“Voice Controlled AI Robotic Assistant for Physically Challenged and Elderly People”

A Project Report
Submitted on Partial Fulfillment of the Requirement
For the Degree of
BACHELOR OF TECHNOLOGY

By
Anup Jaiswal (2022031015)
Akansha Arya (2022031006)
Shambhavi (2022011153)
Rishi Verma (2022031059)
Ayush Kumar (2022031025)

Under the supervision
Of
Prof. L. B. Prasad
Electrical Engineering Department

Department of Electrical Engineering
Madan Mohan Malaviya University of Technology

(U.P. State Gov. University)
Gorakhpur, Uttar Pradesh (273010)
India

ABSTRACT

This project presents a **Voice-Controlled AI Robotic Assistant** designed to enhance the independence, safety, and comfort of physically challenged and elderly individuals. The system utilizes advanced **speech recognition technology**, integrated with ESP32 micro-controller-based robotic platform, to enable seamless hands-free interaction. Voice commands are processed in real time to control robot navigation, perform routine tasks, and assist users in their daily activities.

This project report presents the design and development of a Voice Controlled AI Robotic Assistant aimed at enhancing the autonomy and quality of life for the elderly and physically challenged individuals. As the global population ages, the demand for affordable assistive technologies has surged; however, existing solutions are often characterized by high costs due to expensive onboard computing or limited functionality requiring complex manual control. This project addresses these challenges by proposing a cost-effective, semi-autonomous mobile manipulator that leverages a hybrid edge-cloud architecture. The system integrates a robust four-wheel skid-steer chassis capable of navigating unstructured domestic environments, such as carpets and uneven floors, with a three-degree-of-freedom robotic arm designed for precise pick-and-place operations.

The core technical innovation lies in the distributed computing framework, where low-level control tasks are handled by an ESP32-S3 microcontroller, while resource-intensive processes—including Natural Language Processing (NLP) and Computer Vision—are offloaded to a local server. A custom-built React web application captures user voice commands, which are processed by a Python-based backend to interpret intent and identify target objects using real-time video streaming from an onboard ESP32-CAM. To ensure reliable interaction with the physical world, the system employs a sensor fusion strategy that combines visual feedback for angular alignment with ultrasonic distance measurements for precise depth estimation, allowing the robot to autonomously approach and grasp objects without human intervention.

Ultimately, this project demonstrates the feasibility of creating intelligent service robots without relying on expensive hardware. by utilizing standard

commercial off-the-shelf components and leveraging the processing power of external devices, the system achieves a high degree of functionality and reliability. The resulting prototype serves not only as a functional assistant capable of retrieving household items on command but also as a scalable platform for future research in accessible home automation, proving that advanced AI-driven robotics can be made accessible to the demographic that needs them the most.

ACKNOWELEDGMENT

We would like to express our sincere gratitude to all those who have helped us in the successful completion of our final year project titled *"Voice Controlled AI Robotic Assistant for Physically Challenged and Elderly People."* This project would not have been possible without their support, guidance, and encouragement.

We extend our heartfelt thanks to our project supervisor **Prof. L.B. Prasad**, Department of Electrical Engineering, for his valuable insights, constant support, expert guidance, and continuous encouragement throughout the course of this project. His motivation and feedback have played a crucial role in shaping the direction and quality of our work.

We are grateful to all the faculty members of the Electrical Engineering Department for their support and for providing us with the necessary resources and knowledge base that helped in realizing this project.

We would also like to thank our institution, **Madan Mohan Malaviya University of Technology, Gorakhpur**, for providing us with the facilities and infrastructure necessary to carry out our project.

A special thanks to our teammates **Anup Jaiswal, Akansha Arya, Ayush Kumar, Rishi Verma**, and **Shambhavi** for their active participation, excellent collaboration, teamwork, and mutual support throughout the project work. Their dedication and creative inputs were essential for the successful execution of this idea.

Lastly, we would like to acknowledge the constant support from our families and friends who encouraged us throughout this journey.

Submitted By:

Anup Jaiswal (2022031015)

Akansha Arya (2022031006)

Shambhavi (2022011153)

Rishi Verma (2022031059)

Ayush Kumar (2022031025)

CANDIDATE DECLARATION

We declare that this written submission represents my/our work and ideas in my/our own words and where other's ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my/our submission. We understand that any violation of the above will be cause for disciplinary action by the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Signature of Candidate

Anup Jaiswal
Roll No – 2022031015

Akansha Arya
Roll No - 2022031006

Shambhavi
Roll No – 2022011153

Rishi Verma
Roll No – 2022031059

Ayush Kumar
Roll No – 202203025

CERTIFICATE

DEPARTMENT OF ELECTRICAL ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the Bonafide work of **Anup Jaiswal (2022031015), Akansha Arya (2022031006), Rishi Verma (2022031059), Shambhavi (2022011153), Ayush Kumar (2022031025),** who carried out the project entitled **Voice Controlled AI Robotic Assistant for Physically Challenged and Elderly People** under my supervision from _____ to _____.

Internal Guide
Prof. L. B. Prasad
Electrical Engineering Department

Head of Department
Dr. Prabhakar Tiwari
Electrical Engineering Department

APROVAL SHEET

This thesis entitled Voice Controlled AI Robotic Assistant for Physically Challenged and Elderly People by Anup Jaiswal, Akansha Arya, Ayush Kumar, Rishi Verma, and Shambhavi is approved for the degree of Bachelor of Technology in Electrical Engineering from Madan Mohan Malaviya University of Technology Gorakhpur.

Examiner

_____

Supervisor

Prof. L. B. Prasad

_____

Head of Department

Prof. Prabhakar Tiwari

_____

Date: _____

Place: _____

TABLE OF CONTENT

TABLE OF FIGURES

INTRODUCTION

1. Background And Motivation

The field of service robotics has witnessed a paradigm shift in recent years, moving from per-programmed industrial automation toward intelligent assistants capable of operating in unstructured human environments. Despite this progress, a significant gap remains between high-end research robots and accessible consumer devices. Most existing assistive robots are either prohibitively expensive due to the need for powerful onboard computers or are limited to basic remote-controlled functionality that requires complex manual operation. The primary motivation behind this project is to democratize access to smart assistive technology by developing a Voice Controlled AI Robotic Assistant that combines affordability with advanced autonomy. By replacing traditional joystick interfaces with natural language processing, the system addresses the accessibility needs of users with limited mobility or technical expertise. Furthermore, this project seeks to validate a distributed computing architecture where resource-intensive tasks such as object detection and intent recognition are offloaded to a local server, allowing a low-cost micro controller like the ESP32-S3 to perform complex manipulation tasks usually reserved for much more expensive hardware. This approach not only reduces the physical cost of the robot but also creates a scalable framework for future developments in home automation and elderly care.

2. Problem Statement

Physically challenged individuals and the elderly often face significant hurdles in performing daily tasks such as retrieving objects, which impacts their independence and quality of life. While robotic assistants have the potential to alleviate these challenges, current market solutions present a critical dichotomy: they are either rudimentary devices requiring fine motor skills to operate

complex joystick controllers, or they are fully autonomous systems with prohibitive costs that make them inaccessible to the average consumer. Furthermore, standard mobile robots often fail to navigate the unstructured environments of a typical home, such as carpets or uneven floors, rendering them ineffective for practical use. There is a distinct lack of affordable, intuitive systems that bridge this gap by offering robust mobility and intelligent manipulation without the need for manual dexterity. This project addresses the urgent need for a cost-effective, voice-activated solution that leverages modern AI to interpret natural human intent, thereby removing the physical barrier to interaction and empowering users to command a robotic assistant as easily as speaking to a companion.

3. Objective Of The Project

The primary objective of this project is to design and develop a cost-effective, voice-actuated mobile robotic assistant capable of autonomously identifying, tracking, and retrieving objects in a home environment. Specifically, the project aims to implement a rugged four-wheel skid-steer chassis to ensure reliable navigation over rough or uneven surfaces, overcoming the mobility limitations of standard caster-wheel designs. A core technical goal is to engineer a hybrid edge-cloud architecture that offloads computationally intensive tasks—such as natural language processing and computer vision—to a local server, enabling complex AI functionality on a low-cost ESP32 microcontroller platform. Additionally, the project seeks to integrate a three-degree-of-freedom robotic arm with a sensor-fusion control loop, combining visual data for alignment and ultrasonic feedback for depth, to achieve precise object manipulation. Ultimately, this system intends to provide a seamless, hands-free user interface that empowers physically challenged and elderly users to perform pick-and-place tasks with independence and ease.

LITERATURE REVIEW

1. Introduction

The growth of artificial intelligence (AI), human–computer interaction (HCI), and assistive robotics has transformed how individuals with disabilities and older adults interact with technology. With the rise of smart homes, IoT devices, and advanced speech recognition systems, voice-controlled robotic assistants have become a vital research domain. These systems aim to improve independence, enhance safety, and reduce dependency on caregivers. The existing literature presents significant work on speech processing, autonomous navigation, assistive robotics, and AI-driven decision-making. This review consolidates major findings, evaluates technological progress, and identifies gaps that motivate the development of a voice-controlled AI robotic assistant for physically challenged and elderly individuals.

2. Evolution Of Voice Recognition Systems

Early research in voice recognition began with template-matching approaches such as Dynamic Time Warping (DTW,1978) and rule-based speech processing. These systems had limited flexibility and struggled with variations in accent, speed, and background noise. The development of statistical models like the Hidden Markov Model (HMM,1989) marked a major improvement, enabling continuous speech recognition with better accuracy.

With the emergence of machine learning and neural networks, speech recognition advanced significantly. Deep Neural Networks (DNNs), Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) models, and Convolutional Neural Networks (CNNs) became the backbone of commercial systems like Google Speech API(2008) and Microsoft Cortana.

Recent advancements introduced Transformer-based architectures such as Whisper (2022), BERT-based ASR models(2024), and self-supervised learning systems, which enhanced robustness, multilingual support, and noise resilience.

Studies highlight that voice interfaces are the most suitable communication method for individuals with mobility impairments, as they eliminate the need for physical interaction and offer intuitive control over assistive systems.

A considerable amount of research focuses on developing voice-controlled assistive tools for Motor impaired users, individuals with paralysis, patients with muscular dystrophy, visually impaired users and elderly people with limited mobility.

3. Assistive Robotics

Assistive robots can be broadly classified into physical-assist robots, companion or social robots, and service robots. Research shows that these robots can perform tasks such as:

- Fetching and carrying objects
- Medication reminders
- Environmental monitoring
- Emergency assistance
- Navigation support within homes
- Interaction and emotional comfort

Robots such as robotic arms, exoskeletons, and automated wheelchairs have been widely studied for individuals with motor impairments. Systems like the *JACO robotic arm (Kinova Robotics, 2010)* and *iARM* allow users to perform daily tasks such as lifting, grabbing, or opening doors with minimal physical involvement.

Research on SARs emphasizes the importance of emotional engagement, conversation, and companionship for elderly users. Robots such as *PARO (2004)*, *Pepper (2014)*, *NAO (2011)*, and *ELLBO* have demonstrated improvements in reducing loneliness, offering cognitive stimulation, and supporting mental well-being.

Mobile service robots like *Care-O-bot*, *PR2*, *TurtleBot (2012)*, and *Robovie (2020)* focus on household support. They integrate sensors, cameras, and AI modules for indoor navigation and task automation. Studies show that service robots significantly reduce caregiver workload and enable safe living environments for elderly individuals.

4. AI Driven perception, Decision Making and Context Awareness

A key theme in the literature is the implementation of AI algorithms that enhance robotic intelligence. Researchers highlight the importance of context awareness, allowing robots to understand user behavior, preferences, and environmental situations.

Robotic systems commonly use:

- Ultrasonic sensors for obstacle detection
- Infrared sensors for proximity monitoring
- LiDAR and SLAM for mapping and navigation
- Cameras and depth sensors for visual perception

- IoT sensors for home automation

Studies confirm that multi-sensor fusion significantly improves accuracy, reduces navigation errors, and enhances robot responsiveness in dynamic indoor environments.

Research in activity recognition and human intention prediction reveals that machine learning model-scan understand patterns such as:

- Daily movement routines
- Medication schedules
- Activity levels
- Health irregularities
- Speech patterns and preferences

Adaptive learning allows the robot to provide personalized support, improving usability, especially for elderly users with varied mobility and speech clarity.

Decision-making approaches in assistive robots include:

1. Rule-based systems (simple, interpret-able, limited adaptability)
2. Machine learning models (more flexible but require training data)
3. Hybrid AI systems integrating both approaches

Literature suggests hybrid systems perform better, offering reliability with the ability to learn from user behaviour.

5. Voice Interaction Challenges for Elderly Peoples

Voice interaction has become one of the most convenient communication methods for elderly users due to its hands-free and intuitive nature. However, numerous studies published between 2010 and 2024 consistently highlight that speech recognition systems often perform poorly with elderly speakers. This is due to a combination of physiological, cognitive, and environmental factors. Understanding these challenges is crucial for designing reliable voice-controlled robotic assistants for elderly users. With aging, several biological and

psychological changes occur which influence speech pattern in following manner:

- Weak or low-volume voice
- Irregular speech rhythm
- Mispronunciation
- Cognitive issues affecting command clarity

As a result, literature suggests implementing:

- Noise suppression algorithms
- Voice adaptation and personalization
- Repetition-confirmation strategies
- Emotion detection for user monitoring

Research also highlights that speech recognition accuracy drops by 10–20% for senior users, making adaptive AI essential for reliable performance.

6. Navigation and Safety in Indoor Assistive Robots

Safe mobility is a critical requirement for robotic assistants in home environments. Studies evaluate several navigation approaches:

a) Localization And Mapping

Robots commonly use:

- Simultaneous Localization and Mapping (SLAM)
- Extended Kalman Filters (EKF)
- Particle filters
- Grid mapping

These allow robots to create accurate indoor maps, avoid obstacles, and reach desired locations through voice commands.

b) Human-Robot Interaction Safety

Past research stresses the importance of low-speed operation, collision avoidance, emergency stop mechanisms, soft materials to prevent injury and

redundant sensors for improved safety. Such safety protocols are especially crucial for visually impaired or physically weak individuals.

7. Role of IoT and Cloud Connectivity in Assistive Robots

The integration of Internet of Things (IoT) principles and cloud connectivity serves as the technological backbone of this project, fundamentally transforming the robotic assistant from a simple mechanical device into an intelligent, connected system. By leveraging IoT protocols, the robot functions as an edge node that bridges the physical world of motors and sensors with the digital capabilities of high-performance computing. This connectivity is essential for overcoming the hardware limitations of low-cost microcontrollers like the ESP32, as it allows the system to offload resource-intensive tasks such as speech-to-text conversion, natural language understanding, and complex computer vision algorithms to a local server or cloud environment. This architecture ensures that the robot can perform sophisticated AI-driven operations, like identifying specific objects in a cluttered room or interpreting varied human speech patterns, without requiring expensive, power-hungry onboard processors. Consequently, the role of cloud connectivity is pivotal in making this assistive technology affordable and accessible; it reduces the cost and weight of the physical unit while enabling powerful, scalable intelligence that can adapt to the unique needs of physically challenged and elderly users, providing them with a responsive and capable companion for daily independence.

Modern literature shows an increasing shift toward IoT-enabled assistive systems. Integrating robotics with IoT allows real-time monitoring, remote caregiver alerts, health tracking, cloud-based data analysis, integration with smart appliances.

Studies on cloud-assisted robots demonstrate enhanced processing capability using cloud AI models, though challenges like latency, data privacy, and network dependence remain.

8. Identified Research Gaps

The literature reveals several challenges that existing systems have not fully addressed:

1. Limited affordability – High-end robots are expensive.
2. Lack of personalized AI – Most systems do not adapt to elderly speech patterns.
3. Insufficient mobility support – Many AI assistants remain stationary (e.g., Alexa).
4. Integration issues – Fewer systems combine robotics + IoT + voice AI.
5. Complex user interfaces – Difficult for elderly individuals to operate.
6. Safety constraints – Indoor navigation needs further improvement.
7. Limited independence – Robots often require manual supervision.

MODEL OVERVIEW

1. Concept Design and Requirements

Everything starts with an imagination, Sci-Fi Movies have excited this idea long before like *Jarvis* from IRONMAN movie etc. But they look fancy on screen only, In real world its quite challenging to achieve same.

Our Idea is not only limited to current scope instead its have the potential to drive future, for simplicity and practically representing our idea we decided to use *Object Detection and Voice Commanding* feature up-to a limited scope.

First step was to transform the idea into a picture, we used Google's Gemini AI and got this result :

This represent the concept design.

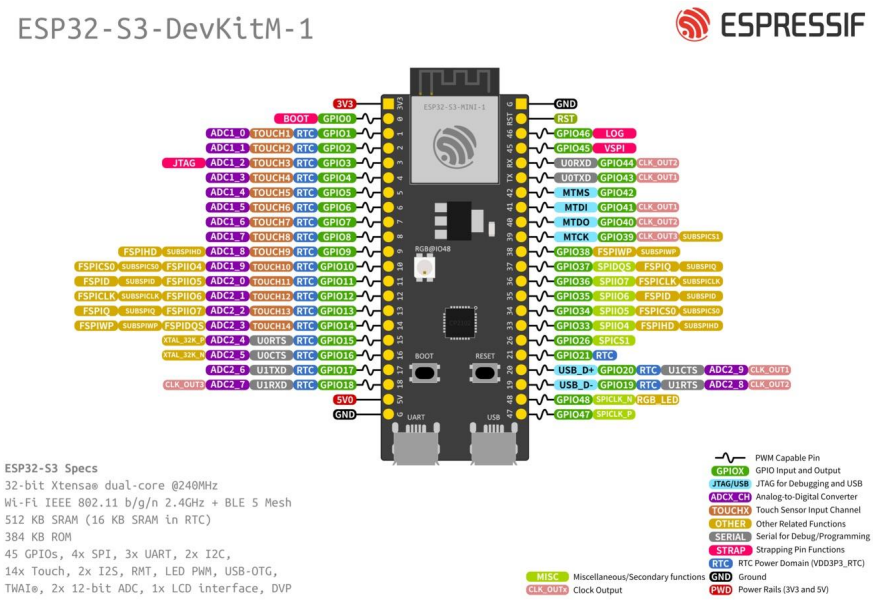
And We built the actuall model as :

Requirements form this model is quite simple and straight forward, we have tried our best to keep things as less complex as possible :

- Model should understand basic voice commands to navigate
- Model should recognize basic objects with good confidence value
- Model should able to Navigate to the target, pick, place and drop the object as instructed.
- Model should reply non actionable commands in a user friendly manner.

2. Components and their Specification

i. ESP32 S3 Microcontroller Development Board



The ESP32-S3 is a highly integrated, low-power system-on-chip (SoC) microcontroller developed by Espressif Systems, designed specifically for the Internet of Things (IoT) and Artificial Intelligence of Things (AIoT) applications. Unlike its predecessors, the ESP32-S3 is built with a focus on accelerating neural network computing and signal processing workloads directly on the edge. It combines a powerful dual-core processor with integrated Wi-Fi and Bluetooth connectivity, making it a robust platform for devices that require both high computational performance and seamless wireless communication. Its architecture supports a wide array of peripherals, allowing it to interface easily with sensors, motors, and cameras, which establishes it as an industry-standard choice for developing smart, connected robotic systems.

- Model No. YD ESP32 S3 N8R2
- Flash : 2 MB
- Processor Type : Xtensa 32-bit LX7 Dual-Core
- Clock Frequency : Upto 2.4 GHz
- CPU : Xtensa LX7
- Connector : USB Type-C
- Operating Temperature : -40 to 65 deg.

While the "ESP32" is a large family, the term most often refers to the original, classic series (like the chip in the ESP32-WROOM-32 module).

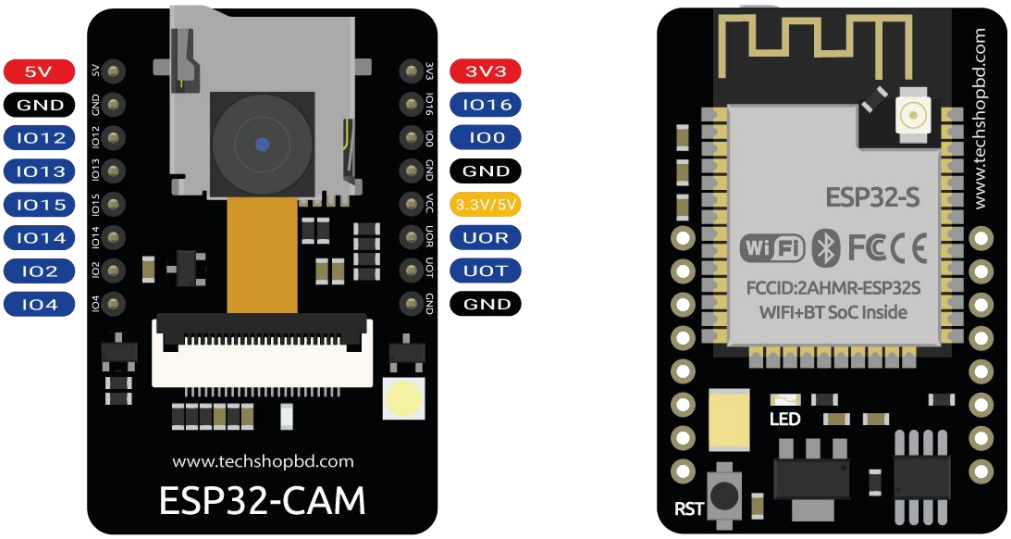
- CPU: Tensilica Xtensa Dual-Core LX7 microprocessor, running up to 240 MHz. (Having two cores allows for one to handle the Wi-Fi/Bluetooth stack while the other runs user code.)
- Wireless:
  - Wi-Fi: 802.11 b/g/n (2.4 GHz)
  - Bluetooth: v4.2 BR/EDR and Bluetooth Low Energy (BLE)
- Memory:
  - SRAM: 520 KB (for program variables and state)
  - Flash Memory: Typically 4 MB to 16 MB (external, on the module, for storing the program code)
  - ROM: 448 KB (for the bootloader and core functions)

- Power: Operates at 3.3V. It includes advanced power-saving features, most notably a deep-sleep mode that allows it to run on batteries for extended periods.

The ESP32 has a very flexible development ecosystem, which is a major reason for its success.

1. Arduino IDE / PlatformIO: This is the most popular entry point for hobbyists. You can write code in C++ using the familiar Arduino setup() and loop() functions. A massive number of libraries are available.
2. ESP-IDF (Espressif IoT Development Framework): This is the official and most powerful framework from Espressif. It's built on C/C++ and the FreeRTOS real-time operating system. It gives you full control over every aspect of the chip but has a steeper learning curve.
3. MicroPython / CircuitPython: This allows you to write code for the ESP32 in the Python programming language. It's excellent for rapid prototyping, education, and projects that are less performance-critical.

ii. ESP32 Wifi WebCAM Module



The ESP32-CAM-MB is a versatile development board that combines the power of the ESP32 microcontroller with built-in WiFi and Bluetooth capabilities and an integrated OV2640 camera module. This board is designed for various applications, including IoT projects, robotics, surveillance systems, and more, where wireless communication and image capture are essential. Below is a description of the key features and components of the ESP32-CAM-MB development board

Features:

1. The heart of the ESP32-CAM-MB is the ESP32
2. WiFi and Bluetooth Connectivity
3. OV2640 Camera Module
4. Micro USB Connection
5. I/O Pins and GPIO Headers
6. MicroSD Card Slot

iii. Pro-Range 7.4V 35 Kg cm DC Digital Servo Motor (OT5330M)

The Pro-Range OT5330M **7.4V 35Kg.cm 180° Metal Gear Digital Servo Motor** with any servo code, library, and hardware. It rotates approximately 120-degree, 60 degrees in each direction. This motor package comes with a selection of hardware and arms.

Wire Description:

- RED – Positive
- Brown – Negative
- Orange – Signal

Features:

1. Steel Gears Construction
2. The connection cable is thicker
3. Equips high-quality motor
4. High resolution
5. Fast control response

6. Constant torque throughout the servo travel range
7. Excellent holding power
8. CE & RoHS Approved

Specifications :

1. Model No. OT5330M
2. Operating Voltage : 4 to 8.4V
3. No load Speed : 42@6.0V (0.185sec/60 deg.), 52@7.4V (0.151sec/60 deg)
4. Stall Torque : 28.8 Kg cm to 35 Kg cm
5. Operating Temperature : -20 to 60 deg. C
6. Motor Supply Current : 3.8A
7. Dead Band Width : <= 4
8. Gear Material : Steel
9. Rotation : 180 deg. Continous Rotation
10. Shaft Type : 6mm (25T-M3)
11. Cable Length : 30cm
12. Command Signal : PWM
13. Connector Type : JR
14. Dimension : 55 mm x 20 mm x 43 mm
15. Weight : 70 gm

iv. Pro-Range 25 Kg cm DC Digital Servo Motor (DS3225)

Pro-Range DS3225 25kg·cm Metal Gear Digital Servo Motor – 180° is a powerful, high-torque servo designed for demanding applications in robotics, RC vehicles, and automation projects. With an impressive torque output of 25kg·cm, Digital Servo Motor provides the strength needed for heavy-duty tasks while maintaining smooth and precise movement.

Its all-metal gear construction ensures durability and long-lasting performance under stress, and the digital control system delivers accurate, stable positioning. Supporting a 180° rotation range and operating on a wide voltage range of 5 ~ 8.4, the DS3225 offers excellent versatility and reliability. Whether you're building a robotic arm or upgrading and RC system, this servo motor is built to deliver consistent and efficient performance.

Wire Description:

- RED – Positive
- Brown – Negative
- Orange – Signal

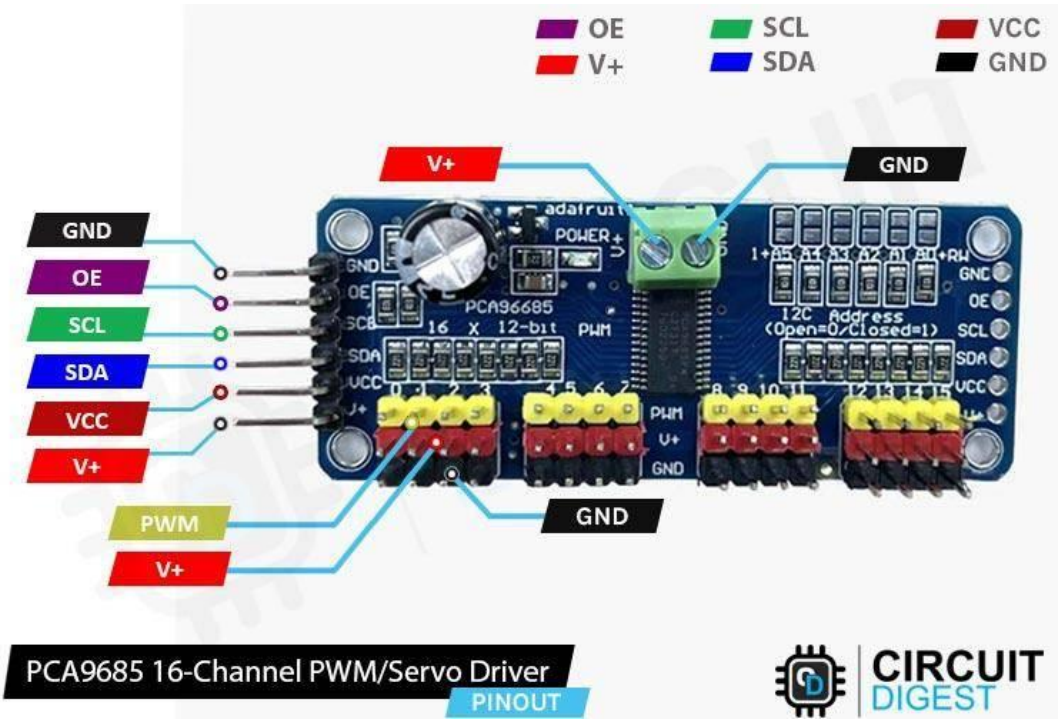
Features:

1. Steel Gears Construction
2. The connection cable is thicker
3. Equips high-quality motor
4. High resolution
5. Fast control response
6. Constant torque throughout the servo travel range
7. Excellent holding power
8. CE & RoHS approved

Specifications :

1. Model No. DS3225
2. Operating Voltage : 5 to 8.4V
3. No load Speed : 0.14 sec /60 deg to 0.09 sec/ 60 deg
4. Stall Torque : 25 Kg cm
5. Operating Temperature : -20 to 60 deg. C
6. Motor Supply Current : 3.8A
7. Dead Band Width : <= 4
8. Gear Material : Steel
9. Rotation : 180 deg. Continous Rotation
10. Shaft Type : 6mm (25T-M3)
11. Cable Length : 30cm
12. Command Signal : PWM
13. Connector Type : JR
14. Dimension : 40 mm x 20 mm x 38.5 mm
15. Weight : 60 gm

v. PCA9685 16 Channel 12 Bit Servo Driver



The PCA9685 is a 16 channel PWM servo driver commonly used to control multiple servos with precise timing using only two I2C pins from a microcontroller. It generates stable PWM signals independently of the main processor, which reduces the load on the microcontroller and allows smooth and consistent servo movement. The driver can run at a selectable frequency, usually around 50 to 60 hertz for standard servos, and each channel provides 12 bit resolution for accurate angle control. It is widely used in robotics projects for controlling robotic arms, legs, and other mechanisms that require many servos at the same time.

Core Specifications & Features

- **16 Channels:** It provides **16 independent PWM outputs**, allowing you to control up to 16 servos, LEDs, or other PWM-driven devices simultaneously.
- **12-bit Resolution:** Each output has **12-bit resolution**, meaning the pulse width can be set to 4096 discrete steps.
    - *For Servos:* This offers very **fine control** over the servo motor's position.
- **I2C Interface (Two-Wire Control):**

    - It uses the **I2C communication protocol**, which requires only **two pins** (SDA and SCL) on your microcontroller (Arduino/Raspberry Pi) to control all 16 outputs.
    - This is a massive advantage as it **saves valuable I/O pins** on your main board.
- **Built-in Clock (Free-Running PWM):**
    - It has an internal oscillator, allowing the PCA9685 to run the PWM signals independently.
    - This means your microcontroller doesn't need to continuously generate the PWM signals, **freeing up its processing power** for other tasks.
- **Adjustable Frequency:** The PWM output frequency is **programmable** from about 24 Hz up to 1526 Hz. Servos typically require about 50 Hz or 60 Hz.
- **Chainable:** The chip includes 6 hardware address pins, allowing you to **chain up to 62 PCA9685 modules** on a single I2C bus.
- **Output Configuration:** Outputs can be configured as either **push-pull** (default for servo control) or **open-drain** (useful for LED control).
- **Power:**
    - **Logic Voltage (VCC):** Operates from 2.3V to 5.5V, making it compatible with both 3.3V (Raspberry Pi, some Arduinos) and 5V (Arduino Uno) microcontrollers.
    - **Servo Power (V+):** Modules typically have a separate input for the V+ pin, which powers the servos or LEDs directly, preventing high current draw from damaging the microcontroller.

Primary Applications

The PCA9685 module is primarily used in applications that require a large number of precisely controlled PWM outputs:

- **Robotics:** Controlling multiple **servo motors** in robotic arms, hexapods, quadcopters, and other complex mechanical systems.
- **Lighting:** Controlling the brightness (dimming) or color mixing of a large array of **LEDs** or **RGB/RGBA LEDs**.
- **Automation:** Driving various electronic loads that require precise duty-cycle control.

vi. Pro-Range 11.1V 6000 mAH 3C 3S2P, 3 Cell Li-Ion Battery

The Pro-Range 11.1V 6000mAh 3C Li-Ion battery is a high-capacity rechargeable power source designed specifically for applications requiring long operational endurance rather than extreme bursts of power. Unlike standard Lithium Polymer (LiPo) batteries often used in racing drones, this Lithium-Ion pack utilizes cylindrical 18650 cells arranged in a 3S2P configuration. The designation 3S2P indicates that the internal architecture consists of three sets of cells connected in series to achieve the nominal voltage of 11.1 volts, with each set containing two cells connected in parallel to double the capacity to 6000 milliamp-hours. This structure offers an excellent balance between energy density and physical durability, making it significantly more robust and safer for educational or ground-based robotics projects compared to fragile soft-pouch batteries.

In terms of technical specifications, this battery pack provides a nominal voltage of 11.1V, which charges up to a peak of 12.6V, and holds a total energy capacity of approximately 66 Watt-hours. The 3C discharge rating is a critical specification for system design, indicating that the battery can safely deliver a continuous current of three times its capacity, which translates to 18 Amperes (3 x 6A). While lower than the 50C ratings found in racing batteries, this 18A limit is intentional and advantageous for this project, as it provides a stable and safe current supply sufficient to drive multiple DC motors and servos without the risk of extreme current spikes that could damage sensitive electronics. The pack typically utilizes a JST-XH connector for balance charging, ensuring individual cell health, and a high-current discharge connector like an XT60 to minimize resistance during operation.

Core Specfications

- Capacity : 6000 mAH
- Discharge : 3C
- BMS : Yes
- Connector : DC Jack Female
- Max. Charging Voltage : 12.6V
- Max. Discharging voltage : 8.7V
- Charging Current : 1C
- Dimension : 72mm x 48mm x 65mm
- Model Series : 3
- Weight : 0.4kg

vii. TowerPro MG946R 10 Kg cm Digital Servo Motor

TowerPro Servo Motors are optimum-quality and affordable cost servos.! They are suitable for a wide range of applications, including RC aircraft, automobiles, and robotics, Or just to have some fun with whatever crazy project you're working on.

Wire Description:

- RED – Positive
- Brown – Negative
- Orange – Signal

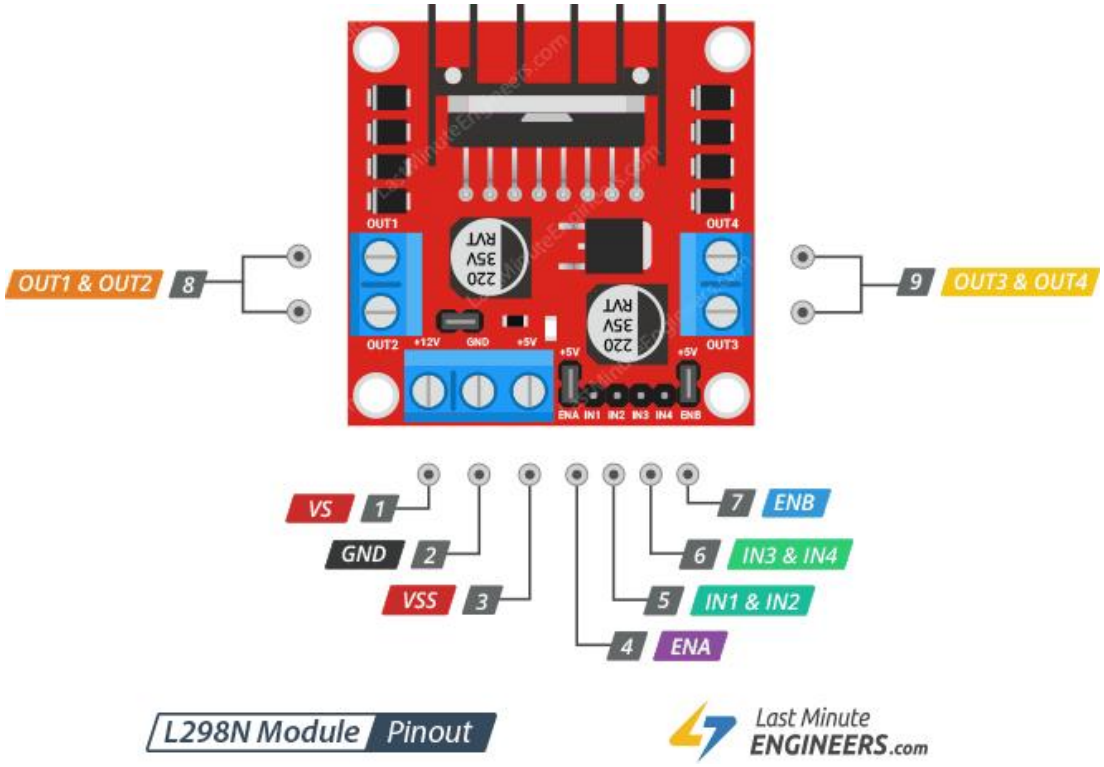
Features:

1. Steel Gears Construction
2. The connection cable is thicker
3. Equips high-quality motor
4. High resolution
5. Fast control response
6. Constant torque throughout the servo travel range
7. Excellent holding power
8. CE & RoHS approved

Specifications :

1. Model No. MG Series
2. Operating Voltage : 4.8V to 6.6V
3. No load Speed : 0.19sec/60 deg
4. Stall Torque : 9.4 to 11 Kg cm
5. Operating Temperature : 0 to 55 deg. C
6. Motor Supply Current : 3A
7. Dead Band Width : <= 4
8. Gear Material : Semi Metal
9. Rotation : 180 deg. Continous Rotation
10. Shaft Type : 6mm (25T-M3)
11. Cable Length : 30cm
12. Command Signal : PWM
13. Connector Type : JR
14. Dimension : 40 mm x 19 mm x 41 mm
15. Weight : 56 gm

viii. L298N DC Motor Driver



The L298N motor driver is a dual H bridge module used to control the speed and direction of two DC motors using a microcontroller. It can handle relatively high current and voltage, making it suitable for small robots, cars, and other DIY projects. The driver allows forward, reverse, and braking control by switching the internal transistors, and it works using simple input signals from devices like Arduino or ESP32. It also includes an onboard voltage regulator and screw terminals that make wiring easier. Although it is not very efficient and generates heat, it remains one of the most popular and beginner friendly motor drivers.

Core Specifications

- Driver Type : Dual H-Bridge (MOSFET Based)
- Motor Voltage : 2.5V to 13.5 V (upto 15V)
- Logic Voltage : 2.7V to 5.5V
- Continuous Current : 1.2A per channel
- Peak Current : 3.2A per Channel
- PWM Frequency : Upto 100 kHz

Key Features & Performance

The module is specifically highlighted for its **performance** and **ultra small volume**, especially when compared to the older **L298N** (which the text mentions).

1. **Ultra Small Volume & Efficiency**
   - **Size:** The TB6612FNG is significantly **smaller** and **lighter** than L298N modules.
   - **High Efficiency:** It uses **MOSFETs** (Metal-Oxide-Semiconductor Field-Effect Transistors) instead of the older Bipolar Junction Transistors (BJTs) used by the L298N.
     - This design has a much lower internal resistance, resulting in **higher efficiency** (often over 90%) and **less power lost as heat**.
     - **No Heatsink Required:** Unlike the L298N, the TB6612FNG typically runs cool and does **not** require a large, bulky heatsink.
   - **Lower Voltage Drop:** The voltage drop across the driver is very low (e.g., less than 0.5V), meaning almost the full motor supply voltage reaches the motors, improving performance for low-voltage motors.

2. **Motor Control & Safety**
   - **Dual H-Bridge:** Allows for **bidirectional control** (forward/reverse) of two independent motors.
   - **Control Modes:** Supports four function modes: **CW (Clockwise), CCW (Counter-Clockwise), Short Brake, and Stop**.
   - **Standby Control (STBY):** Includes a dedicated pin to put the chip into a **low-power sleep mode**, which is excellent for **battery-powered applications**.
   - **Built-in Protection:** Features internal protection circuits:
     - **Thermal Shutdown Circuit:** Shuts down the output if the chip overheats.
     - **Low Voltage Detecting Circuit:** Protects against operation at unsafe supply voltages.

3. **Comparison to L298N (The "Ultra L298N" Performance Match)**

The comparison in the text suggests this module offers a **modern, high-performance, compact alternative** to the widely used L298N:

- **TB6612FNG:** Small, highly efficient, low heat, excellent for low-to-medium current motors.
- **L298N:** Large, low efficiency, requires a heatsink, but can typically handle higher voltages and continuous currents (around 2A) than the TB6612FNG.

ix. DC Gear Motor 12V 200RPM

A 200 RPM DC gear motor is a commonly used motor in robotics and automation projects where moderate speed and good torque are required. It combines a small DC motor with a built in gear reduction system that lowers the output speed to around 200 revolutions per minute while increasing torque. This makes it suitable for driving robot wheels, small mechanisms, and other applications that need controlled and steady movement. These motors typically run on 6 to 12 volts and offer reliable performance with simple control using motor drivers like the L298N.

x. Pro-Range 3S Li-ion 12.6V 2A Battery Charger

The Pro-Range 3S Li-ion 12.6V 2A Battery Charger is a specialized power supply unit engineered specifically for safely recharging 3-cell (3S) Lithium-Ion battery packs. Unlike standard DC adapters that output a fixed voltage regardless of load, this charger implements a dedicated Constant Current / Constant Voltage (CC/CV) charging algorithm, which is critical for the health and safety of lithium-based chemistries. It is designed to take a standard AC wall input (100V-240V) and convert it into a precise 12.6V DC output, matching the full-charge voltage of a 3S battery pack (4.2V per cell x 3 = 12.6V). The "2A" specification indicates that it delivers a charging current of 2 Amperes, allowing it to recharge the project's 6000mAh battery from empty to full in approximately 3 to 4 hours, which is an optimal speed that balances convenience with cell longevity.

xi. Jumper Wires

Jumper Wires are low power single core cables , widely used for prototyping purpose and Embedded Applications.

xii. Fiber Wheels 10 cm diameter (Rear)

Heavy Fibre Wheels with Rubber Grip have been used in rear wheels providing better grip and driving speed.

xiii. Swivel Caster Wheels (Front)

Caster Wheels are heavy duty and Omni Directional , instead of using the complex steering mechanism , we decided to utilize Caster Wheels with differential drive for changing the direction.

xiv. Aluminium Chasis

Custom Build Aluminium Chassis, for better rigidity and strength. It’s a handmade chassis build from Aluminum frames and tied together with screws and bolts , along with m-seal putty.

Dimension : 33 cm x 23 cm x 4.7cm

xv. 3D Printed Gripper Mechanism

3D Printed Robotic Gripper Mechanism, designed using freeCAD software and Operates on parallel jaw mechanism.

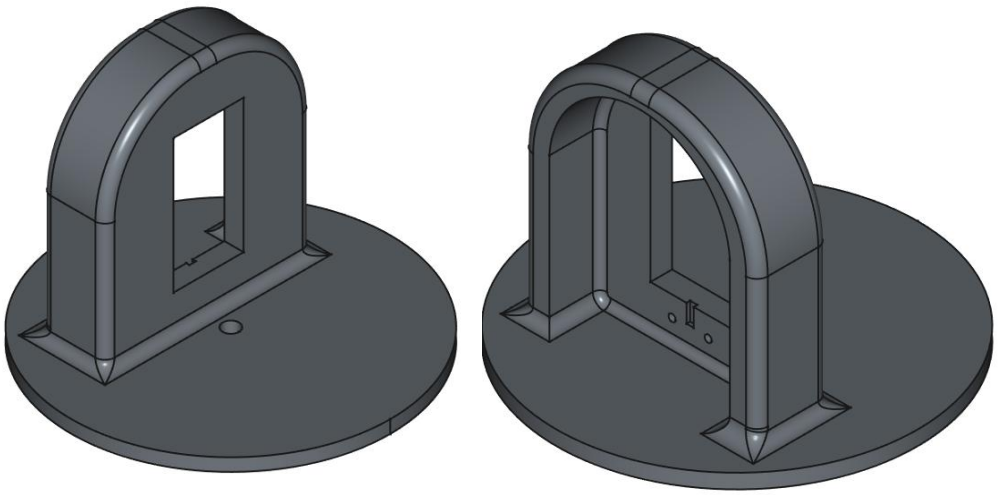
3D CAD designed is made such that it can accommodate MG946R servo motor fro providing gripping force. Two interlocked gear, among which one is driver gear (connected to servo motor’s shaft). Interlocking betweening gear ensures synchronized operation and equal force distribution. Both jaw are connected to gear links over a movable joint, along with that Link are provided to support jaws and ensures their parallel operation.

3. 3D CAD Design

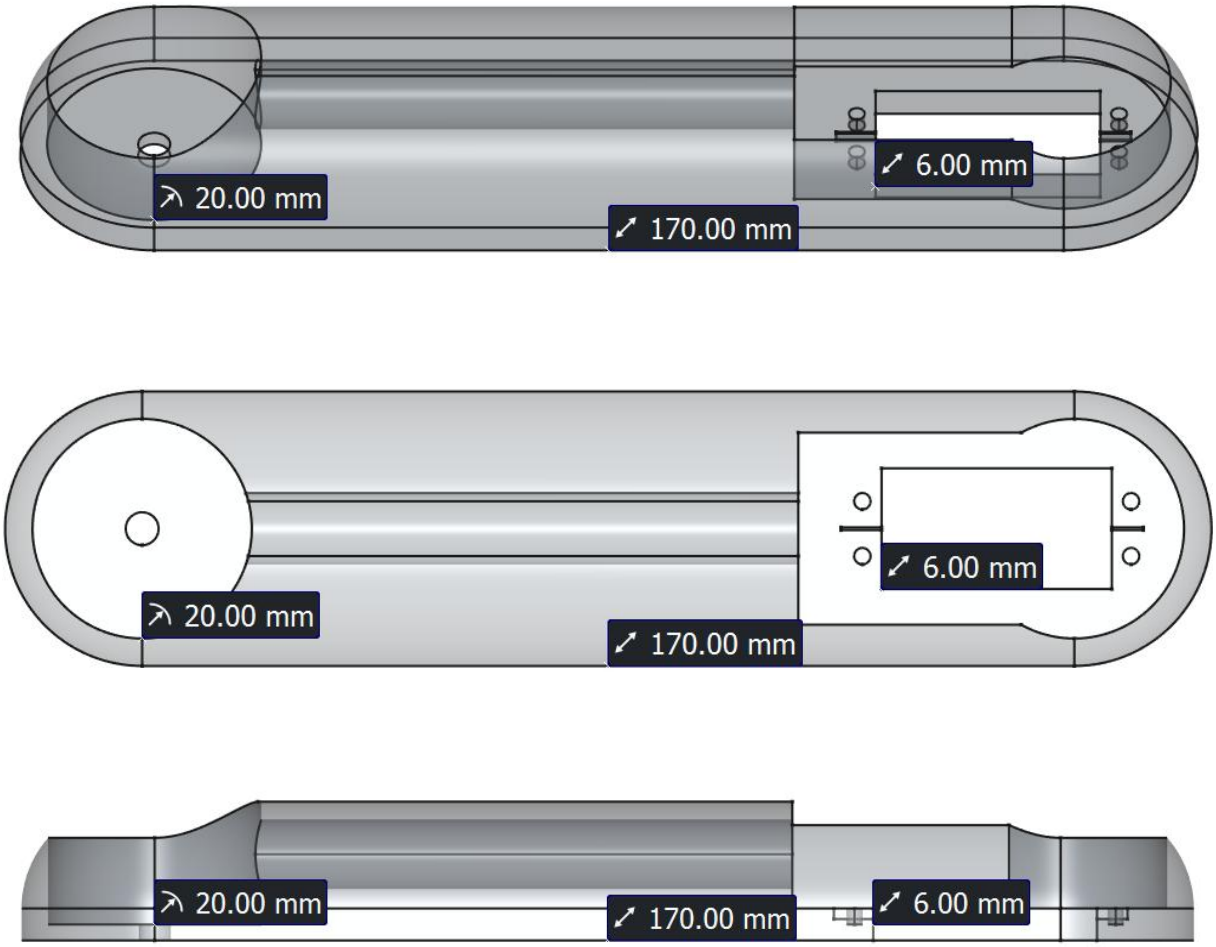
FreeCAD is a general-purpose, open-source parametric 3D computer-aided design (CAD) modeler that serves as the primary tool for the mechanical design and simulation aspects of this project. Unlike direct modelers, FreeCAD allows for a feature-based workflow where designs are defined by parameters and constraints, making it exceptionally powerful for iterative engineering. This means that if a dimension of the robot's chassis needs to change—for example, to accommodate a slightly larger battery—the user can simply modify a single parameter in the history tree, and the entire 3D model will automatically regenerate to reflect this update without needing to be redrawn from scratch. The software is built on the robust Open CASCADE technology kernel and supports a modular architecture through "workbenches," which provide specialized tools for different tasks such as 2D sketching, 3D part design, architectural modeling, and robot simulation.

We have used FreeCAD software for 3D cad design of 3 Axis robotic arm, its assembly and testing. Although, actual model uses alumium links (arms), due to its low weight to strength ratio. This CAD design are intended to be implemented but due to some time constrained, we found it better to use aluminum frames for it, and surprisingly it proved to be a better decision. CAD models can be used to print the arm links using suitable material.

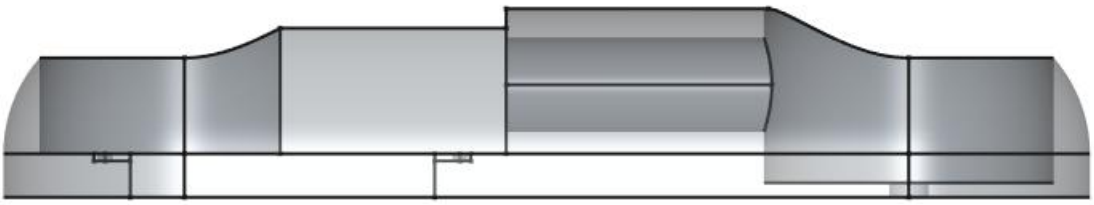
Base :

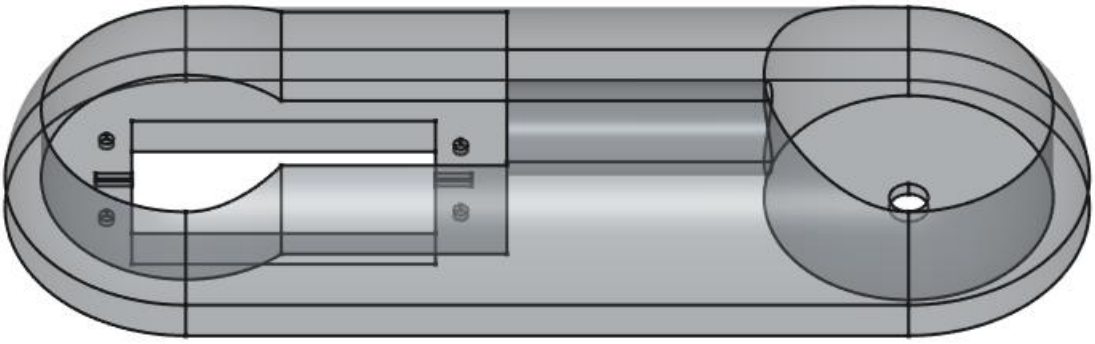


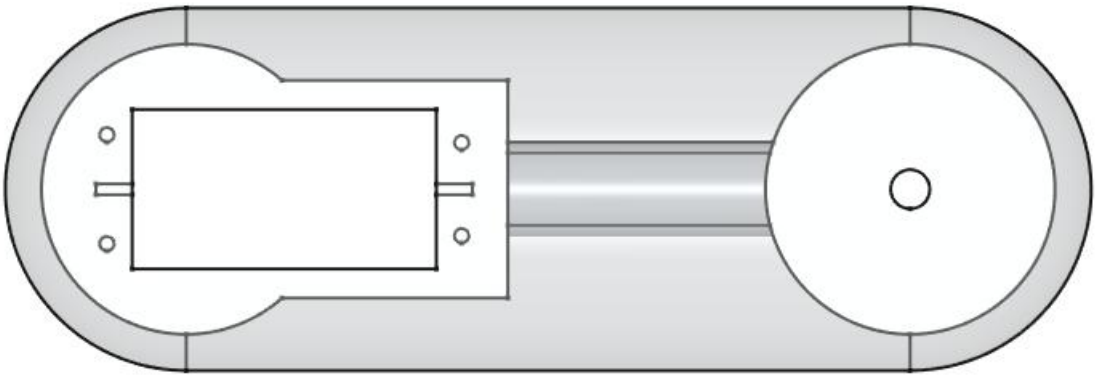
Shoulders Link :



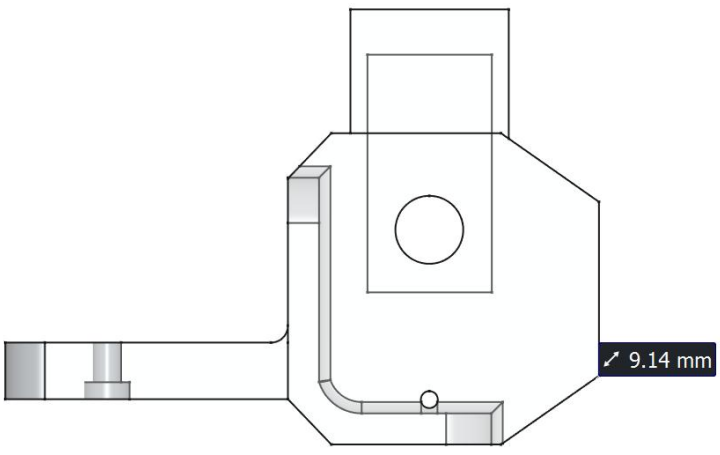
Elbow Link :

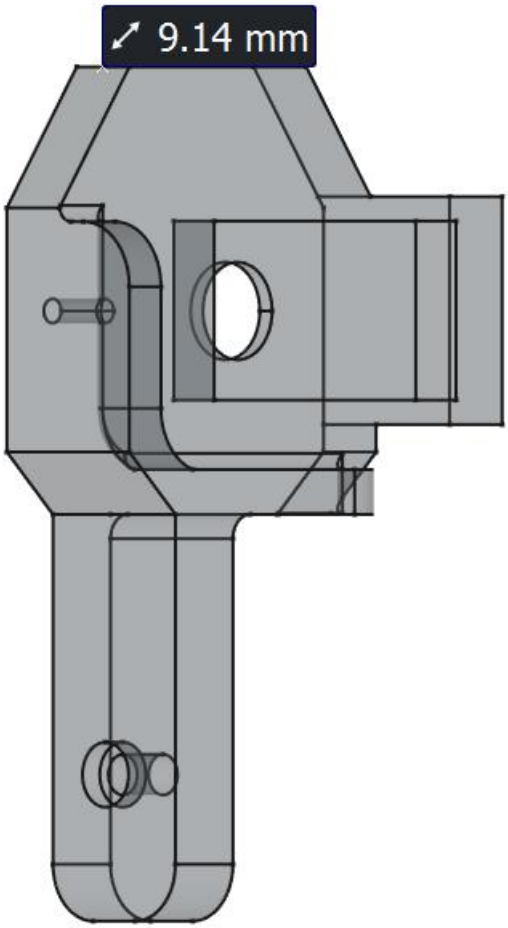


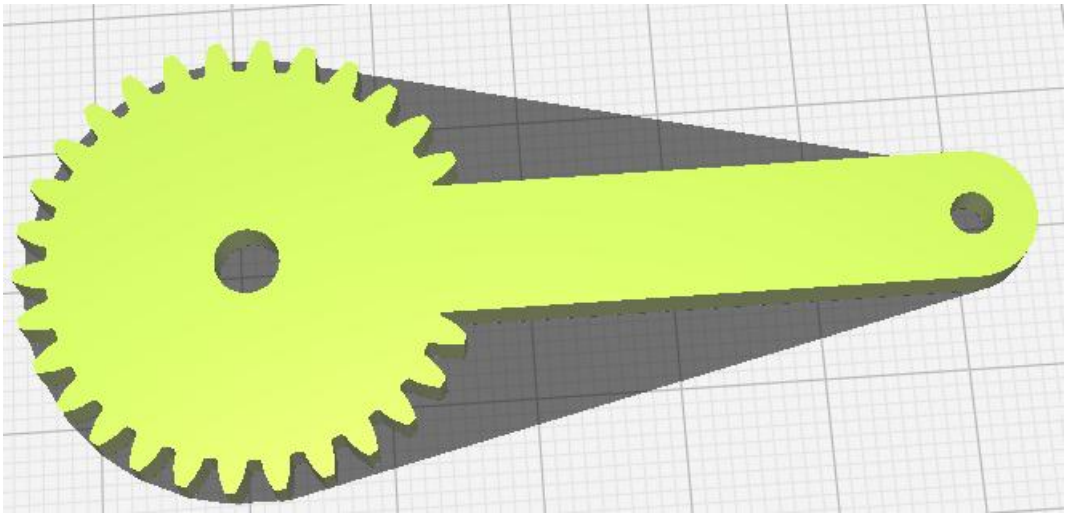


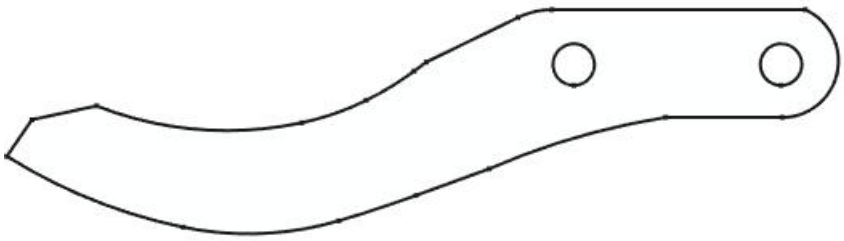


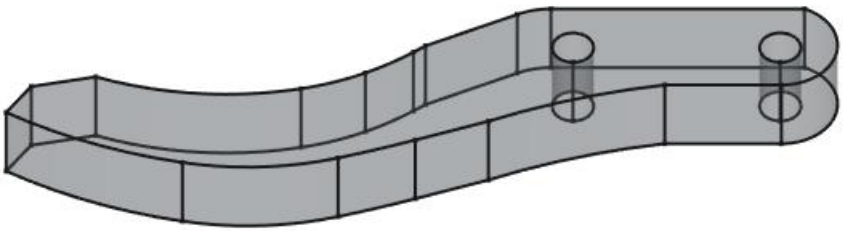
Gripper :



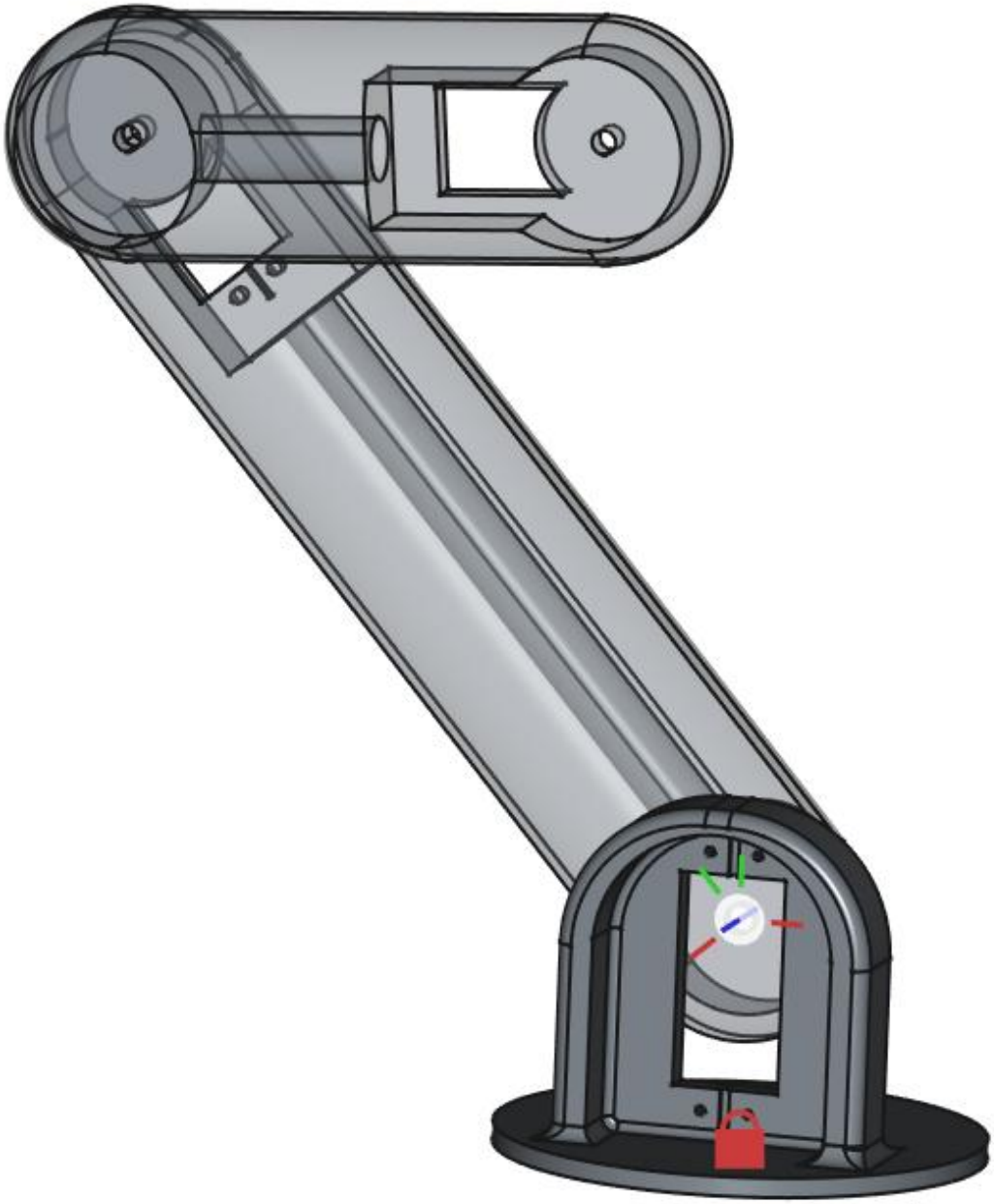

9.14 mm







4. 3-Axis Robotic ARM and Inverse Kinematics



Inverse Kinematics and Why its important

Inverse kinematics (IK) is a fundamental concept in robotics, computer graphics, and animation, playing a crucial role in how machines and characters move within their environments. At its core, inverse kinematics involves determining the necessary joint configurations of a mechanism to achieve a desired position and orientation of its end effector, such as a robot's hand or a character's limb. This is the inverse of forward kinematics, where joint parameters are given, and the position of the end effector is computed. Inverse kinematics is essential for creating realistic and functional movements, making it a key area of study and application.

The Mechanism Behind Inverse Kinematics :

To understand inverse kinematics, it's important to delve into the mechanics behind it. Consider a robotic arm with several joints and links. Each joint can move in specific ways, such as rotating or sliding, which in turn affects the position of the end effector. In inverse kinematics, the challenge lies in figuring out the angles and movements required at each joint to position the end effector at a given point in space. This task becomes increasingly complex as the number of joints increases, resulting in numerous possible configurations.

The mathematics involved typically includes solving sets of equations that relate joint angles to the position of the end effector. Techniques such as Jacobian matrices, optimization algorithms, and iterative solvers are often used to find solutions. The complexity of these calculations varies depending on the system's degrees of freedom and constraints imposed by its physical structure.

Application Of Inverse Kinematics :

Inverse kinematics is widely used in various fields, owing to its versatility and practicality. In robotics, IK is crucial for tasks such as pick-and-place operations, assembly, and surgical procedures, where precise control of the robot's end effector is necessary. It enables robots to perform complex tasks with accuracy, enhancing their utility in industrial and medical settings.

In the realm of computer graphics and animation, inverse kinematics is employed to create lifelike movements for characters. Animators use IK to ensure that a character's limbs move naturally, adhering to physical constraints and enhancing the realism of animations. This is particularly important in video games and films, where believable character motion is essential for immersive experiences.

Furthermore, inverse kinematics finds applications in biomechanics for analyzing human movements and designing prosthetics. By understanding the kinematics of the human body, researchers can develop better assistive devices and rehabilitation techniques, improving the quality of life for individuals with mobility impairments.

Challenges and Limitations :

Despite its wide-ranging applications, inverse kinematics presents several challenges. One of the primary difficulties is the existence of multiple solutions for the same end effector position, particularly in systems with many joints. This ambiguity can lead to undesirable or inefficient movements, requiring careful selection of solutions based on additional criteria such as energy efficiency or avoidance of obstacles.
Additionally, computational complexity can be a hurdle, especially in real-time applications where rapid responses are necessary. High-dimensional systems or those with complex constraints may require advanced algorithms and significant computational resources to achieve practical solutions.

Moreover, inverse kinematics fosters innovation in fields such as virtual reality, where it is used to track and replicate human movements in digital environments. As technology continues to advance, the significance of inverse kinematics is likely to grow, opening new avenues for exploration and application.

3.1. General Inverse Kinematics Problem

The general problem of IK is to find a solution or multiple solutions when a 4 × 4 homogeneous transformation matrix is given:

$$H_n^0 = \begin{bmatrix} R_n^0 & o_n^0 \\ 0 & 1 \end{bmatrix} \in \text{SE (3)} \qquad 3.1$$

As we know, the above homogenous transformation matrix provides us with the end-effector orientation (3 × 3 matrix) and position (a 3×1 vector providing the coordinates of the end-effector origin) with respect to the base frame.

We know from the forward kinematics chapter (Chapter 2) that for an n – DoF manipulator,

$$H = T_n^0 = A_1(q_1). \ . \ . \ A_n(q_n) \qquad 3.2$$

By using this relation, we can find the solution for joint variables $q_1, q_2, q_3, \ldots, q_n$

Example: To better understand the inverse kinematic problem, let us consider a 2 – link planar manipulator

Assumptions:

- $a_1 > a_2$
- revolute joints can revolve complete 180°

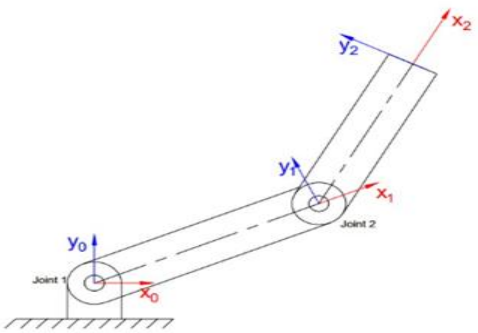


Fig 3.1. (a)

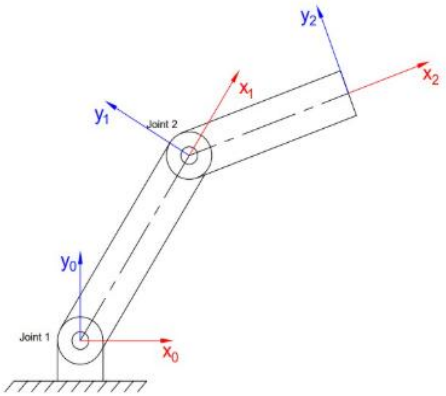
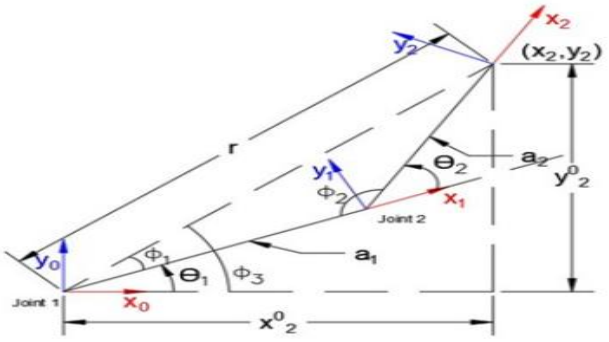


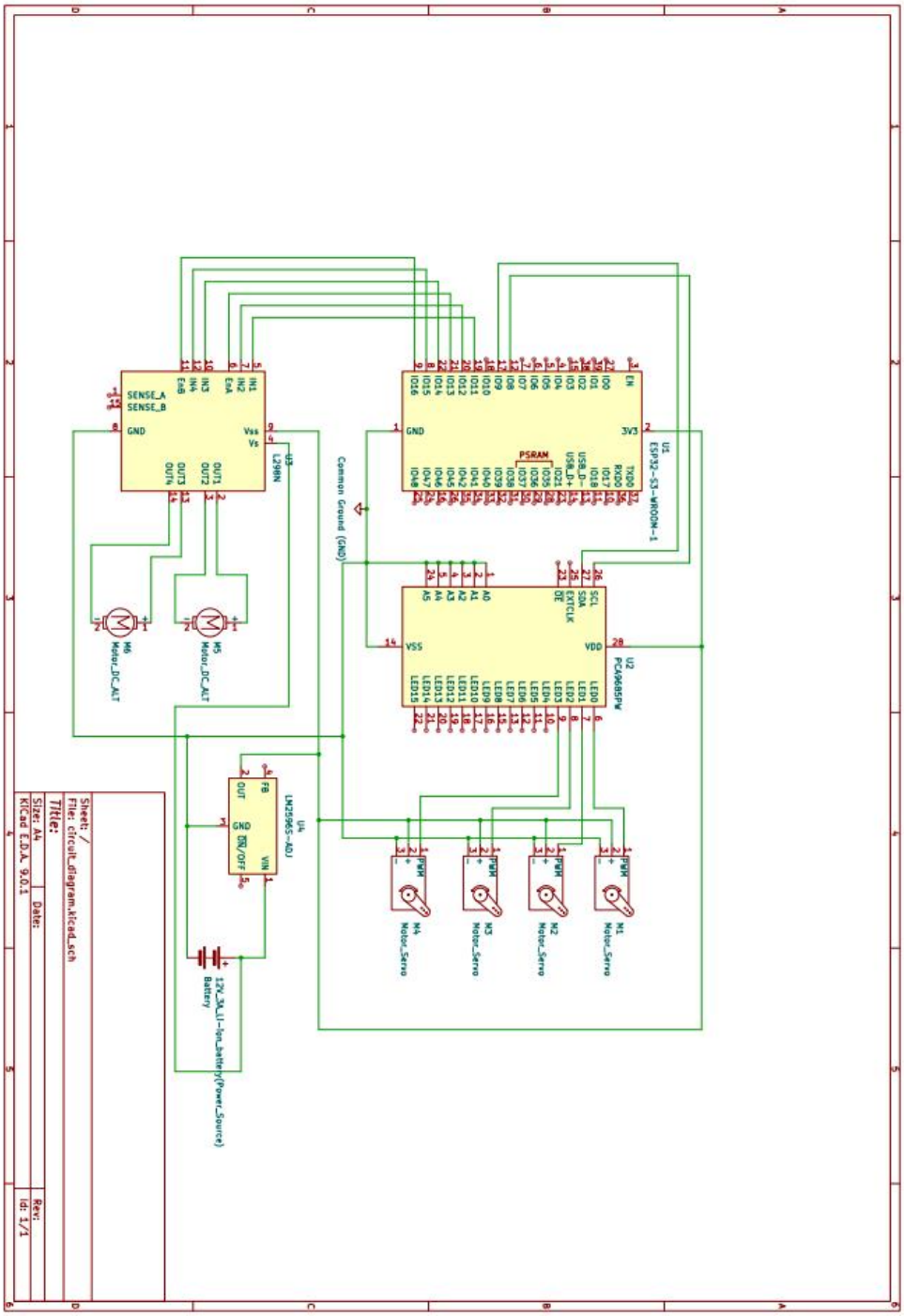
Fig 3.1. (b)

Fig 3.1. Revolute-Revolute (RR) planar robot manipulator where (a) Elbow-down configuration, (b) Elbow-up configuration

Denavit-Hartenberg (DH) Table:



5. Circuit Diagram



TECHNICAL SPECIFICATIONS

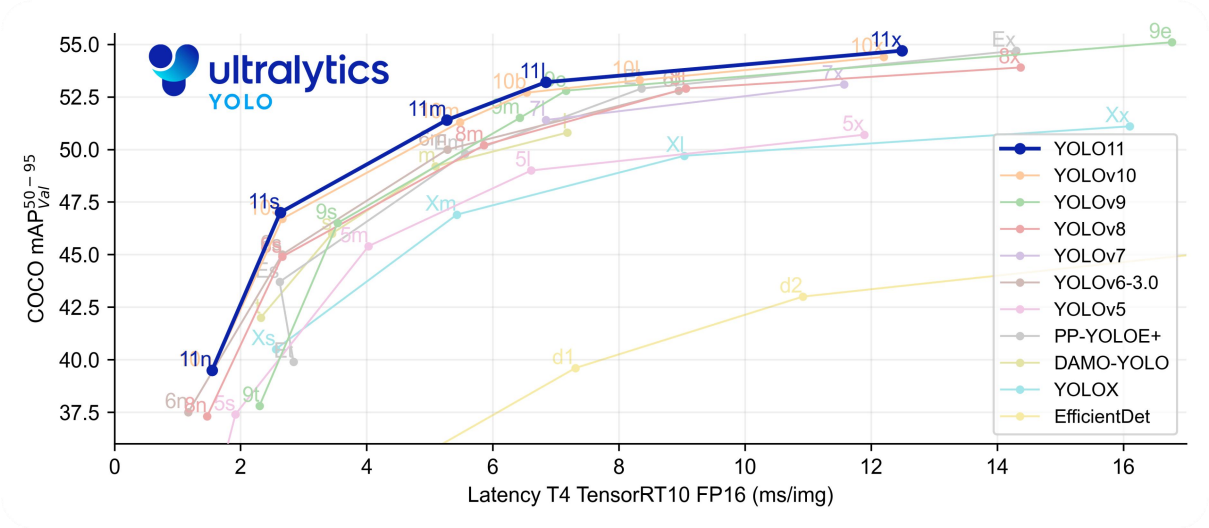
1. Object Detection and Image Classification

Object detection and image classification play a central role in the vision system of this robotics project. The robot relies on a dedicated ESP32-CAM module to capture a continuous video stream of its surroundings, which is sent to a Python-based processing server running on a laptop. This server functions as the primary vision and machine learning engine, responsible for interpreting what the robot sees in real time. Using modern deep learning models such as YOLO or MobileNet-based classifiers, the server analyzes each incoming frame to identify objects, classify items, and understand their position within the environment. This allows the robot to gain contextual awareness, enabling it to recognize common objects, obstacles, and target items relevant to user commands.

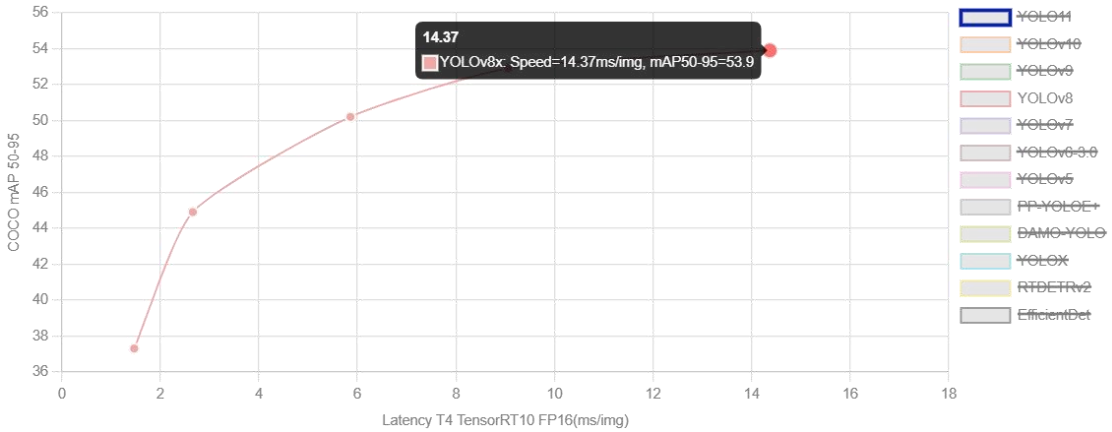
The object detection process goes beyond simply labeling what is present in the scene. It also locates objects with bounding box coordinates, extracts confidence values, and assesses spatial orientation. These details are essential because they allow downstream systems to make actionable decisions. For example, if a user instructs the robot to pick up an object, the vision module must accurately identify the item, confirm its presence, and provide positional cues so the robotic arm can be guided to the correct coordinates. By combining object detection with basic image classification, the system forms a reliable perception layer that transforms raw video data into structured and meaningful information.

As the Python server completes its analysis, it sends the classification and detection results back to the Express.js backend in the form of a compact JSON object. The backend then integrates these findings with a language model to convert them into user friendly descriptions. These descriptions help the user understand what the robot is seeing at any given moment. This interaction forms a continuous feedback loop between the robot and the user, where visual information is interpreted, summarized, and communicated through the web interface.

In this project, object detection and image classification are not used in isolation but rather as key components of a larger decision making pipeline. The goal is not only to identify objects but also to support navigation, manipulation, and higher level task execution. The vision system allows the robot to interact with its surroundings in an intelligent way by providing accurate environmental awareness. This capability is essential for enabling autonomous or semi autonomous actions, enhancing safety, and improving responsiveness to user commands.



Performance Matrix



Supported tasks and modes of YOLOv8 Model

The YOLOv8 series offers a diverse range of models, each specialized for specific tasks in computer vision. These models are designed to cater to various requirements, from object detection to more complex tasks like instance segmentation, pose/key-points detection, oriented object detection, and classification.

Each variant of the YOLOv8 series is optimized for its respective task, ensuring high performance and accuracy. Additionally, these models are compatible with various operational modes including Inference, Validation, Training, and Export, facilitating their use in different stages of deployment and development.

Model ↓↑	Filenames	Task	Inference	Validation	Training	Export
YOLOv8	`yolov8n.pt` `yolov8s.pt` `yolov8m.pt` `yolov8l.pt` `yolov8x.pt`	Detection	✅	✅	✅	✅
YOLOv8-seg	`yolov8n-seg.pt` `yolov8s-seg.pt` `yolov8m-seg.pt` `yolov8l-seg.pt` `yolov8x-seg.pt`	Instance Segmentation	✅	✅	✅	✅
YOLOv8-pose	`yolov8n-pose.pt` `yolov8s-pose.pt` `yolov8m-pose.pt` `yolov8l-pose.pt` `yolov8x-pose.pt` `yolov8x-pose-p6.pt`	Pose/Keypoints	✅	✅	✅	✅
YOLOv8-obb	`yolov8n-obb.pt` `yolov8s-obb.pt` `yolov8m-obb.pt` `yolov8l-obb.pt` `yolov8x-obb.pt`	Oriented Detection	✅	✅	✅	✅
YOLOv8-cls	`yolov8n-cls.pt` `yolov8s-cls.pt` `yolov8m-cls.pt` `yolov8l-cls.pt` `yolov8x-cls.pt`	Classification	✅	✅	✅	✅

Definition and Scope :

Object detection is a key computer vision task that extends beyond basic image classification. While image classification assigns a single label to an entire image (e.g., *"Car"*), object detection performs two simultaneous operations:

- **Classification:** Identifying what the object is (its category).
- **Localization:** Determining where the object is located within the image (its spatial coordinates).

Localization is typically achieved by predicting a **bounding box**, defined by coordinates (x, y, w, h), where (x, y)represent either the center or the top-left corner of the box, and (w, h)specify its width and height. Through this combination of classification and localization, object detection converts raw pixel data into structured, actionable information.

Core Architectures :

Modern object detection systems rely heavily on **Deep Learning**, particularly **Convolutional Neural Networks (CNNs)**. These architectures are broadly classified into two major categories:

Two-Stage Detectors (e.g., R-CNN, Faster R-CNN) Mechanism:

Two-stage detectors operate by first generating a set of *region proposals*—areas

likely to contain objects. A classifier then evaluates each proposed region to determine the object category and refine the bounding box.

Pros: High accuracy, strong localization precision, excellent performance in complex or cluttered environments.

Cons: Higher computational cost and latency, making them less suitable for real-time or high-speed applications.

Single-Stage Detectors (e.g., YOLO, SSD) Mechanism:

Single-stage detectors such as **YOLO** and **SSD** simplify the detection process by treating it as a single regression problem. The input image is divided into a grid, and for each grid cell, the model directly predicts bounding boxes, class probabilities, and confidence scores in a single forward pass.

Pros: Extremely fast, capable of processing video streams at high frame rates (FPS), ideal for safety-critical real-time systems like autonomous driving.

Cons: Slightly lower accuracy compared to two-stage models, especially for small or closely packed objects.

2. Voice Recognition System

The voice recognition system in this project serves as the primary human-robot communication interface, allowing users to control the robot and its mounted robotic arm using natural spoken language. The process begins at the React web application, where the browser's built-in speech recognition engine converts the user's spoken words into raw text. This text is then transmitted to the Express.js backend, where a language model interprets the sentence and converts it into a structured command format. The goal of this system is to translate unstructured human speech into actionable robotic instructions with high accuracy and minimal ambiguity.

From a technical and mathematical perspective, voice recognition can be described as a sequence-to-text mapping problem. The incoming voice signal x(t) is first sampled at a frequency Fs, typically in the range of 16 kHz to 44.1 kHz, producing a discrete signal x[n]. This discrete signal is processed using feature extraction techniques such as Mel Frequency Cepstral Coefficients (MFCC), where the frequency spectrum is mapped to the mel scale using the transformation mel = 2595 * log10(1 + f / 700). These features represent the spectral and temporal properties of the speech signal and form the input for the speech-to-text model, which may internally rely on sequence models or transformer based architectures to estimate the most probable word sequence W

for a given acoustic feature sequence A. Mathematically, the model attempts to find W* = argmax P(W|A), where P(W|A) is the conditional probability of the word sequence given the acoustic signal.

Once the speech is converted to text, the system performs intent recognition and command parsing. The language model analyzes the sentence structure, tokens, verbs, and entities to determine whether the instruction represents an actionable command for the robot. It extracts parameters such as movement direction, arm joint targets, object names, or reference positions. For example, a command such as "move the arm to the left" is converted into parameterized data like {"action": "arm_move", "direction": "left", "magnitude": 1}. This ensures that high level user language is mapped to low level robotic control variables that can be executed reliably by the ESP32-S3 microcontroller.

The voice recognition system is tightly integrated with the robot's perception and action pipeline. If a user's instruction involves objects or locations, the parsed text command is combined with real-time visual feedback coming from the object detection module. The system is designed so that voice commands can reference what the robot sees in its camera feed. In such cases, mathematical parameters derived from vision, such as bounding box coordinates (x, y, w, h) or detection confidence p in the range 0 to 1, are merged with parsed voice instructions before the robot begins executing a task. This coupling of voice and visual data enables more natural and effective control, turning simple speech input into nuanced and context-aware robotic actions.

Overall, the voice recognition system forms an intelligent gateway between the human operator and the robot's control architecture. By combining speech processing, probabilistic modeling, feature extraction, and structured command generation, it transforms raw human speech into precise control signals. This allows the robot to operate in a more intuitive and human-friendly manner, supporting hands-free interaction and enhancing the usability and versatility of the entire robotic platform.

3. Firmware and Controller Design

Setup And Configure Files :

A. pinConfig.h

```
#pragma once


// PCA9685 servo driver pins
#define SERVO_DRIVER_SCL 9
#define SERVO_DRIVER_SDA 8


// Arm control pins
#define ARM_BASE 0
#define ARM_ELBOW 1
#define ARM_WRIST 2
#define ARM_CLAW 3

// motor driver control pins
// motor 1
#define MOTOR_A_IN1 5
#define MOTOR_A_IN2 6
#define MOTOR_A_SPEED 10
//motor2
#define MOTOR_B_IN3 7
#define MOTOR_B_IN4 12
#define MOTOR_B_SPEED 11

```

B. Constants.h

```
#pragma once

// servo driver constants
#define SERVO_DRIVER_MIN 115
#define SERVO_DRIVER_MAX 615
#define SERVO_DRIVER_FREQ 50
```

c. Platformio.ini

```
; PlatformIO Project Configuration File
;
;   Build options: build flags, source filter
;   Upload options: custom upload port, speed and extra flags
;   Library options: dependencies, extra library storages
;   Advanced options: extra scripting
;
; Please visit documentation for the other options and examples
; https://docs.platformio.org/page/projectconf.html

[env:rymcu-esp32-s3-devkitc-1]
platform = espressif32
board = rymcu-esp32-s3-devkitc-1
framework = arduino
upload_port = COM8
monitor_speed = 115200
lib_deps = adafruit/Adafruit PWM Servo Driver Library@^3.0.2
build_flags =
    -I src/modules
    -I src/config
```

Motor Driver Class and Implementation :

A. Class Declaration :

```
#pragma once
#include <Arduino.h>
#include "pinConfig.h"

class MotorDriver {
private:
    // --- Configuration ---
    int maxSpeed;           // The speed limit (0-255)
    int accelerationStep;   // How much to change speed per update (Higher = Jerkier, Lower = Smoother)
    unsigned long lastUpdate;
    int updateInterval;     // Time in ms between speed updates

    // --- State Variables ---
    // We use signed integers (-255 to +255)
    // Positive = Forward, Negative = Backward, 0 = Stop
    int targetSpeedA;
    int currentSpeedA;

    int targetSpeedB;
    int currentSpeedB;
    int pushSpeed;

    // --- Hardware Helper ---
    // Handles the logic of IN1/IN2/EN pins for the L298N
    void applyHardware(int speed, int pin1, int pin2, int pinPWM);

public:
    // Constructor
    // defaultSpeed: The starting max speed (e.g., 230 for ~90%)
    MotorDriver(int defaultSpeed = 230);

    // Initialization
    void begin();

    // The Heartbeat: Must be called in loop() constantly
    void update();

    // --- Movement Commands ---
    void moveForward();
    void moveBackward();
    void stop(); // Smooth braking

    // --- Differential Steering ---
```

```
    // --- Differential Steering ---
    void turnLeft();  // Left wheel back, Right wheel fwd
    void turnRight(); // Left wheel fwd, Right wheel back

    // --- Settings ---
    void setMaxSpeed(int speed);
    void setPushSpeed(int speed);
};
```

B. Implementaion :

```cpp
#include "MotorDriver.h"

// Constructor
MotorDriver::MotorDriver(int defaultSpeed) {
    // Initialize settings
    maxSpeed = constrain(defaultSpeed, 0, 255);
    pushSpeed = 220;

    // TUNING PARAMETERS:
    accelerationStep = 5;   // Add 5 PWM units per tick

    // Reset states
    lastUpdate = 0;
    targetSpeedA = 0;
    currentSpeedA = 0;
    targetSpeedB = 0;
    currentSpeedB = 0;
}

void MotorDriver::begin() {
    // Setup Motor A (Left)
    pinMode(MOTOR_A_IN1, OUTPUT);
    pinMode(MOTOR_A_IN2, OUTPUT);
    pinMode(MOTOR_A_SPEED, OUTPUT); // PWM Pin

    // Setup Motor B (Right)
    pinMode(MOTOR_B_IN3, OUTPUT);
    pinMode(MOTOR_B_IN4, OUTPUT);
    pinMode(MOTOR_B_SPEED, OUTPUT); // PWM Pin

    // Ensure stopped at boot
    stop();
}

// The Non-Blocking Update Logic
void MotorDriver::update() {
    delay(20);

    // Motor A
    if (currentSpeedA < targetSpeedA) {
        currentSpeedA += accelerationStep;
        if (currentSpeedA > targetSpeedA) currentSpeedA = targetSpeedA; // Don't overshoot
    }
    else if (currentSpeedA > targetSpeedA) {
        currentSpeedA -= accelerationStep;
        if (currentSpeedA < targetSpeedA) currentSpeedA = targetSpeedA;
    }

    // Motor B
    if (currentSpeedB < targetSpeedB) {
        currentSpeedB += accelerationStep;
        if (currentSpeedB > targetSpeedB) currentSpeedB = targetSpeedB;
```

```
52          if (currentSpeedB > targetSpeedB) currentSpeedB = targetSpeedB;
53      }
54      else if (currentSpeedB > targetSpeedB) {
55          currentSpeedB -= accelerationStep;
56          if (currentSpeedB < targetSpeedB) currentSpeedB = targetSpeedB;
57      }
58
59
60      applyHardware(currentSpeedA, MOTOR_A_IN1, MOTOR_A_IN2, MOTOR_A_SPEED);
61      applyHardware(currentSpeedB, MOTOR_B_IN3, MOTOR_B_IN4, MOTOR_B_SPEED);
62  }
63
64  // Helper to translate -255 to 255 into L298N signals
65  void MotorDriver::applyHardware(int speed, int pin1, int pin2, int pinPWM) {
66      int pwmValue = abs(speed); // PWM must be positive
67
68      if (speed > 0) {
69          // Forward Configuration
70          digitalWrite(pin1, HIGH);
71          digitalWrite(pin2, LOW);
72      }
73      else if (speed < 0) {
74          // Backward Configuration
75          digitalWrite(pin1, LOW);
76          digitalWrite(pin2, HIGH);
77      }
78      else {
79          // Stop / Brake
80          digitalWrite(pin1, LOW);
81          digitalWrite(pin2, LOW);
82      }
83
84      // Send speed signal
85      analogWrite(pinPWM, pwmValue);
86  }
87
88  // --- Command Implementations ---
89
90  void MotorDriver::setMaxSpeed(int speed) {
91      maxSpeed = constrain(speed, 0, 255);
92  }
93  void MotorDriver::setPushSpeed(int speed){
94      pushSpeed = constrain(speed,0,255);
95  }
96
97  void MotorDriver::stop() {
98      targetSpeedA = 0;
99      targetSpeedB = 0;
100 }
101
```

```
101
102 void MotorDriver::moveForward() {
103     applyHardware(pushSpeed, MOTOR_A_IN1, MOTOR_A_IN2, MOTOR_A_SPEED);
104     applyHardware(pushSpeed, MOTOR_B_IN3, MOTOR_B_IN4, MOTOR_B_SPEED);
105     delay(500);
106     targetSpeedA = maxSpeed;
107     targetSpeedB = maxSpeed;
108 }
109
110 void MotorDriver::moveBackward() {
111     applyHardware(-pushSpeed, MOTOR_A_IN1, MOTOR_A_IN2, MOTOR_A_SPEED);
112     applyHardware(-pushSpeed, MOTOR_B_IN3, MOTOR_B_IN4, MOTOR_B_SPEED);
113     delay(500);
114     targetSpeedA = -maxSpeed;
115     targetSpeedB = -maxSpeed;
116 }
117
118 // Differential Steer: Rotate in place
119 void MotorDriver::turnLeft() {
120     // Left wheel goes back, Right wheel goes forward
121     applyHardware(-pushSpeed, MOTOR_A_IN1, MOTOR_A_IN2, MOTOR_A_SPEED);
122     applyHardware(pushSpeed, MOTOR_B_IN3, MOTOR_B_IN4, MOTOR_B_SPEED);
123     delay(500);
124     targetSpeedA = -maxSpeed;
125     targetSpeedB = maxSpeed;
126 }
127
128 void MotorDriver::turnRight() {
129     // Left wheel goes forward, Right wheel goes back
130     applyHardware(pushSpeed, MOTOR_A_IN1, MOTOR_A_IN2, MOTOR_A_SPEED);
131     applyHardware(-pushSpeed, MOTOR_B_IN3, MOTOR_B_IN4, MOTOR_B_SPEED);
132     delay(500);
133     targetSpeedA = maxSpeed;
134     targetSpeedB = -maxSpeed;
135 }
```

Robotic ARM Class and Implementation :

A. Class Decleration (roboticArm.h)

```
2  #pragma once
3
4  #include <Adafruit_PWMServoDriver.h>
5  #include "pinConfig.h"
6  #include "constants.h"
7
8
9  class RoboticArm {
10 private:
11
12     Adafruit_PWMServoDriver pwm;
13     int stepDelay ;
14     int homeConfig[];
15
16     // helper to convert angle into PWM pulse
17     int AngleToPulse(int angle);
18     // common servo handler
19     void setServo(int channel, int step,int targetAngle);
20
```

```
public:
    // Constructor
    RoboticArm();

    // Initialization
    void begin();

    // Basic Movements
    void setBase(int angle);
    void setELbow(int angle);
    void setWrist(int angle);
    void setClaw(int angle);

    // home position
    void homePosition();
};
```

B. Implementation (roboticArm.cpp)

```
// src/modules/RoboticArm.cpp
#include "roboticArm.h"
#include <Wire.h>
#include <Arduino.h> // Needed for 'map' function

// Constructor: Initialize the Adafruit driver object
RoboticArm::RoboticArm()
{
    pwm = Adafruit_PWMServoDriver();
    stepDelay = 50;
    homeConfig[0] = 140;
    homeConfig[1] = 135;
    homeConfig[2] = 20;
}

void RoboticArm::begin()
{
    pwm.begin();
    pwm.setOscillatorFrequency(27000000);
    pwm.setPWMFreq(SERVO_DRIVER_FREQ);
    Serial.println("Robotic arm check : ok , taking homeConfig");

    // Move to safe position immediately on startup
    homePosition();
}
void RoboticArm::setServo(int servoNum, int stepDelay, int targetAngle)···
// Helper definition
int RoboticArm::AngleToPulse(int angle)
{

    // Map degrees to pulse length (standard Arduino map function)
    return map(angle, 0, 180, SERVO_DRIVER_MIN, SERVO_DRIVER_MAX);
}

// Joints control
void RoboticArm::setBase(int angle)
{
    // pwm.setPWM(ARM_BASE, 0, AngleToPulse(angle));
    setServo(ARM_BASE,stepDelay,angle);
}

void RoboticArm::setELbow(int angle)
{
    setServo(ARM_ELBOW,stepDelay,angle);
}
```

```
81
82  void RoboticArm::setWrist(int angle)
83  {
84      setServo(ARM_WRIST,stepDelay,angle);
85  }
86
87  // --- High Level Actions ---
88  void RoboticArm::setClaw(int angle)
89  {
90      setServo(ARM_CLAW,stepDelay,angle);
91  }
92
93  void RoboticArm::homePosition()
94  {
95      setBase(homeConfig[ARM_BASE]);
96      setELbow(homeConfig[ARM_ELBOW]);
97      setWrist(homeConfig[ARM_WRIST]);
98      setClaw(0);
99  }
```

Main Entry Point (main.cpp) :

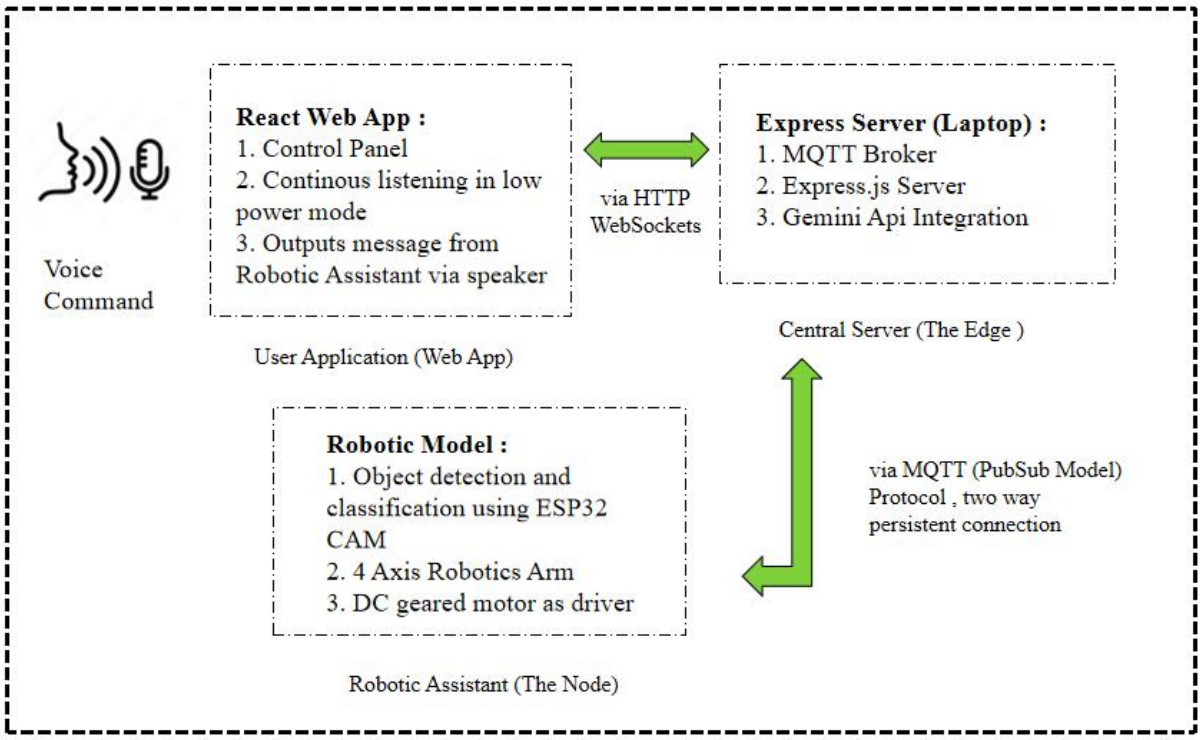
Main entry point of this program includes, route handling and command execution.

```ino
#include <Arduino.h>
#include <WiFi.h>
#include <WebServer.h>
#include "roboticArm.h"
#include "motorDriver.h"

RoboticArm arm;
MotorDriver engine(230);

const char *ssid = "ANUPJAISWAL 8480";     // projectee
const char *password = "anupj1234"; // 12345678ee
// ----------------------------------------

//WebServer object that will listen on port 80
WebServer server(80);

// Arm related req handler
void handleServoControl()
{

  String response = "OK"; // Default response to send back to the app

  if (server.hasArg("s") && server.hasArg("a"))
  {
    int servoNum = server.arg("s").toInt();
    int angle = server.arg("a").toInt();
    int step = server.arg("step").toInt();

    Serial.print("Received command for Servo ");
    Serial.print(servoNum);
    Serial.print(" -> Angle ");
    Serial.println(angle);

    switch (servoNum)
    {
    case 0:
      arm.setBase(angle);
      break;
    case 1:
      arm.setELbow(angle);
      break;
    case 2:
      arm.setWrist(angle);
      break;
    case 4:
      arm.setClaw(angle);
      break;
    case -1:
      arm.homePosition();
    }
  }
  else
```

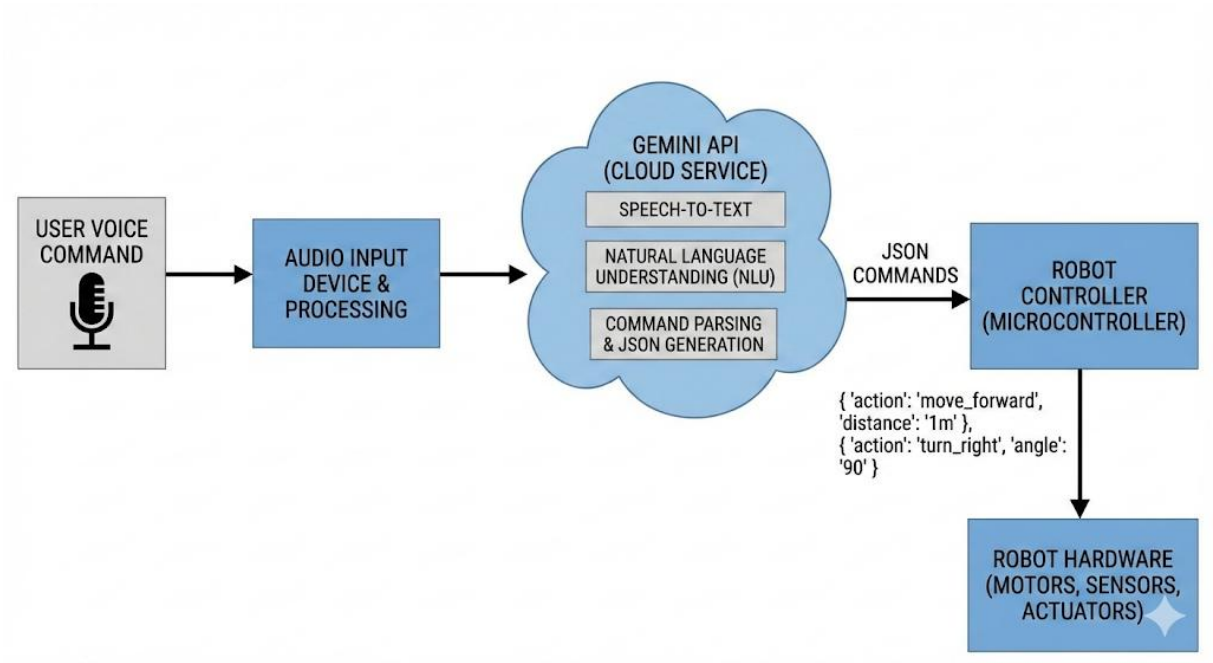
```
  {
    response = "Error: Missing parameters";
  }

  // Send a response back to the phone app
  server.send(200, "text/plain", response);
}

void handleMove()
{
  Serial.println("Moving engine !!");
  if (server.hasArg("d") && server.hasArg("s") && server.hasArg("ps"))
  {
    int d = server.arg("d").toInt();
    int s = server.arg("s").toInt();
    int ps = server.arg("ps").toInt();
    engine.setMaxSpeed(s);
    engine.setPushSpeed(ps);
    if (d == 1)
      engine.moveForward();
    else if (d == -1)
      engine.moveBackward();
    else if(d== 0)
      engine.stop();

    server.send(200, "text/plain", "OK");
  }
}

void handleTurn()
{
  Serial.println("turning engine !!");
  if (server.hasArg("d") && server.hasArg("s") && server.hasArg("ps"))
  {
    int d = server.arg("d").toInt();
    int s = server.arg("s").toInt();
    int ps = server.arg("ps").toInt();
    engine.setMaxSpeed(s);
    engine.setPushSpeed(ps);
    if (d == 1)
      engine.turnLeft();
    else if (d == -1)
      engine.turnRight();
    else if(d==0)
      engine.stop();

    server.send(200, "text/plain", "OK");
```

```
    server.send(200, "text/plain", "OK");
  }
}

void setup()
{
  Serial.begin(115200);
  delay(10);

  // --- Connect to Wi-Fi ---
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected!");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP()); // <-- IMPORTANT! WRITE THIS DOWN!

  server.enableCORS(true);
  server.on("/",HTTP_GET, [](){
    server.send(200,"text/plain","Server is running");
  });
  server.on("/servo",HTTP_GET,handleServoControl );
  server.on("/move", HTTP_GET, handleMove);
  server.on("/turn", HTTP_GET, handleTurn);

  // This catches the browser's security check
  server.onNotFound([]()
                    {
      if (server.method() == HTTP_OPTIONS) {
          server.send(200); // OK, go ahead!
      } else {
          server.send(404, "text/plain", "Not found");
      } });

  server.begin();

  // Initialize the PCA9685
  Wire.begin(8, 9);
  arm.begin();
  engine.begin();
  Serial.println("HTTP server started");

}

// ...existing code...

void loop()
{
  server.handleClient();
  engine.update();
}
```

4. Flow of Information



5. Command generation and Interpretation



SUMMARY AND CONCLUSION

1. Future Scope

The "Future Scope" section of your project report is crucial because it demonstrates that you are thinking beyond the prototype stage. It shows the evaluators that you understand the current limitations and have a roadmap for real-world deployment.

Here are three distinct paragraphs covering **Hardware Evolution**, **Software/AI Enhancements**, and **Real-World Integration**.

1. Transition to Fully Edge-Based AI Computing
Currently, the system relies on a local server (laptop) to perform heavy computational tasks such as speech recognition and object detection. While effective for a prototype, this dependency limits the robot's portability and range. A primary future objective is to migrate these AI workloads directly onto the robot using more powerful embedded computers, such as the NVIDIA Jetson Nano or the Raspberry Pi 5 with an AI accelerator. This transition would eliminate the need for a separate laptop, making the robot a fully self-contained, autonomous unit. By processing data locally on the device, the system would achieve lower latency, greater reliability in areas with poor Wi-Fi connectivity, and enhanced privacy by keeping voice and video data within the robot itself.
2. Integration of Advanced Depth Sensing and SLAM
The current reliance on a 2D camera and ultrasonic sensors provides basic functional navigation but lacks true environmental awareness. Future iterations will integrate sophisticated depth-sensing technologies, such as LiDAR (Light Detection and Ranging) or RGB-D stereo cameras (like Intel RealSense). These sensors would enable the implementation of Simultaneous Localization and Mapping (SLAM) algorithms, allowing the robot to create a digital map of a user's home in real-time. With SLAM, the robot could navigate complex floor plans more intelligently, plan optimal paths around dynamic obstacles, and understand specific locations (e.g., "Go to the kitchen" or "Bring this to the bedroom"), vastly increasing its utility as a home assistant.
3. Enhanced Manipulation and Human-Robot Interaction
The existing 3-DOF robotic arm is sufficient for simple pick-and-place tasks but is limited in its dexterity and reach. Future developments will focus on upgrading to a 5-DOF or 6-DOF manipulator with a smart adaptive gripper capable of handling a wider variety of household objects, from delicate glasses to heavy books. Furthermore, the human-robot interaction layer will be expanded to include bi-directional communication. Instead of just receiving commands, the robot will be equipped with a text-to-speech engine to provide verbal feedback (e.g., "I have found the medicine," or "The path is blocked"). This conversational capability will make the system more intuitive and

comforting for elderly users, fostering a sense of companionship alongside its functional assistance.

2. Improvements

Here are descriptive paragraphs for the "Improvements and Advancements" section of your report. These focus on refining the *current* prototype's performance and functionality, distinguishing them from the broader "Future Scope."

Improvements and Advancements

1. Implementation of Closed-Loop Odometry and Slip Compensation
A significant improvement for the current skid-steer chassis would be the integration of wheel encoders and a 9-axis Inertial Measurement Unit (IMU). Currently, the robot relies on open-loop timing for turns (e.g., "rotate for 500ms"), which is prone to errors due to wheel slippage on different terrains like carpet or tile. By adding magnetic encoders to the motor shafts and fusing this data with gyroscope readings from an IMU, the control system could dynamically calculate and correct for wheel slip in real-time. This "closed-loop" control strategy would allow the robot to execute precise turns (e.g., exactly 90 degrees) regardless of the floor surface, significantly improving the accuracy of the arm alignment process without requiring expensive visual navigation sensors.
2. Integration of Force Feedback and Soft Robotics
To enhance the robotic arm's capability to handle delicate or irregular objects, the gripper mechanism can be advanced by incorporating force-sensitive resistors (FSRs) or current-sensing algorithms. In the current iteration, the servo closes the claw to a fixed angle, which risks crushing fragile items like plastic cups or dropping heavy ones. By implementing force feedback, the robot can sense the pressure being applied to an object and adjust its grip strength automatically—stopping the motor the instant it feels sufficient resistance. Furthermore, replacing the rigid 3D-printed claws with silicone-based soft robotic fingers would increase friction and adaptability, allowing the assistant to securely grasp objects of unpredictable shapes, such as fruits or glass bottles, which are common in a household setting.
3. Expansion into a Centralized Smart Home Hub
While the robot currently operates as a standalone device, a major advancement would be to integrate it into the broader Smart Home IoT ecosystem using protocols like MQTT or Zigbee. Instead of just physically manipulating objects, the robot could act as a mobile voice interface for the entire house. For example, if a user says, "It is too dark to read," the robot could communicate directly with smart bulbs to increase brightness rather than trying to physically switch on a lamp. This advancement transforms the robot from a simple fetch-and-carry machine into a holistic home automation manager, capable of bridging the gap between physical tasks and digital environmental control for users with limited mobility.
4. Autonomous Charging and Energy Management

To achieve true independence from human intervention, the power system can be improved by developing an autonomous docking station. Currently, the user must manually plug in the charger, which defeats the purpose of an assistant for someone with severe motor disabilities. An advanced iteration of this project would feature a contact-based charging dock with an infrared (IR) beacon or ArUco marker. When the battery voltage drops below a critical threshold (monitored by the ESP32), the robot would automatically interrupt its current task, navigate to the charging station using visual homing, and align its charging contacts. This self-sustaining energy cycle would allow the robot to remain operational indefinitely, ensuring it is always ready to assist when needed.

REFERENCES

[1] J. Williams and P. H. Chen, “Assistive robotics for independent living: A review of technological trends,” *IEEE Transactions on Human-Machine Systems*, vol. 49, no. 4, pp. 314–326, 2019.

[2] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, and G. Moore, *The HTK Book (for HTK Version 3.5)*, Cambridge University Engineering Department, 2015.

[3] A. Graves, N. Jaitly, and A. Mohamed, “Hybrid speech recognition with deep bidirectional LSTM networks,” in *Proceedings of IEEE Automatic Speech Recognition and Understanding (ASRU)*, 2013, pp. 273–278.

[4] A. Vaswani et al., “Attention is all you need,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 5998–6008.

[5] M. Tan, T. Le, and Q. Zhang, “Voice-controlled assistants for elderly and disabled users: A survey,” *International Journal of Human–Computer Interaction*, vol. 36, no. 17, pp. 1625–1648, 2020.

[6] N. Sharkey and A. Sharkey, “Granny and the robots: Ethical issues in robot care for the elderly,” *AI & Society*, vol. 26, pp. 327–337, 2011.

[7] Y. Li and S. Hee, “Speech recognition challenges among elderly individuals: A systematic review,” *Journal of Voice*, vol. 35, no. 4, pp. 618–627, 2021.

[8] T. Ince and H. Kilic, “Voice-controlled wheelchair for disabled people using Arduino,” in *International Conference on Engineering Technologies (ICENTE)*, 2018, pp. 1–5.

[9] S. Ghosh, P. K. Dutta, and S. Banerjee, “Deep learning-based speech-to-text system for assistive applications,” *Procedia Computer Science*, vol. 167, pp. 1971–1980, 2020.

[10] https://eureka.patsnap.com

[11] https://opentextbooks.clemson.edu/wangrobotics

[12] https://robu.in/product/16-channel-12-bit-pwmservo-driver-i2c-interface-pca9685-arduino-raspberry-pi/

[13] https://www.coimbatronics.in/index.php?route=product/product&product_id=105

[14] https://dronebotworkshop.com/tb6612fng-h-bridge/

[15] https://grabcad.com/library/robotic-arm-420