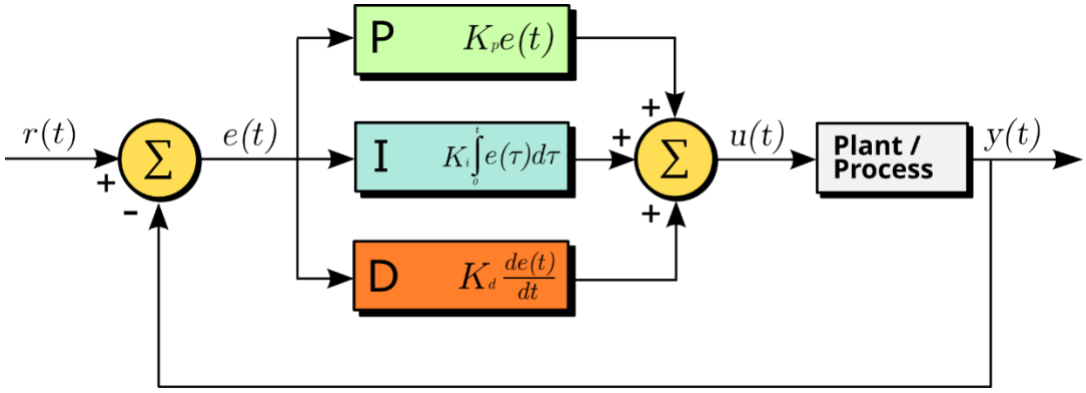


Fig 2.27 PID Controller Diagram

2.7 Kinematics of the arm:

To effectively control a 7 degrees of freedom robot arm utilizing three IMUs (Inertial Measurement Units), it is essential to carefully consider the kinematics of the arm. This involves understanding how each joint and link interacts to produce the desired movements and

orientations. The kinematic models will help in calculating the positions and orientations based on the data received from the IMUs, which provide critical information about the angles and motions of the arm. By integrating this kinematic analysis with the IMU data, we can achieve precise control and coordination of the robot arm's movements, ensuring that it functions accurately and efficiently within its operational environment.

- The concept is that each IMU will provide us with the rotation matrix for 3 degrees of freedom, derived from the Euler matrix (Fig 2.28).

$$
\begin{aligned}
R &= R_z(\psi)R_y(\theta)R_x(\phi) \\
&= \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \\
&= \begin{bmatrix} \cos\psi\cos\theta & \sin\phi\sin\theta\cos\psi - \sin\psi\cos\phi & \sin\theta\cos\phi\cos\psi + \sin\phi\sin\psi \\ \sin\psi\cos\theta & \sin\phi\sin\psi\sin\theta + \cos\phi\cos\psi & \sin\psi\sin\theta\cos\phi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}
\end{aligned}
$$

Fig 2.28 Euler Rotation Matrix

- The DH table of the arm will give us a rotation matrix, with the joint angles represented as unknown variables.
- The Euler matrix will be equivalent to a 3 degrees of freedom rotation matrix.
- By implementing inverse kinematics, to the two equivalent matrixes (Euler matrix, and the full rotation matrix), we could find the angle that the robotic arm should moves in in order to mimic the human arm.

Several important factors need to be considered:

1. Kinematic Model Accuracy: Ensure that the kinematic model accurately represents the physical structure of the robotic arm, including lengths and joint configurations.

2. Joint Restrictions: Consider the limits on the angles of each joint to prevent the arm from reaching impossible configurations.

3. Singularities: Identify and handle potential singularities in the kinematic chain, where the arm could lose a degree of freedom or become unresponsive.

4. Sensor Calibration: Ensure that the inertia measurement units (IMUs) are properly calibrated for accurate readings of the rotation matrices.

5. Reference Frames: Establish a consistent reference frame for both the robotic arm and the human arm to ensure correct mapping of movements.

6. Data Fusion: If multiple IMUs are used, implement data fusion techniques to merge their outputs for a more accurate representation of the arm's orientation.

7. Computational Load: Consider the computational resources required for real-time processing of inverse kinematics calculations.

8. Control Algorithms: Develop robust control algorithms that can effectively translate the calculated angles into smooth movements for the robotic arm.

9. Feedback Mechanisms: Incorporate feedback systems to improve the accuracy of the arm's movements based on real-time performance.

10. Environment Interaction: Consider how the arm will interact with its environment and any external forces that may affect its movement.

2.8 Additive production strategies:

Most of the components were produced using a resin 3D printer (Fig 2.20). Resin 3D printers are a type of additive manufacturing technology that utilize photosensitive resin to create objects layer by layer. Instead of extruding plastic filament like FDM (Fused Deposition Modeling) printers, resin printers use a light source—typically an LED or a laser—to cure the resin in selective areas, solidifying it into the desired shape.

Key features of resin 3D printers include:

- High Detail: They are capable of producing highly detailed and intricate models with smooth surfaces, making them ideal for applications such as jewelry, dental models, and miniatures.

- Speed: Although print times can vary based on the complexity of the model, many resin printers are faster than traditional FDM printers due to the quick curing process.

- Variety of Resins: There are various types of resins available, including flexible, rigid, and special formulations that can simulate the properties of different materials.

- Post-Processing (Curing Prosses): After printing, objects typically require cleaning in isopropyl alcohol and curing under UV light to ensure strength and stability.

Fig 2.29 Elegoo Resin 3D Printer

Ther Materials That Were Used:

- ABS like resin: ABS-like resin materials are specially formulated photopolymer resins designed to mimic the properties of ABS (Acrylonitrile Butadiene Styrene) plastic.

Key Characteristics of ABS-like resins:

Table 2.3 Characteristics of Elegoo ABS-like Resin and FDMs' ABS [6] :

Performances	ABS-Like Resin Resin 3.0+	ABS-Like Resin Resin 3.0 Pro	FDMs' ABS (depending on the specific formulation and print conditions)
Max force	12.53 ± 10% MPa	16.46 ± 10% MPa	30-50 MPa
Viscosity (25°C)	90-170 MPa.s	175-275 MPa.s	200-400 mPa.s
Tensile strength	30.13 ± 10% MPA	39.52 ±10% MPA	20-40 MPa
Impact strength	50.48 ±10% j/m	27.96 ±10% j/m	15-30 kJ/m²
Elongation at break	25.00 ± 10%	22.19 ±10%	4-10%

The specs of the ABS-like Resin can vary depending on the curing prosses.

Resin 3.0 Pro was used to print most of components.

2.9 Selection of Gearbox:

2.9.1 Type of gears:

The A2212 BLDC motor operates at a high rotational speed, achieving up to 12,000 revolutions per minute; however, it exhibits limited torque characteristics. To enhance torque output, it is necessary to reduce the motor's speed, which can be accomplished through the implementation of a gearbox system. The following candidates are being considered for this purpose:

1. Harmonic gearboxes

Gear Ratio Equation:

$$GearRatio = \frac{N_{cs} - N_{fs}}{N_{fs}}$$

Where:

- N_{cs} : Number of teeth on Circular Spline
- N_{fs}: Number of teeth on Flexspline

Speed and Torque:

- **Output Speed:**

$$\omega_{out} = \omega_{in} \cdot \frac{N_{fs}}{N_{cs} - N_{fs}}$$

- **Output Torque:**

$$T_{out} = T_{in} \cdot \frac{N_{cs} - N_{fs}}{N_{fs}}$$

Notes: Typical gear ratios range from **30:1 to 160:1** in a compact, lightweight format.

2. Cycloidal gearboxes

Gear Ratio Equation (Single Stage):

$$Gear\ Ratio\ = \frac{N_p - N_d}{N_d}$$

Where:

- N_p: Number of pin rollers
- N_d: Number of lobes on the cycloidal disc

Speed and Torque:

- **Output Speed:**

  $\omega_{out} = \omega_{in} \cdot \frac{N_d}{N_p - N_d}$

- **Output Torque:**

$$T_{out} = T_{in} \cdot \frac{N_p - N_d}{N_d}$$

Notes: Ratios typically range from **30:1 to 300:1**. Highly efficient and robust under shock loads.

3. Planetary gearboxes

Configuration A: Sun Input, Carrier Output, Ring Fixed (Reduction Mode)

- **Gear Ratio:**

  $Gear\ Ratio\ =\ 1\ +\ \frac{N_r}{N_s}$

- **Speed:**

$$\omega_{carrier} = \frac{\omega_{sun}}{1 + \frac{N_r}{N_s}}$$

- **Torque:**

$$T_{carrier} = T_{sun} \cdot \left(1 + \frac{Nr}{Ns}\right)$$

Configuration B: Ring Input, Carrier Output, Sun Fixed

- **Gear Ratio:**

$$Gear\ Ratio\ =\ 1 + \frac{N_s}{N_r}$$

Configuration C: Carrier Input, Sun Output, Ring Fixed (Overdrive)

- **Gear Ratio:**

$$Gear\ Ratio\ =\ \frac{N_r}{N_r + N_s}\ <\ 1$$

Notes: Planetary gearboxes are valued for **high torque density** and **coaxial alignment**.

4. Split ring compound planet epicyclic gearboxes [7]:

Gear Ratio Equation:

$$Gear\ Ratio\ =\ 1 + \frac{N_{r1} \cdot N_{s2}}{N_{r2} \cdot N_{s21}}$$

Where:

- N_{r1}, N_{r2}: Teeth on ring gears
- N_{s1}, N_{s2}: Teeth on sun gears

Speed and Torque:

- **Speed:**

ωout =f(ωin ,gear configuration)

- **Torque:**

Tout =f(Tin ,gear configuration)

Notes: This design allows combining harmonic-like ratios with planetary strength and precision.

- **Harmonic gearbox:**

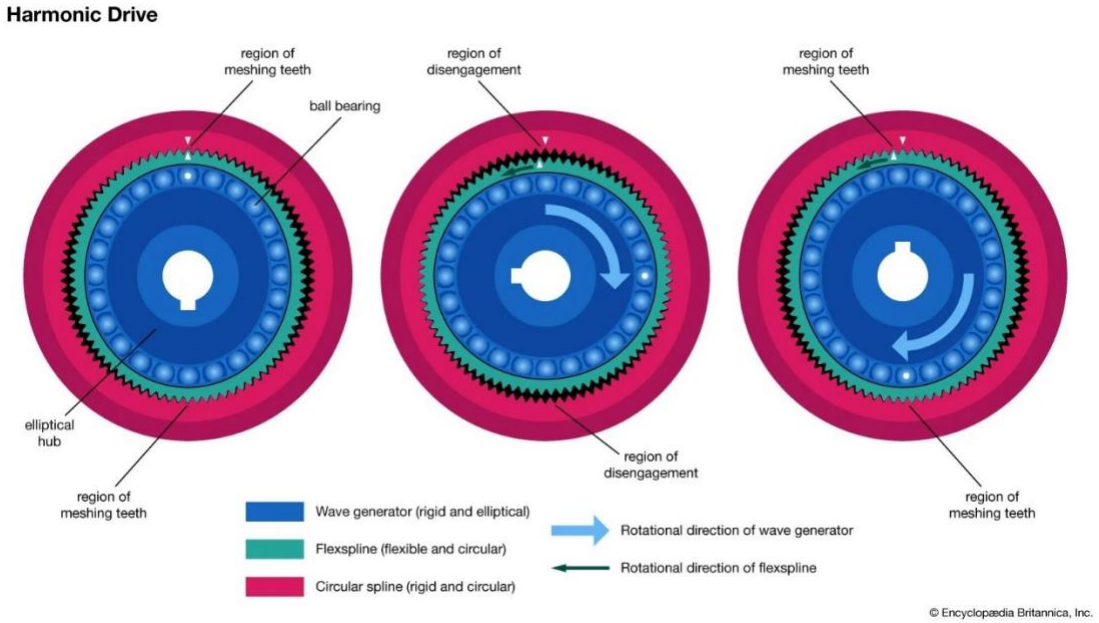


Fig 2.30 Harmonic gearbox

It is an advanced type of transmission system, characterized by its support for precise high-torque applications in a small space. This system relies on the principle of flexible movement, which changes the shape of a flexible ring, allowing for a significant reduction ratio with a small size.

Advantages of Harmonic Gear:

- High reduction ratio: The harmonic gear can achieve reduction ratios ranging from 30:1 to 100:1 or more.

- High precision: It is used in applications that require precision in movement, such as robotics and precision machinery.

- Compact design: Designed to save space, making it attractive for mechanical applications that require lightweight solutions.

- Long lifespan: It reduces wear due to the minimal friction between moving components.

Working principle: The harmonic gear typically consists of several main components: - Flexible Spline: This forms the main flexible element.

- Circular Spline: This is fixed and does not move.

- Wave Generator: It is used to generate flexible movement through mechanical effects.

This type has not been utilized mainly due to the unavailability of a flexible material for the flexible spline.

- **Cycloidal gearbox:**

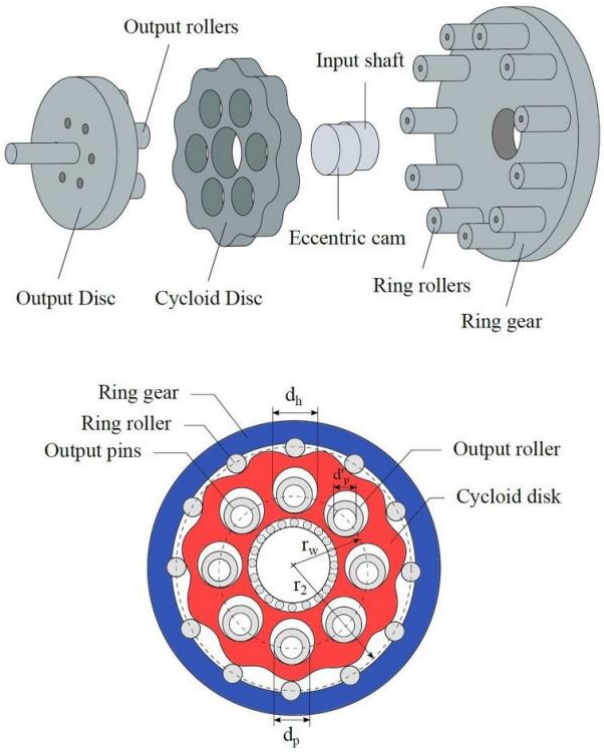


Fig 2.31 Cycloidal gearbox

The cycloidal gearbox is distinguished by its unique kinematic design, operating on the principle of rotary motion.

A tilted cam moves the input shaft, leading to the rotation of a cycloidal disc within the output housing. This distinctive mechanism allows for a compact design while achieving high reduction ratios in the gears, typically ranging from 10:1 to 100:1 or more.

The rotary motion is translated into smooth and efficient torque transmission, making these gearboxes particularly suitable for applications that require high torque and a high degree of precision, such as robotics, transportation systems, and industrial machinery. Moreover, the inherent design of cycloidal gearboxes reduces backlash and enhances durability, leading to lower maintenance requirements and increased operational lifespan.

As a result, cycloidal gearboxes are increasingly adopted across various sectors, reflecting their versatility and efficiency in power transmission applications.

This type of gearbox has not been used due to the complexity and difficulty of the design, as well as the presence of numerous moving parts.

- **Planetary gearbox:**

  The planetary gearbox, also referred to as a ring gearbox, is a highly adaptable and efficient variable.

  It consists of a central sun gear, multiple planetary gears, and an outer ring gear. As the planetary gears rotate around the sun gear and revolve around the central axis, the term "planetary" is justified. This gear arrangement allows for high torque transmission within a compact framework. The applications of planetary gearboxes are extensive, including automotive and robotics applications.

Fig 2.32 Planetary gearbox

- **Split ring compound planet epicyclic gearbox:**

  Split ring compound planet epicyclic gearboxes are advanced mechanical systems that utilize a unique design to enhance torque output and efficiency. These gearboxes feature a split ring mechanism, which allows for multiple stages of power transmission and increased load capacity.

  Split ring compound planet epicyclic gearboxes represent a sophisticated integration of harmonic and planetary gearing technologies, resulting in a highly efficient power transmission system. The unique split ring design facilitates a compact structure while delivering an impressive high transfer ratio, which enhances torque output without requiring excessive space. This innovative combination allows the gearbox to achieve high load capacities and operational efficiency, making it ideal for this project.

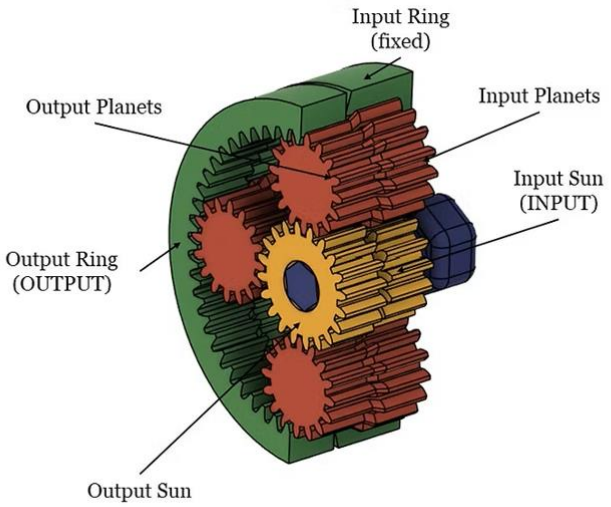


Fig 2.33 Split ring compound planet epicyclic gearbox

2.9.2 Types of gears:

There are three primary types of gear designs commonly used in mechanical systems: spur gears, helical gears, and bone gears.

Spur gears are the simplest type, featuring straight teeth that are parallel to the gear's axis, making them ideal for transmitting motion and power between parallel shafts.

Helical gears have angled teeth that create a smoother and quieter operation compared to spur gears, and they are capable of handling higher loads due to the gradual engagement of the teeth.

Bone gears, on the other hand, are a less common type featuring curved, bone-shaped teeth that can provide unique meshing characteristics, often used in specialized machinery where specific contact patterns are beneficial. Each of these gear types offers different advantages depending on the specific requirements of the mechanical application.

Table 2.4 Comparision between spur, helix, and bone gears:

Feature	Spur Gears	Helical Gears	Bevel Gears
Gear Teeth Orientation	Parallel to the axis of rotation	Angled at a helix angle to the axis of rotation	Teeth are conically shaped and intersect at an angle
Efficiency	High efficiency in parallel axis applications	High efficiency, quieter than spur gears	Can be less efficient depending on the angle
Noise & Vibration	Noisy, generates vibrations	Quieter operation, less vibration	Moderate noise due to angular contact
Load Distribution	Uneven load distribution (point contact)	More even load distribution (line contact)	Load distributed along angled teeth
Torque Capacity	Lower torque compared to helical gears	Higher torque capability due to better contact	Moderate torque capacity depending on the application
Applications	Simple, low-speed applications	High-speed, high-torque, and precision applications	Angular motion transmission (e.g., differential, right angle)
Complexity	Simple to design and manufacture	More complex design and manufacturing	Moderate complexity, requires precise angles
Cost	Lower cost	Higher cost due to manufacturing complexity	Moderate cost depending on design and material
Efficiency in Power Transmission	Generally lower compared to helical gears	Generally higher, smoother meshing	Can lose efficiency at high angles or speeds
Common Uses	Lifts, conveyors, low-speed applications	Automotive, conveyors, elevators, pumps	Differential drives, right-angle gearboxes
Size	Compact, simpler sizes	Larger, more robust sizes for heavy-duty use	Compact but often customized for specific angles

Chapter 3

The designing of the project

The objective of this chapter is to refine the project's design, as successful execution necessitated a multifaceted approach that involved a thorough analysis of several critical factors. This encompassed an extensive review of strain measurements, maximum torque capacities, model design parameters, power consumption profiles, and gear modeling dynamics. Developing a comprehensive understanding of these elements was essential for creating a robust, efficient, and reliable prototype that aligns with the project's objectives.

3.1 Project Hypotheses:

3.1.1 Primary Hypothesis (System-Level):

The integration of a wearable IMU-based motion capture suit with a robotic arm and neck system will enable accurate and real-time replication of human arm and neck movements, with minimal latency and acceptable error margins.

3.1.2 Mechanical Design Weight Hypotheses:

The robotic arm is required to achieve a total weight of approximately 1.276 kilograms, in order to maintain a balance between structural integrity, functional performance, and wearability. This weight constraint is critical to ensure that the arm remains lightweight enough for practical applications—particularly in wearable assistive technologies—while still accommodating the necessary actuators, sensors, and mechanical components for accurate and responsive movement replication.

3.1.3 Material Optimization Hypothesis:

In order to reduce production costs and enhance the practicality of the robotic arm, the project will focus on minimizing the use of 3D-printed materials while utilizing standardized, commercially available materials for structural components. It is hypothesized that this approach will strike an effective balance between material efficiency, cost-effectiveness, and the mechanical properties necessary for the arm's durability and functionality. By leveraging standard materials for key structural parts, it is expected that the overall weight of the system can be reduced without compromising its strength, stability, or performance.

3.1.4 Mechanical Performance Hypothesis:

The 3D-printed robotic arm, actuated by a combination of BLDC, stepper, and servo motors, will reproduce natural human arm and wrist movements with a functional range of motion comparable to that of the human limb.

3.1.5 User Comfort Hypothesis:

The wearable motion capture system, including IMUs and potentiometers, will be lightweight and comfortable enough to be worn continuously for at least 30 minutes without significant discomfort, as rated by test users.

3.2 Mechanical design:

3.2.1 Force analysis:

The torque acting on each individual component of the system will be determined based on the formulation provided in Equation 3.1:

$$T = F \times d \quad \text{(Eq3.1)}$$

F= force (Newton) d=distance(meter)

In order to compute the forces on each component, it is necessary to first determine the weight of each individual part.

The following are the weights of the individual components:

- The Servo 45 kg motor has a mass of 80 g.
- The MG996R servo motor weighs 60 g.
- The long carbon fiber rod (16 cm) has a mass of 14 g.
- The two short carbon fiber rods (12.5 cm) collectively weigh 22 g.
- The 24 cm T-slot aluminum bar weighs 100 g.
- The printed parts in the elbow assembly have a total mass of 100 g.
- The printed components of the shoulder link weigh 50 g.
- The printed parts of the hand assembly have a mass of 150 g and the printed differential the wrist is 120g.

The gravity will be considered 10 m/s^2

$$T_{\times 2 short\ carbon} = 0.08_m \times 0.22_N = 0.0176\ N.m$$

$$T_{45kg\&shoulder\ link} = 0.12_m \times 1.3_N = 0.156\ N.m$$

$$T_{long\ carbon} = 0.2_m \times 0.14_N = 0.028\ N.m$$

$$T_{servo\ 45kg} = 0.14_m \times 0.8_N = 0.112\ N.m$$

$$T_{MG996R} = 0.28_m \times 0.6_N = 0.168\ N.m$$

$$T_{Elbow} = 0.28_m \times 1_N = 0.28\ N.m$$
$$T_{2\times MG996R} = 0.35_m \times 1.2_N = 0.42\ N.m$$
$$T_{6\times MG996R} = 0.35_m \times 3.6_N = 1.26\ N.m$$
$$T_{differential} = 0.55_m \times 1.2_N = 0.66\ N.m$$
$$T_{hand} = 0.65_m \times 1.5_N = 0.975\ N.m$$

From these torques we find the total torque of the body:

$$T_{Body} = \Sigma T = 4.235\ N.m$$

The load that the arm will carry is 1kg.

$$T_{load} = \Sigma T + (0.65_m \times 10_N) = 10.735\ N.m$$

3.2.2 Cad Designs:

In this section, multiple approaches for designing each part of the project will be discussed.

3.2.2.1Finger motion capture Design:

The capturing mechanism is based on the four-bar link mechanism it is designed to be mounted directly to a joystick to capture the finger movement the design consist of four part the base is the joystick the fourth link is the last link is the finger itself:

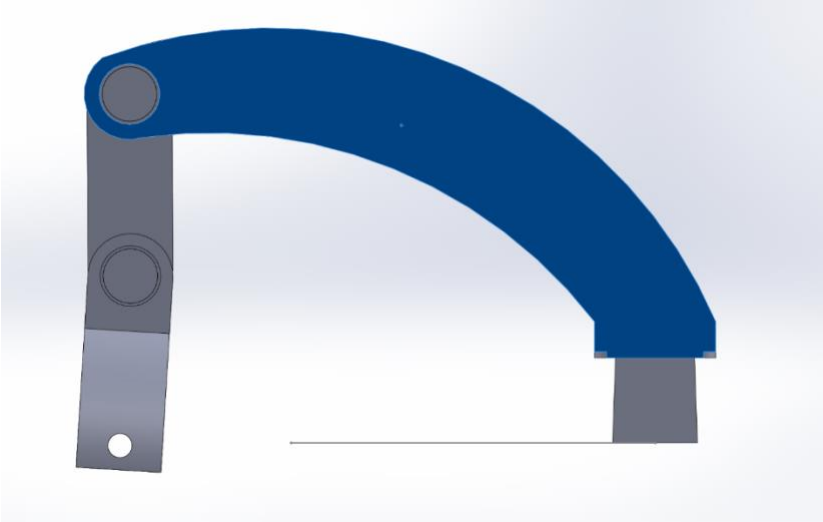


Fig3.1 Finger motion capturing design

This mechanism will be integrated with potentiometers strategically positioned on the hand according to the proposed design. The Inertial Measurement Unit (IMU) will be mounted at the base of the structure to provide stable reference data for hand orientation and movement.

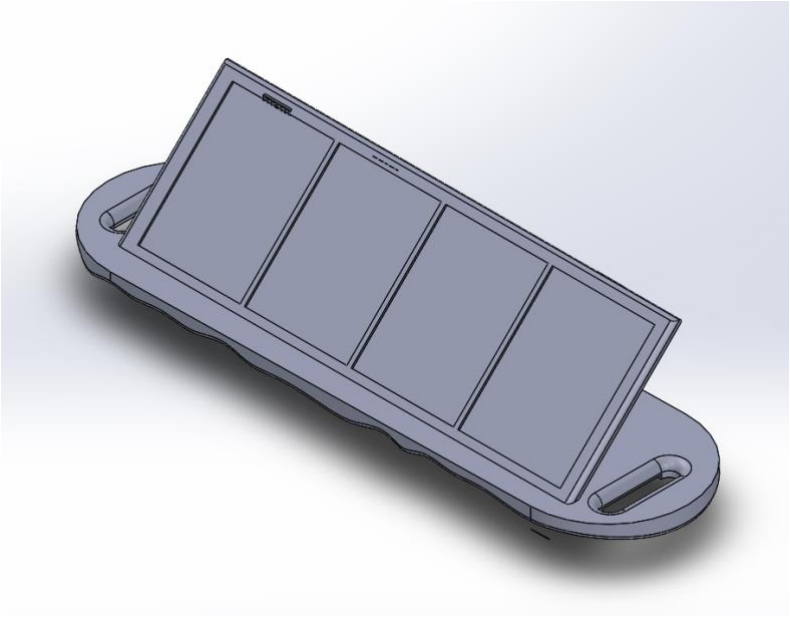


Fig 3.2 The Base That the POTs are position at

3.2.2.2 Arm motion capture Design:

A dedicated casing will be designed for each Inertial Measurement Unit (IMU) -except the hand- to ensure proper protection, secure mounting, and optimal performance. The design process will take into consideration factors such as durability, material selection, environmental conditions, and ease of integration with other components of the project. This approach aims to enhance the reliability and longevity of the IMUs within the overall system.

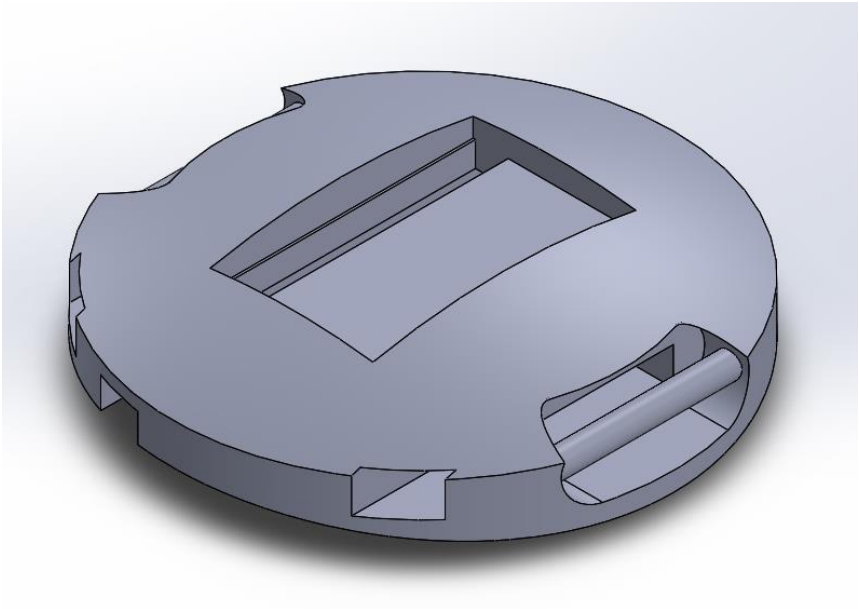


Fig 3.3 The Base that the IMU will be Mounted on

3.2.2.3 Shoulder Design:

There are two primary designs for implementing the shoulder joint, both utilizing an A2212 BLDC motor. The key difference between them lies in the materials used: the first design is entirely 3D printed, which results in a heavier assembly. The second design, however, employs carbon fiber rods for the arm's structure, connected by 3D printed links, significantly reducing the overall weight, both designs provide three degrees of freedom for the shoulder.

First design:

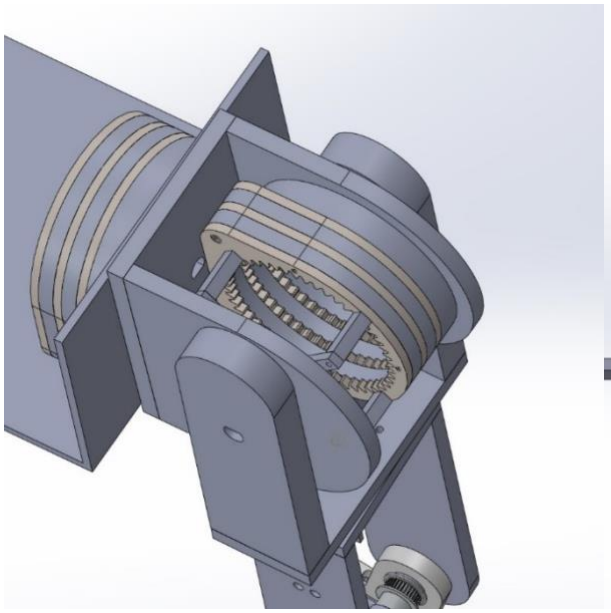


Fig3.4 Isometric view of first design shoulder

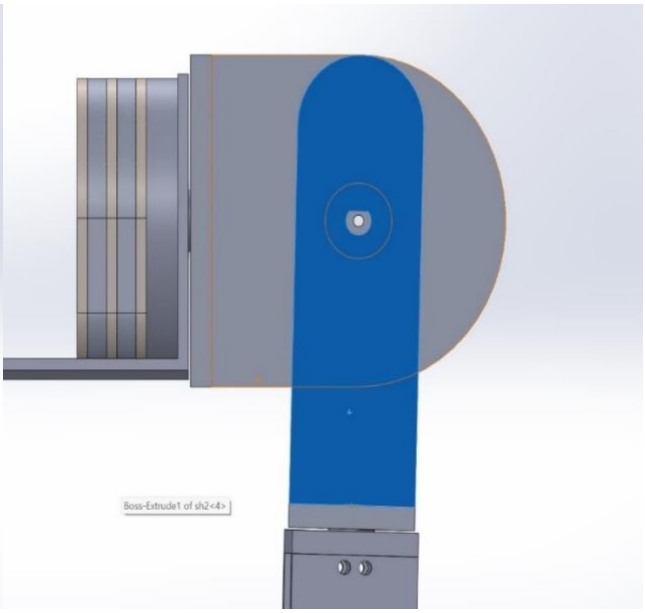


Fig3.5 Frontal view of first design shoulder

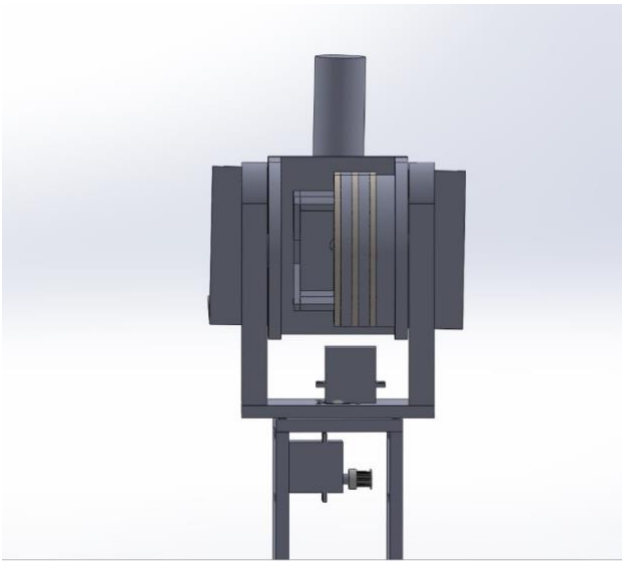


Fig3.6 Side view of shoulder

Second design:

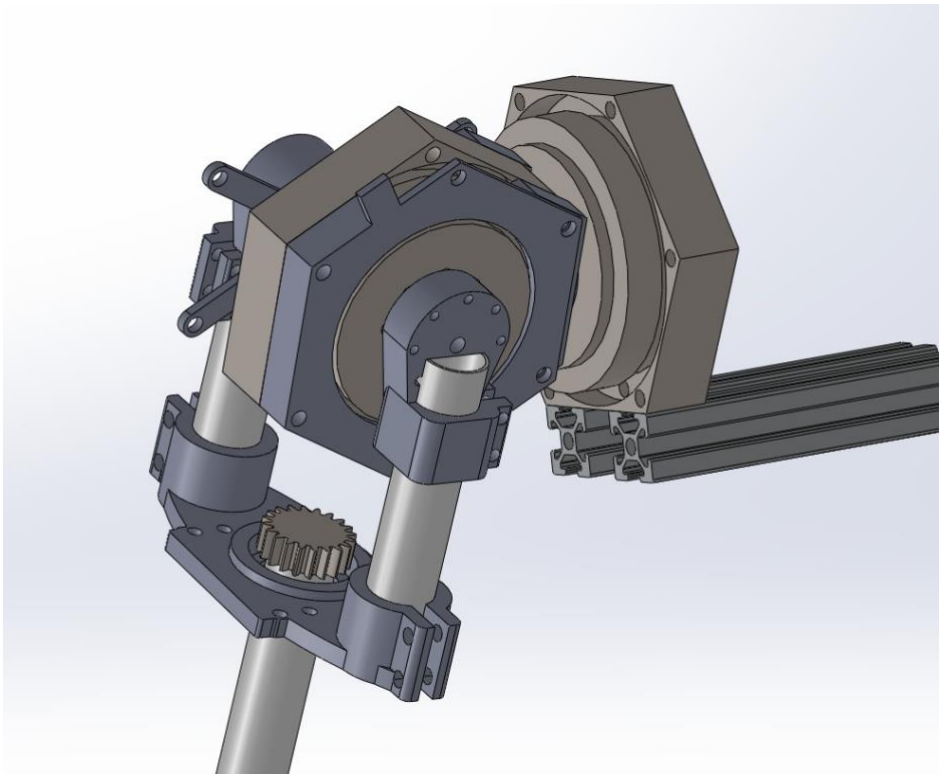


Fig 3.7 Isometric View of second design shoulder

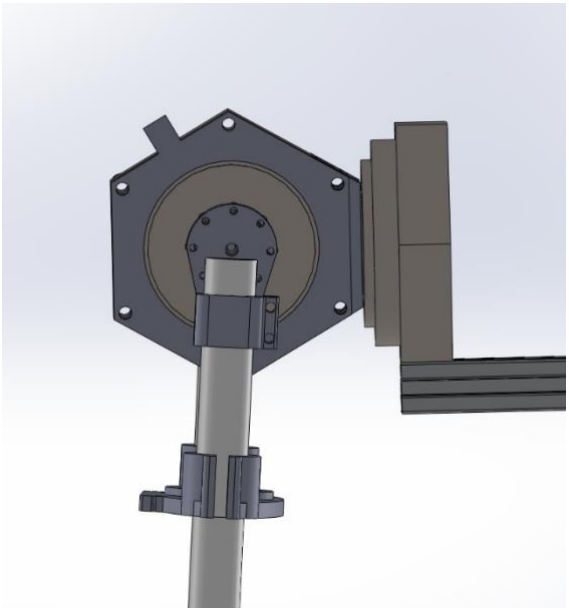


Fig 3.8 Frontal view of second design shoulder

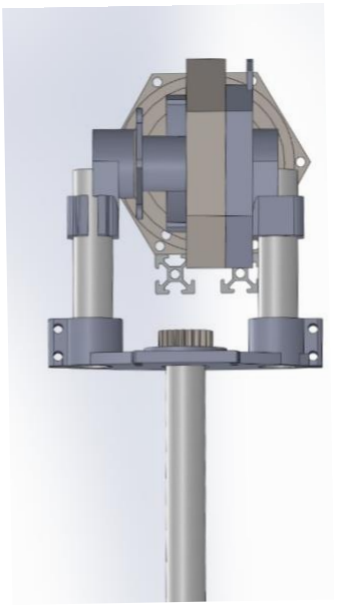


Fig 3.9 side view of second design shoulder

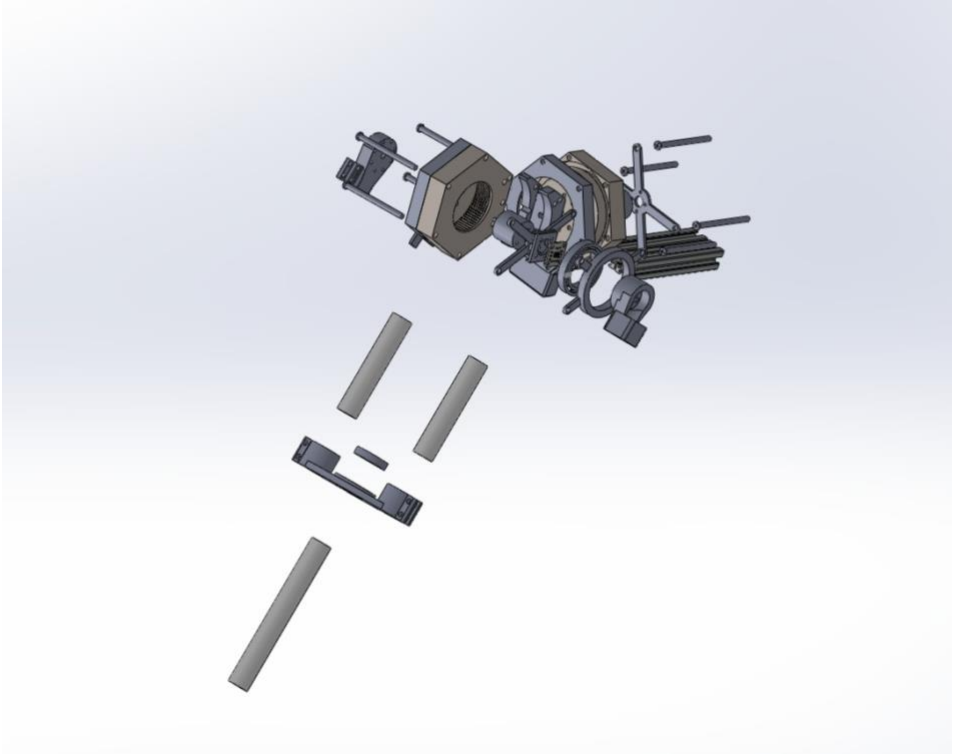


Fig 3.10 Explosion view of shoulder

3.2.2.4 Elbow Design & Forearm:

There are three proposed designs for the elbow joint, all primarily 3D printed.

The main differences between them are as follows:

- The first design places a high torque demand on the motor.

- The second design incorporates a pulley system to reduce the torque load.

- The third design differs from the first two in the construction of the forearm; while the first and second designs feature a forearm that is entirely 3D printed, the third design uses a T-slot aluminum bar for the forearm.

These variations in mechanical configuration and material choice significantly affect the structural characteristics and overall performance of each design.

First design:

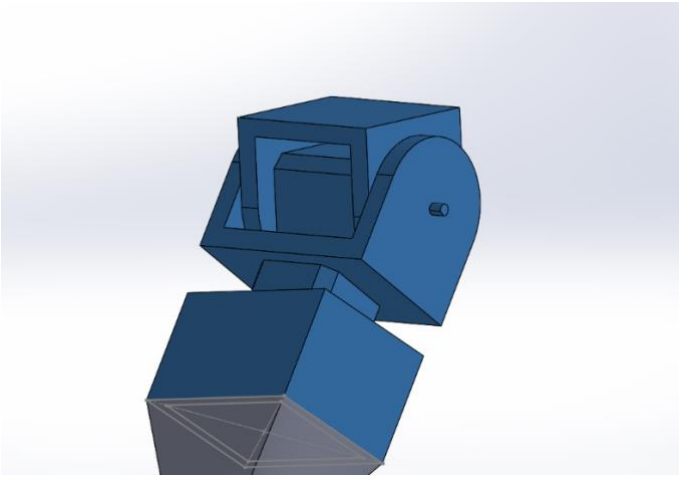


Fig 3.11 First design of Elbow

Secound design:

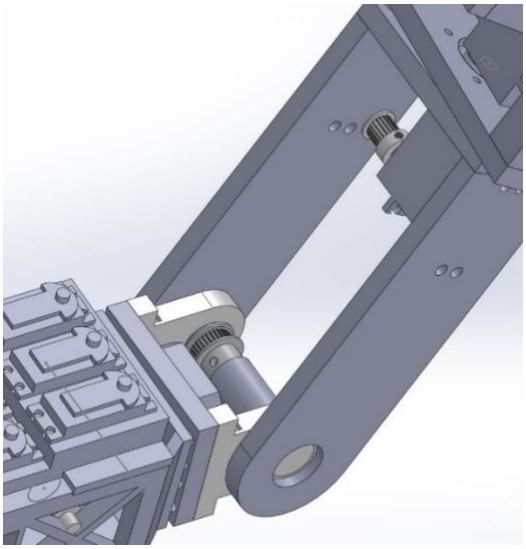


Fig 3.12 second design of elbow

Third design:

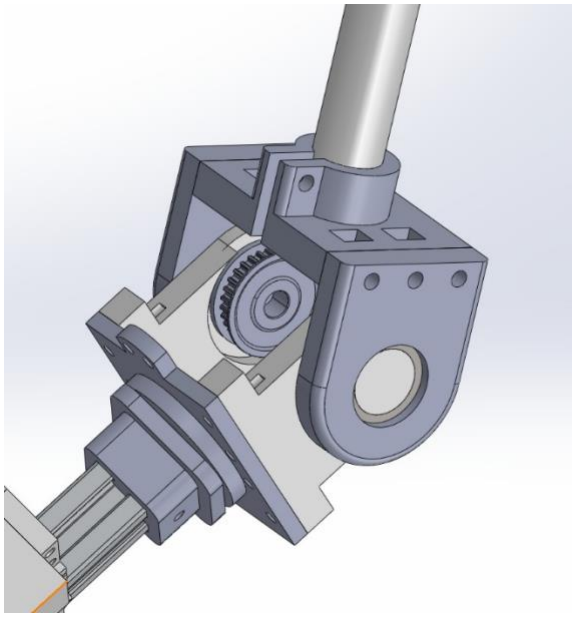


Fig 3.13 Third design of elbow

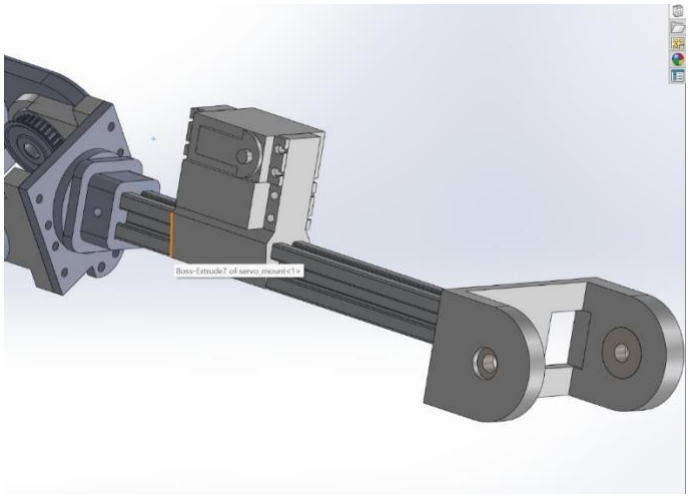


Fig 3.14 Third design of elbow

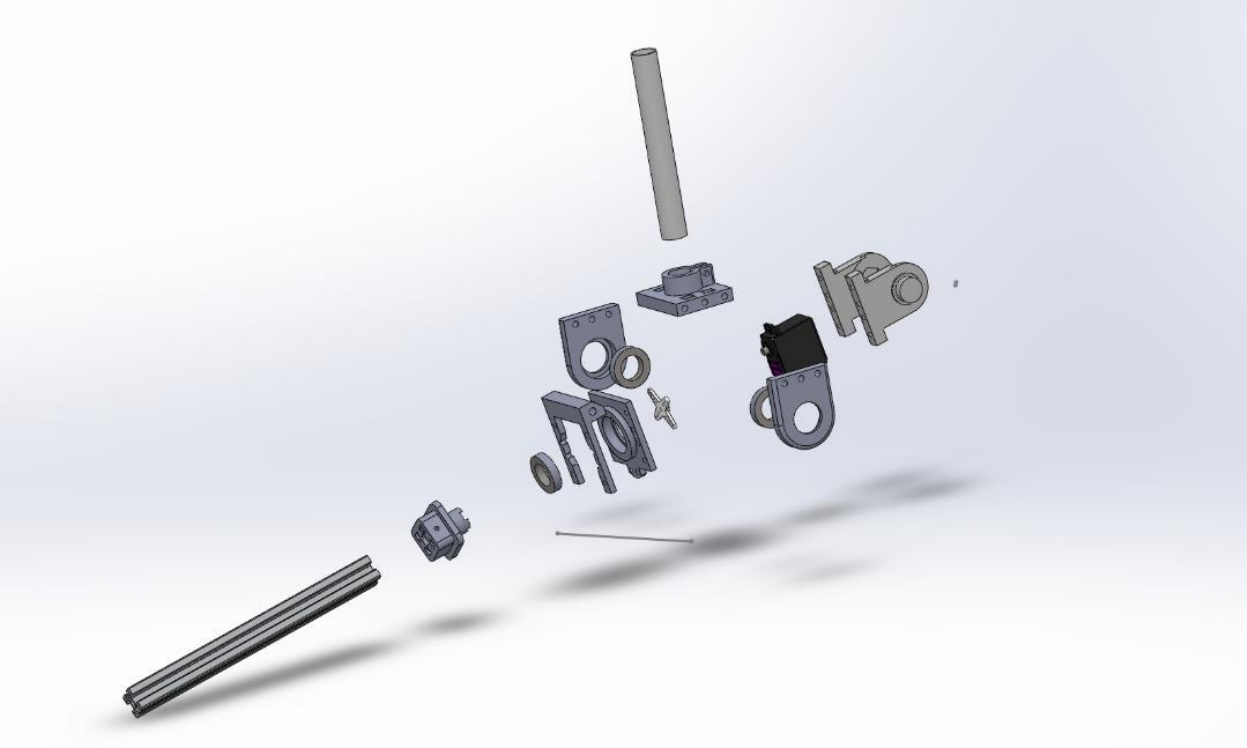


Fig 3.15 Explosion view Elbow

3.2.2.5 Wrist Design:

Several mechanisms employing indirect drive methods were designed to reduce the load on the actuator while ensuring smooth and robust movement. All designs are based on the MG996R servo motor.

The first design connects the servo motors to spherical joints, enabling joint movement along their respective axes.

However, this design was rejected due to its excessive weight and large size compared to the other alternatives.

The second design incorporates a bevel gear to rotate the wrist along the x-axis and uses spherical joints to achieve rotation along the z-axis. This design was also rejected owing to its heavier weight and larger size relative to the third design.

The third design employs a differential gearbox mechanism to provide motion along two axes. The wrist rotates around the x-axis when the motors operate in the same direction and around the z-axis when the motors move in opposite directions. The gearbox consists of three bevel gears with a module of 1.5, and pulley drives are utilized instead of direct drives to reduce the load on other actuators. This design was selected due to its robustness, compact and lightweight construction, and smooth operational performance.

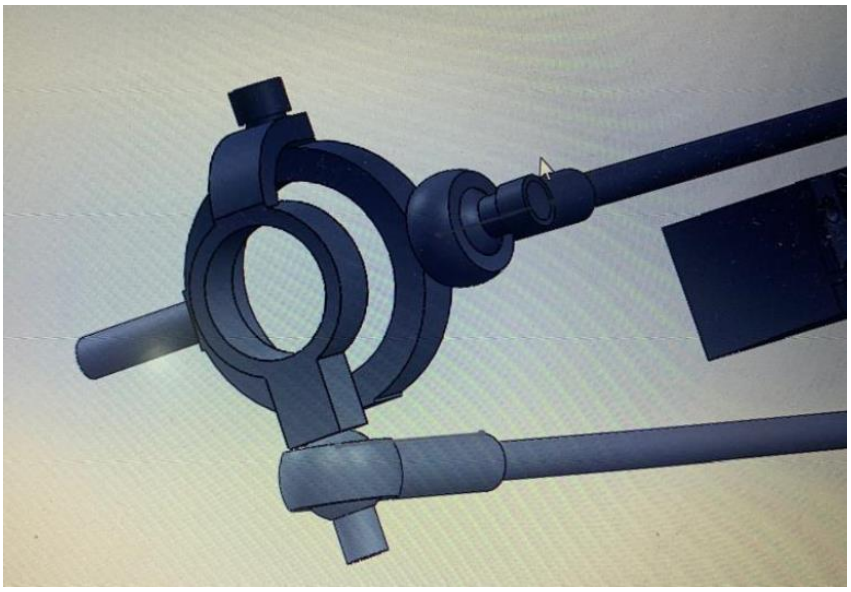


Fig 3.16 wrist first design

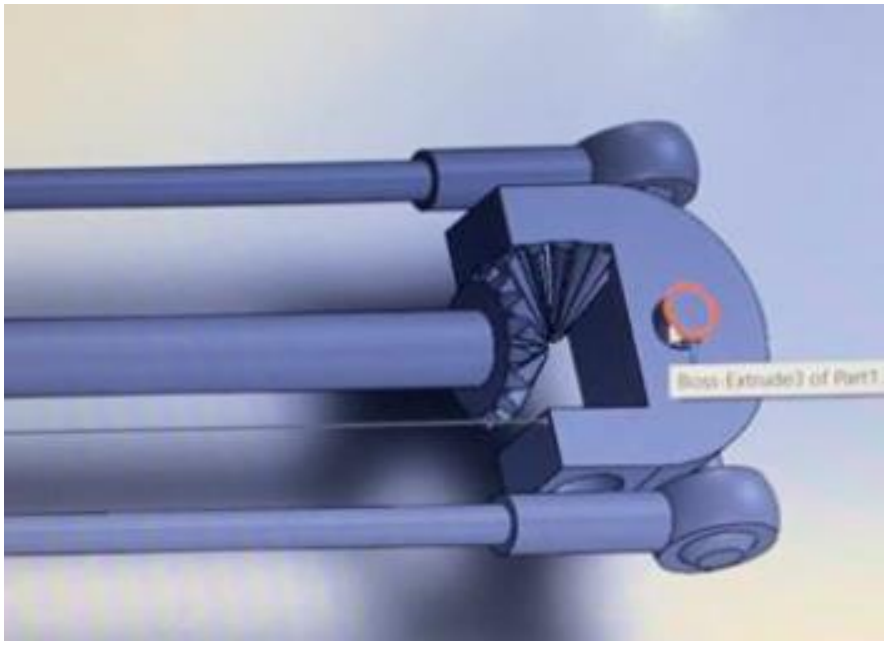


Fig 3.17 wrist second design

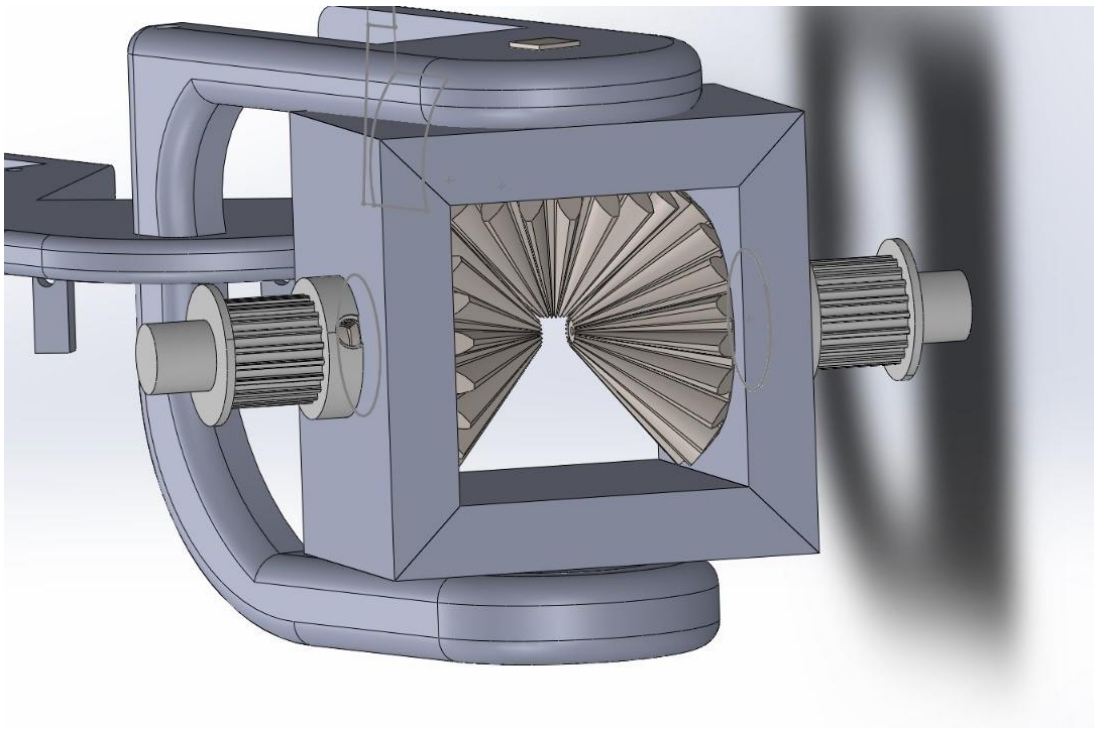


Fig3.18 Wrist third design

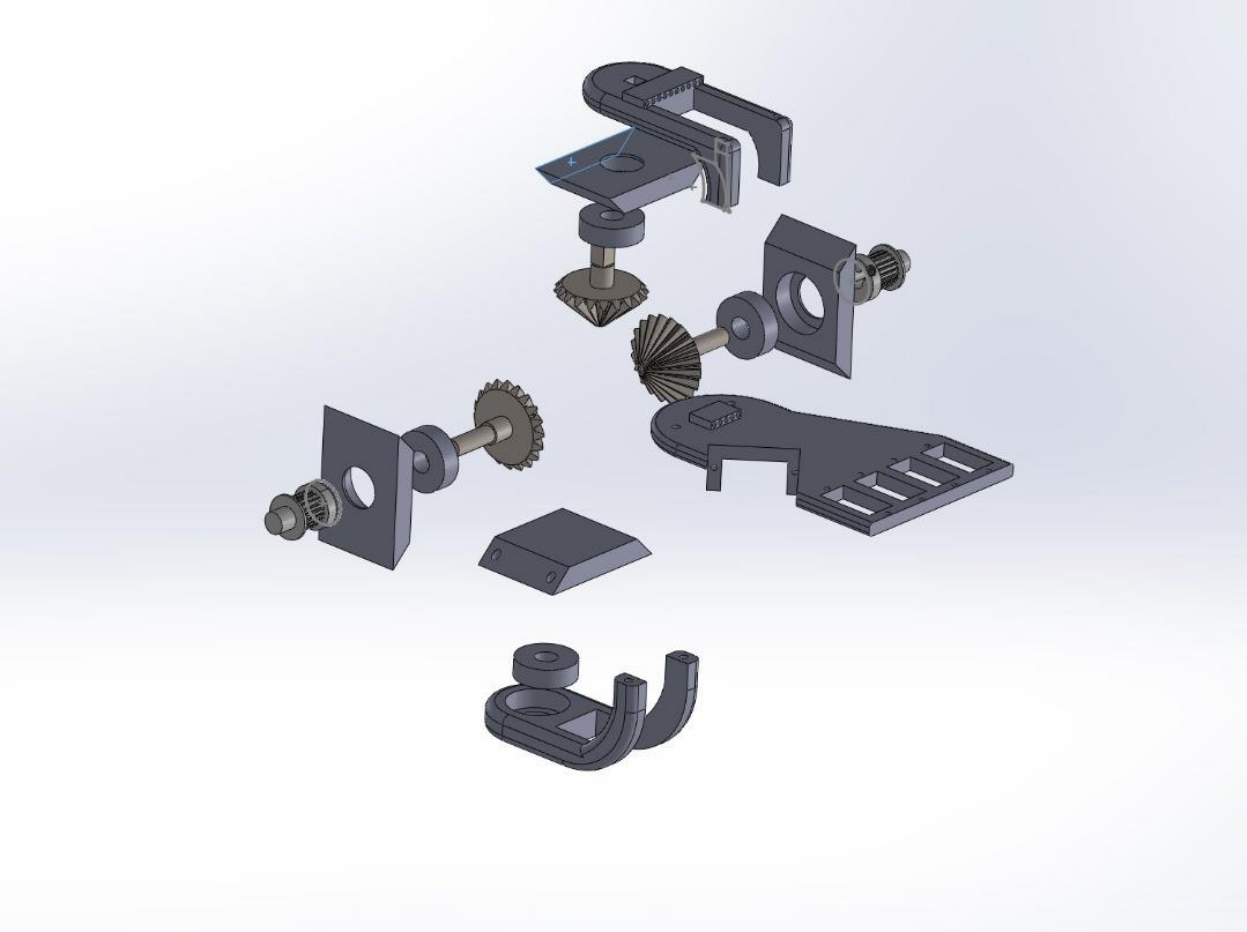


Fig 3.19 Explosion view of the wrist

3.2.2.6 Hand Design:

The hand was designed to replicate a human hand the design contains mounting holes for servos to be placed in.

The thumb base is shifted 30 degrees, to replicate a real human thumb and there are paths designed for lubricating the finger strings, so it reduces wearing and friction on the wrist joint.

3.2.2.7 Finger Design:

Multiple underactuated mechanisms were conceptualized for finger articulation, with emphasis on robustness, weight reduction, compactness, and biomimicry of human finger movement.

The first design employs a pulley mechanism to facilitate motion, with bearings used to connect components.

The second design utilizes a string mechanism to generate movement, also incorporating bearings for part connections.

The third design is based on a four-bar linkage mechanism actuated via a string connection to the actuators. This configuration demands high manufacturing precision and tight tolerances to ensure smooth operation and proper assembly.

The fourth design features rolling contact joints to link components. This specialized joint is favored in precision and compact applications due to its minimal size, lightweight nature, and tight tolerance requirements. Movement is actuated through a string mechanism, and notably, the finger is designed to be fabricated as a single piece via 3D printing. To enhance positional accuracy, an end-stroke sensor may be integrated at the fingertips.

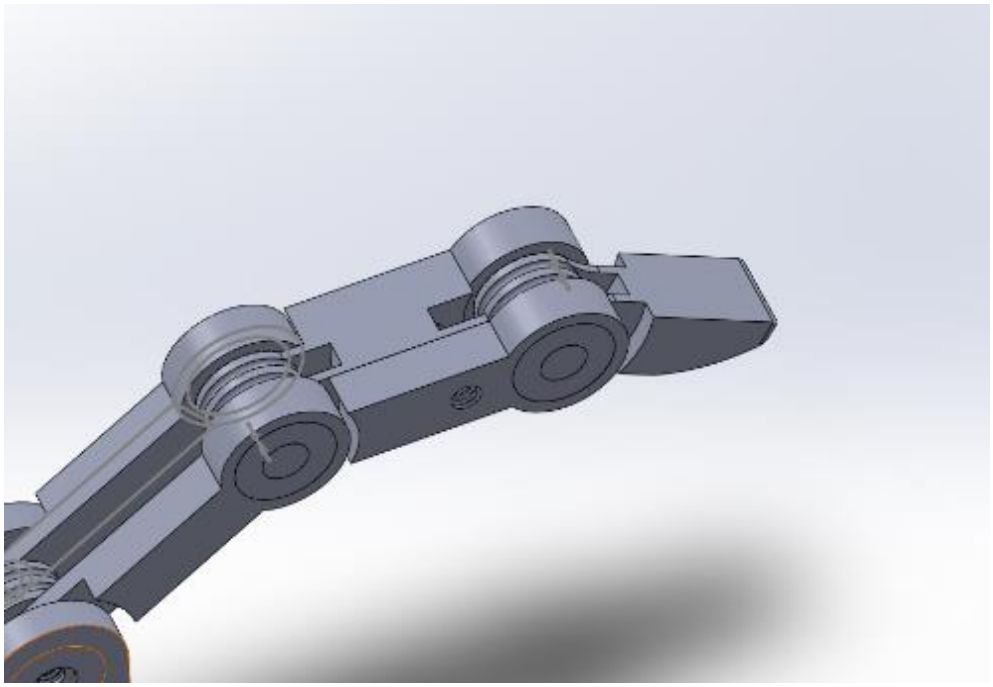


Fig 3.20 First finger design

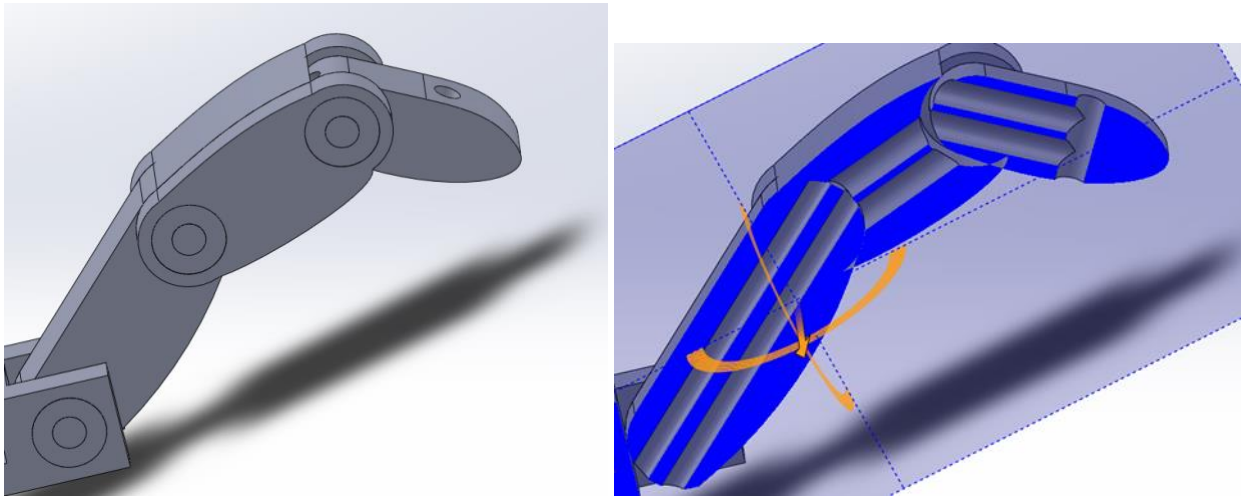


Fig 3.21 Second finger design

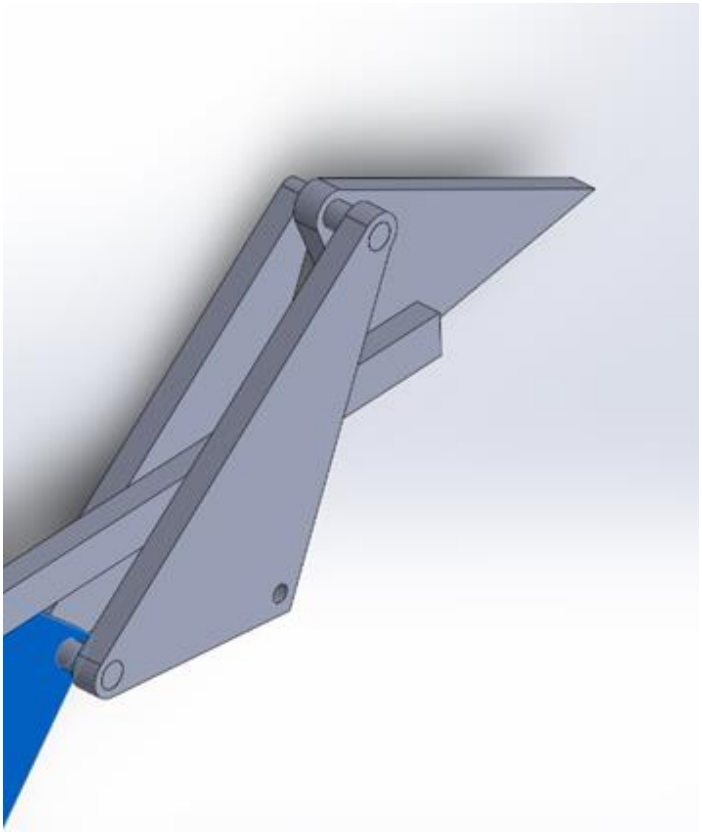


Fig 3.22 Third finger design

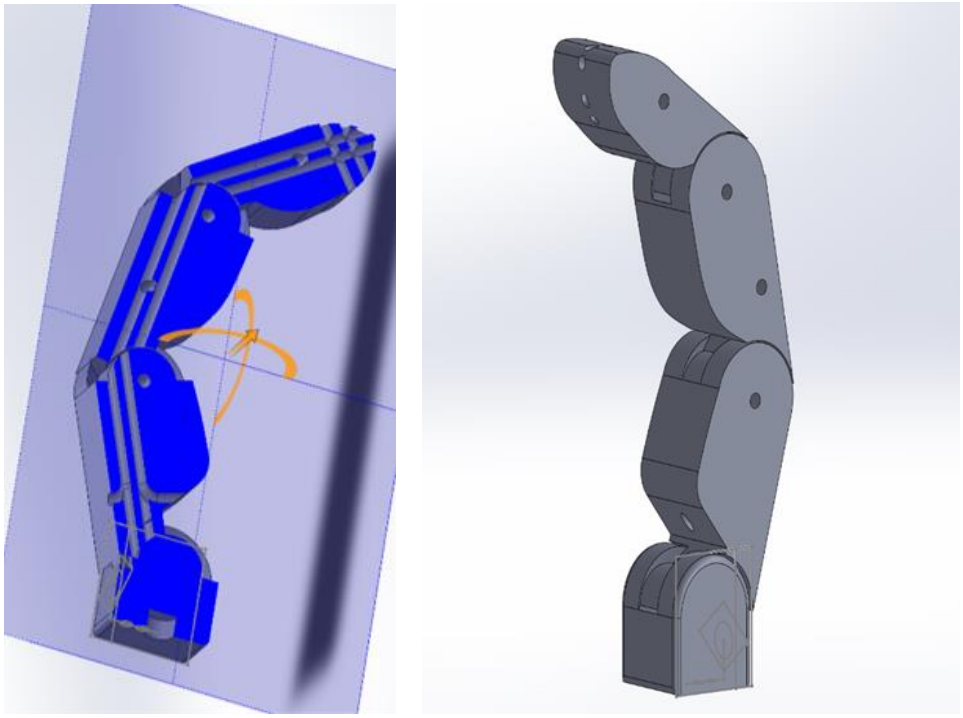


Fig 3.23 Fourth finger design

Table 3.2 Finger design compression:

	1st design	2nd design	3rd design	4th design
Pros	Easy to manufacture		Easy to manufacture	Easy to manufacture
Cons	Large number of parts	Large number of parts	Hard to manufacture	Must be 3d printed in one peace

3.2.2.8 Final Design Of The Robot:

Fig 3.24 Explosion view of the entire hand

3.2.3 Gearbox Design:

To achieve the objectives of the project, a comprehensive analysis of the gearbox was required,

A split ring compound planet gearbox was decided for the project; the ratio will be divided into two stages the first will be determent from the diminutions that the gearbox must be in:

The full ratio must be determent –based on the project requirements- in this case the full ratio that is required is 255.

For determining the first stage the inner gearg must be 90 teeth as imposed with a module 0.8, from the Equation 3.1 the second stage gear ratio will be calculated.

$$2_{end} = \frac{innergear}{3} \text{ (Eq 3.2)}$$

$$2_{end} = \frac{90}{3} = 30$$

(3 is the deference between number of teeth of the first stage and the second stage)

From the Equation 3.2 the first stage ratio will be calculated;

$$Full_{ratio} = First_{ratio} \times Secound_{ratio} \text{ (Eq 3.3)}$$

$$First_{ratio} = \frac{255}{30} = 8.5$$

From the Equation 3.3 the number of sun gear teeth are;

$$First_{stage} = \frac{inner\ gear_{first\ stage}}{sun\ gear_{first\ stage}} + 1 \text{ (Eq 3.4)}$$

$$sun\ gear_{first\ stage} = \frac{First_{stage} - 1}{inner\ gear_{first\ stage}} = 12\ teeth$$

The module multiplied by the number of teeth, will give us the dimeter of the gears.

Table 3.1 Gears Parameters:

	Number of teeth	Dimeter=module (0.8) * Number of teeth [mm]
Sun Gear 1est stage	12	9.6
Inner Gear 1est stage	90	72
Inner Gear 2end stage	87	69.6
Sun Gear 2end stage	15	12

In the split ring configuration, there exists an additional sun gear designated for the second stage of the transmission system. The primary function of this sun gear is to maintain the structural integrity of the second stage. Consequently, it is not necessary to consider it in the initial analysis. However, it will be subjected to calculations to ensure that it is appropriately sized to fit within the parameters of the second stage.

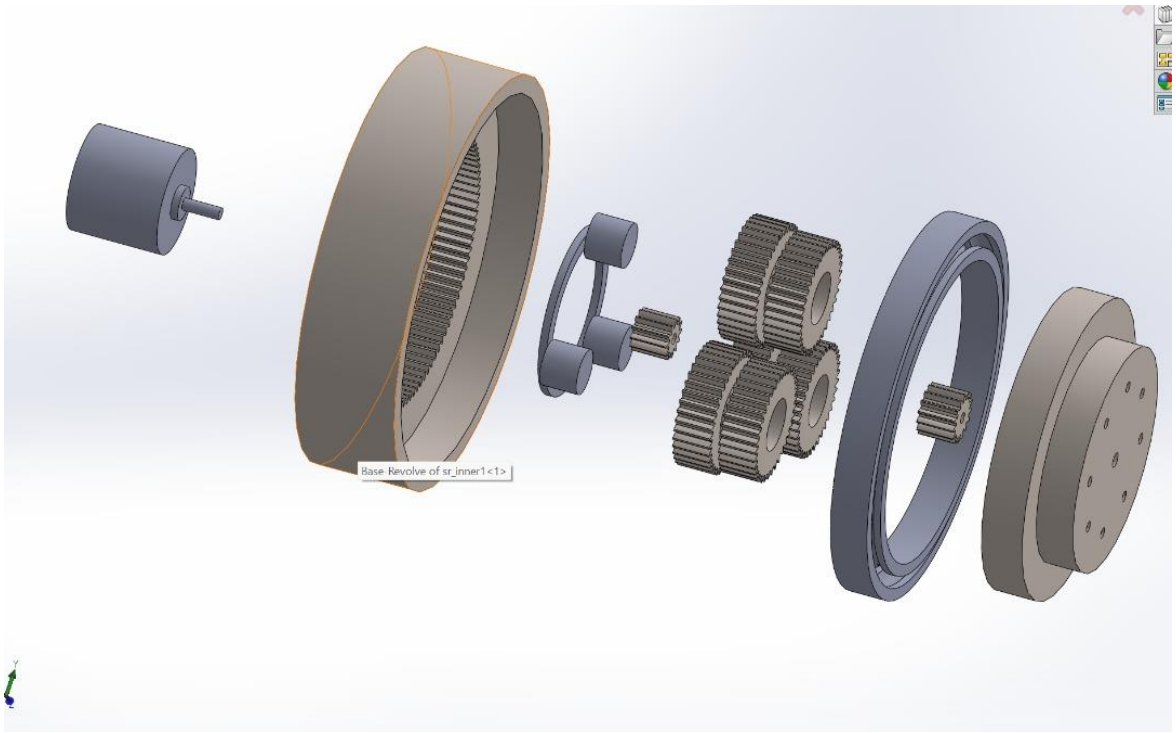


Fig 3.25 Split ring gearbox exploded view

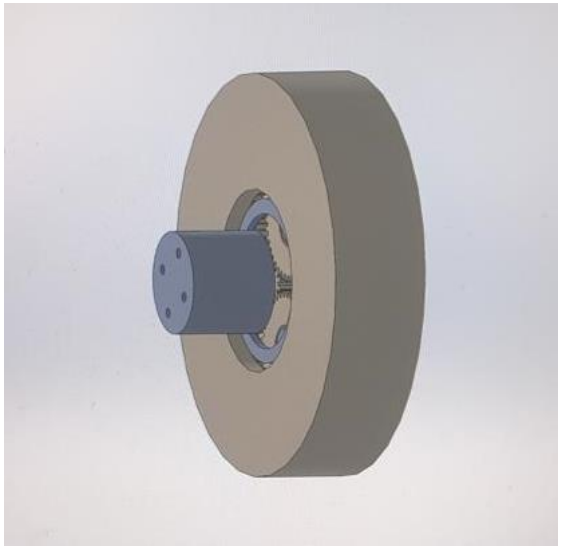


Fig 3.26 isometric Gearbox rear view

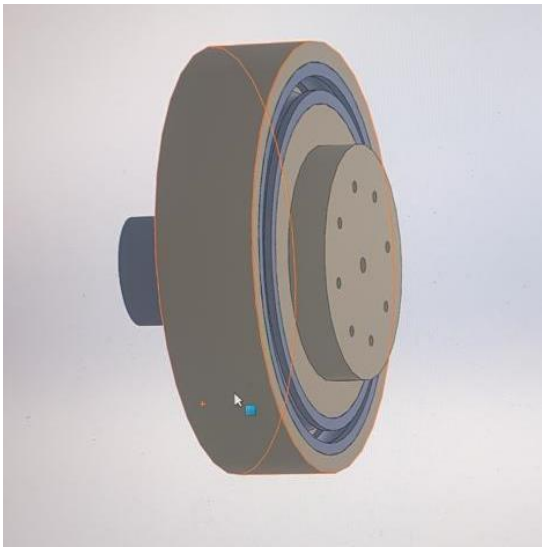


Fig 3.27 Gearbox isometric frontal view

3.2.3Belt Drives Design:

The belt drives were used in the design instead of a direct drive in order to reduce the load on the shoulder actuators

The slipping is ignored because the belts that are used are toothed belts

The speed ratio for the elbow belt is 1.6:1 we used GT2 belt with 6mm width the safety factor of the torque applied on the elbow is:

$$safety\ Factor = \frac{T_{actuator}}{T} = \frac{6}{4.61} = 1.3 \text{(Eq 3.5)}$$

the max force for each mm is 86N

The calculation of the first belt:

T1=5 T2=8

$$F_{belt} = \frac{T}{R_{Pulley}} \text{ (Eq 3.6)}$$

F1=806.45 N F2=784.3125N

$$\sqrt{F_i} = 0.5\left(\sqrt{F_1} + \sqrt{F_2}\right) \text{ (Eq 3.7)}$$

Fi = 795.185

$$F_{max} = F * 2 = (86 * 6) * 2 = 1032\ N$$

$$Safety\ Factor = \frac{F_{max}}{F_i} = 1.3 \text{ (Eq 3.8)}$$

$$\sigma = \frac{F}{A} = 87.19MPa \text{ (Eq 3.7)}$$

$$\varepsilon = \frac{\sigma}{E} = 0.174 \text{ (Eq 3.8)}$$

A=9.12∗10^−6m^2

To calculate the belt length the following formula is used: D=13cm ,R1=0.8cm ,R2=1.25cm

$$L = 20 + \pi(R_1 + R_2) - \frac{(R_2 - R_1)^2}{D} \text{ (Eq 3.9)}$$

$$L_1 = 34\ cm$$

The speed ratio for the wrist belt is 1:1 because the pulleys have the same diameter, we used GT2 belt with 3.5mm width the second and the third pulley are the same

The calculation of the second and the third belts:

$$F_i = F_1 = 201.6N$$

$$F_{max} = F * 2 = (86 * 3.5) * 2 \ = 63\ N$$

$$Safety\ Factor\ = \frac{F_{max}}{F_i} = 2.98$$

$$A = 5.47 * 10^{-6} m^2$$

$$\sigma = \frac{F}{A} = 36.86\ MPa$$

$$\varepsilon = \frac{\sigma}{E} = 0.073$$

$$L_2 = L_3 = 48.83\ cm$$

$$R_1 = R_2 = 0.8\ cm$$

3.3 Electrical Design:

The power consumption:

To make sour that the right power supply was chosen. We had to calculate and know the max power that the motors drain.

A2212 brushless motor:

There is used 2 A2212 brushless motor 1400kv, that are running on 12v with an output torque of 0.071 [N.m] . Using the following equation we can calculate the currant;

(1)n=V×kv=12×1000=12000 [rpm] , (2) ω=12000×2π60=1256.6 [rads] $I = \frac{T \times \omega}{V \times \eta} = 9.9 \approx 10\ A$ (Eq 3.10), The power that is drained from one A2212 brushless motor is P=120 watt, so from both motors the power would be P=240 watt.

Servo 50kg:

3 servo 50kg are used on the elbow, that runs on 12v, with an output torque of T=4.44 [N.m], the current would be 5A (from datasheet).

The power consumption of the system was calculated to be 60 Watts, based on the product of the voltage (12V) and current (5A) requirements. (3) P = 12V x 5A = 60W. Furthermore, the system incorporates three servo motors, each with an identical power consumption profile, resulting in a cumulative power drain of 180Watt. This leads to a total power consumption of 180Watt, underscoring the significance in the overall system design and implementation.

Nema 17 stepper motor:

Nema 17 (single stack) is a power-full stepper motor, that operates on 12v, generating torque of 0.015 [N.m], the current would be I=2 A (from data sheet).

The power would be, P=24 watt.

moving the fingers:

For moving the hall fingers close together, MG996R servo motor was chosen. this motor generates 0.921N.m, for 5v and 500mA, P=2.5 watt. For all eight motors P=20 watt.

For opening and closing the fingers, SG90 servo motor was chosen. This motor generates 0. 15N.m, for 5v and 60mA, that makes P=0.3 watt. And for five motors 1.5 watt.

3.4 Capturing data from sensors:

Capturing data from the sensors is a critical step in the motion capture process. This involves collecting data from the sensors embedded in the motion capture suit and transmitting it to the ESP (Microcontroller). The data is then packed and sent to the computer via serial communication, where it is processed and utilized for 3D model animation.

Microcontroller Programming:

In this research project, the Arduino IDE development environment was used to program the ESP-32s microcontroller, which is a powerful and suitable processor for collecting sensory data. The primary role of this processor is to collect sensory values from a variety of sensors connected to it. After the collection process, the processor organizes these values into a structured data packet (Packet) to be sent to the central computer via the serial port.

Data Packet Structure:

Start Byte:

This byte represents the initial signal that informs the computer that data transmission has begun. This byte is crucial for synchronizing the transmission and reception processes, ensuring that the computer reads the incoming data correctly.

Destination Byte:

This byte identifies the sensor to which the value belongs, making it easier to determine the source of the data on the receiving computer.

Value Byte:

This part of the packet contains the measured value from the sensor, spanning four bytes. The data is encoded in this format to ensure accurate reading, as four bytes allow for a wide range of numerical values, enhancing the precision of the transmitted measurements.

Transmission Termination:

After sending the data packet, the system sends a Carriage Return and Line Feed. These two characters are essential for properly terminating the transmission, ensuring that the receiving computer recognizes that the packet is complete and can process the received data or wait for a new packet.

3.4 Receiving sensors data and moving the robot:

Several microcontrollers are dedicated to actuating the robotic arm; however, the primary control unit is an ESP-32. This main controller is responsible for receiving data transmitted from the motion capture suit via the ESP-Now protocol.

Upon reception, the ESP-32 processes the incoming data and determines the appropriate action: either directly controlling the motors or relaying the commands to another microcontroller tasked with operating the miniplate motors. This hierarchical architecture ensures efficient distribution of control signals and optimizes the responsiveness of the overall system.

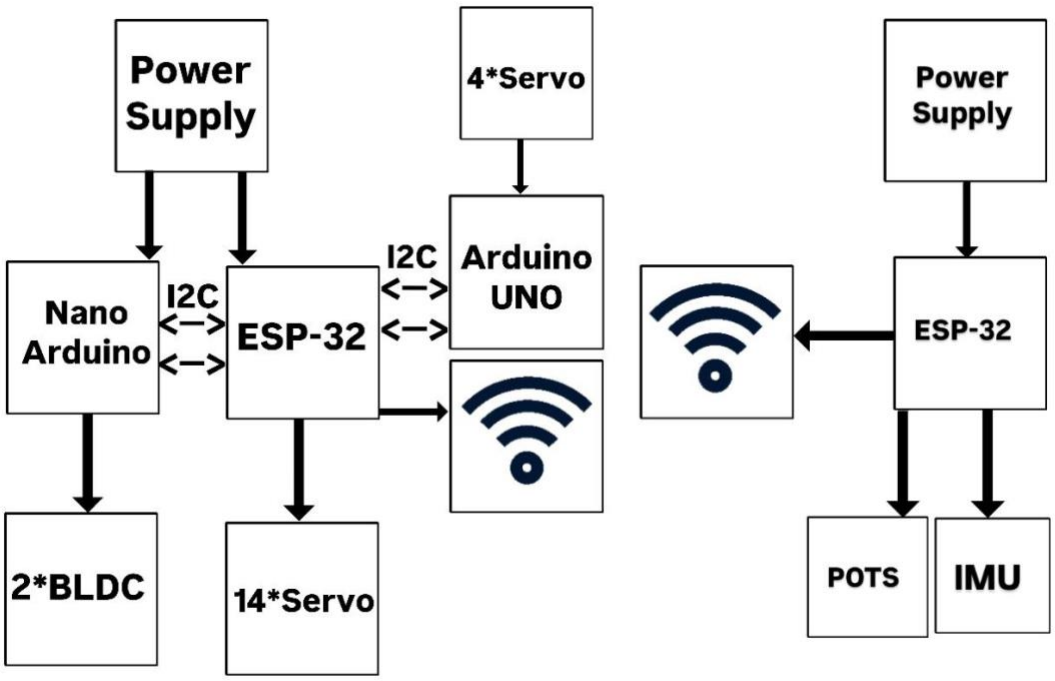


Fig 3.28 Block Diagram of Capturing Sensors Data and Receiving Them

3.4 Kinematics of the arm:

To control a 7-degrees-of-freedom (7-DOF) robot arm, the first step is to build its kinematic model. This starts by creating the Denavit-Hartenberg (DH) table, which helps describe how each part of the arm is connected. From this table, we can find the homogeneous transformation matrices that show the position and rotation of each link.

The kinematics of the proposed robotic arm is inspired by the kinematics of the human upper limb. Therefore, the structure of its Denavit-Hartenberg (DH) table is designed to resemble that of the human arm. In the work of Andrea Maria, Paolo Rocco, and others on human arm kinematics [11], a representative DH table is presented. Their model serves as a reference for the development of the DH parameters used in this project.

Table3.2 Denavit-Hartenberg parameters of the kinematic model

axis, i	a_i	d_i	α_i	θ_i
1	0	0	$\pi/2$	q_1
2	0	0	$\pi/2$	q_2
3	0	d_3	$-\pi/2$	q_3
4	0	0	$-\pi/2$	q_4
5	0	d_5	$\pi/2$	q_5
6	0	0	$\pi/2$	q_6
7	0	0	0	q_7

However, in our robotic design, an additional degree of freedom for the as the body of the robot was introduced, making the system 8-DOF. For the purpose of kinematic modeling, this joint was assigned a fixed value of 0°, effectively excluding it from the transformation calculations. As a result, the DH table was modified accordingly to reflect this adjustment.

Table3.3: DH-Table of the project

	θ	d	a	α
1	0	L1	0	$-\frac{\pi}{2}$

2	$-\frac{\pi}{2}$	L2	0	$-\frac{\pi}{2}$
3	$-\frac{\pi}{2}$	0	a1	$-\frac{\pi}{2}$
4	0	L3	0	$\frac{\pi}{2}$
5	0	0	0	$-\frac{\pi}{2}$
6	0	L4	0	$\frac{\pi}{2}$
7	$\frac{\pi}{2}$	0	a2	$\frac{\pi}{2}$
8	$-\frac{\pi}{2}$	0	a3	0

From this table, the translation matrix for the first three degrees of freedom can be derived, starting from the second link. The first link represents the stationary base (the robot's body) and is assigned a fixed angle of 0°, thus it does not contribute to the transformation chain.

C24 +C3*S24	-S23	C2*S4-C34*S2	-30*S23
S34	C3	-C4*S3	30*C3+5
-C23*S4-C4S2	C2S3	C234-S24	30C2S3+20
0	0	0	1

Chapter 4

Results of the study

4.1 Introduction

This chapter presents the results from the testing and evaluation of the wearable motion capture system and its integration with the robotic arm and neck. It focuses on the accuracy of motion tracking, the responsiveness of the robotic system, and the effectiveness of the material and design choices. The results are organized to highlight key performance metrics such as sensor data accuracy, robotic movement replication, real-time data transmission, and user comfort. These findings offer insights into the system's strengths and areas for improvement, forming the foundation for the conclusions in the final chapter.

4.2 stress results

The stress analysis is performed in SolidWorks 2021. To start the study, First the Simulation add-in is activated by navigating to Add-Ins and checking the box next to SOLIDWORKS Simulation. Then, a new study is created by clicking on the Simulation tab in the Command Manager, selecting New Study, and choosing the Static study type.

Material properties are assigned by right clicking on the part in the Simulation study tree and selecting Apply Material to All Bodies. Fixtures are applied by clicking Fixtures in the Simulation tab, selecting the faces, edges, or vertices to fix, and specifying the type of restraint (Table 3.4). Similarly, loads are applied by clicking External Loads in the Simulation tab, selecting the load type, and defining the magnitude and direction Then, Run is clicked in the Simulation tab. Finally, the results are reviewed by selecting the result plots in the Simulation study tree and using the Stress Plot to view stress distribution. By examining the results, we can confirm that the applied forces are safe. The tables below display the values that were obtained.

Multiple studies have been made for the part after a study is made. Here are the results:

4.2.1 Shoulder Stress Analysis:

We will analyze the stress applied to the shoulder using three main designs: a fully 3D-printed model, an aluminum alloy 1060 design, and a carbon fiber design:

- **Fully 3D-printed model:**

Table 4.1 stress test:

Name	Type	Min	Max
Stress σ	VON: von Mises Stress	9.869e+01N/m^2	3.539e+06N/m^2
Displacement	URES: Resultant Displacement	0.000e+00mm	5.702e-01mm
Strain ε	ESTRN: Equivalent Strain	6.518e-08	1.115e-03

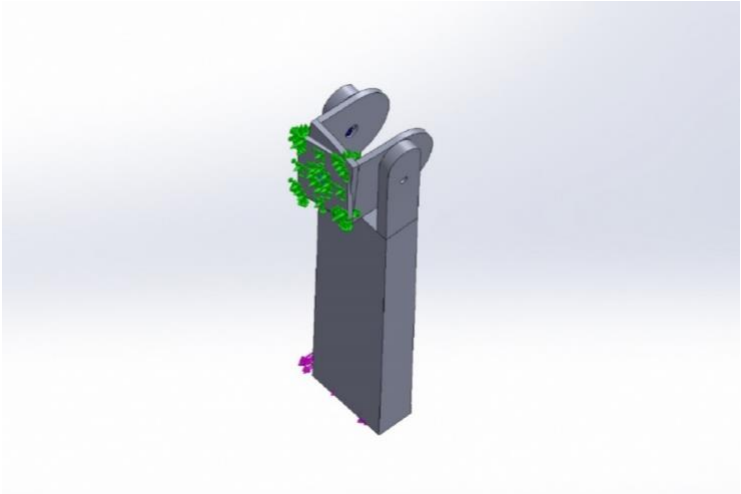


Fig 4.1 stress study full 3D model

- **Aluminum alloy 1060 design:**

Table 4.2 stress test:

Name	Type	Min	Max
Stress σ	VON: von Mises Stress	0.000e+00N/m^2 Node: 216967	3.113e+07N/m^2 Node: 110754

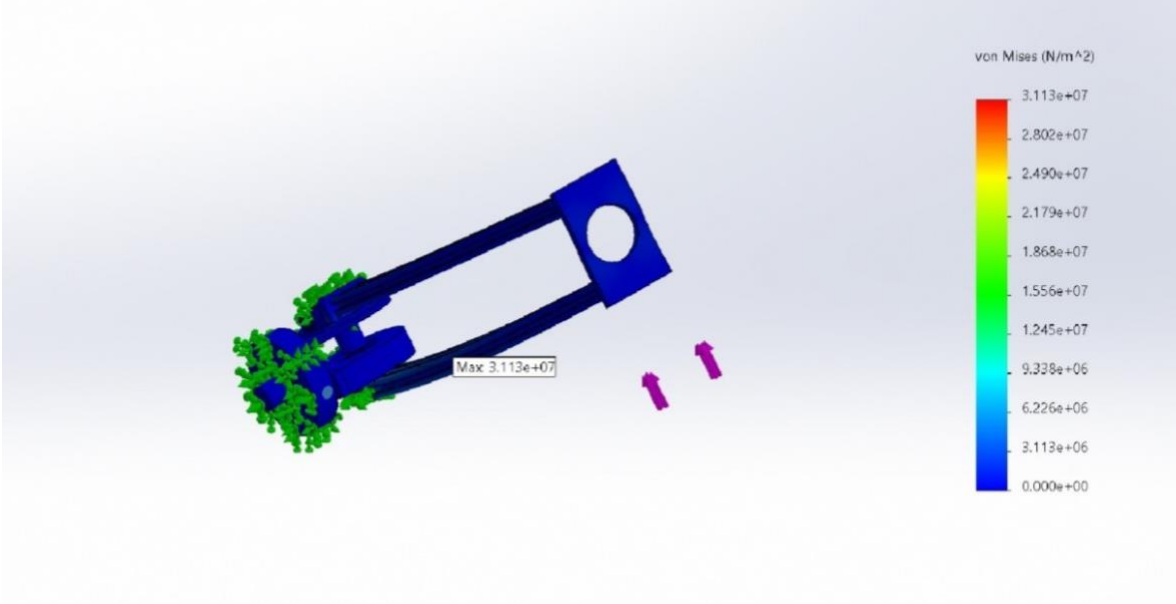


Fig 4.2 Stress study aluminum 1060 stress graph

Table 4.3 stress test:

Name	Type	Min	Max
Displacement	URES: Resultant Displacement	0.000e+00mm Node: 216967	1.555e+00mm Node: 209111

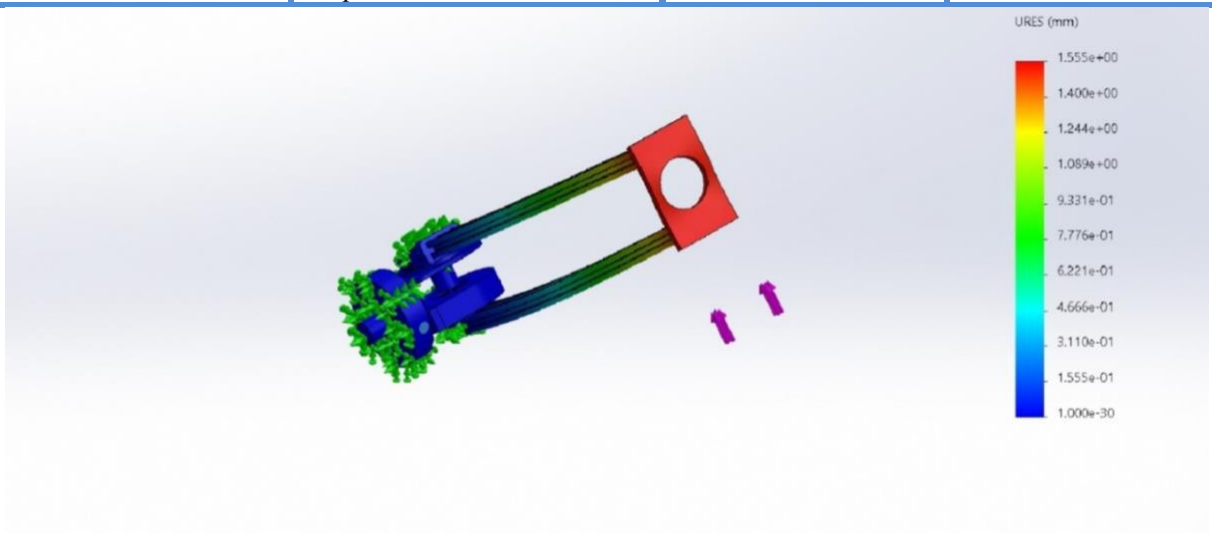


Fig 4.3 Stress Study aluminum 1060 displacement

Table 4.4 stress test:

Name	Type	Min	Max
Strain ε	ESTRN: Equivalent Strain	0.000e+00 Element: 123577	2.289e-03 Element: 128433

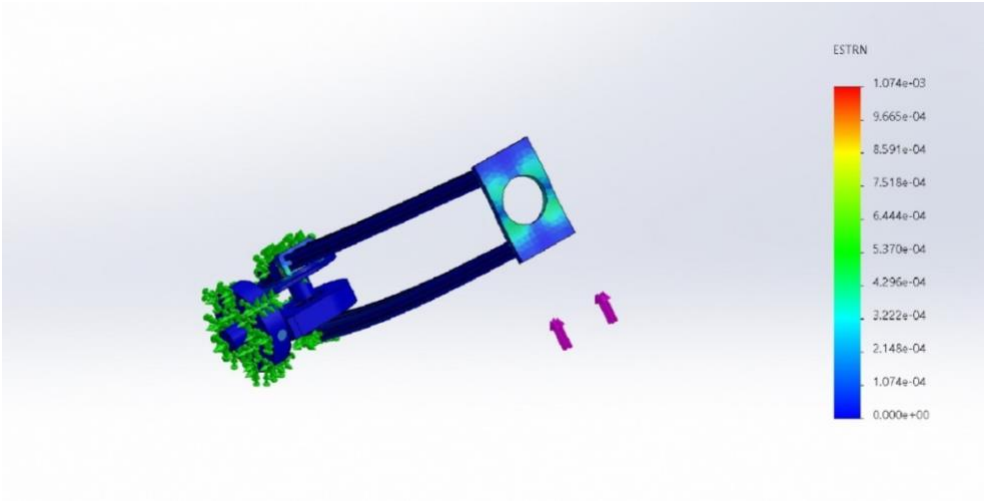


Fig 4.4 Stress Study aluminum 1060 strain

- **Carbon fiber design:**

Table 4.5 stress test:

Name	Type	Min	Max
Stress σ	VON: von Mises Stress	0.000e+00N/m^2 Node: 65734	1.299e+08N/m^2 Node: 503

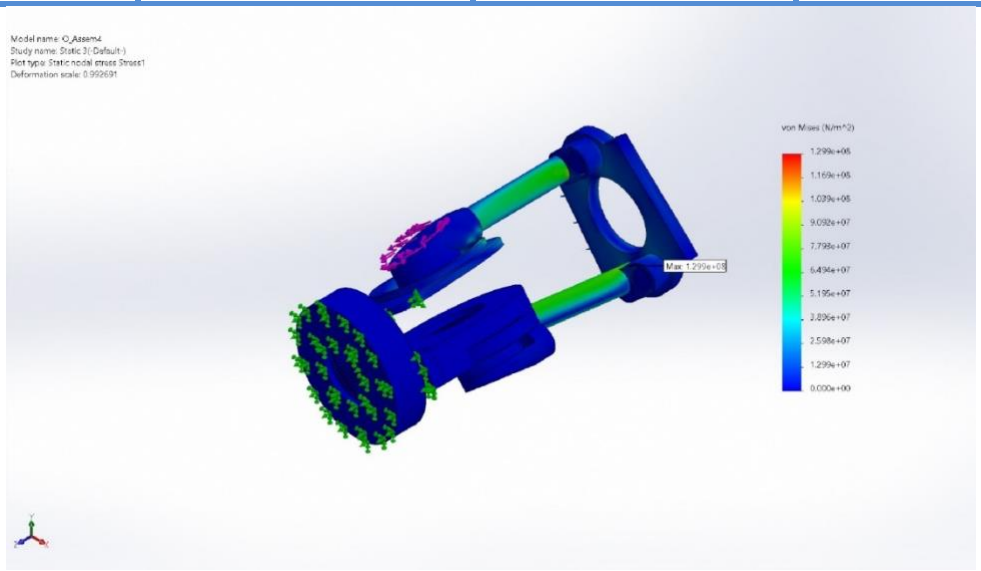


Fig 4.5 Stress study carbon stress graph

Table 4.6 stress test:

Name	Type	Min	Max
Displacement	URES: Resultant Displacement	0.000e+00mm Node: 65734	3.265e+01mm Node: 150713

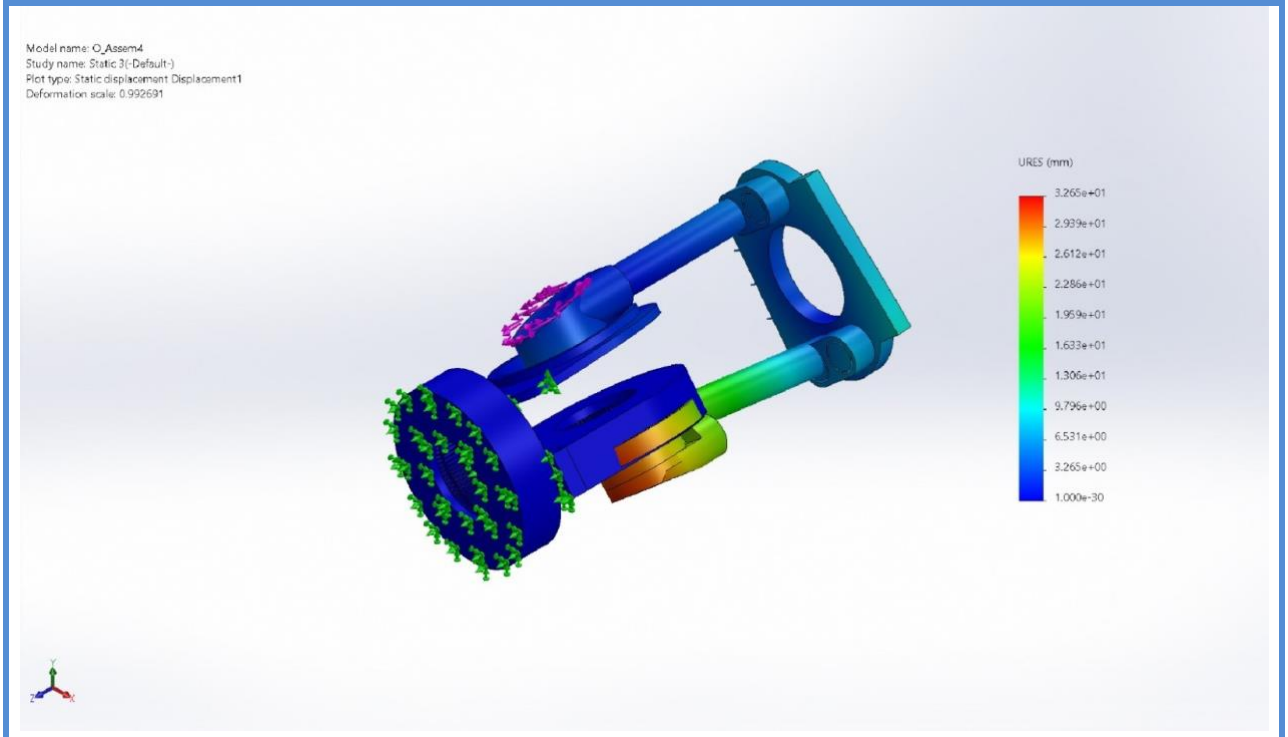


Fig 4.6 Stress Study carbon displacement

Table 4.7 stress test:

Name	Type	Min	Max
Strain ε	ESTRN: Equivalent Strain	0.000e+00 Element: 40424	3.432e-02 Element: 105480

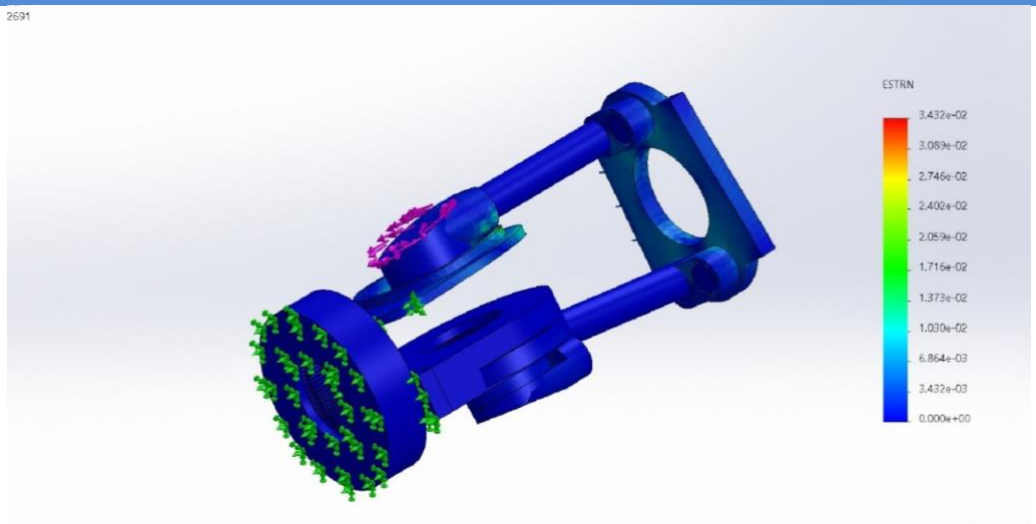


Fig 4.7 Stress Study aluminum 1060 strain

4.2.2 Forearm Stress Analysis:

We will analyze the stress applied to the forearm using two main designs: a fully 3D-printed model, and an aluminum alloy 1060 design:

- **Fully 3D-printed model:**

Table 4.8 stress test

Name	Type	Min	Max
Stress σ	VON: von Mises Stress	1.876e+01Pascal	6.746e+06 Pascal

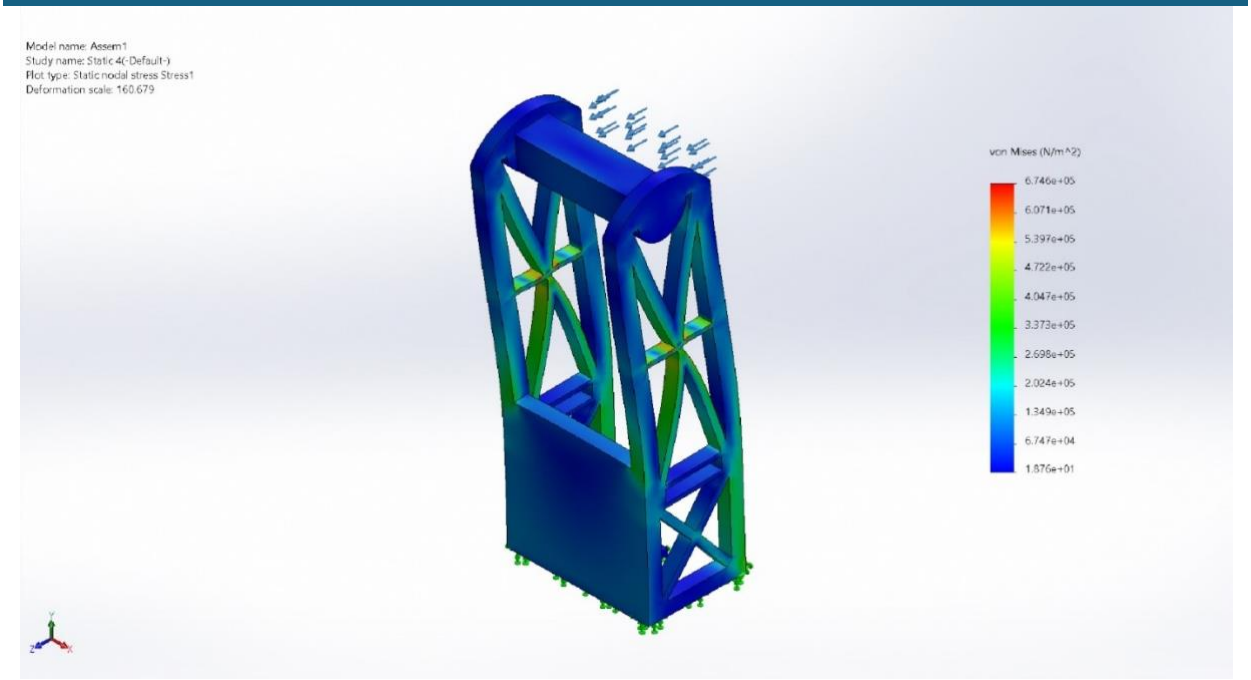


Fig 4.8 stress study 3D model forearm

Table 4.9 stress test:

Name	Type	Min	Max
Displacement	URES: Resultant Displacement	0.000e+00mm	1.735e-01mm

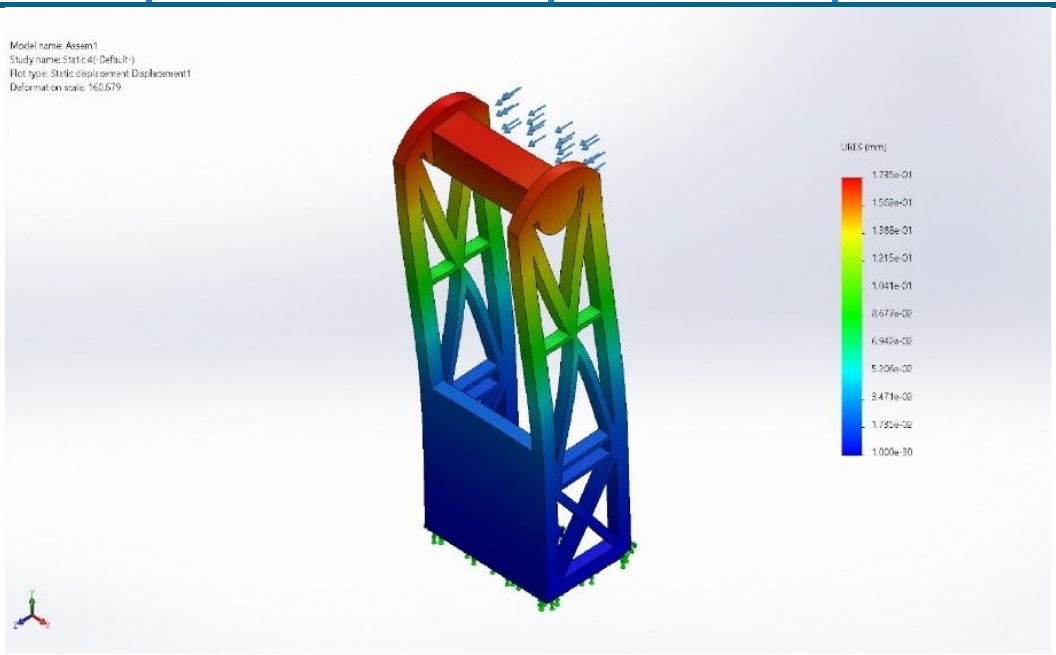


Fig 4.9 Displacement study 3D model forearm

Table 4.10 stress test:

Name	Type	Min	Max
Strain ε	ESTRN: Equivalent Strain	6.960e-09	2.235e-04

- **Aluminum alloy 1060 design:**

Table 4.11 stress test:

Name	Type	Min	Max
Stress σ	VON: von Mises Stress	4.351e+01N/m^2 Node: 36836	4.734e+06N/m^2 Node: 2649

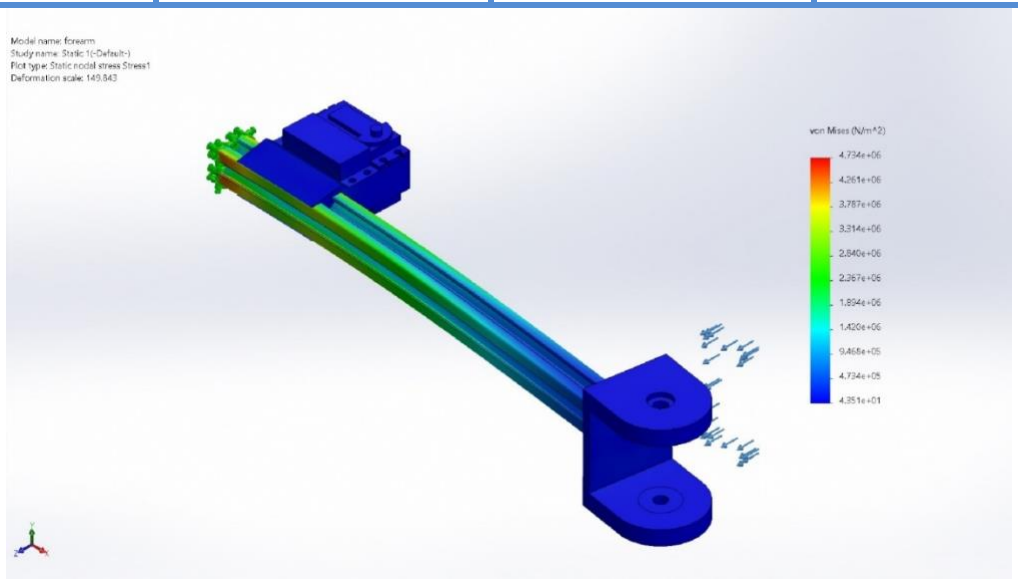


Fig 4.10 Stress study aluminum 1060 stress graph forearm

Table 4.12 stress test:

Name	Type	Min	Max
Displacement	URES: Resultant Displacement	0.000e+00mm Node: 51	2.369e-01mm Node: 49168

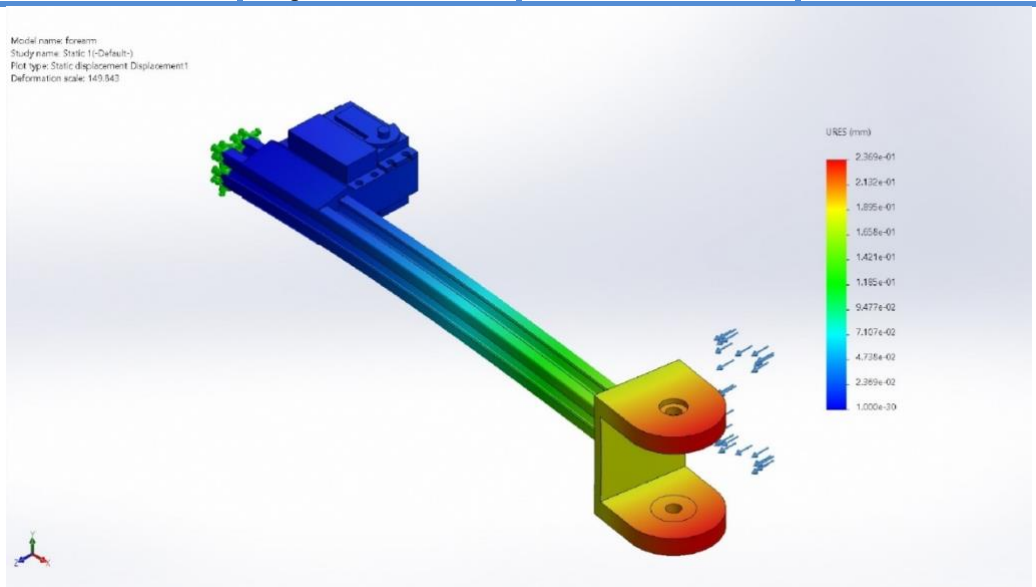


Fig 4.11 Displacement study aluminum 1060 stress graph forearm

Table 4.13 stress test:

Name	Type	Min	Max
Strain ε	ESTRN: Equivalent Strain	1.086e-08 Element: 20407	1.150e-04 Element: 26210

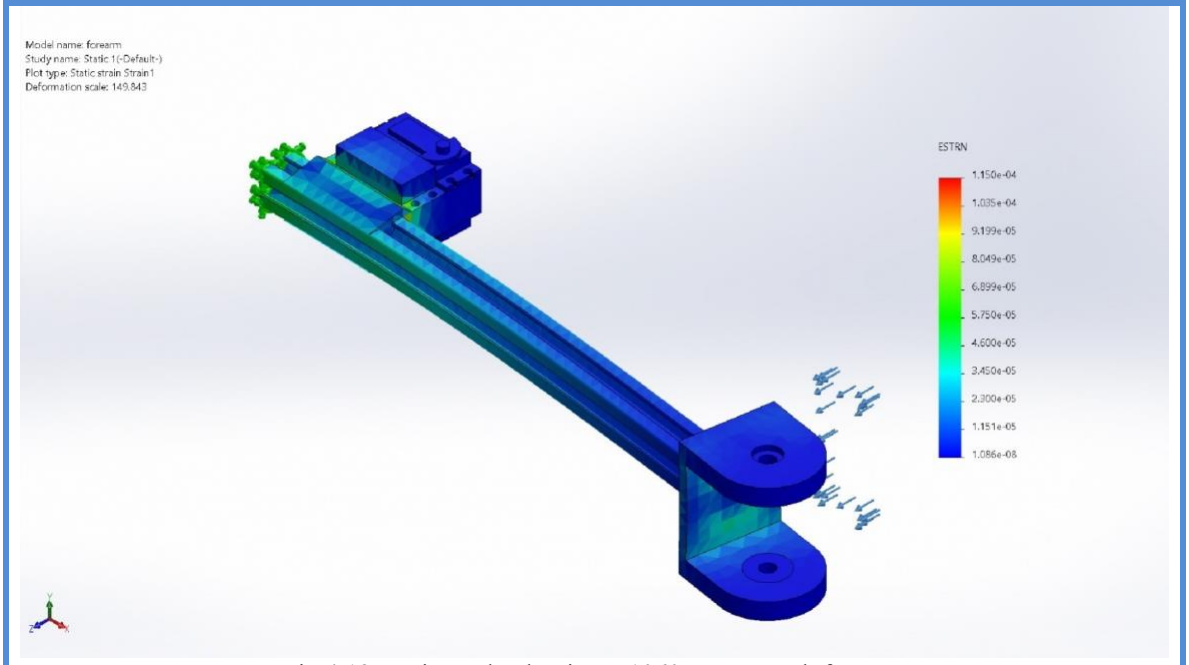


Fig 4.12 Strain study aluminum 1060 stress graph forearm

4.2.3 Comparison between Aluminum 6060 vs Carbon Fiber: [9]

Table 4.14 Comparison between Aluminum 6060 vs Carbon Fiber:

Property	Carbon Fiber	Aluminum 6061
Density (g/cm³)	1.55 - 1.75	2.70
Tensile Strength (MPa)	2,400 - 5,500	290 - 350
Ultimate Stress (MPa)	2,500 - 6,000	270 - 310
Young's Modulus (GPa)	120 - 250	68.9
Yield Strength (MPa)	1,000 - 3,000	240
Poisson's Ratio	0.2 - 0.3	0.33
Elongation (%)	1.5 - 2.5%	12 - 17%
Shear Modulus (GPa)	5 - 7	26.0
Fatigue Strength	Very high (depends on the fiber lay-up)	Moderate (depends on alloy purity)
Deformation	Very low deformation due to high rigidity	Higher deformation under load due to lower stiffness
Strain at Failure (%)	1 - 2%	10 - 15%
Creep Resistance	Excellent, low creep	Moderate (Can creep at high temperatures)
Impact Resistance	Low (brittle)	Moderate (ductile)
Corrosion Resistance	Very high (does not rust)	Good (but can corrode; protective treatments available)
Thermal Conductivity (W/m·K)	Low (0.6 - 1.0)	High (167)
Thermal Expansion (10^{-6}/K)	Low (0.5 - 1.0)	High (23.6)
Weight-to-Strength Ratio	Excellent, lighter for the same strength	Moderate (heavier for the same strength)

4.3 The Electrical result:

Total power consumption:

The total power consumption of the system is the sum of the power drained by all motors:

P = 465 Watt

Power Supply Selection

To ensure reliable operation, it's recommended to select a power supply that can provide at least 10-20% more power than the total power consumption of the system. This allows for some headroom in case of unexpected power spikes or increases in motor current.

Let's calculate the minimum required power supply rating:

(1) P_supply = P + (10% to 20% of P)

= 465 Watt + (46.5 Watt to 93 Watt)

= 511.5 Watt to 558 Watt

Recommended Power Supply for one arm:

Based on the calculation, a power supply with a rating of at least 550 Watt to 600 Watt would be suitable for this system. This will provide a comfortable margin to ensure reliable operation and minimize the risk of power supply failure.

Some possible power supply options could be:

- 550 Watt switching power supply
- 600-Watt redundant power supply (for high-reliability applications)

4.4 Result of sensor fusion in IMU:

As mentioned in theoretical frame, it is important to apply senser fusion algorithms when implementing MPU-6500.

The MPU-6500 is capable of measuring orientation; however, since it does not include a magnetometer, it cannot directly measure yaw (rotation around the z-axis). Therefore, the MPU-6500 is limited to measuring roll (rotation around the x-axis) and pitch (rotation around the y-axis).

4.4.1 Sensors data without using any filter:

The MPU-6500 provides 3 degrees of freedom from the accelerometer and 3 degrees of freedom from the gyroscope. Roll (rotation around the x-axis) and pitch (rotation around the y-axis) can be obtained without applying sensor fusion techniques by either integrating the gyroscope data for each axis or by calculating the angles from the accelerometer measurements.

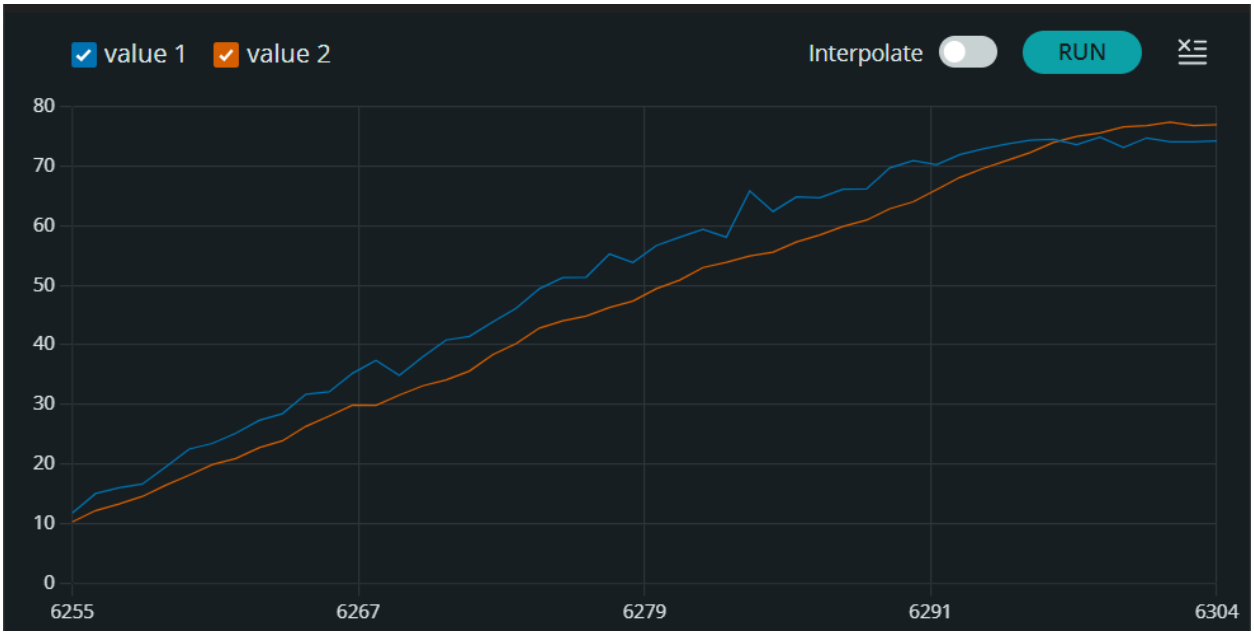


Fig 4.13 gyro and accelerometer pitch sensor data

In Fig we can see the roll value from 10° to 75°.

Value 1: is the roll value computed from the accelerometer.

Value 2: is the roll value computed from the gyro scope.

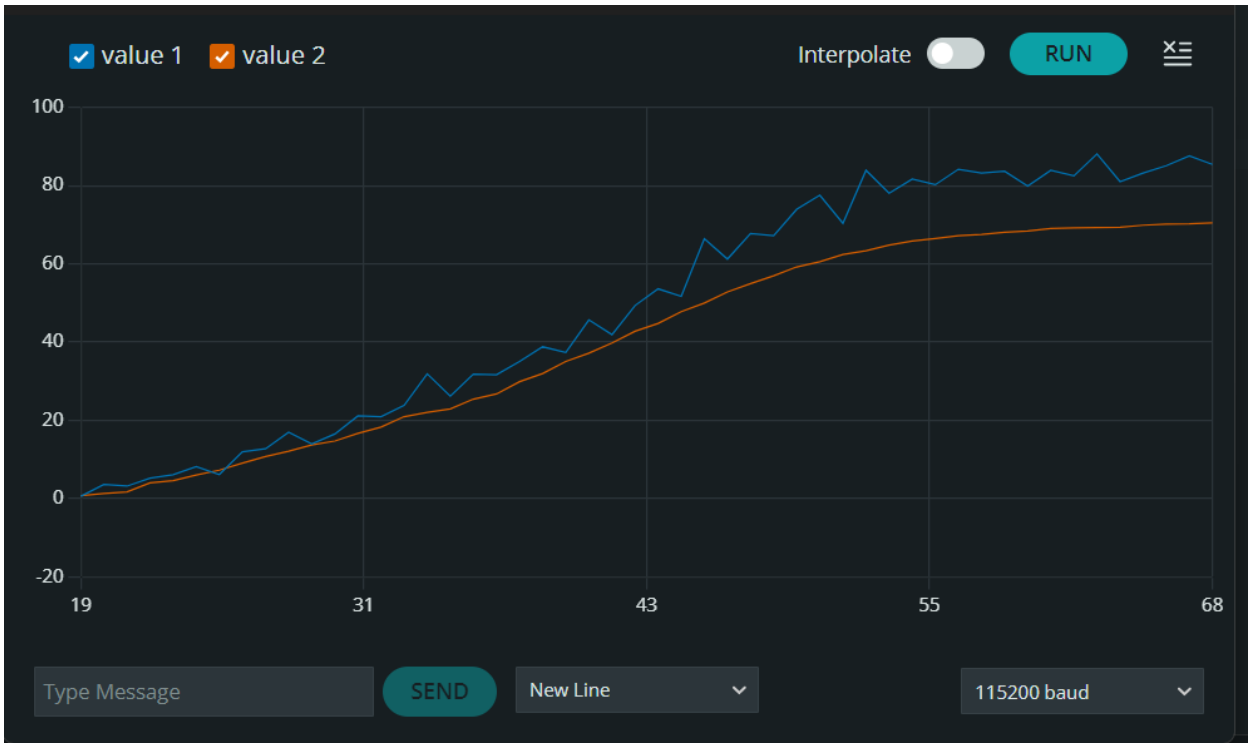


Fig 4.14 gyro and accelerometer roll sensor data

In Fig we can see the roll value from 0° to 70°.

Value 1: is the roll value computed from the accelerometer.

Value 2: is the roll value computed from the gyro scope.

It was found that the gyro measurements are more stable and accurate, but they still exhibit some drift. This causes instability, which is why it cannot be used on its own.

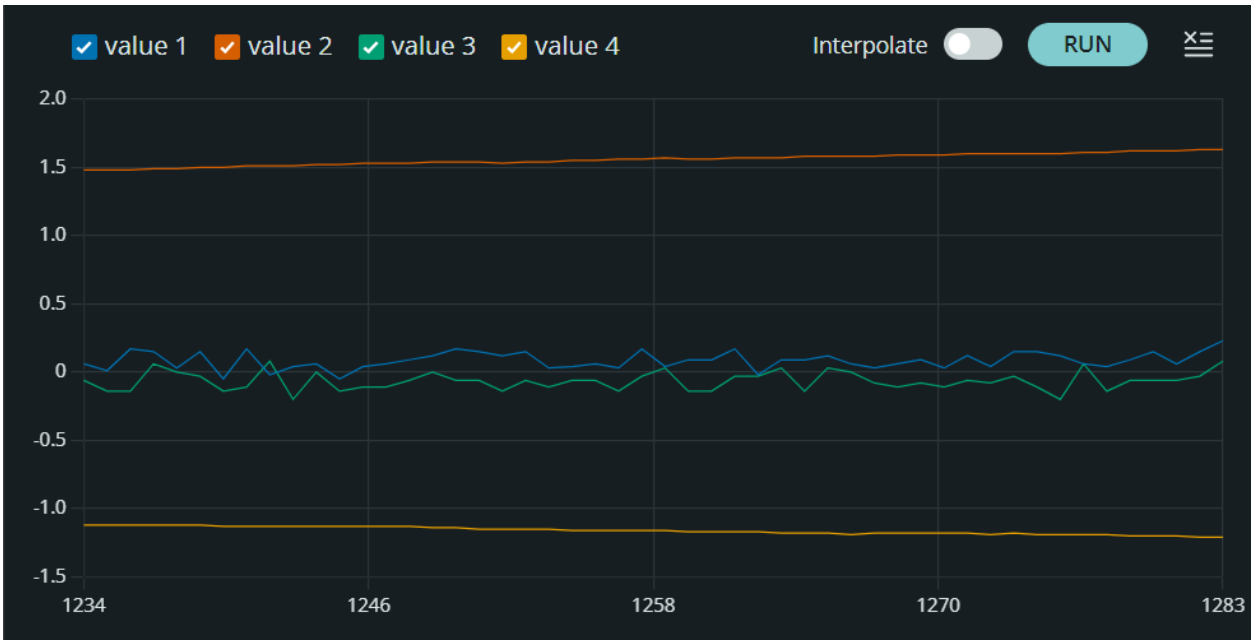


Fig 4.15 Gyro drift

In the Fig, the IMU is placed stationary on a table for a duration of 30 seconds. As illustrated, the gyro drift is represented by the yellow and orange lines, which exhibit significant deviation over time. This pronounced drift highlights the inherent instability of the gyro measurement when left uncorrected.

4.4.2 Complementary filter results:

The gain of the complementary filter will be implemented as α=0.9.

The gain for the gyro would be 90%, and the gain for the accelerometer would be 10%.

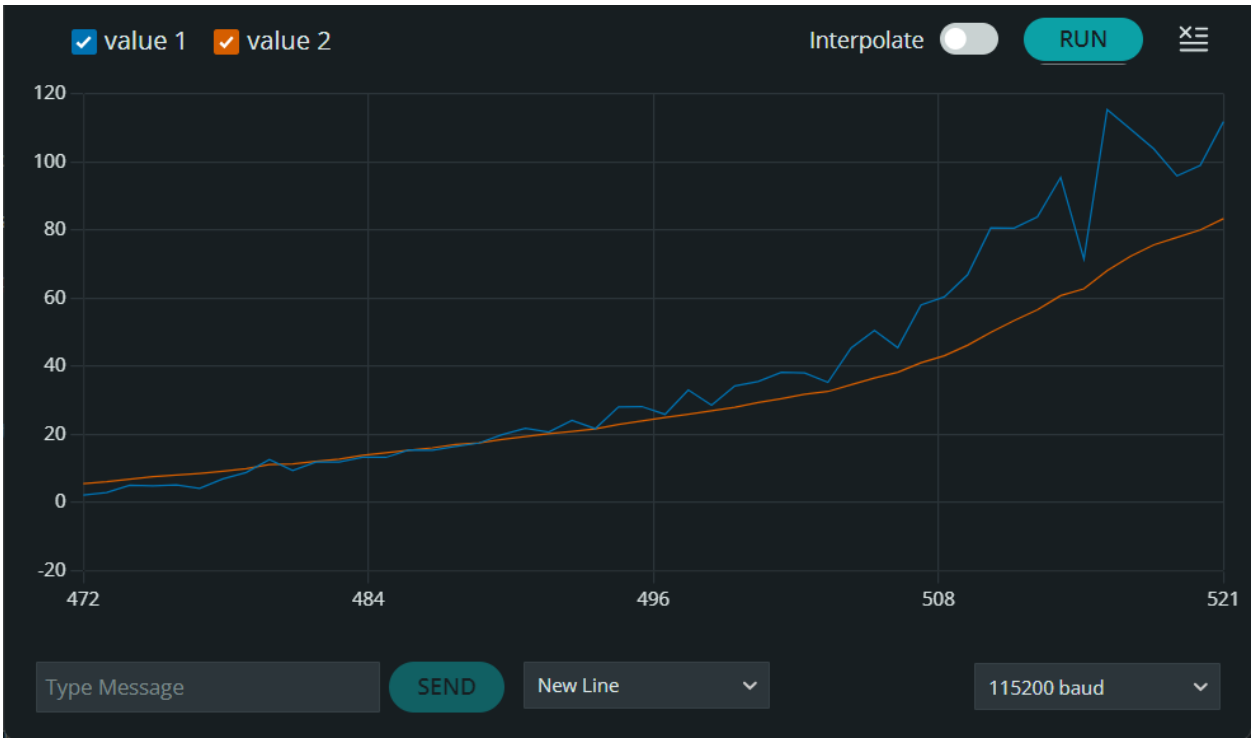


Fig 4.16 roll Complementary filter results

In Fig we can see the roll value from 5° to 82°.

Value 1: is the roll value without implementing Complementary filter (accelerometer data only).

Value 2: is the roll value with implementing Complementary filter.

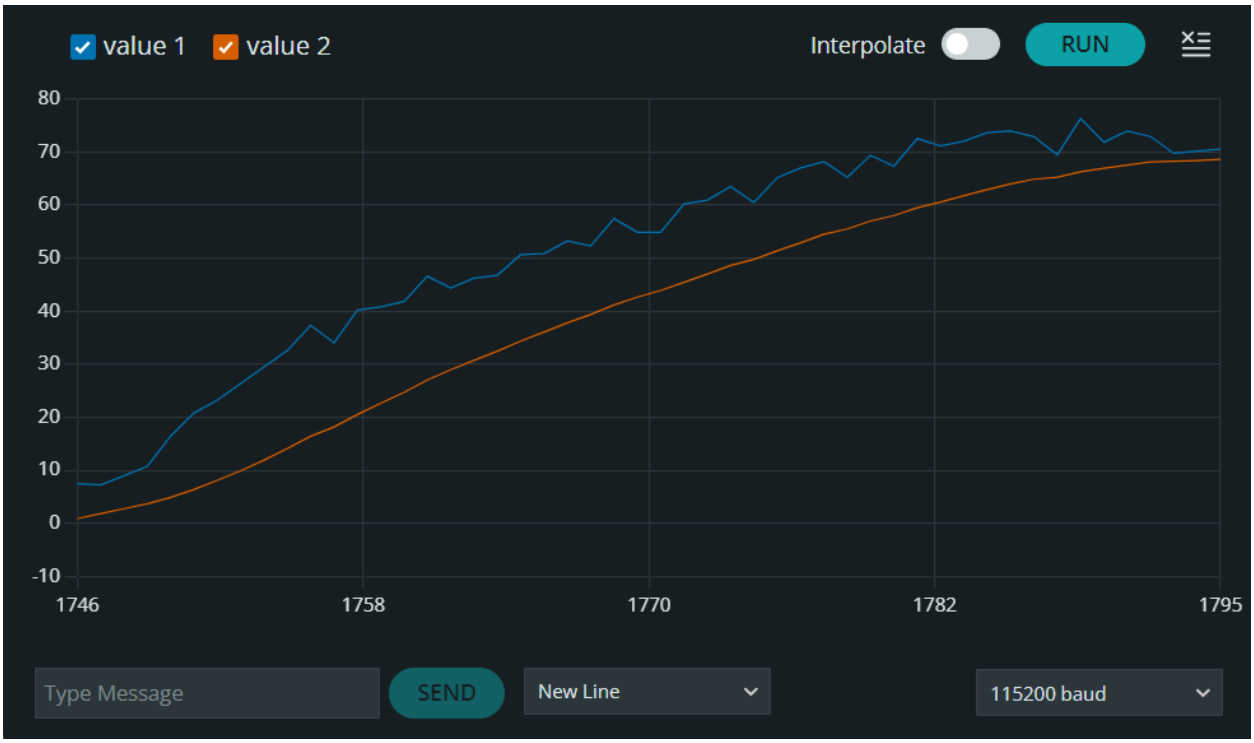


Fig 4.17 pitch Complementary filter results

In Fig we can see the pitch value from 0° to 70°.

Value 1: is the pitch value without implementing Complementary filter (accelerometer data).

Value 2: is the pitch value with implementing Complementary filter.

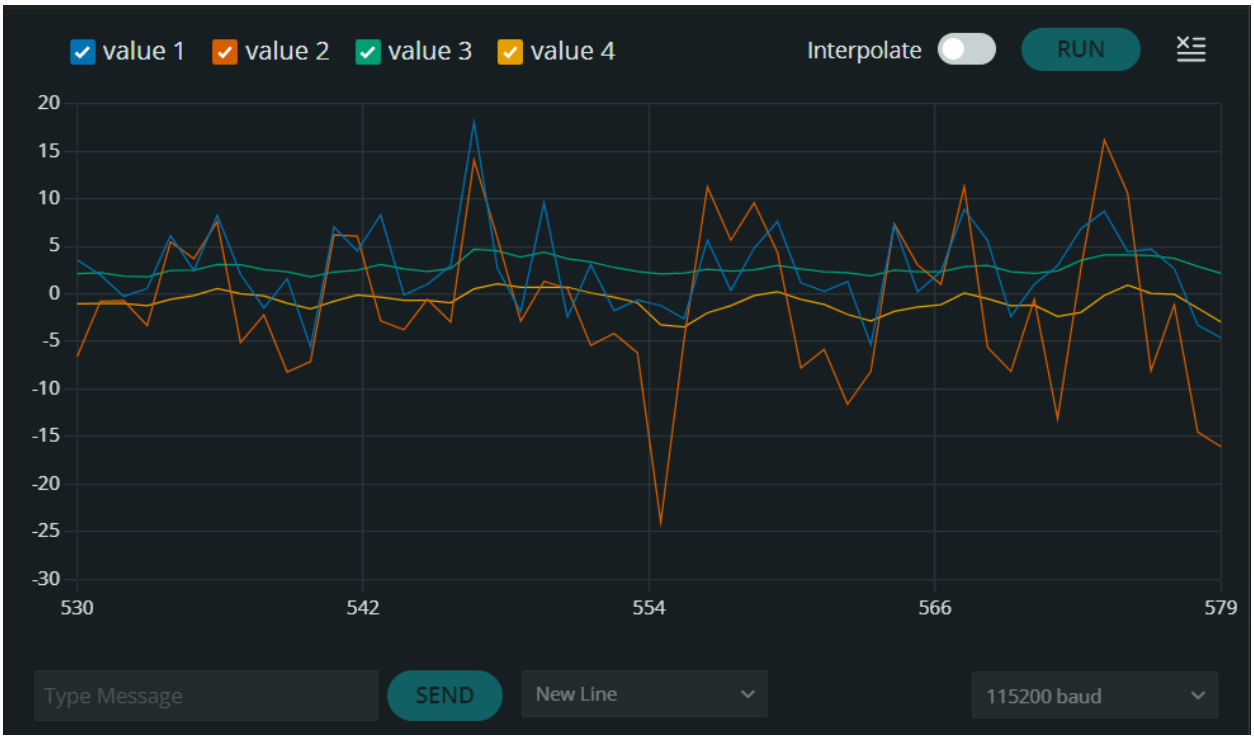


Fig 4.18 noise Complementary filter results

In Fig, the IMU will be rotated sideways along the z-axis, as a noise/vibration test.

Value 1: is the pitch value without implementing Complementary filter (accelerometer data).

Value 2: is the roll value without implementing Complementary filter (accelerometer data).

Value 3: is the pitch value with implementing Complementary filter.

Value 4: is the roll value with implementing Complementary filter.

4.4.3 Kalman filter results:

Fig 4.19 pitch Kalman filtre Resuls

In Fig we can see the pitch value from -5° to -90°.

Value 1: is the pitch value without implementing Kalman filter (accelerometer data).

Value 2: is the pitch value with implementing Kalman filter.

Fig 4.20 roll Kalman filter results

In Fig we can see the roll value from -9° to -71°.

Value 1: is the roll value without implementing Kalman filter (accelerometer data).

Value 2: is the roll value with implementing Kalman filter.

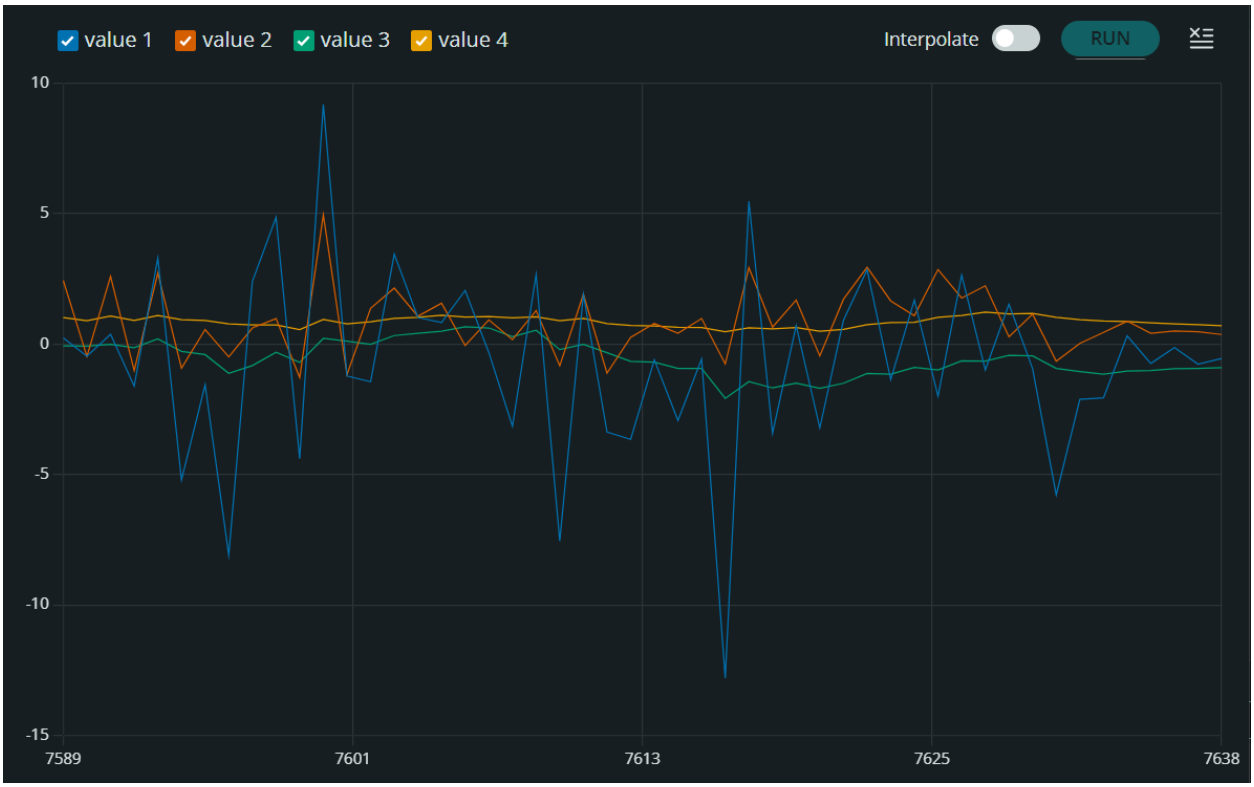


Fig 4.21 noise Kalman filter results

In Fig, the IMU will be rotated sideways along the z-axis, as a noise/vibration test.

Value 1: is the roll value without implementing Kalman filter (accelerometer data).

Value 2: is the pitch value without implementing Kalman filter (accelerometer data).

Value 3: is the roll value with implementing Kalman filter.

Value 4: is the pitch value with implementing Kalman filter.

We can observe that the Kalman filter responds very effectively to noise and vibration. In other words, the presence of noise does not significantly affect the output of the Kalman filter, demonstrating its robustness in filtering out unwanted disturbances.

4.4.4 Complementary filter vs Kalman filter:

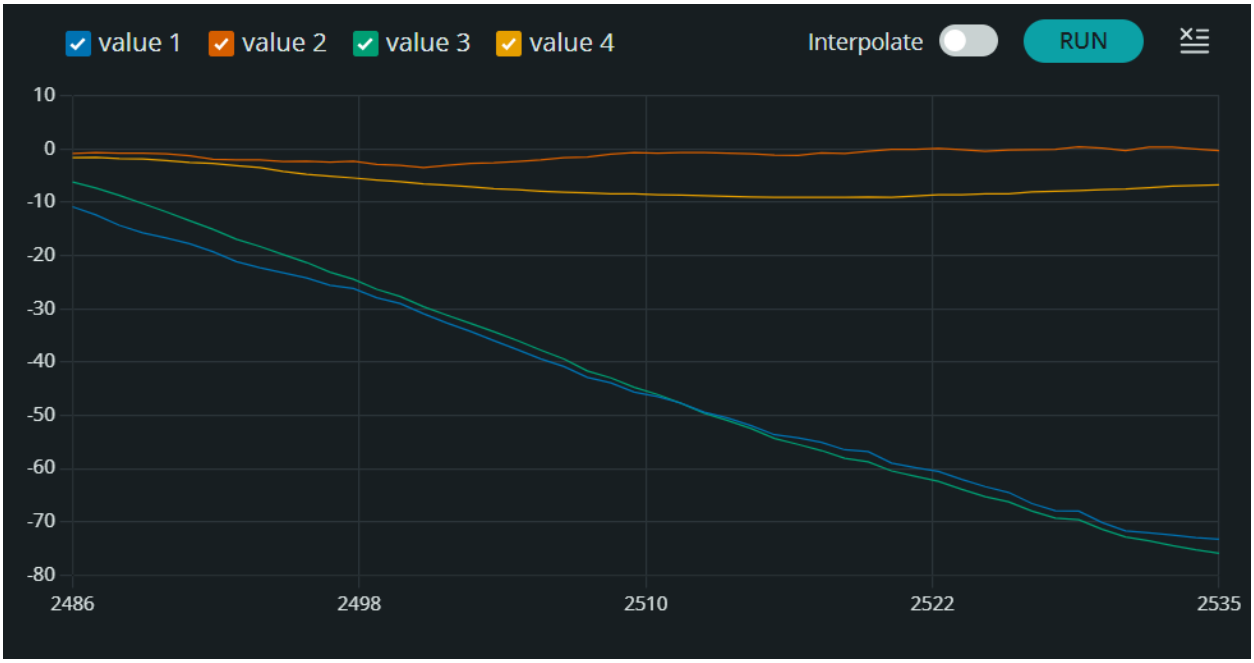


Fig 4.22 Kalman vs Complementary for a stable system

In Fig we can see the pitch value from -5° to -75°.

Value 1: is the roll value with implementing a Complementary filter.

Value 2: is the pitch value with implementing a Complementary filter.

Value 3: is the roll value with implementing a Kalman filter.

Value 4: is the pitch value with implementing a Kalman filter.

Both filters yield equivalent results.

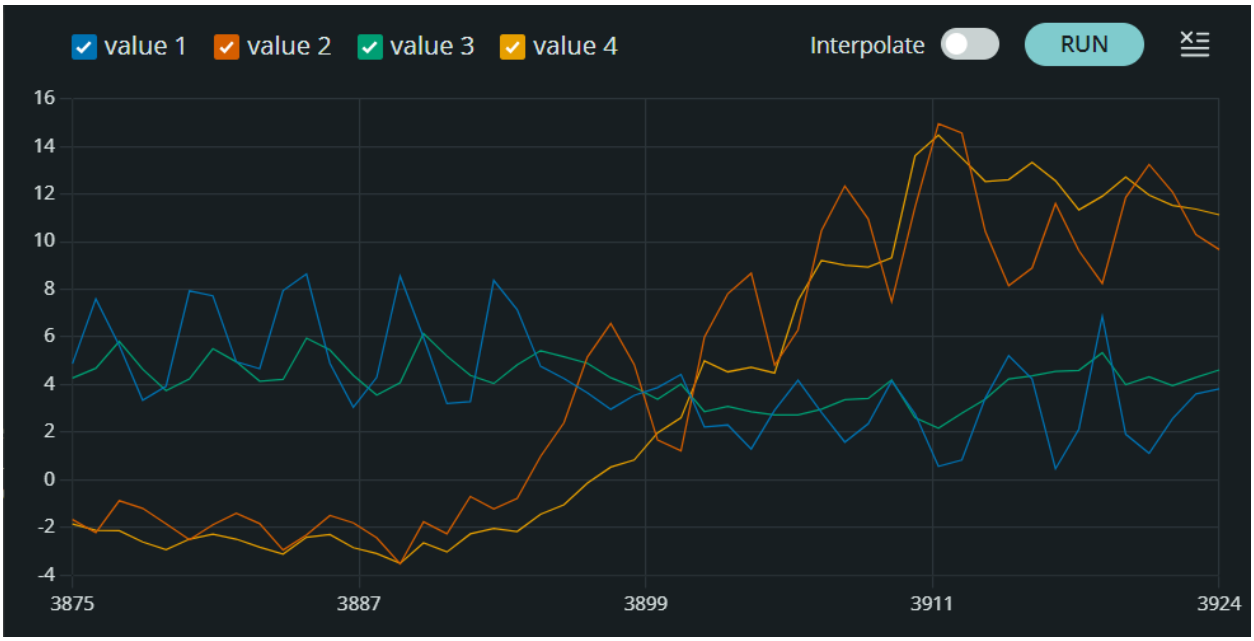


Fig 4.23 Kalman vs Complementary for noise/vibration

In Fig, the IMU will be rotated sideways along the z-axis, as a noise/vibration test.

Value 1: is the roll value with implementing a Complementary filter.

Value 2: is the pitch value with implementing a Complementary filter.

Value 3: is the roll value with implementing a Kalman filter.

Value 4: is the pitch value with implementing a Kalman filter.

The Complementary filter has more spikes than Kalman filter

Conclusion:

Both the Kalman filter and the complementary filter produce similar outcomes when applied to a stable system. However, a key distinction lies in their ability to manage disturbances: the Kalman filter demonstrates superior performance in handling noise and vibration due to its optimal recursive estimation capabilities. Consequently, while the complementary filter may suffice for simpler or more predictable environments, the Kalman filter offers enhanced robustness and accuracy in dynamic and noisy conditions, making it the preferred choice for complex real-world applications.

4.5 Gearbox results:

After fabricating the split ring gearbox, the performance outcomes exceeded expectations. The gearbox exhibited an absence of backlash, indicating a high degree of precision in gear meshing and alignment. Furthermore, there was no observable back drive, demonstrating effective mechanical locking and efficient transmission of torque without unintended reverse motion. These results underscore the success of the design and manufacturing processes in achieving a robust and reliable gearbox suitable for applications requiring precise and stable motion control. The elimination of backlash and back drive contributes significantly to the overall accuracy and responsiveness of the mechanical system.

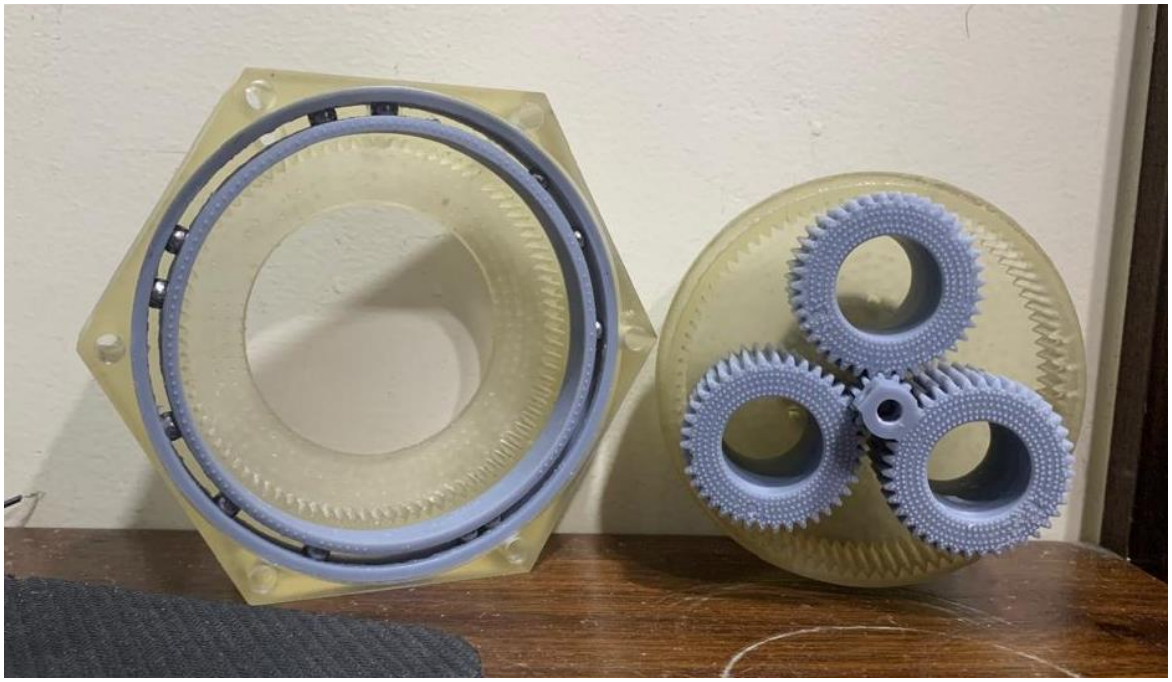


Fig 4.24 Split ring gearbox

In comparison to the planetary gearbox, the split ring gearbox demonstrates a more compact and space-efficient design. This reduced size not only facilitates easier integration into systems with limited installation space but also contributes to a lighter overall assembly. Consequently, the split ring gearbox can offer advantages in applications where minimizing volume and weight is critical, without compromising mechanical performance.

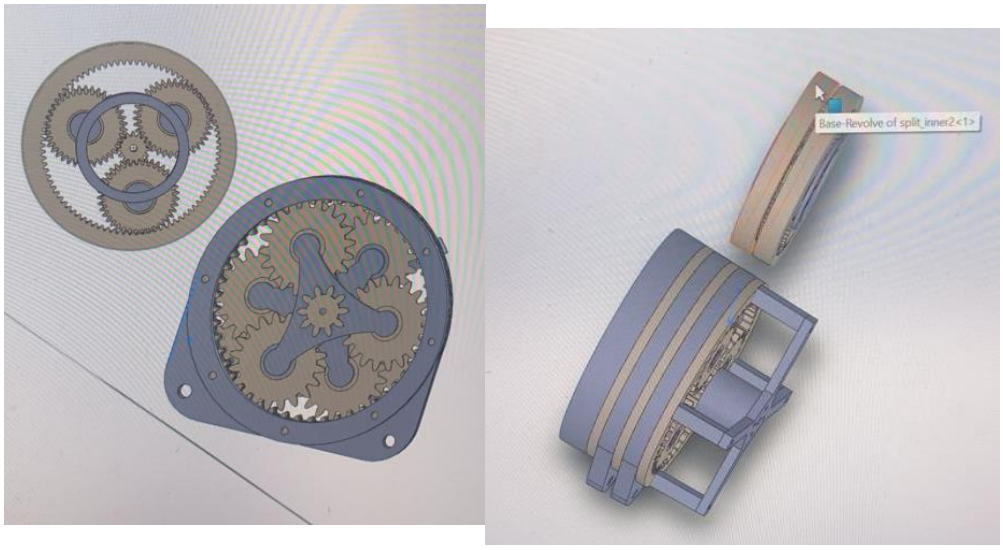


Fig 4.25 split ring gearbox vs planetary gearbox

4.6 Kinematics results:

In the results of the kinematic analysis, the use of the Euler angle orientation system proved to be suboptimal for our application. One key limitation is the occurrence of **gimbal lock**, which leads to a loss of one degree of freedom in certain configurations, making smooth and accurate motion representation difficult. To overcome this issue, the **quaternion orientation system** was adopted instead. Quaternions offer a more stable and continuous representation of orientation, avoiding singularities and enabling more reliable calculations during both forward and inverse kinematics. This change significantly improved the consistency and accuracy of the robot's motion, especially in complex or highly dynamic movements.

The homogenous matrix will be computed using:

$$M_R = \begin{pmatrix} 1-2y^2-2z^2 & 2xy-2sz & 2xz+2sy & 0 \\ 2xy+2sz & 1-2x^2-2z^2 & 2yz-2sx & 0 \\ 2xz-2sy & 2yz+2sx & 1-2x^2-2y^2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\text{where } q = s + ix + jy + kz, \quad |q| = 1$$

Fig4.26 Quaternion to Rotation Matrix

The transition to quaternions was implemented in the orientation component of the homogeneous transformation matrices, replacing the rotation matrices derived from Euler angles. This allowed for smoother interpolation between orientations and eliminated instability near singularities. As a result, the robot's motion planning and control became more robust, particularly during tasks involving large or complex joint movements. Additionally, the use of quaternions simplified the integration with modern control algorithms and simulation environments, many of which natively support quaternion-based orientation handling.

Chapter 5

Future Prospects

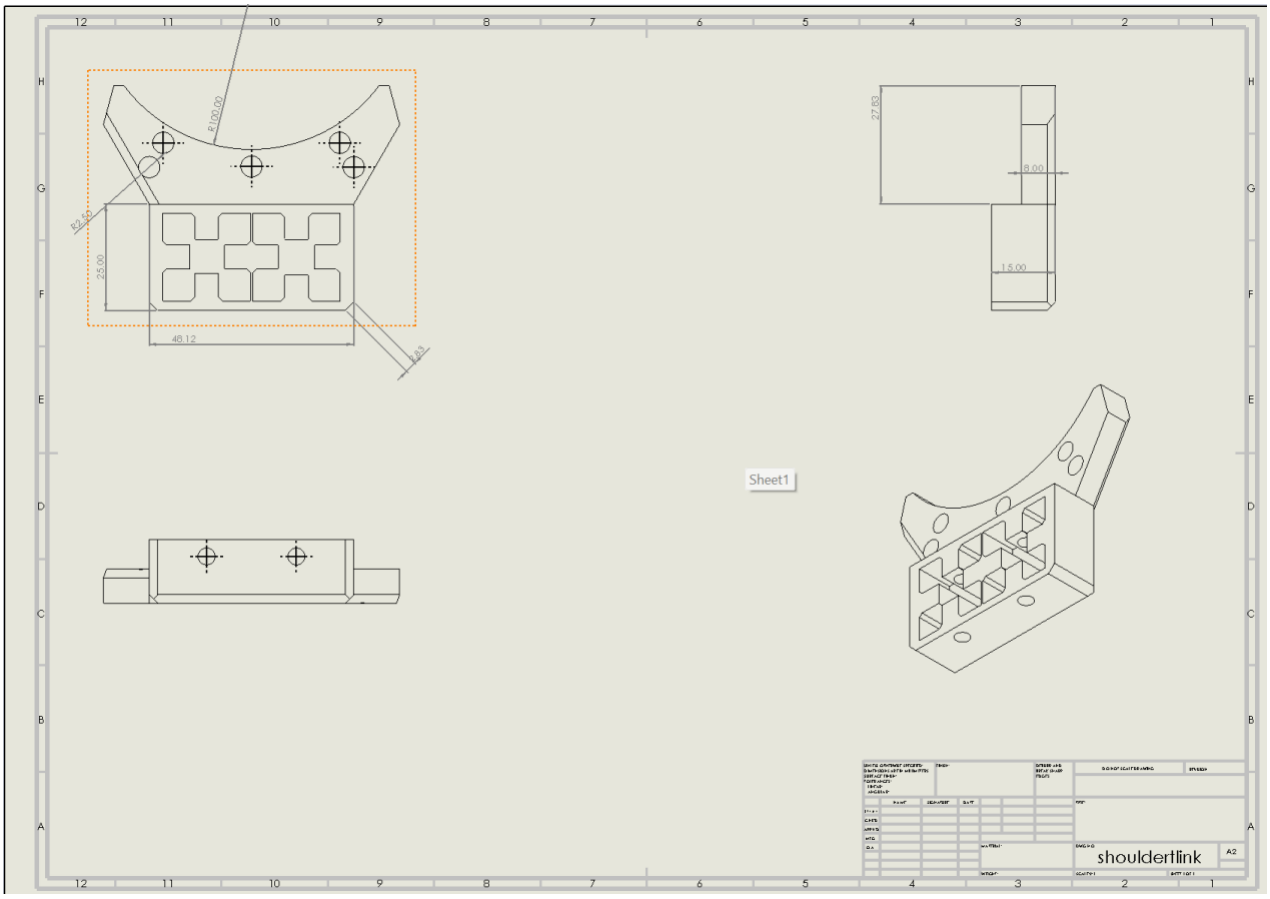
There are multiple prospects and upgrades that can be implemented and expand on to further enhance the project some of them are:

- **Substitute all the servos with BLDC motors, by implementing FOC to them.**
- **Implementing machine learning algorithms.**
- **Make the robot mobile (design lower body), integrate battery with BMS technology.**
- **Implement metal helical gears instead of plastic spur gears.**
- **Integrate more powerful SBC like nvidia's jetson.**
- **Adding Meta's artificial fingers to the robot which is called Digit360.(8)**
- **Using accurate EMG signal sensors for sensing finger movements.(10)**

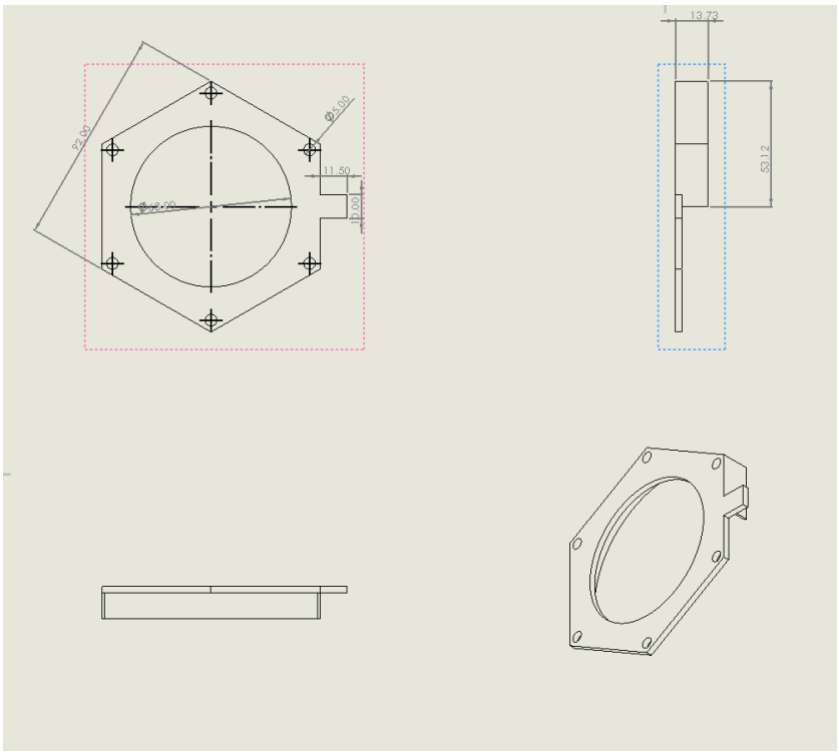
References

1. J Biomech . A Method for Defining Carpometacarpal Joint Kinematics. 2013 .NIH.USA
2. Moore, K. L., Lal, K., & Agur, A. M. R. (2013). Clinically Oriented Anatomy. Lippincott Williams & Wilkins
3. Thrun, S., Burgard, W., & Fox, D. (2005)
4. Original Kalman Filter Paper (1960)
5. Field Oriented Control (FOC) - A Deep Dive ~ Chuck Lewin
6. https://www.elegoo.com/pages/compare-products
7. https://github.com/IdeaPropulsionSystems/CoolEpicyclicGearing
8. Mike Lambeta.Digitizing Touch with an Artificial Multimodal Fingertip.(2024)
9. MatWeb - Material Property Data. -Carbon Fiber (PAN-based and pitch-based composites). -Aluminum 6061-T6:
10. Machine learning for hand pose classification from phasic and tonic EMG signals during bimanual activities in virtual reality~ Cedric Simar , Martin Colot, Ana-Maria Cebolla, Mathieu Petieau, Guy Cheron, Gianluca Bontempi.
11. Kinematic motion analysis of the human arm during a manipulation task~Andrea Maria & Paolo Rocco.

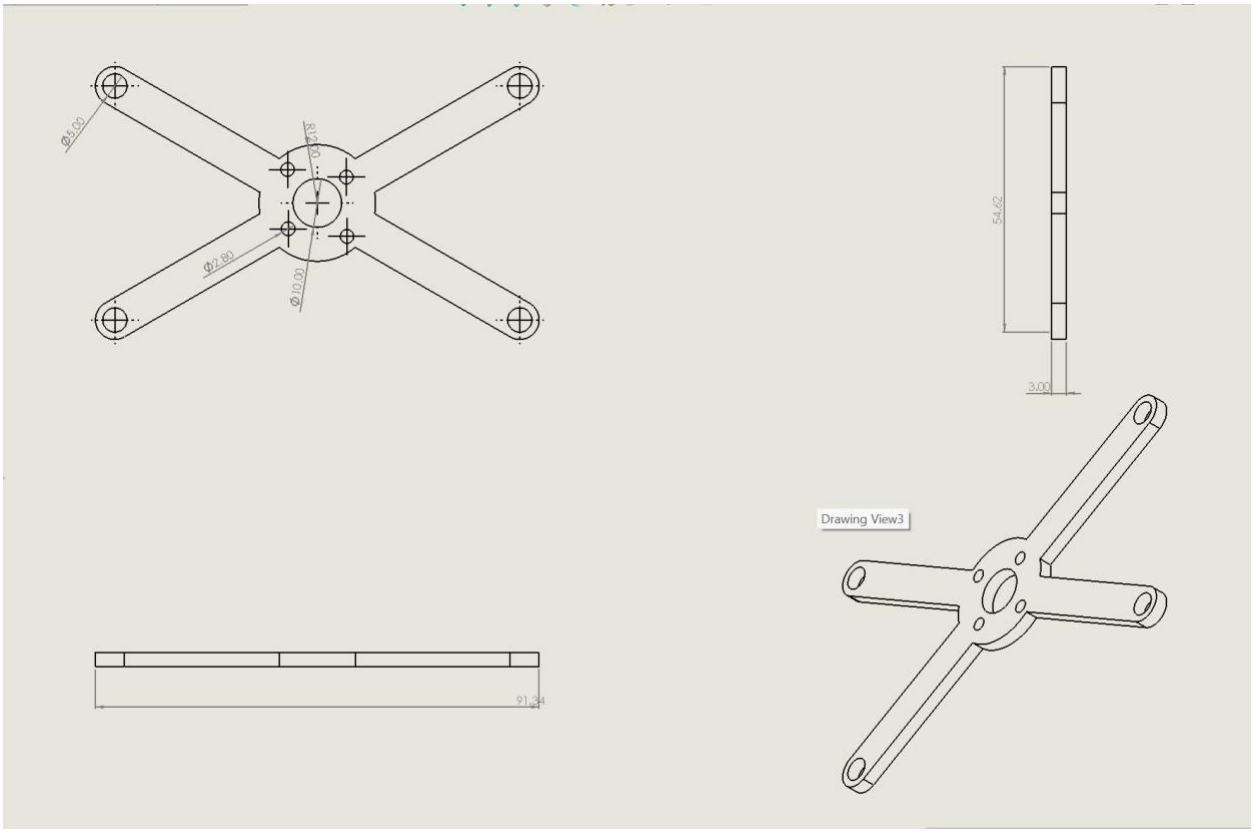
Appendix



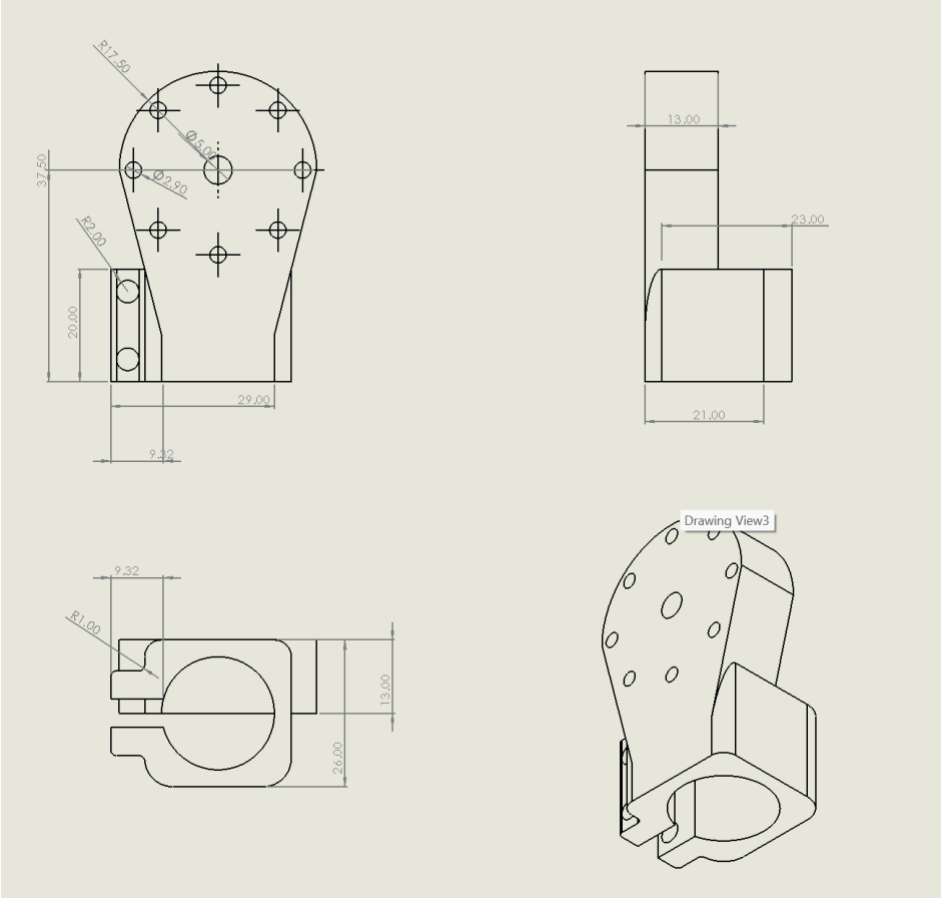
The Mount of the Gearbox on the Base



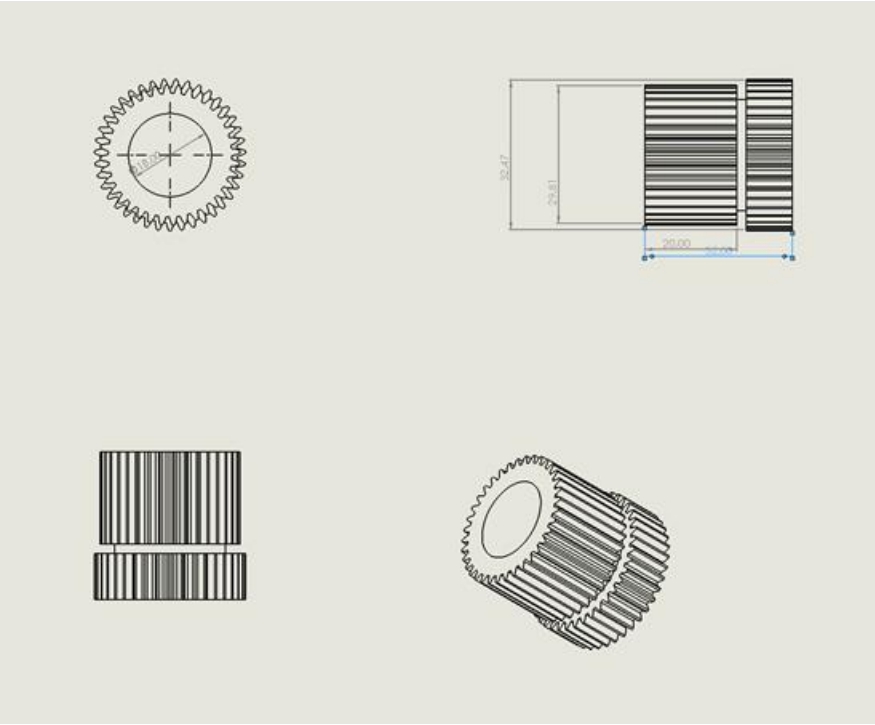
Gearbox Cover



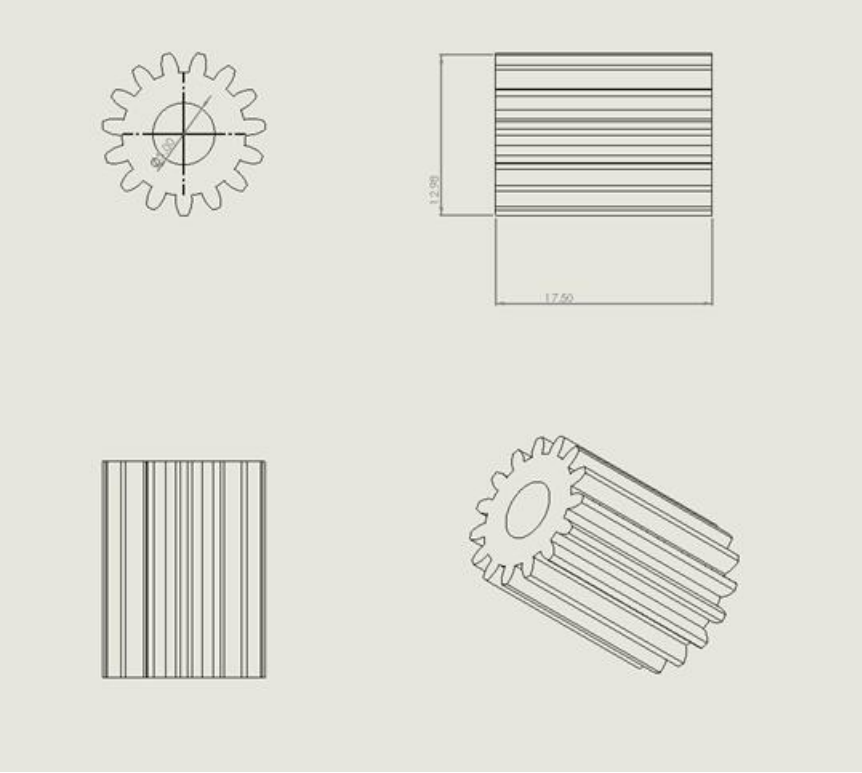
BLDC Motor Mount



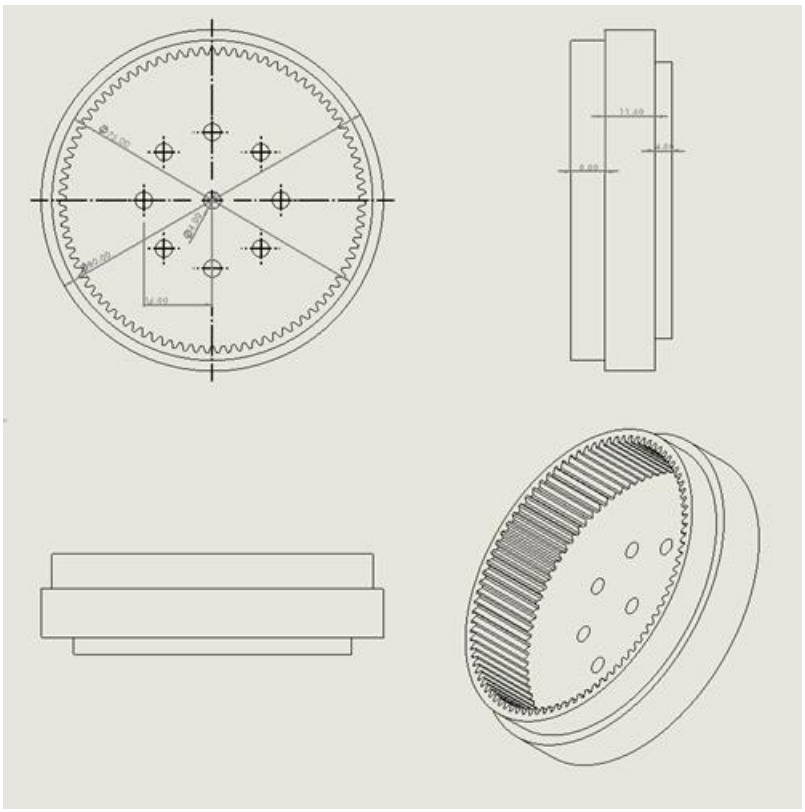
Shoulder Gearbox And Carbon Fiber Rod Link 1



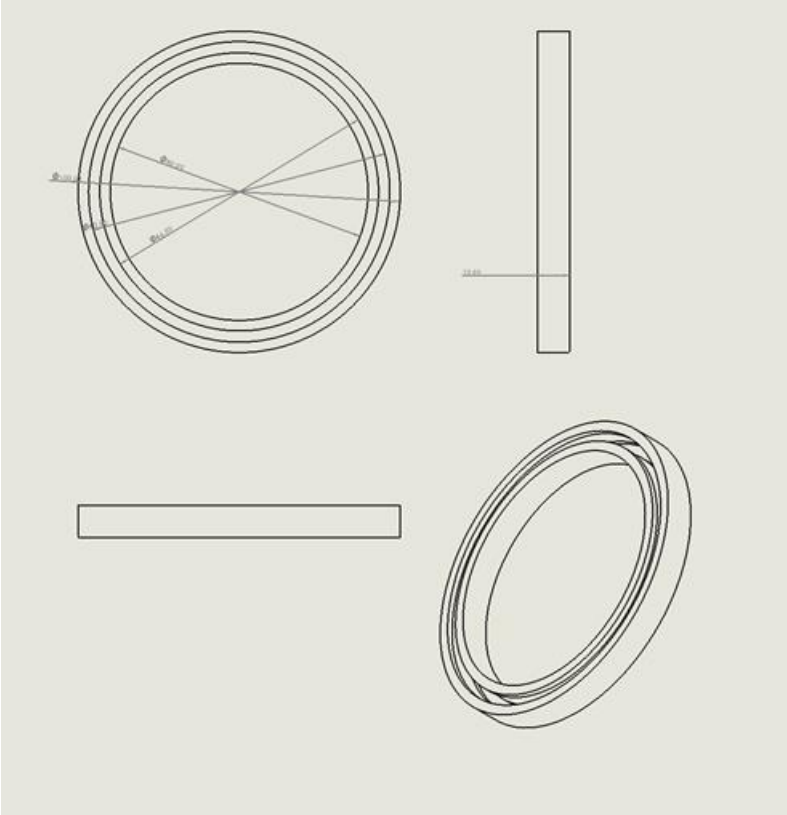
Planet Gears of Gearbox



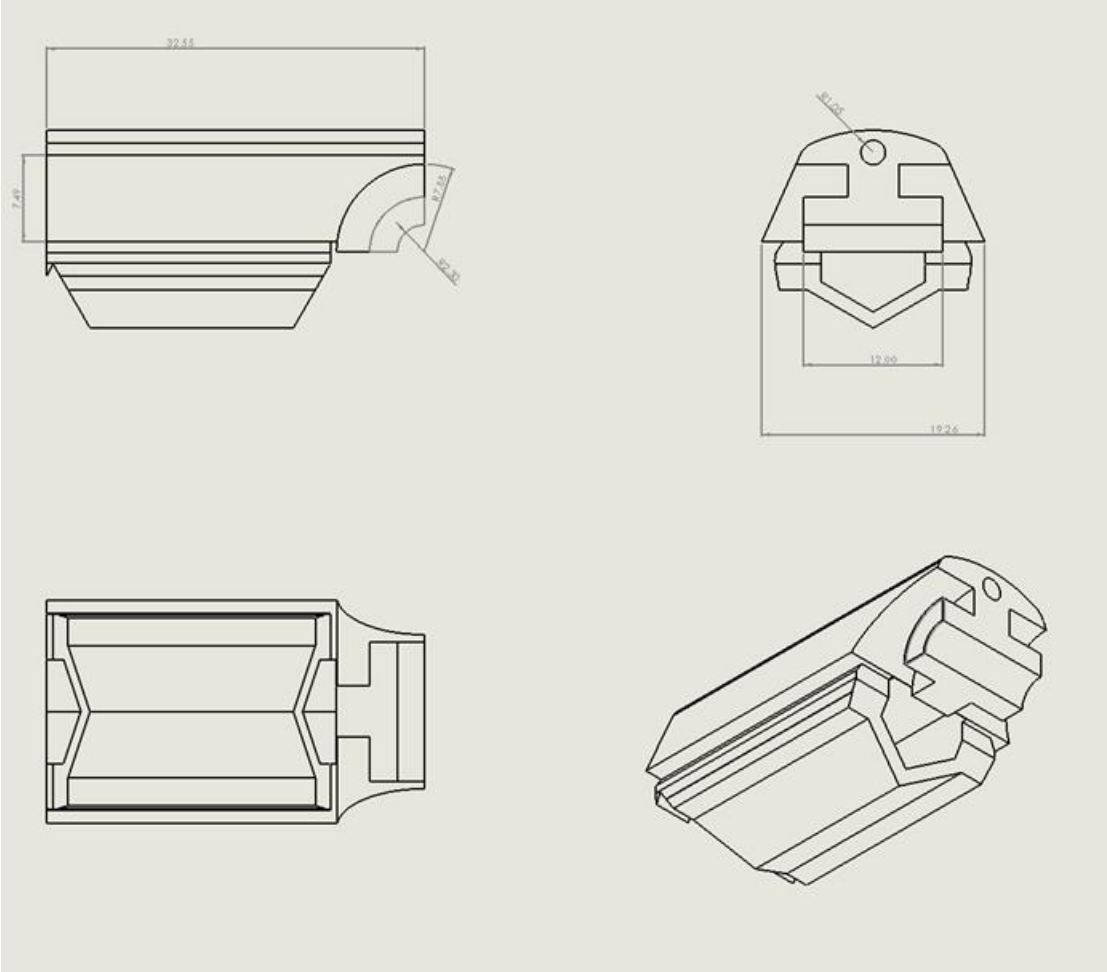
Sun Gear of Gearbox

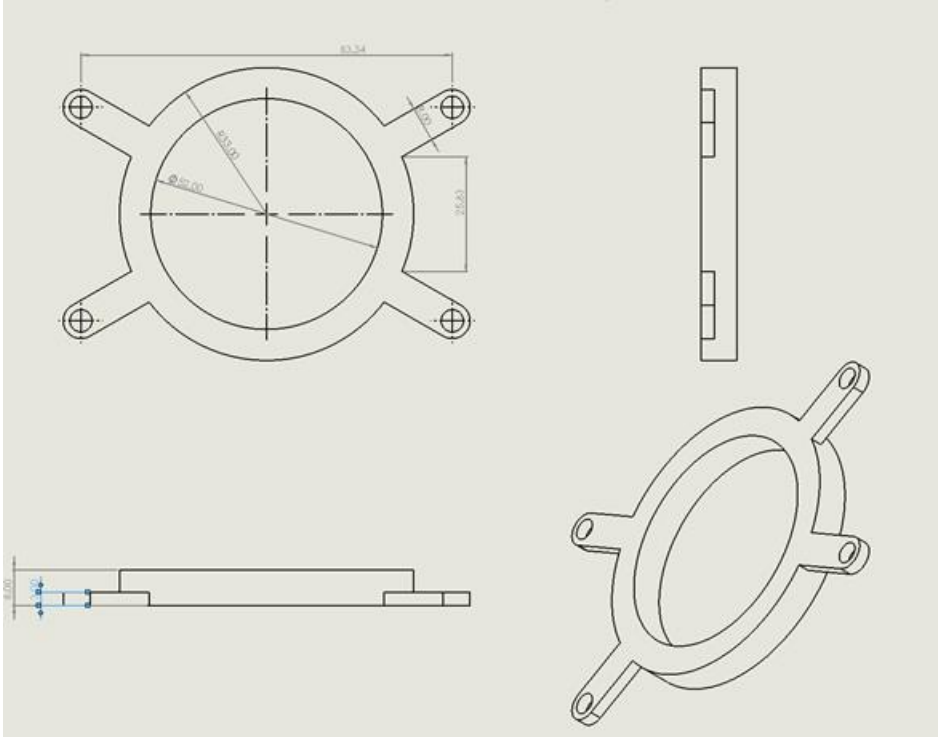


Gearbox Output Gear

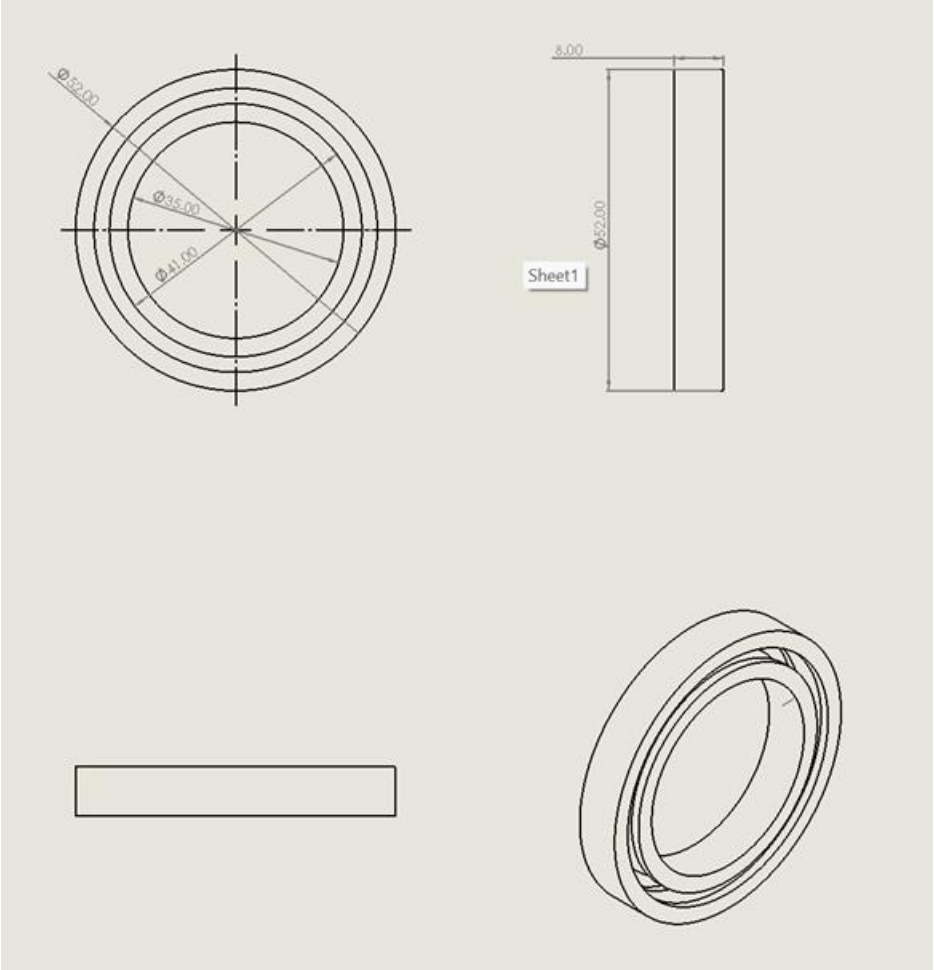


Gearbox Bearing

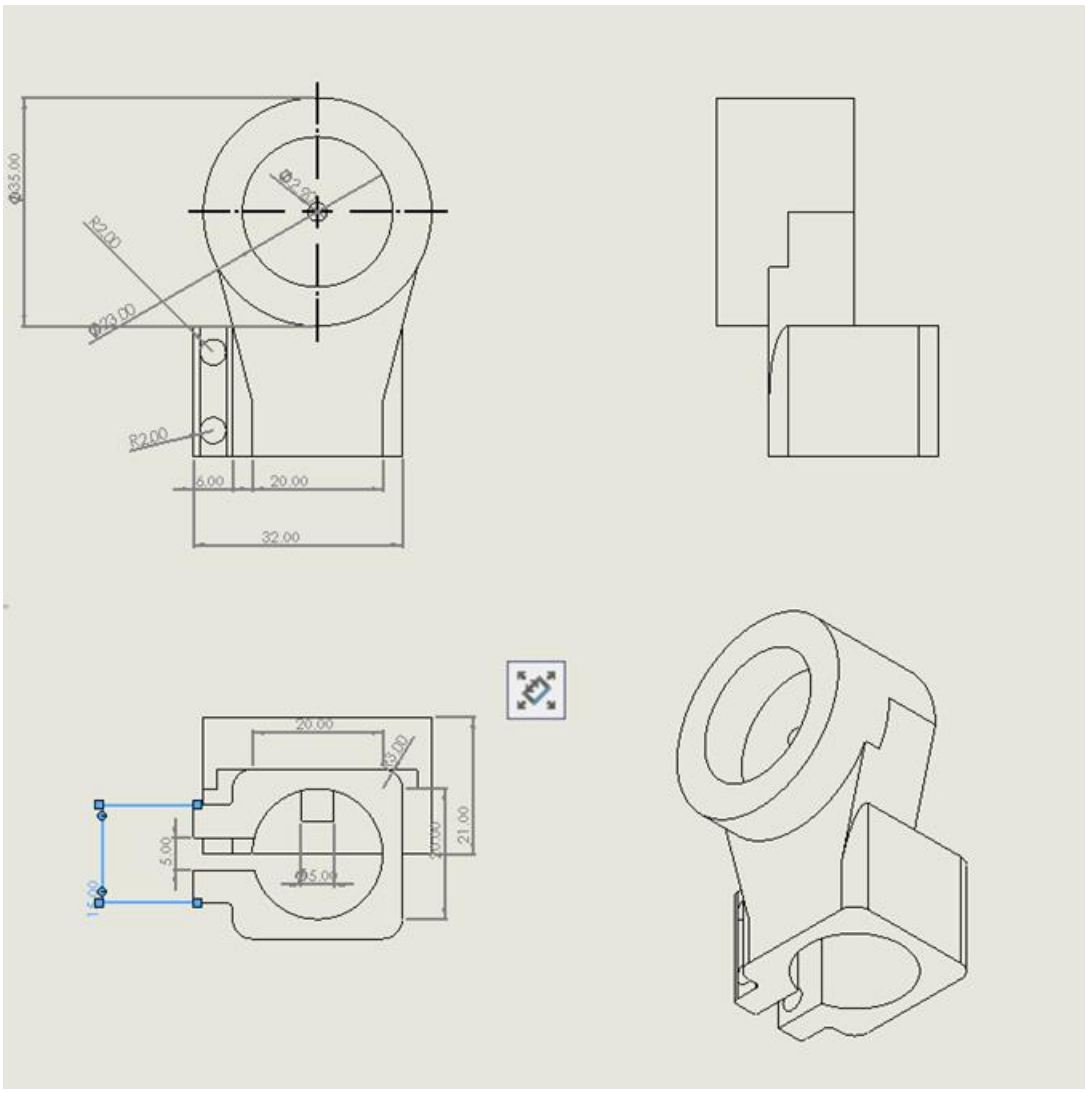

32.55
7.49
12.09
19.26



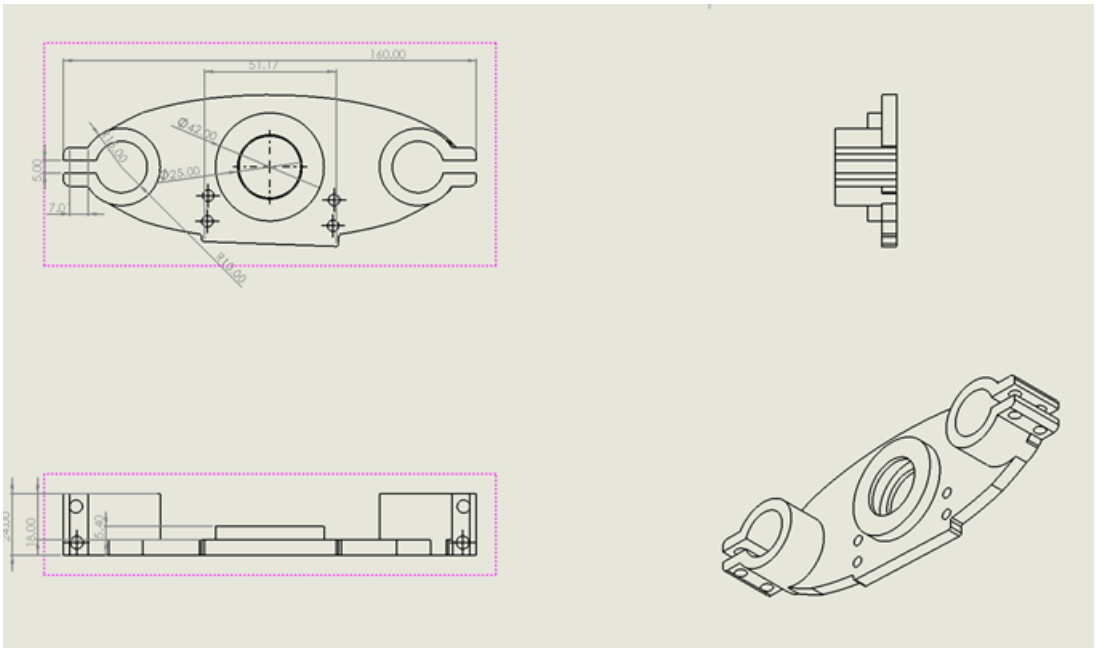
Shoulder Encoder Cover



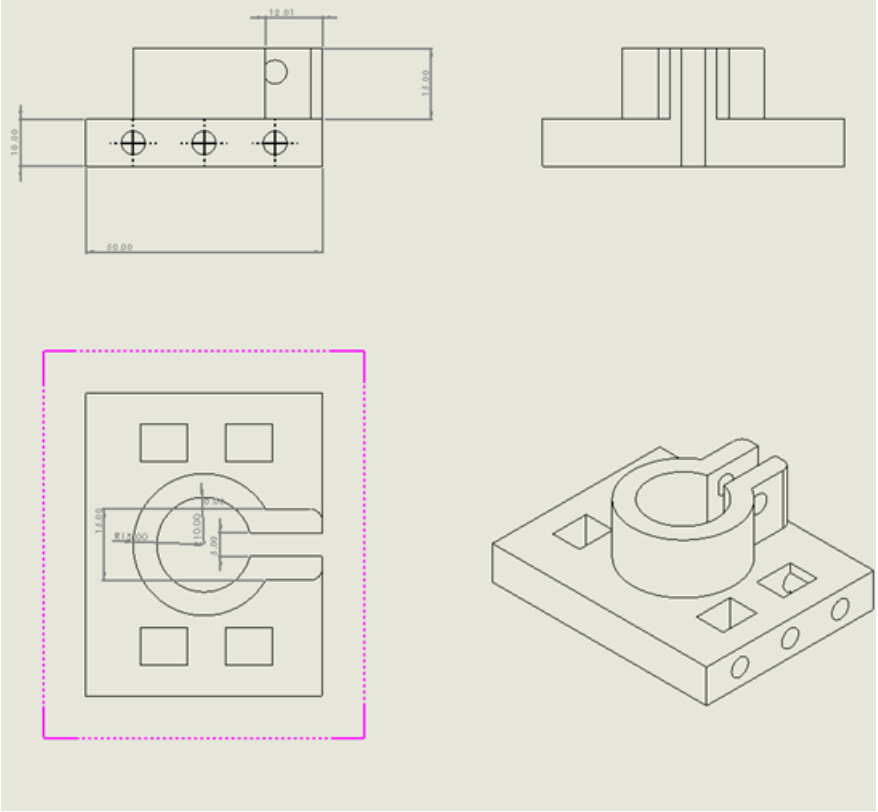
Shoulder Bearing



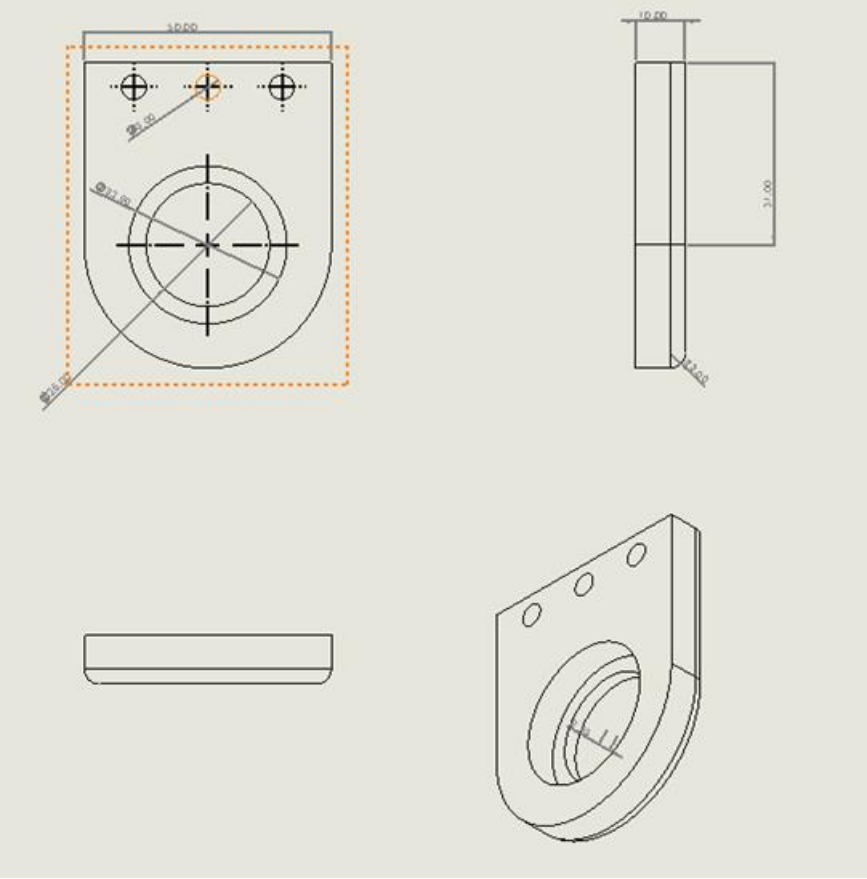
Shoulder Gearbox And Carbon Fiber Rod Link 2



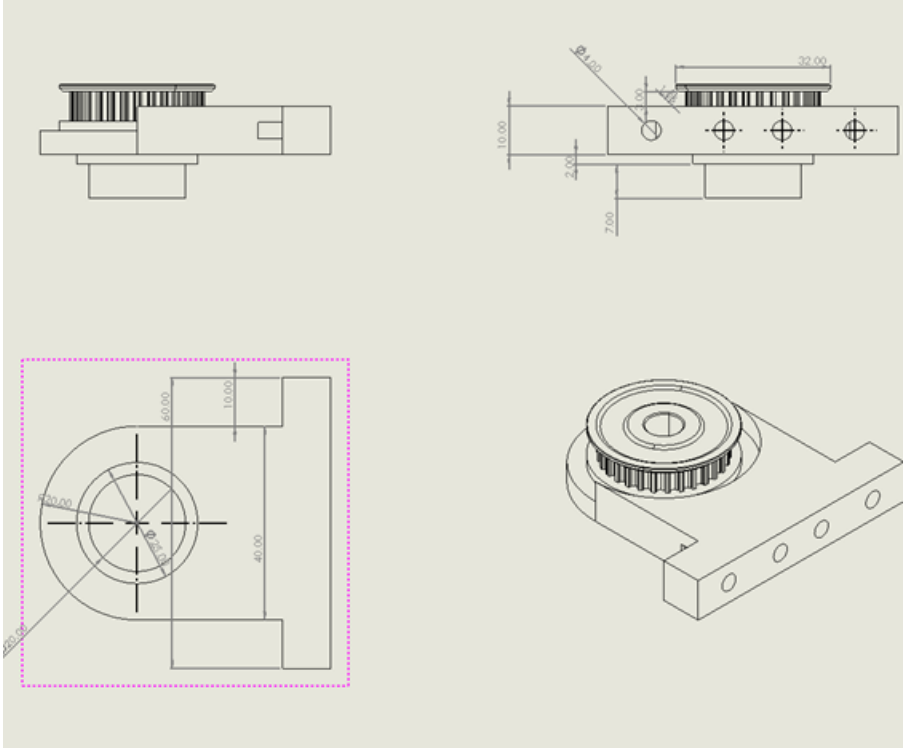
Shoulder Rods Link



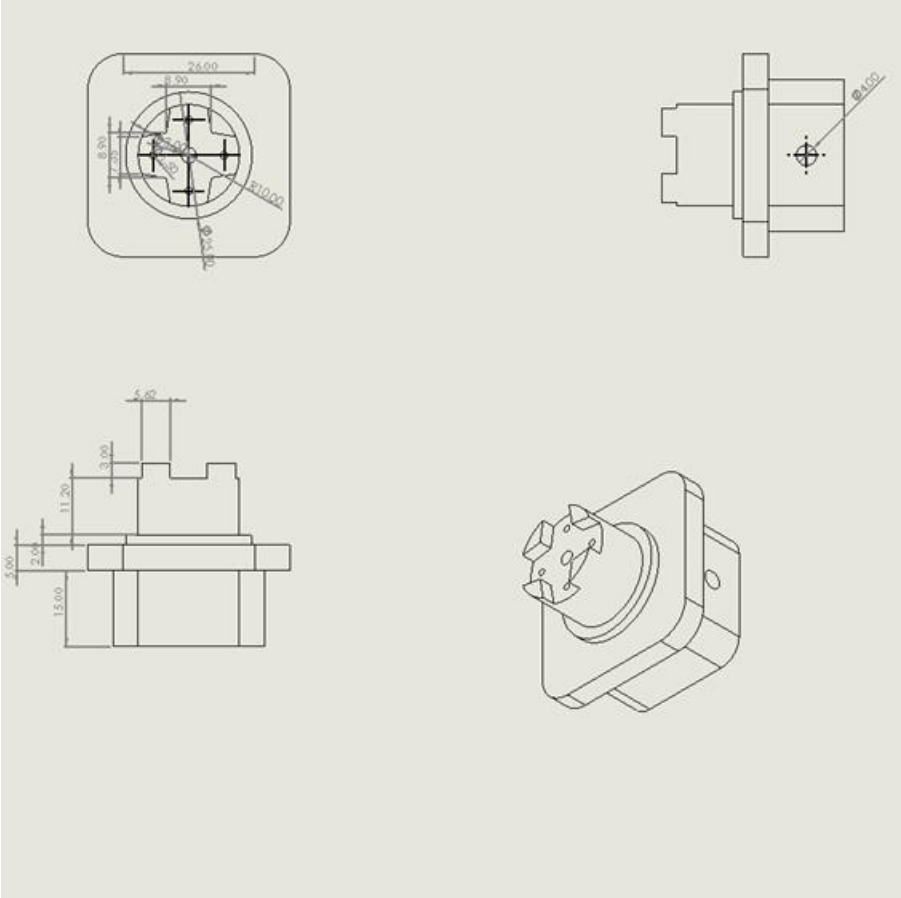
Elbow Linking plate



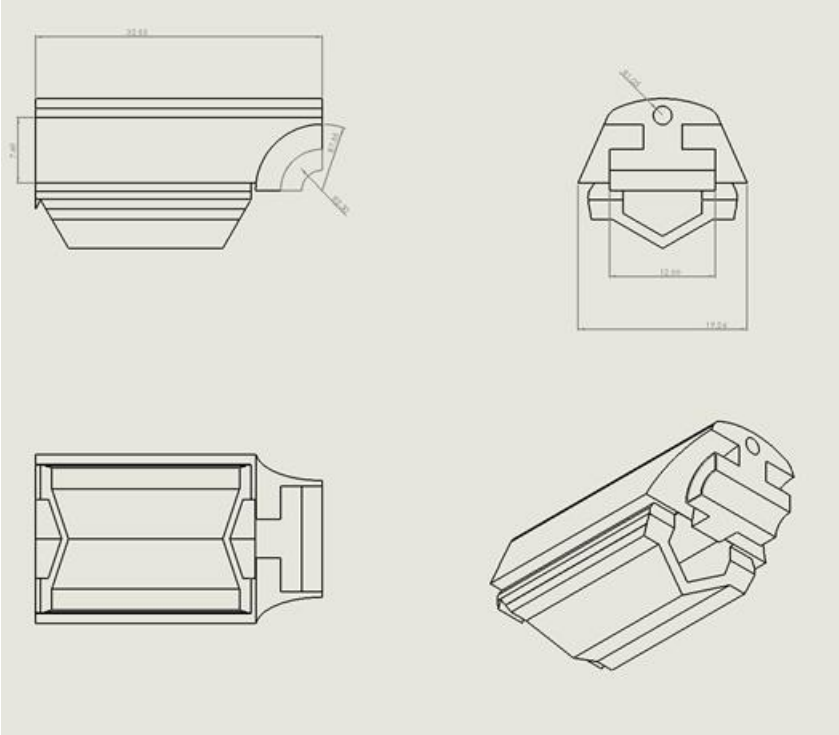
Elbow Joint 1



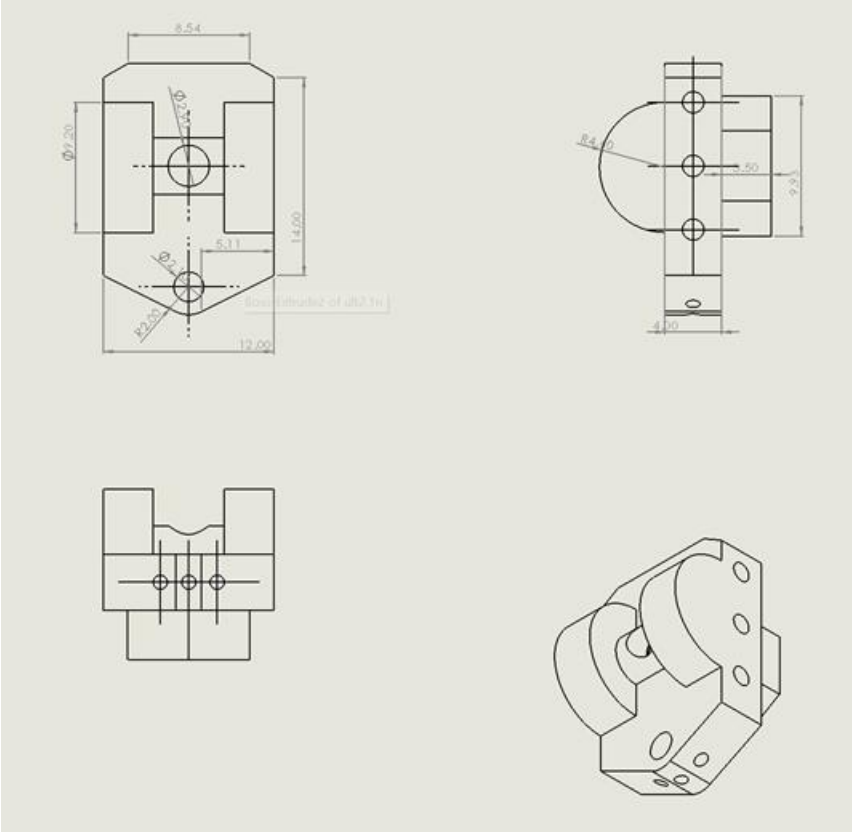
Elbow Joint 2



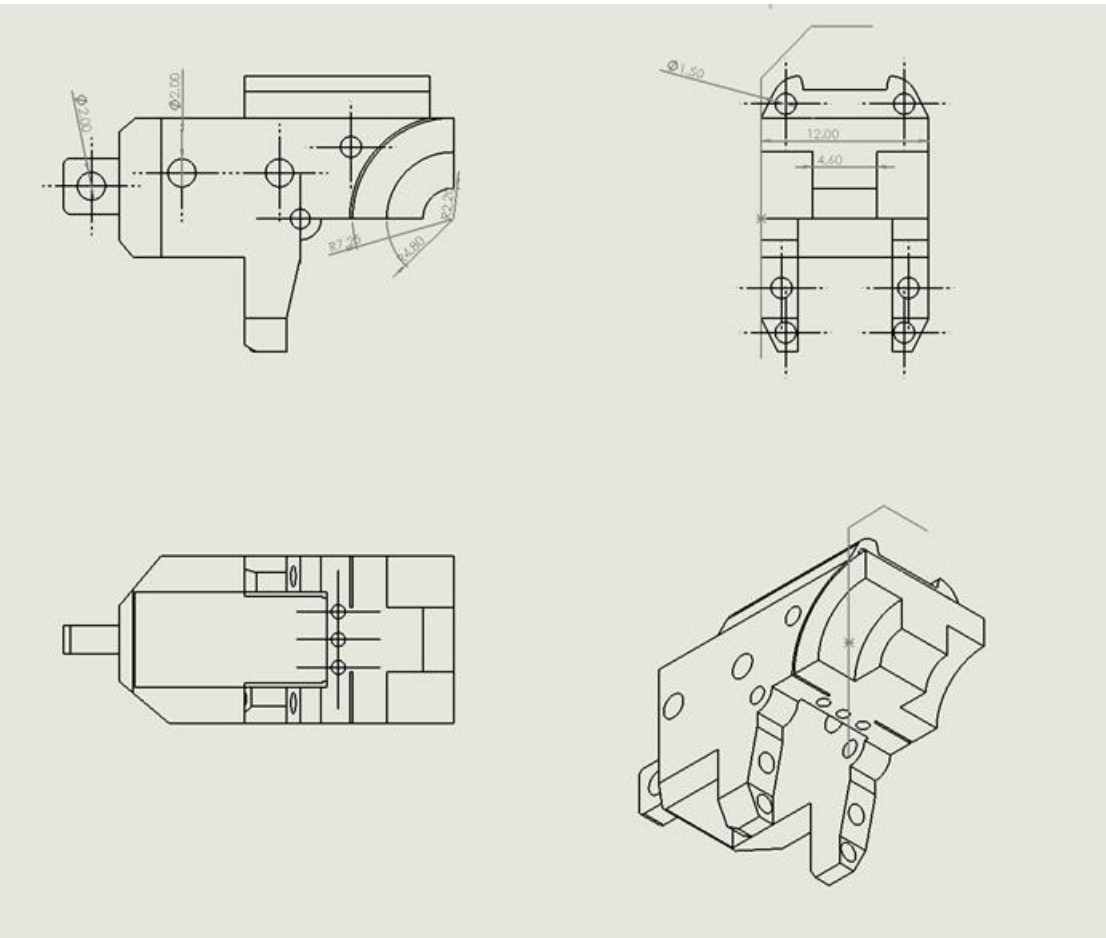
Robot Supinator



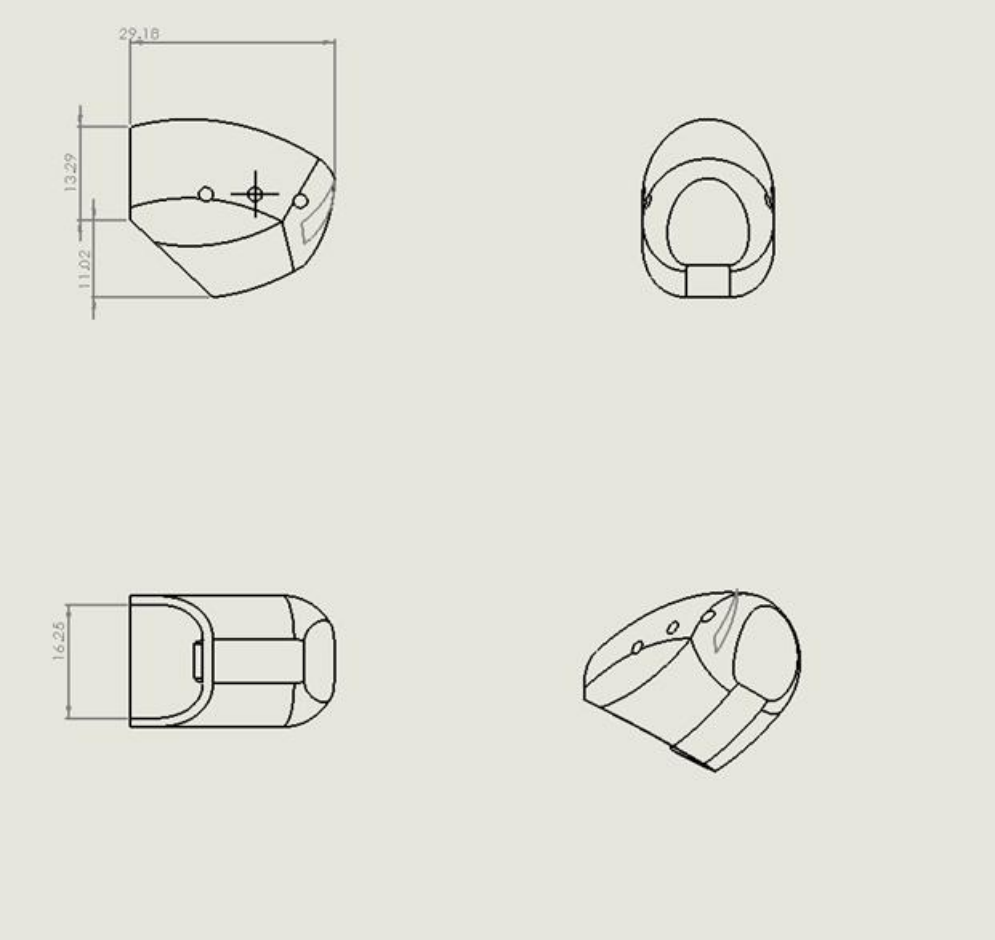
Index middle Phalanx



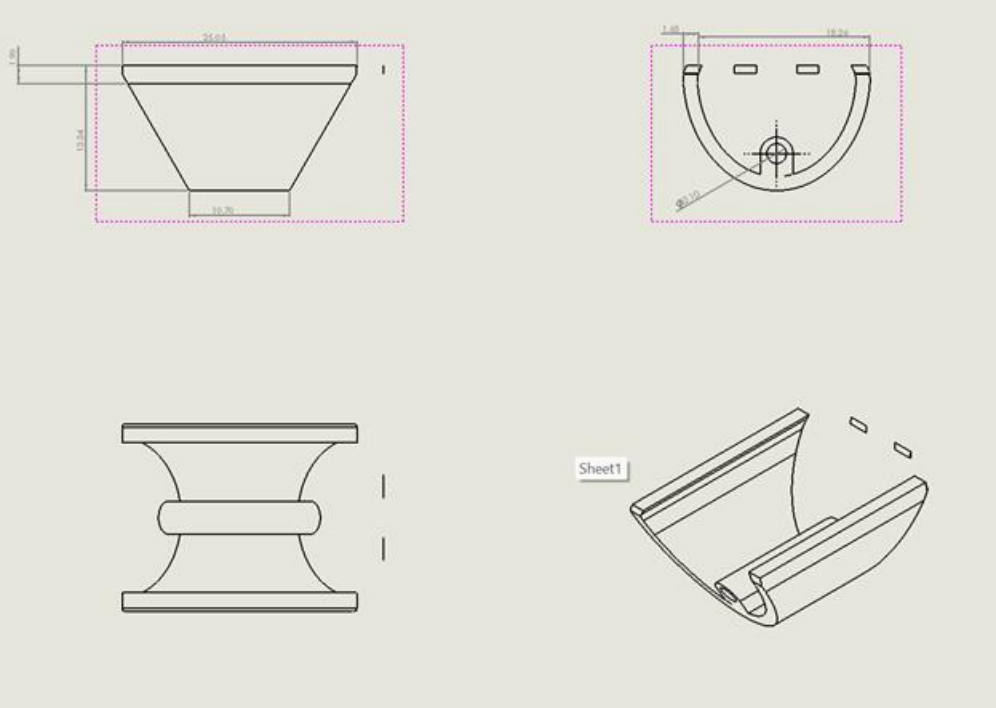
Index Proximal Phalanx



Index Distal Phalanx



Index Distal Phalanx's Rubber



Index Middle Phalanx Rubber

IMU Neck Sensing and Moving

IMU_Neck.ino

```ino
#include "FastIMU.h"
#include <Wire.h>
#include <Servo.h>

#define IMU_ADDRESS 0x68
#define IMU_ADDRESS2 0x69

#define PERFORM_CALIBRATION //Comment to disable startup calibration
MPU6500 IMU;               //Change to the name of any supported IMU!
MPU6500 IMU2;

calData calib = { 0 };  //Calibration data
AccelData accelData;
GyroData gyroData;

AccelData accelData2;
GyroData gyroData2;

//*********Servos********
Servo myservo1;
Servo myservo2;
Servo myservo3;
float t1,t2,t3;

// --- FILTER VARIABLES ---
double pitch = 0, roll = 0, com_pitch=0; // Orientation angles (degrees)
double dt = 0.02;                        // Loop time in seconds (50 Hz)

// --- KALMAN FILTER PARAMETERS ---
double Q_angle = 0.001;  // Process noise variance for the accelerometer
double Q_bias = 0.003;   // Process noise variance for the gyroscope bias
double R_measure = 0.03; // Measurement noise variance
double angle = 0, bias = 0, rate = 0; // Kalman filter state variables
```

```
double angle = 0, bias = 0, rate = 0; // Kalman filter state variables
double P[2][2] = {{0, 0}, {0, 0}};    // Error covariance matrix

unsigned long lastTime; // For calculating loop time



void setup() {
  myservo1.attach(9);
  myservo2.attach(10);
  myservo3.attach(11);

  Wire.begin();
  Wire.setClock(400000); //400khz clock
  Serial.begin(115200);
  while (!Serial) {
    ;
  }

  int err = IMU.init(calib, IMU_ADDRESS);
  if (err != 0) {
    Serial.print("Error initializing IMU: ");
    Serial.println(err);
    while (true) {
      ;
    }
  }

  int err2 = IMU2.init(calib, IMU_ADDRESS2);
  if (err2 != 0) {
```

```
    Serial.print("Error initializing IMU: ");
    Serial.println(err2);
    while (true) {
      ;
    }
  }

}

void loop() {
  IMU.update();
  IMU2.update();

  IMU.getGyro(&gyroData);
  double gx = (gyroData.gyroX)/131;
  double gy = (gyroData.gyroY)/131;

  IMU2.getGyro(&gyroData2);
  double gx2 = (gyroData2.gyroX)/131;
  double gy2 = (gyroData2.gyroY)/131;
  double gz2 = (gyroData2.gyroY)/131;

  IMU.getAccel(&accelData);
  double ax = accelData.accelX;
  double ay = accelData.accelY;
  double az = accelData.accelZ;

  IMU2.getAccel(&accelData2);
  double ax2 = accelData2.accelX;
  double ay2 = accelData2.accelY;
```

```
94
95    // --- CALCULATE TIME STEP ---
96    unsigned long currentTime = millis();
97    dt = (currentTime - lastTime) / 1000.0; // Calculate time in seconds
98    if (dt == 0) dt = 0.001;               // Prevent division by zero
99    lastTime = currentTime;
100
101   // --- KALMAN FILTER FOR PITCH AND ROLL ---
102   pitch = Kalman_filter(pitch, gx, atan2(-ax, sqrt(ay * ay + az * az)) * 180.0 / PI
103
104   roll = Kalman_filter(roll, gy, atan2(ay, az) * 180.0 / PI);
105
106   // --Complementary filter for roll on yaw--
107
108   com_pitch += gz2 * dt * 180 / PI;
109   float alpha = 0.9; // adjust this value to change the filter's responsiveness
110   com_pitch = alpha * (com_pitch + gx2 * dt) + (1 - alpha) * atan2(-ax2, sqrt(az2 *
111
112
113   Serial.print(com_pitch);
114   Serial.print(" , ");
115   //Serial.print(pitch);
116   Serial.println();
117   neck(roll,pitch);
118
119 }
120 // --- KALMAN FILTER FUNCTION ---
121 double Kalman_filter(double angle, double gyroRate, double accelAngle) {
122   // Predict
123   rate = gyroRate - bias;
124   angle += dt * rate;
125
```

```
  P[0][0] += dt * (dt * P[1][1] - P[0][1] - P[1][0] + Q_angle);
  P[0][1] -= dt * P[1][1];
  P[1][0] -= dt * P[1][1];
  P[1][1] += Q_bias * dt;

  // Update
  double S = P[0][0] + R_measure; // Estimate error
  double K[2];                    // Kalman gain
  K[0] = P[0][0] / S;
  K[1] = P[1][0] / S;

  double y = accelAngle - angle; // Angle difference
  angle += K[0] * y;
  bias += K[1] * y;

  double P00_temp = P[0][0];
  double P01_temp = P[0][1];

  P[0][0] -= K[0] * P00_temp;
  P[0][1] -= K[0] * P01_temp;
  P[1][0] -= K[1] * P00_temp;
  P[1][1] -= K[1] * P01_temp;

  return angle;
}
void neck(int x,int y)
{
t1=90+x/2;
t2=90-x/2+y/2;
t3=90-x/2-y/2;
  myservo1.write(t1);
  myservo2.write(t2);                  // sets the servo position according to the s
```

```
  myservo1.write(t1);
  myservo2.write(t2);                  // sets the servo position according to the s
  myservo3.write(t3);

}
```

Arduino Nano Code which is for controlling shoulder

NANO.ino

```
#include <Wire.h>
#include<Servo.h>
//Arduino: SDA: A4 SCL: A5
Servo ESC ;
#define relay_pin 5

#define NANO_ADD 0x69
String data;
int angle=0;
double error = 0;

//**********Magnetic sensor things***************
int magnetStatus = 0; //value of the status register (MD, ML, MH)

int lowbyte; //raw angle 7:0
word highbyte; //raw angle 7:0 and 11:8
int rawAngle; //final raw angle
float degAngle; //raw angle in degrees (360/4096 * [value between 0-4095])

int quadrantNumber, previousquadrantNumber; //quadrant IDs
float numberofTurns = 0; //number of turns
float correctedAngle = 0; //tared angle - based on the startup value
float startAngle = 0; //starting angle
float totalAngle = 0; //total absolute angular displacement
float previoustotalAngle = 0; //for the display printing
/*********************************************/

//********for PID******//
double dt, last_time;
double integral, previous, output = 0;
double kp, ki, kd; // PID gains
double setpoint = 10; //the target for PID
/*********************************************/

```

```
34
35  void setup() {
36    ESC.attach (9,1000,2000);
37    ESC.write(0); // initialising the BLDC
38    delay(5000); // Wait for intit to finish
39
40    pinMode(relay_pin,OUTPUT); // For controlling the direction of the motor
41
42
43    Serial.begin(115200);
44    Wire.onReceive(onReceive);
45    Wire.begin((uint8_t)NANO_ADD);
46
47   Serial.println("setup began!");
48
49
50    Serial.println("Wire began!");
51    //Wire.setClock(800000L); //fast clock
52    Serial.println("checking for magnet");
53    checkMagnetPresence(); //check the magnet (blocks until magnet is found)
54
55    ReadRawAngle(); //make a reading so the degAngle gets updated
56    startAngle = degAngle; //update startAngle with degAngle - for taring
57    Serial.println(startAngle);
58    delay(500);
59    kp = 0.5; //p=0.5
60    ki = 0.1; //i=0.2
61    kd = 0;
62    last_time = 0;
63
64  }
65
66  void loop() {
67  double now = millis();
68    dt = (now - last_time)/1000.00; // 1000 to convert time from msec to sec
```

```
68   dt = (now - last_time)/1000.00; // 1000 to convert time from msec to sec
69   last_time = now; // updating the last time
70
71
72   ReadRawAngle(); //ask the value from the sensor
73   correctAngle(); //tare the value , the value of it is stored in : correctedAngle
74   checkQuadrant(); //check quadrant, check rotations, check absolute angular positio
75    error = setpoint - totalAngle;
76
77        if (setpoint < error )
78       {
79        CLKwise();
80        setpoint = angle;
81       }
82     else if (setpoint > error)
83       {
84         countCLKwise();
85        setpoint = angle;
86       }
87
88
89   double output = pid(error);
90
91   // Make sure the value is within the expected range (0 to setpoint)
92   output = constrain(output, 0, setpoint); // Constrain the speed to be between 0 an
93
94   // Map the speed (0-setpoint) to the PWM range (1000-2000 microseconds)
95   double pwmValue = map(output, 0, setpoint, 1473, 1518); // Map speed to PWM range
96
97   ESC.writeMicroseconds(pwmValue);
98
99   Serial.print(setpoint);
100  Serial.print(",");
101  Serial.print(totalAngle);
```

NANO.ino

```
  Serial.print(totalAngle);
  Serial.print(",");
  Serial.println(pwmValue);
  delay(50);

  /*if (error < 20 &&  -20 < error )
    while(1)
    {
      Serial.println("motor reached it's target....");
      ESC.writeMicroseconds(1000);
      if (Serial.available() > 0)
        break;
      delay(1000);
    }*/

}

void onReceive(int len) {

data = ""; // clear previous data

while (Wire.available()) {
  char c = Wire.read();
  data += c;  // build the full string
}

Serial.println(data);
angle =handle(data);


}

int handle(String data) {
  if (data.startsWith("#W")) {
    String numberPart = data.substring(2);
```

```
    String numberPart = data.substring(2);
    if (numberPart.length() == 0) return -1;
    int value = numberPart.toInt();
    return value;
  }
  return -1;
}

void countCLKwise()
{
  digitalWrite(relay_pin,HIGH);

}
void CLKwise()
{
  digitalWrite(relay_pin,LOW);

}
double pid(double error)
{
  double proportional = error;
  integral += error * dt;
  double derivative = (error - previous) / dt;
  previous = error;
  double output = (kp * proportional) + (ki * integral) + (kd * derivative);
  return output;
}
void ReadRawAngle()
{
  ///Serial.println("Reading raw angle...");
  //7:0 - bits
  Wire.beginTransmission(0x36); //connect to the sensor
  Wire.write(0x0D); //figure 21 - register map: Raw angle (7:0)
```

```
  Wire.endTransmission(); //end transmission
  Wire.requestFrom(0x36, 1); //request from the sensor

  while(Wire.available() == 0); //wait until it becomes available
  lowbyte = Wire.read(); //Reading the data after the request

  //11:8 - 4 bits
  Wire.beginTransmission(0x36);
  Wire.write(0x0C); //figure 21 - register map: Raw angle (11:8)
  Wire.endTransmission();
  Wire.requestFrom(0x36, 1);

  while(Wire.available() == 0);
  highbyte = Wire.read();

  //4 bits have to be shifted to its proper place as we want to build a 12-bit numb
  highbyte = highbyte << 8; //shifting to left
  //What is happening here is the following: The variable is being shifted by 8 bit
  //Initial value: 00000000|00001111 (word = 16 bits or 2 bytes)
  //Left shifting by eight bits: 00001111|00000000 so, the high byte is filled in

  //Finally, we combine (bitwise OR) the two numbers:
  //High: 00001111|00000000
  //Low:  00000000|00001111
  //      ----------------
  //H|L:  00001111|00001111
  rawAngle = highbyte | lowbyte; //int is 16 bits (as well as the word)

  //We need to calculate the angle:
  //12 bit -> 4096 different levels: 360° is divided into 4096 equal parts:
  //360/4096 = 0.087890625
  //Multiply the output of the encoder with 0.087890625
  degAngle = rawAngle * 0.087890625;
```

```
201
202     /*Serial.print("Deg angle: ");
203     Serial.println(degAngle, 2); //absolute position of the encoder within the 0-360
204
205   }
206   void correctAngle()
207   {
208     //recalculate angle
209     correctedAngle = degAngle - startAngle; //this tares the position
210
211     if(correctedAngle < 0) //if the calculated angle is negative, we need to "normal
212     {
213     correctedAngle = correctedAngle + 360; //correction for negative numbers (i.e. -
214     }
215     else
216     {
217       //do nothing
218     }
219     ///Serial.print("Corrected angle: ");
220     ///Serial.println(correctedAngle, 2); //print the corrected/tared angle
221   }
222
223   void checkQuadrant()
224   {
225     /*
226     //Quadrants:
227     4  |  1
228     ---|---
229     3  |  2
230     */
231
232     //Quadrant 1
233     if(correctedAngle >= 0 && correctedAngle <=90)
234     {
```

```
244   //Quadrant 3
245   if(correctedAngle > 180 && correctedAngle <=270)
246   {
247     quadrantNumber = 3;
248   }
249
250   //Quadrant 4
251   if(correctedAngle > 270 && correctedAngle <360)
252   {
253     quadrantNumber = 4;
254   }
255   //Serial.print("Quadrant: ");
256   //Serial.println(quadrantNumber); //print our position "quadrant-wise"
257
258   if(quadrantNumber != previousquadrantNumber) //if we changed quadrant
259   {
260     if(quadrantNumber == 1 && previousquadrantNumber == 4)
261     {
262       numberofTurns++; // 4 --> 1 transition: CW rotation
263     }
264
265     if(quadrantNumber == 4 && previousquadrantNumber == 1)
266     {
267       numberofTurns--; // 1 --> 4 transition: CCW rotation
268     }
269     //this could be done between every quadrants so one can count every 1/4th of tr
270
271     previousquadrantNumber = quadrantNumber;  //update to the current quadrant
272
273   }
274   //Serial.print("Turns: ");
```

```
283  void checkMagnetPresence()
284  {
285    //This function runs in the setup() and it locks the MCU until the magnet is not p
286
287    while((magnetStatus & 32) != 32) //while the magnet is not adjusted to the proper
288    {
289      magnetStatus = 0; //reset reading
290
291      Wire.beginTransmission(0x36); //connect to the sensor
292      Wire.write(0x0B); //figure 21 - register map: Status: MD ML MH
293      Wire.endTransmission(); //end transmission
294      Wire.requestFrom(0x36, 1); //request from the sensor
295
296      while(Wire.available() == 0); //wait until it becomes available
297      magnetStatus = Wire.read(); //Reading the data after the request
298
299      Serial.print("Magnet status: ");
300      Serial.println(magnetStatus, BIN); //print it in binary so you can compare it to
301      delay(1000);
302    }
303
304    //Status register output: 0 0 MD ML MH 0 0 0
305    //MH: Too strong magnet - 100111 - DEC: 39
306    //ML: Too weak magnet - 10111 - DEC: 23
307    //MD: OK magnet - 110111 - DEC: 55
308
309    Serial.println("Magnet found!");
310    //delay(1000);
311  }
```

ESP-32 motion capture Receiver

Reseve_data.ino

```ino
#include <esp_now.h>
#include <WiFi.h>
#include <ESP32Servo.h>
#include <Wire.h>

#define bluepin 17
#define MG99pin 16
#define NANO_ADD 0x69

String data;
Servo blue;  //for the 45kg servo
Servo MG99; //for mg99 servo

void setup() {

  Serial.begin(115200);
  Wire.begin();

  blue.attach(bluepin);
  MG99.attach(MG99pin);

  ESPnow();
}

void loop() {
  // put your main code here, to run repeatedly:

}
```

```
29
30  void OnDataRecv(const esp_now_recv_info* recvInfo, const uint8_t *incomingData, in
31      memcpy(&data, incomingData, sizeof(data));
32     // Serial.print("Data received: ");
33     // Serial.println(len);
34     Serial.println(data);
35     Serial.println();
36     handle(data);
37   }
38
39   void handle(String data)
40   {
41     if (data.startsWith("#A"))
42     {
43       String numberPart = data.substring(2); // Skip '#' and 'A'
44       int value = numberPart.toInt()*1.6;
45       blue.write(value);
46
47     }else if ((data.startsWith("#B")))
48     {
49       String numberPart = data.substring(2);
50       int value = numberPart.toInt();
51       MG99.write(value);
52     }else if ((data.startsWith("#W")))
53     {
54       Wire.beginTransmission(NANO_ADD);
```

```
55       Wire.print(data);
56       uint8_t error = Wire.endTransmission(true);
57     }
58   }
59
60   void ESPnow()
61   {
62       WiFi.mode(WIFI_STA);
63
64     if(esp_now_init() != ESP_OK) {
65       Serial.println("Error initializing ESP-NOW");
66       return;
67     }
68     esp_now_register_recv_cb(OnDataRecv);
69   }
70
```

ESP-32 motion capture Sender

```
#include <esp_now.h>
#include <WiFi.h>
#include <Arduino.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>
#include <math.h>

#define CHANNEL 1 //the same channel should be on both esps

String startByte = "#";
String destROLL[3] = {"A","D", "G"}; // Destination bytes for roll
String destPITCH[3] = {"B","E","H"}; // Destination bytes for pitch
String destYAW[3] = {"C","F","I"}; // Destination bytes for yaw

/* Set the delay between fresh samples */
uint16_t BNO055_SAMPLERATE_DELAY_MS = 100;
Adafruit_BNO055 bno; // Declare an array of Adafruit_BNO055 objects

// MAC Address of responder - edit as required
uint8_t broadcastAddress[] = {0x9C, 0x9C, 0x1F, 0xC7, 0x69, 0xEC}; //For the Dev ki
String data;
esp_now_peer_info_t peerInfo;// Peer info
```

```
void setup() {
  Serial.begin(115200); // Set serial baud rate
  Serial.println("Setup started");

  ESPnow();

  while (!Serial) delay(10);  // wait for serial port to open!

   bno = Adafruit_BNO055(55, 0x29, &Wire);
    Serial.print("Initializing BNO055 ");
    if (!bno.begin())/* Initialise the sensor */
    {
    /* There was a problem detecting the BNO055 ... check your connections */
    Serial.print("Ooops, no BNO055 detected ... Check your wiring or I2C ADDR!");
    while (1);
    }
    bno.setExtCrystalUse(true);

}

esp_err_t result;
```

```
void loop() {
  /* Get a new sensor event */
  sensors_event_t event;
  bno.getEvent(&event);

  float yaw =  (float)event.orientation.x;
  if (yaw < 0) {
    yaw = -yaw;
   } else if (yaw > 180) {
    yaw -= 360;
  }

  float pitch = (float)event.orientation.y;


  float roll = (float)event.orientation.z;

  Serial.println(Packet(startByte,pitch,"A"));
  result = esp_now_send(broadcastAddress , (uint8_t*) &data, sizeof(data));
  delay(100);

}

String Packet(String startBy, float VAL, String dest) {
```

```
String Packet(String startBy, float VAL, String dest) {
  String packet = "";
  packet += String(startBy);
  packet += String(dest);
  packet += String(VAL, 2);
  data=packet;
  return data;
}

void OnDataSent(const uint8_t *mac_addr , esp_now_send_status_t status){
  Serial.print("\r\nLast Packet Send Status: \t");
  Serial.println(status == ESP_NOW_SEND_SUCCESS ? "Delivery Success" : "Delivery Fai
}
void ESPnow()
{
    WiFi.mode(WIFI_STA);
  // Initialize ESP-NOW
  if (esp_now_init() == ESP_OK) {
    Serial.println("ESP-NOW initialized successfully");
  } else {
    Serial.println("Error initializing ESP-NOW");
  }
  // Register the send callback
  esp_now_register_send_cb(OnDataSent);
```

```
  esp_now_register_send_cb(OnDataSent);
  //Register peer
  memcpy(peerInfo.peer_addr , broadcastAddress,6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;
  //Add peer
  if(esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Fail to add peer!");
  } else
  {
    Serial.println("peer added successfully") ;
  }
}
```