Republic of Yemen
Ministry of Higher Education
And Scientific Research
Sana'a University
Faculty of Engineering

AI and Computer Vision Based Humanoid Robot

Zyad Abdullah Alshuja ID: 201970233

Supervised by

Prof Eng. Khalil Al-Hatab
Vice Dean for Student's Affairs

Dr. Sami Al-Maqtari
PhD, Computer Science and Control Engineering

A graduation project document submitted to the department of Mechatronics in partial fulfillment to the requirements for Bachelor Degree in Engineering

2023

Summary

This graduation project focuses on the development and implementation of a sophisticated humanoid robot equipped with custom software. By combining various technologies such as object detection, face recognition, emotion analysis, gender and age determination, voice control, natural language processing (NLP), text-to-speech (TTS), and robotics operating system (ROS), the robot aims to serve as an advanced interactive platform.

Human-robot interaction (HRI) has always been a challenge, especially in achieving natural and intuitive communication. The project addresses this challenge by integrating various sensing, recognition, and response mechanisms to make human-like interactions possible, as well as adding a unique feature of sign language translation.

Main Objectives of the Graduation Project:

- Build and implement a robot with various computer vision technologies that has abilities for interaction with people
- Learning & experiencing about robots developing processes
- Design programming controlling
- Using new artificial intelligent technology and get advantages of them in real life applications.
- Applying ROS to implement humanoid robot.
- Implement the first advanced humanoid robot. In our country.
- Applying mechatronics concepts in this project such as experiencing 3d printing, Artificial Intelligent, and computer vision, electronics, and embedded system mechanical cad design and mechanism

This project utilizes a mix of software and hardware tools. The software, created using Python, employs OpenCV for computer vision, TensorFlow for machine learning, and ROS for robot control. On the hardware side, an Humanoid robot serves as the base, equipped with cameras for spotting objects, microphones for voice commands, and various sensors for recognizing gestures. This combination enables the robot to perform tasks like identifying objects, responding to voice instructions, and understanding human gestures.

The Humanoid robot successfully recognized faces with an accuracy of 95%, detected emotions, gender, and age with significant reliability. Voice control and natural conversation were achieved, enabling intuitive interactions. The sign language translator effectively recognized hand gestures, providing a new dimension of accessibility.

The project achieved its goal of creating an advanced Humanoid robot with multiple layers of interaction and recognition. It demonstrates the potential and applicability of such robots in diverse fields. Future work may include refining the algorithms for even higher accuracy, real-world testing, and exploration of additional applications, particularly in assisting individuals with disabilities.

Authorization

We authorize University of Sanaa Faculty of Engineering to supply copies of our graduation project report to libraries, organizations or individuals on request.

Name	I.D. No.	Signature
Mohammed Abdalnasser Alzaghir	201970211	
Ahmed Ramzi Alherwy	201973306	
Asaf Abdulrahman Abdullah	201873036	
Zyad Abdullah Alshuja	201970233	
Mohammed Fouad Alsurihi	201873412	
Ali Abdulrahman Alzaidy	201970346	

Date: September 16, 2025

Dedication

With immense pride and gratitude, we dedicate our graduation project to those who have been by our side throughout this remarkable journey. Your unwavering belief in our potential, your guidance through challenges, and your encouragement to reach higher have been our driving forces.

To our mentors, whose wisdom and expertise have shaped our project and enriched our understanding, we extend our deepest appreciation. Your patience and dedication to our growth have been truly inspiring.

To our families, friends, and loved ones, whose unwavering support and encouragement have lifted us through every hurdle, we dedicate our achievement. Your belief in us has been a constant source of strength.

To the journey itself, with its highs and lows, its lessons and triumphs, we offer our gratitude. It has been a transformative experience that has shaped us into resilient and forward-thinking individuals.

As we stand at this milestone, we carry forward the knowledge, skills, and values imparted by this endeavor. This dedication is a tribute to the collective efforts that have brought us here and a promise to continue pushing boundaries in pursuit of excellence.

With heartfelt gratitude,

Acknowledgement

We would like to express our deepest gratitude and appreciation to all those who have contributed to the successful completion of the Zeta Robot project, which served as our engineering graduation project at the Mechatronics Department of Sanaa University.

First and foremost, we are sincerely grateful to Dr. Khalil Al-Hatab and Dr. Sami Al-Maqtari, our esteemed supervisors, for their invaluable guidance, support, and expertise throughout the entire duration of the project. Their dedication, patience, and insightful feedback have been instrumental in shaping the project's direction and ensuring its successful execution.

We would also like to extend our heartfelt thanks to the faculty members of the Mechatronics Department for their continuous encouragement, valuable insights, and their commitment to fostering a nourishing academic environment. Their vast knowledge and willingness to share have greatly enriched our understanding of the subject matter.

We are indebted to our fellow classmates and colleagues who provided assistance, shared ideas, and collaborated on various aspects of the project. Their collective effort and enthusiasm have played a significant role in the realization of the Zeta Robot.

Furthermore, we are grateful to the technical staff and the administration of Sanaa University for their support and provision of necessary resources, enabling us to carry out the project effectively.

Last but not least, we would like to express our heartfelt appreciation to our families and friends for their unwavering support, encouragement, and understanding throughout this challenging endeavor. Their belief in us has been a constant source of motivation and inspiration.

To all those mentioned above and to anyone else who has contributed in one way or another, although not explicitly mentioned, we extend our sincere thanks and gratitude. Your collective efforts and contributions have been invaluable in making the Zeta Robot project a success.

Thank you all.

Supervisor Certification

We certify that the preparation of this project entitled

AI and Computer Vision Based Humanoid Robot, prepared by

Mohammed Abdalnasser Alzaghir, Ahmed Ramzi Alherwy, Asaf Abdulrahman Abdullah, Mohammed Fouad Alsurihi, Zyad Abdullah Alshuja, and Ali Abdulrahman Alzaidy.

Was made under my supervision at *Mechatronics* department in partial fulfillment to the requirements of /Bachelor Degree in *Engineering*.

Dr. Khalil Al-Hatab
Sign.
Date.

Dr. Sami Al-Maqtari
Sign.
Date.

Examination Committee

Project Title: AI and Computer Vision Based Humanoid Robot

Supervisor

No	Name	Position	Signature
1			
2			

Examination Committee

No	Name	Position	Signature
1			
2			
3			

Dr. Eng. Abdul-Malik E. Momin
Head of Mechatronics Department

Table Of Contents

List of Abbreviations

ROS	Robotic Operating System
AI	Artificial Intelligence
NLP	Natural Language Processing
WABOT	WAseda roBOT
URDF	Unified Robot Description Format
CAD	Computer Aided Design
FDM	Fused Deposition Modeling
PLA	Polylactic Acid
PETG	Polyethylene Terephthalate Glycol
LED	Lighting Emitting Diode
YOLOv8n	You Only Look Once version 8n
CNN	Convolutional Neural Network
API	Application Programming Interface
GPT	Generative Pre-trained Transformer
TTS	Text-To-Speech
AWS	Amazon Web Services
NER	Named Entity Recognition

List Of Tables

List Of Figures

Chapter 1
Introduction

Chapter 1 Introduction

1.1 Overview

This project represents a comprehensive effort to design and build an Humanoid robot that pushes the boundaries of human-robot interaction. Utilizing state-of-the-art technology in computer vision, natural language processing, and robotics, the robot is capable of object and face detection, emotion and gender recognition, age estimation, voice control, sign language translation, and more. This multifaceted integration marks a significant advancement in the field, offering possibilities for application in various domains.

1.2 Problem Statement

The challenge of creating a robot that can communicate and interact naturally with humans has driven the need for a multi-disciplinary approach. Current solutions often lack integration between different functionalities and fail to provide a truly interactive and intuitive experience. The project aims to bridge this gap by:

- Developing a seamless integration of object detection, facial recognition, emotional analysis, voice interaction, and more.
- Creating a system capable of understanding and responding in a human-like manner through speech and gesture.
- Addressing the limitations of existing technologies in providing a comprehensive and adaptable solution.

1.3 Project Objectives

The primary objectives of this project are:

- Build and implement a robot with various computer vision technologies that has abilities for interaction with people
- Learning & experiencing about robots developing processes
- Design programm for controlling
- Using new artificial intelligent technology and get advantages of them in real life applications.
- Applying ROS to implement humanoid robot.
- Implement the first advanced humanoid robot. In our country.
- Applying mechatronics concepts in this project such as experiencing 3d printing, Artificial Intelligent, and computer vision, electronics, and embedded system mechanical cad design and mechanism

1.4 Project Scope and Limitations

Scope:

- Designing and programming the Humanoid robot with diverse features.
- Integrating complex functionalities for a seamless experience.
- Evaluating performance through simulation and controlled environments.

Limitations:

- Real-world application may face challenges due to environmental factors.
- Hardware constraints may limit the robot's mobility or sensory precision.
- Ethical considerations around privacy and security in facial recognition.

1.5 Project Methodology

The project employed an iterative development approach, including:

- **Requirement Analysis**: Identifying and detailing the project's needs.
- **Design Phase**: Outlining the hardware and software components.
- **Development Phase**: Coding, integrating, and building the robot.
- **Testing Phase**: Simulating scenarios to evaluate the robot's performance.
- **Deployment Phase**: Proposing an implementation strategy for real-world application.
- **Tools Used**: Python, OpenCV, TensorFlow, ROS, Raspberry Pi, various sensors.

1.6 ROS (Robotic Operating System)

ROS (Robot Operating System) is an innovative open-source operating system that was initially developed in 2007 by the Stanford Artificial Intelligence Laboratory to support the Stanford AI Robot project [4]. It emerged as a crucial component of the project, aiming to streamline the development of robot software. ROS serves as a middleware that connects robot hardware with a computer's operating system, providing a flexible and efficient platform for software development. By utilizing a Publish/Subscribe model and facilitating interprocess communication through Topics, ROS revolutionizes the creation of robot software [5].

Figure 1-1 ROS Robotics Operating System-Noetic

The primary objective behind the creation of ROS was to empower robot developers by offering a comprehensive set of libraries and tools. Instead of starting from scratch, developers can leverage the existing functionalities and modules provided by ROS. This approach significantly reduces development time and effort, allowing for faster and more efficient creation of robot software. With ROS, developers can focus on building upon the existing framework, enhancing and customizing it to meet the specific requirements of their robot applications [4]-[5].

At the core of ROS lies its unique node-based architecture. In ROS, processes are called Nodes, which can perform various tasks and functionalities. These Nodes communicate with each other through the Publish/Subscribe model. A Node can act as a publisher, broadcasting data through Topics, or as a subscriber, receiving and utilizing data from specific Topics. This decoupled and modular approach enables developers to design and integrate complex robotic systems seamlessly [6].

ROS enables efficient and seamless communication between Nodes through the use of Topics. Topics serve as the communication medium through which data is exchanged. Publisher processes can publish one or multiple Topics, while subscriber processes can subscribe to specific Topics to receive the corresponding data. This flexible communication system allows for the creation of distributed and collaborative robot systems, where different components can exchange information and work in synchronization [6].

One of the key strengths of ROS is its open-source nature. Being an open-source operating system, ROS encourages collaboration, knowledge-sharing, and community-driven development. This fosters a vibrant ecosystem of robot developers, researchers, and enthusiasts, who contribute to the continuous improvement and expansion of ROS. The open-source aspect of ROS enables developers to access a vast collection of resources, including libraries, tools, and pre-existing code, facilitating rapid prototyping and development iterations [4]-[5].

ROS has emerged as a game-changer in the field of robot software development. By providing a powerful and flexible operating system, ROS empowers developers to create sophisticated robot software without reinventing the wheel. Its node-based architecture and efficient interprocess communication through Topics enable the seamless integration of various components and facilitate the development of complex robotic systems. Moreover, the open-source nature of ROS promotes collaboration and knowledge-sharing, driving innovation and advancement in the field. With ROS, the landscape of robot software development has been revolutionized, paving the way for future advancements in robotic technologies [44]-[6].

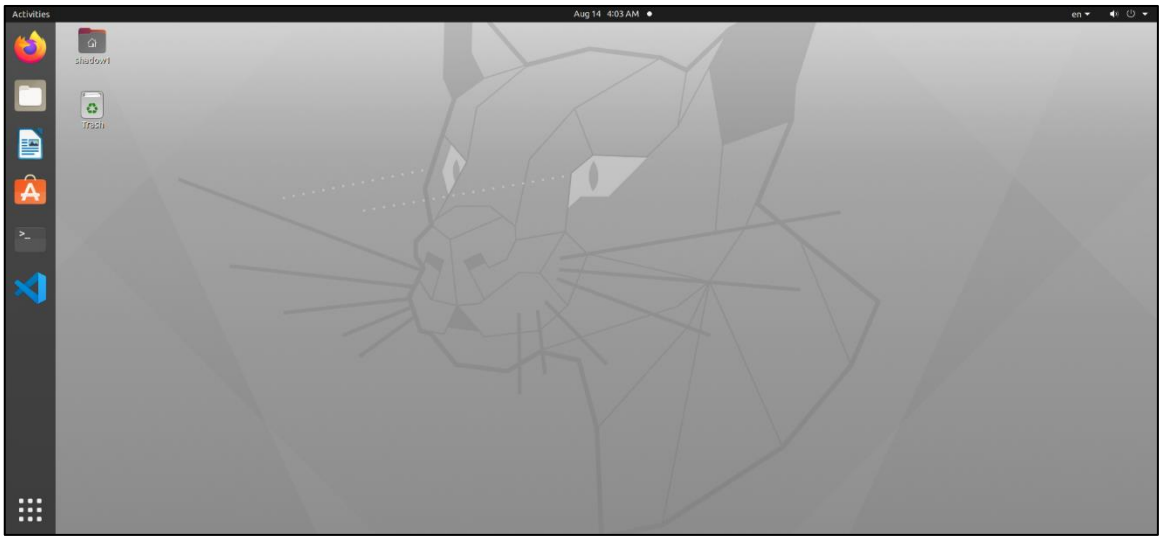


Figure 1-2 ROS Interface

Chapter 2
Literature Review

Chapter 2 Literature Review

2.1 Background

Humanoid robots represent a remarkable convergence of technology, engineering, and artificial intelligence, striving to replicate human-like attributes and behaviors. As advanced machines designed to resemble and interact with humans, humanoid robots have captured the imagination of researchers, engineers, and society at large. Their development has been driven by a desire to bridge the gap between machines and humans, facilitating natural interaction and cooperation.

The concept of humanoid robots traces its roots back to ancient mythology and cultural tales, where automata and animated beings were depicted. However, it wasn't until the mid-20th century that technological advancements enabled the actualization of humanoid robotics. Early attempts focused on creating robotic limbs capable of imitating human motion, gradually evolving to include full-scale humanoid structures.

The field of humanoid robotics gained significant attraction with the advent of artificial intelligence (AI) and computer vision technologies. AI brought the ability to process vast amounts of data, make informed decisions, and learn from experiences which all are qualities essential for providing robots with human-like cognitive abilities. Computer vision, on the other hand, empowered robots with the capability to perceive and interpret their environment through visual information, enabling them to navigate, recognize objects, and interact with their surroundings.

The integration of AI and computer vision into humanoid robots has unlocked unprecedented possibilities. By implementing AI algorithms such as machine learning and neural networks, these robots can autonomously adapt to changing circumstances, refine their actions, and improve their performance over time. Computer vision augments their sensory capabilities, enabling them to understand and respond to visual cues, thus enhancing their interaction with humans and the environment.

The application domains of humanoid robots span a wide spectrum of industries and fields. From healthcare and rehabilitation, where they assist individuals with mobility challenges, to education and entertainment, where they engage and charm audiences. Humanoid robots are revolutionizing the way we interact with technology. Furthermore, in industrial settings, these robots collaborate with humans in complex tasks, contributing to increased efficiency and safety.

The industrial landscape, too, has been transformed by the integration of AI and computer vision into humanoid robots. Collaborative robots or "cobots" work alongside human operators, implementing their AI-enhanced intelligence and computer vision capabilities to streamline manufacturing processes, enhance

precision, and ensure worker safety. This human-robot partnership is reshaping industries and redefining the nature of work.

However, the journey towards humanoid robots is not without challenges. Ethical considerations surrounding human-robot relationships, privacy concerns, and the potential impact on the workforce require careful examination. Technical challenges include the need for robust AI algorithms, efficient processing power and integration of complex sensory data.

In the subsequent sections of this literature review, we discuss and explore the synergy between AI, computer vision, and humanoid robotics. By reviewing existing research, innovations, and real-world applications. We aim to reveal the boundaries and limitations that prevent these technologies from spreading. Our endeavor is to uncover insights and draw a path toward a future where humanoid robots contribute to society and unlock new realms of technological possibilities.

2.2 Motivation

Our motivation is driven by several factors but mainly by the aspiration to establish a pioneer project in the field of robotics. By creating an innovative project that stands as inspiration of excellence, we seek to capture the attention of entrepreneurs and industry leaders, compelling them to recognize the transformative potential of humanoid robotics. This recognition, in turn, has the potential to bring out support for inventors, developers, and visionaries in this field. This way, a dynamic ecosystem will emerge to host innovation and creativity.

Moreover, our project endeavors to ignite a spark of interest among entrepreneurs, encouraging them to hold the possibilities presented by humanoid robotics. As this interest takes root, it may lead to increased investment in research, development, and practical applications within the realm of robotics. By demonstration the power of artificial intelligence, computer vision, and mechatronics engineering in shaping humanoid robots, our project aims to contribute to the emergence of a vibrant and collaborative community of professionals, enthusiasts, and investors.

2.2.1 Other Motivation Factors

In the realm of technological innovation, the development of humanoid robots stands as a testament to humanity's pursuit of pushing the boundaries of what is possible. However, our motivation for embarking on the journey of exploring AI and Computer Vision-Based Humanoid Robots extends beyond the realm of technological fascination. Rather, it is rooted in a fusion of societal aspirations, global participation, and a desire to apply the knowledge acquired throughout our academic study in the field of mechatronics engineering.

2.2.2 Lack of Interest in Robotics Development

We have recognized a noticeable gap in the advancement of robotics, particularly humanoid robots, within Yemen. Despite the rapid progression of robotics technology on a global scale, the domain of humanoid robots has received comparatively limited attention within our local context. This shortage of interest has not only delayed the potential for innovation but has also highlighted the need for dedicated efforts to surround this crucial technology with high interest and awareness.

2.2.3 A Global Aspiration for Evolution

The aspiration to participate in the evolution of humanoid robot development on a global stage further drives our endeavors. The wide impact of humanoid robots, from revolutionizing industries to enhancing human lives, resonates with us as a potential for positive change. By aligning ourselves with the international community of researchers, engineers, and enthusiasts in this field, we aim to contribute to the ever-expanding knowledge base and share insights to contribute in shaping the future of humanoid robotics.

2.2.4 Application of Mechatronics Engineering Knowledge

Our motivation also finds its roots in the conclusion of our dedicated study in mechatronics engineering. Throughout our academic journey, we have combined a comprehensive understanding of the interplay between mechanics, electronics, and computer science. It is with a sense of eagerness and responsibility that we seek to implement this accumulated knowledge and apply it to a concrete endeavor that involve the essence of modern interdisciplinary engineering – the creation of AI and Computer Vision-Based Humanoid Robots.

2.3 Foundations of Humanoid Robotics and AI

2.3.1 Definition of Humanoid Robots

A humanoid robot is a type of robot designed to resemble and imitate human characteristics and behaviors to varying degrees. It typically possesses physical attributes such as a human-like body structure, including arms, legs, a head, and often hands and feet, that allow it to interact with the environment and objects in a manner similar to humans. Additionally, humanoid robots often are intended to replicate human-like movement and gestures, enabling them to navigate and perform tasks in diverse settings.

While the level of human-likeness can vary widely, humanoid robots generally incorporate technologies from multiple disciplines, including mechanical engineering, electronics, artificial intelligence (AI) and computer vision. These technologies enable humanoid robots to perceive and understand their surroundings,

interact with humans and objects, and exhibit certain cognitive capabilities, often leading to more intuitive and natural human-robot interactions.

Humanoid robots have diverse applications, ranging from research and exploration to industry, healthcare, education, entertainment, and beyond. They have the potential to assist humans in various tasks, provide companionship, enhance human-machine communication, and contribute to advancements in AI, robotics, and human-robot interaction.

humanoid robots represent a convergence of engineering, technology, and human-inspired design, seeking to bridge the gap between machines and humans through their physical form and interactive abilities.

2.3.2 Characteristics of Humanoid Robots

1. **Human-like Appearance:** Humanoid robots are designed to resemble human anatomy to some extent, often featuring a head, torso, arms, legs, hands, and feet. Their appearance may include facial features, eyes, and even expressive elements to facilitate human-like interactions.
2. **Bipedal Locomotion:** Humanoid robots are typically built with the ability to walk on two legs, mimicking human locomotion. This bipedal movement allows them to navigate a wide range of environments and perform tasks that require human-like mobility. Humanoid robots can be seen in Fig.2-1.

Figure 2-1 Humanoid Robots credit by Boston dynamics

3. **Dexterous Manipulation:**

Figure 2-2 Dexterous Manipulation credit by Boston dynamics

Many humanoid robots are equipped with hands and arms capable of dexterous manipulation, enabling them to grasp, hold, and manipulate objects with a level of precision similar to human hands. Dexterous Manipulation can be seen in Fig 2-2.

4. **Sensory Perception:** Humanoid robots incorporate sensors such as cameras, depth sensors, touch sensors, and accelerometers to perceive their environment. These sensors allow them to detect obstacles, recognize objects, and interact with humans and objects.

Figure 2-3 ASIMO Robot

5. **Artificial Intelligence (AI):** Humanoid robots often integrate AI technologies, including machine learning and neural networks, to process information, make decisions, and learn from experiences. This enables them to adapt and improve their performance over time.
6. **Computer Vision:** Humanoid robots utilize computer vision techniques to interpret visual data from their surroundings. This enables them to recognize

objects, navigate environments, and engage in tasks that require visual perception.

7. **Natural Language Processing (NLP):** Some humanoid robots incorporate NLP to understand and generate human language. This allows them to engage in spoken communication with humans, understand commands, and respond intelligently.
8. **Human-Robot Interaction:** Humanoid robots are designed to facilitate intuitive interactions with humans. This includes features such as gesture recognition, facial expression analysis, and the ability to respond to voice commands.
9. **Autonomous Behavior:** Humanoid robots can operate autonomously to varying degrees, performing tasks and making decisions based on their sensors, AI algorithms, and programmed instructions.
10. **Social and Emotional Interaction:** Advanced humanoid robots are capable of displaying emotions, expressing empathy, and engaging in social interactions. This enhances their ability to provide companionship, support, and assistance in various contexts.
11. **Versatility:** Humanoid robots can be adaptable and versatile, capable of performing a wide range of tasks across different domains, from healthcare and education to entertainment and industry.
12. **Research and Development:** Humanoid robots are often used as platforms for research and experimentation, allowing scientists and engineers to explore AI, robotics, and human-robot interaction in real-world scenarios.

These characteristics collectively contribute to the unique identity of humanoid robots, distinguishing them from other types of robots and highlighting their potential to interact with humans in more human-like ways.

2.4 Evolution of Humanoid Robotics

The evolution of humanoid robots extends for a long period and spans across several eras, from the epic invention of automata to the sophisticated anthropomorphic marvels of today. Each era has left a shining mark on the landscape of robotics, with standout robots demonstrating the progress achieved in humanoid design and technology.

- **Early Mechanical Automata (18th - 19th Century)**

Among the pioneers of humanoid robotics, Pierre Jaquet-Droz's "The Writer" emerged as an overwhelming example of early automata. This intricately designed creation, decorated in a lavish dress, charmed audiences by skillfully writing custom text with a feather. Such early automata showcased the promising possibilities of imitating human gestures and actions, it can be showen in Fig2-4.

There is also the automaton at the Franklin Institute in Philadelphia whose real-life story was made into the movie, Hugo. It draws pictures and writes poetry, driven by two engines and designed with a series of elegant gears.

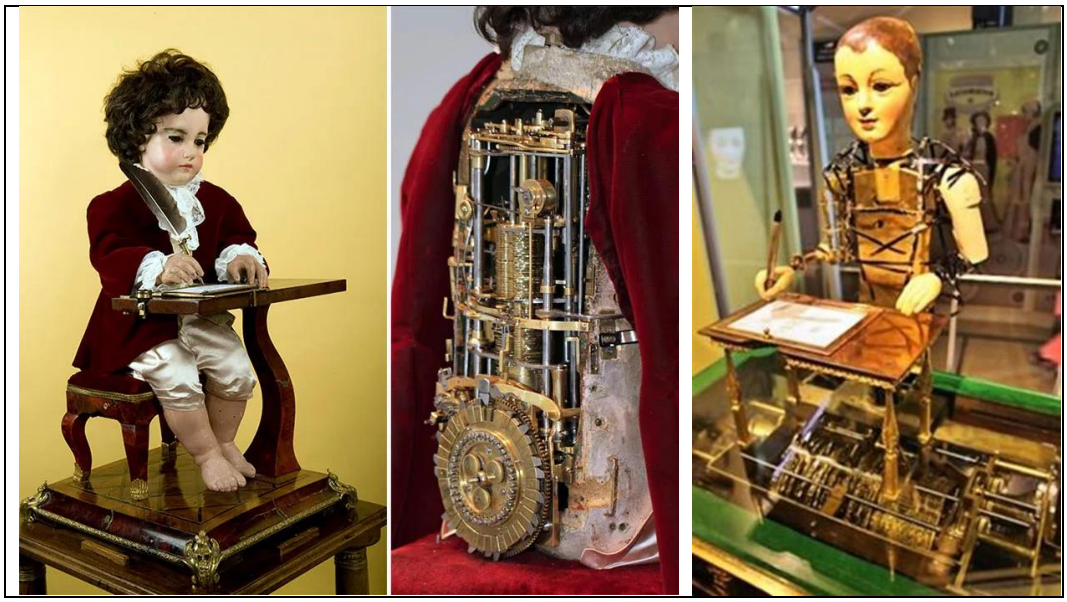


Figure 2-4 Early Mechanical Automata (18th - 19th Century)

- **WABOT Series (1960s - 1970s)**

  In the mid-20th century, the WABOT (WAseda roBOT) series took center stage with WABOT-1, a robot capable of communicating with a person in Japanese, measure distance and direction, and further, it was able to walk and grab objects. However, another notable creation during this period was "Shakey the Robot" developed by SRI International. While not a humanoid in appearance, Shakey demonstrated pioneering capabilities in navigation and reasoning, laying the groundwork for future intelligent robots. This robot shown in the following Fig 2-5.

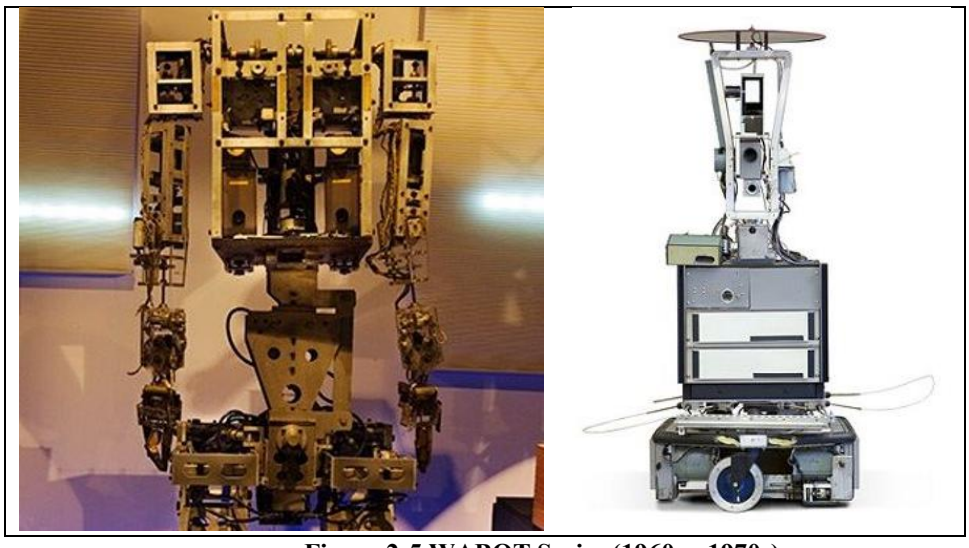


Figure 2-5 WABOT Series (1960s - 1970s)

- **ASIMO (2000s)**

Honda's ASIMO Fig 2-6, a poster child for humanoid robotics in the 2000s, was accompanied by Sony's AIBO, a robotic dog with expressive movements. While ASIMO was able of bipedal walking and advanced mobility, AIBO displayed the potential for human-like interactions through its appealing behavior and communication.

Figure 2-6 ASIMO (2000s)

- **Nao (2006-present)**

Developed by SoftBank Robotics. A small humanoid robot designed for education and research, widely used in academic settings.

- **Pepper (2014-present)**

Also developed by SoftBank Robotics. A social humanoid robot capable of recognizing emotions and engaging in conversations.

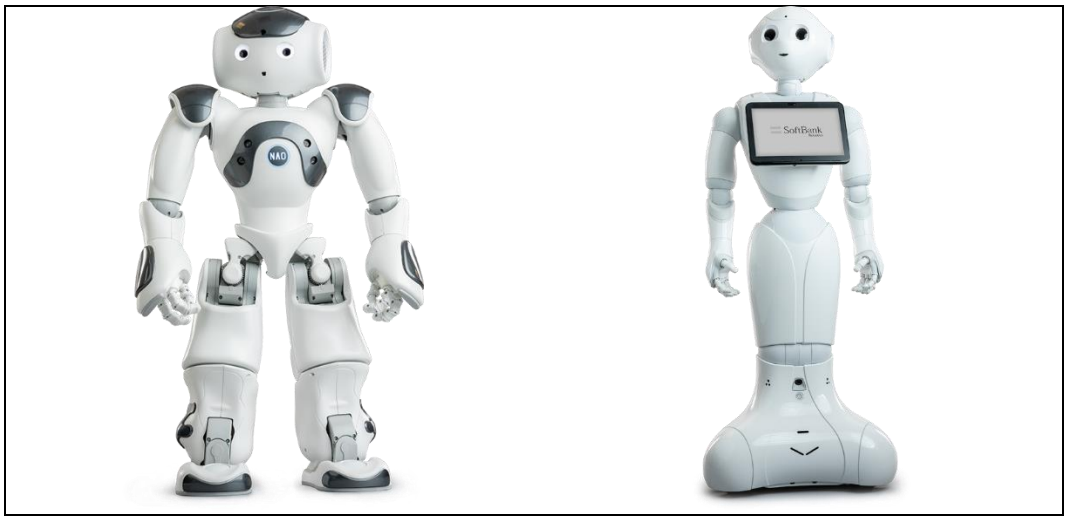


Figure 2-7 Pepper Robots Credit to SoftBank robotics

- **Atlas and Valkyrie (2010s - Present):**

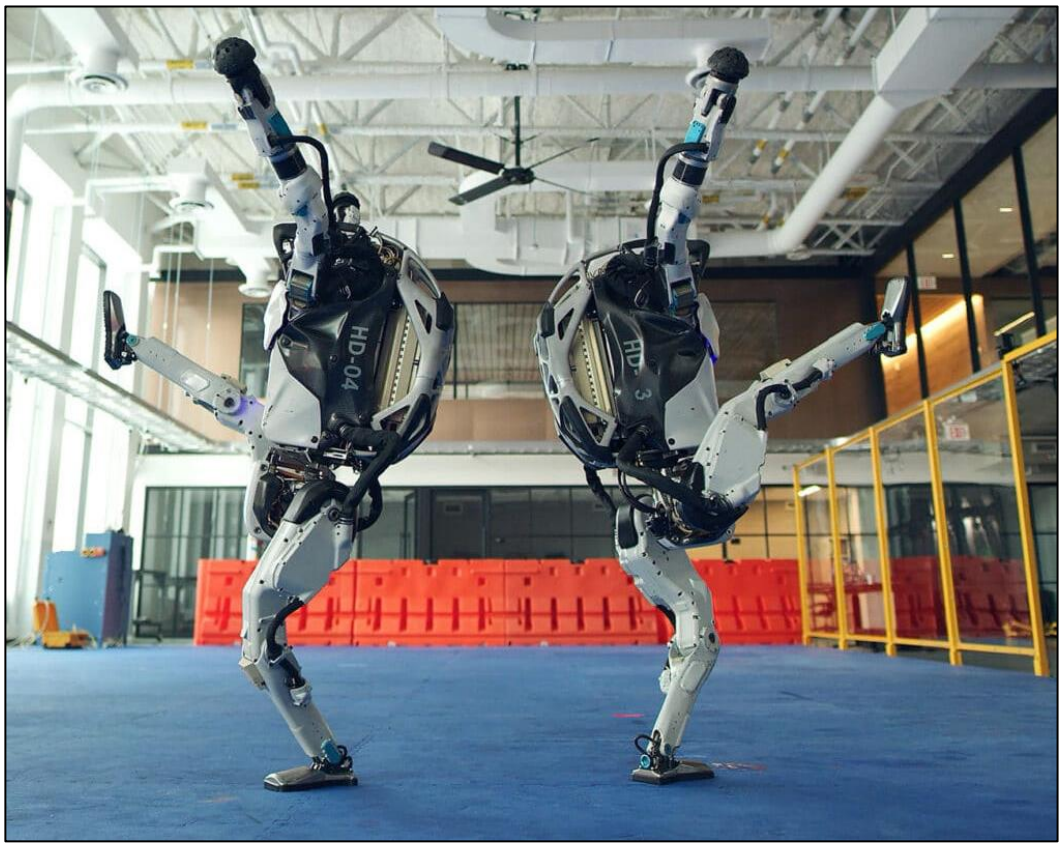


Figure 2-8 Atlas Robot credits to Boston dynamics

Atlas is a versatile robot designed for a variety of tasks, showcasing remarkable mobility and agility(Fig 2-8). It can navigate challenging topographies, perform dynamic movements like backflips, and manipulate objects with its dexterous hands. Originally created for search and rescue missions, Atlas has evolved to demonstrate impressive skills in various fields.

Valkyrie, on the other hand, was initially developed by NASA's Johnson Space Center for potential use in space exploration. It features a sleek design and is built to perform tasks in human environments, making it suitable for tasks like maintenance, repairs, and exploration on other planets or moons.

Figure 2-9 Valkyrie Robot credits to NASA

- **Jia Jia and Sophia (2016 - Present):**

  In the realm of expressive humanoid robots, Sophia shared the stage with Jia Jia, a lifelike creation from the University of Science and Technology of China. Jia Jia exhibited realistic facial features and the ability to engage in natural conversations, advancing the capabilities of social robotics.

Figure 2-10 Jia Jia and Sophia Robots

- **T-HR3**

Figure 2-11 T-HR3

T-HR3 (2017-present)(Fig 2-11): Developed by Toyota. Designed to explore remote control and intuitive human-robot interaction, enhancing teleoperation capabilities.

- **Digit**

Digit (2019-present)(Fiig 2-12): Developed by Agility Robotics. Focuses on dynamic legged locomotion and versatile mobility, with potential applications in logistics and delivery.

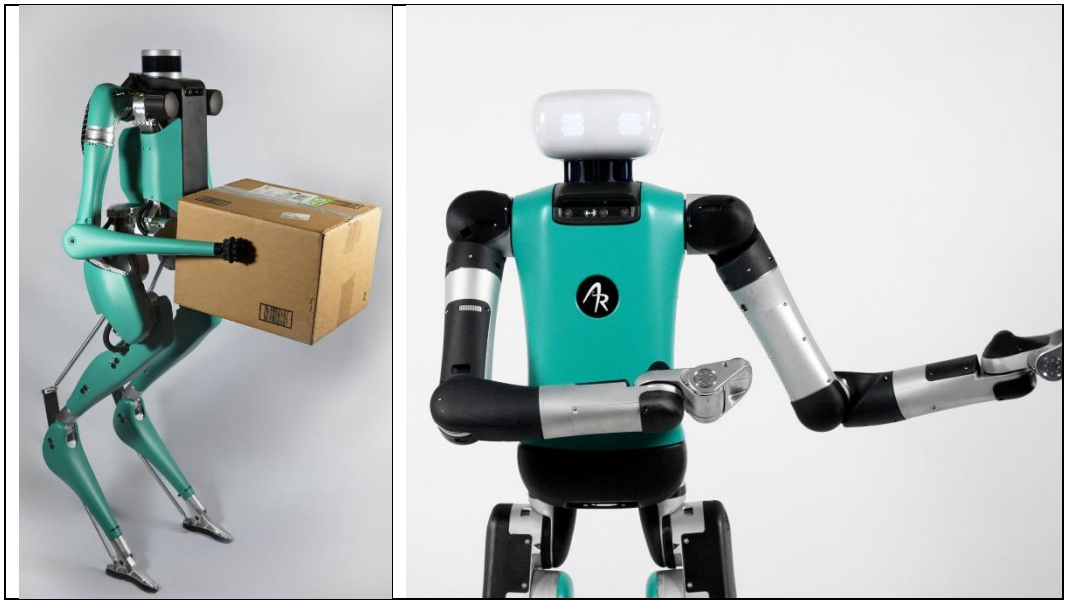


Figure 2-12 Digit Robot

2.5 Notable Projects and Prototypes

2.5.1 Honda's ASIMO

- **Start of Robot Development Modeled on Humans (1986)**
  Studying the fundamental principles of bipedal locomotion. Honda researched and observed all forms of walking, performed numerous experiments, and collected an immense amount of data. At E0, a robot that walks by putting one leg before the other was successfully achieved. But it took nearly five seconds between steps and could only do slow walking in a straight line(Fig 2-13)

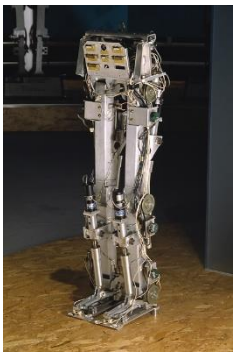


Figure 2-13 First model of Honda's ASIMO

- **Building Human-like, Fast Walking Technology (1987-1991)**
  To achieve fast walking, Honda thoroughly researched and analyzed human walking. Animal and other forms of walking were also studied in addition to human walking, and the movement and location of the joints needed for walking were researched as well. At E2, the first robot that could do fast walking was created, reaching a speed of 1.2 km/h (Fig 2-14).

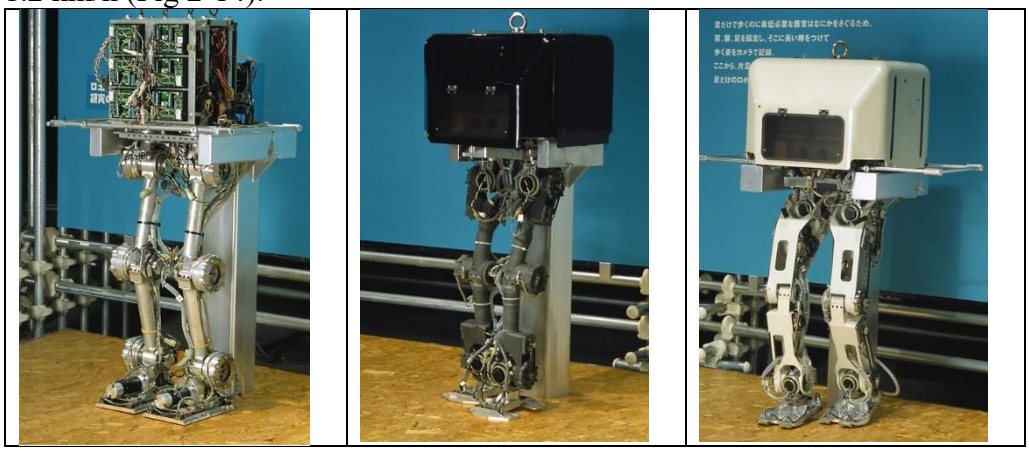


Figure 2-14 The first robot that could do fast walking

- **Achieving Two-legged Walking through Technology to Achieve Stable Walking (1991-1993)**
  Honda conducted research on the technology to achieve stable walking and successfully developed three posture control technologies(Fig 2-15).

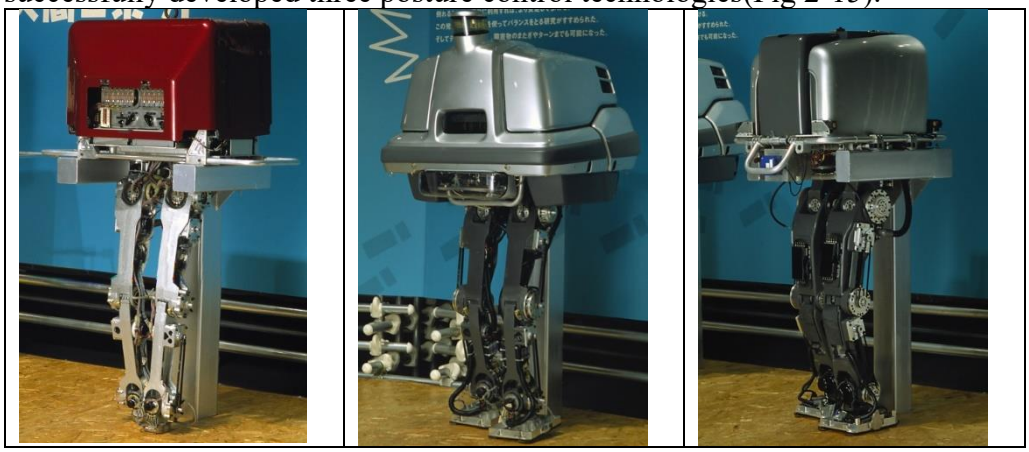


Figure 2-15 ASIMO first stable walking robot model

- **Volving to a Humanoid Robot that Combines an Upper Body with Legs (1993-1997)**
  Studies were carried out to determine what a humanoid robot should be like to function properly in society and in a human living environment. A prototype model of near-human size was completed(Fig 2-16).

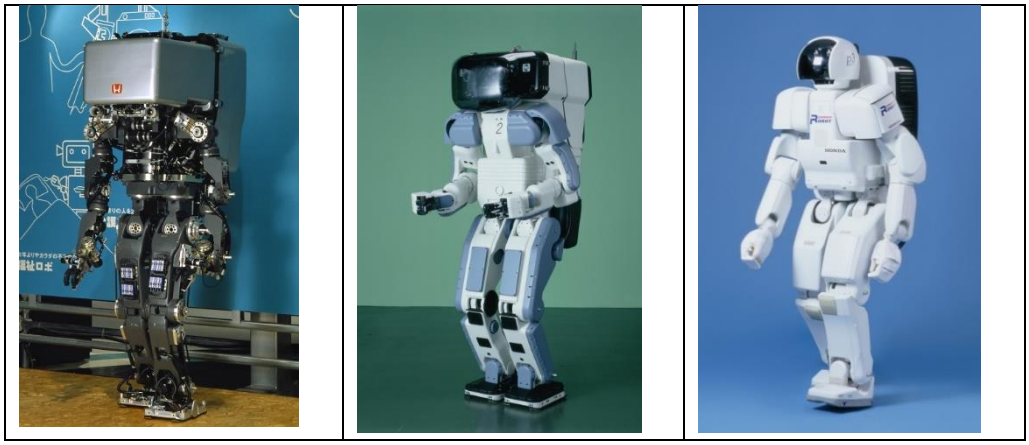


Figure 2-16 Humanoid Robot that Combines an Upper Body with Legs

In 2000, ASIMO was born and the modern humanoid robots started to evolve.

- **ASIMO Comes with Smaller and Lighter with More Advanced "i-WALK" Walking Technology (2000)**
  ASIMO is born in a size that is both capable of operating in a human living environment and is people-friendly, and realizes natural, smooth walking closer to how people walk. ASIMO's arm movements have also been expanded, while operability is improved with a portable controller(Fig 2-17).

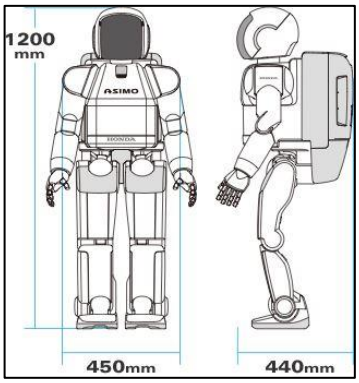


Figure 2-17 ASIMO

- **ASIMO Announced for Rental Services (2001)**
  For use in ordinary living environments, ASIMO's walking capabilities were further enhanced, and the system was simplified. ASIMO could navigate ordinary stairs and freely operate in environments with inclines. and specific tasks and guidance scripts suited to the user could be programmed.

- **New ASIMO With Intelligence Technology Announced (2002)**
  The 2002 ASIMO was announced with intelligence technology capable of interpreting postures and gestures of humans and moving autonomously in response. Human interaction technology was extensively advanced to allow ASIMO to greet approaching people, follow people, move in the direction they indicate, and recognize their faces and address them by name. Utilizing the Internet and other networks, ASIMO could provide information while executing tasks such as reception duties.

- **Aiming For New Mobility to Coexist and Operate with People (2004)**
  The 2004 ASIMO, aiming to become a new type of mobility that coexists and operates with people, was enhanced with technology allowing it to rapidly recognize real-life situations and quickly operate in response.

- **Realizing a Humanoid Robot with Highly Integrated Intelligence and Physical Capabilities (2005)**
  Compared to previous versions, the 2005 ASIMO enhanced functions to operate with people, such as holding hands while walking, and carrying objects with a tray or cart. By developing a unified control system that operates these functions, ASIMO could autonomously conduct reception guidance and delivery services. With drastically improved physical capabilities, ASIMO could run at 6km/h and turn while running.

- **Multiple ASIMOs Can Continuously Provide Services in Environments with Multiple People and ASIMOs (2007)**
  Multiple networked ASIMOs can share each other's task status, and by assigning tasks in the most efficient way, can operate together effectively. By calculating the distance from each ASIMO to the tasks' location and considering battery charge, tasks are assigned to each ASIMO to autonomously perform, maximizing overall time efficiency.

Figure 2-18 ASIMO

- **Newly Equipped with World-first Autonomous Behavior Control Technology (2011)**
  ASIMO becomes more autonomous, capable of continuous behavior without human operation. With a greatly enhanced situation adaptive capacity both intelligently and physically, ASIMO moves one step closer to practical use in crowded public spaces and office environments[23][24].

Chapter 3
Methodology

Chapter 3 Methodology

3.1 Software Design

Humanoid robots are becoming increasingly popular in various fields, including healthcare, entertainment, and manufacturing. However, developing and testing humanoid robots is a complex and time-consuming process that requires significant expertise and resources. To overcome these challenges, engineers and designers can use ROS, an open-source robotics platform, to develop and test humanoid robots in a virtual environment. In this report, we will discuss the process of converting 3D printed parts files into CAD files for simulation and prototyping, creating a URDF file, and launching the robot model in ROS for simulation and testing.

In conclusion, using ROS to develop and test humanoid robots can help to reduce development time and costs while improving the overall quality and performance of the final robot. By following the steps outlined in this report, engineers and designers can create accurate and effective models of parts and products and test them in a virtual environment before moving on to physical prototyping and testing. This can help to identify any issues or areas for improvement early in the development process, ultimately leading to a better final product.

3.1.1 Empowering Design and Assembly for Robotics with URDF Export

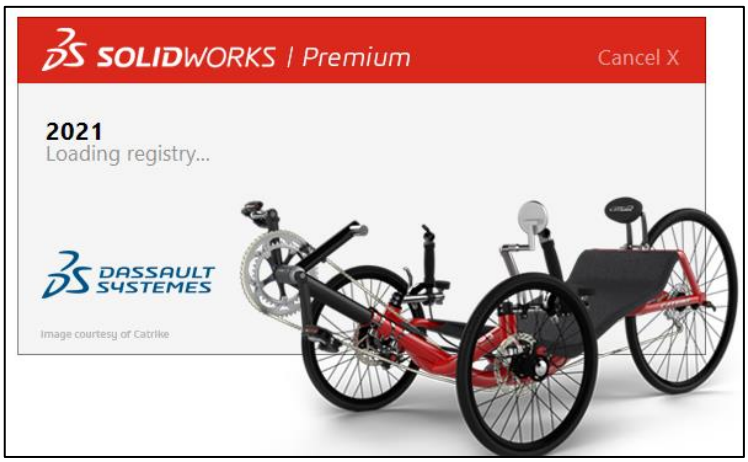


Figure 3-1 SolidWorks

SolidWorks, renowned as a parametric modeler, plays a pivotal role in modern design and assembly processes. As a solid modeler, it employs dimensions, parameters, and relationships to define and manipulate 3D shapes. This unique approach enables efficient editing and modification of parts during construction, making it an ideal tool for creating new designs. In the context of robotics, this essay explores how SolidWorks facilitates the conversion and assembly of robot parts, culminating in the generation of URDF (Unified Robot Description Format) files compatible with ROS (Robot Operating System).

SolidWorks' parametric modeling capabilities empower designers by allowing them to define key dimensions, parameters, and relationships that drive the shape and behavior of 3D objects. This parametric approach enables easy modification and editing of parts throughout the design process. Designers can quickly iterate and refine their designs, making adjustments to dimensions or parameters to achieve desired outcomes. SolidWorks' ability to handle complex geometries and intricate assemblies makes it an indispensable tool for creating new designs with precision and efficiency.

In the realm of robotics, SolidWorks serves as a valuable asset for converting and assembling robot parts. Designers can import existing CAD models or create new ones from scratch. SolidWorks provides a comprehensive set of tools and features to accurately represent and assemble robot components, including joints, links, and sensors. The software's intuitive interface and user-friendly workflows streamline the process of constructing complex robot structures from individual parts. Designers can leverage SolidWorks' capabilities to ensure proper fit and functionality during the assembly phase.

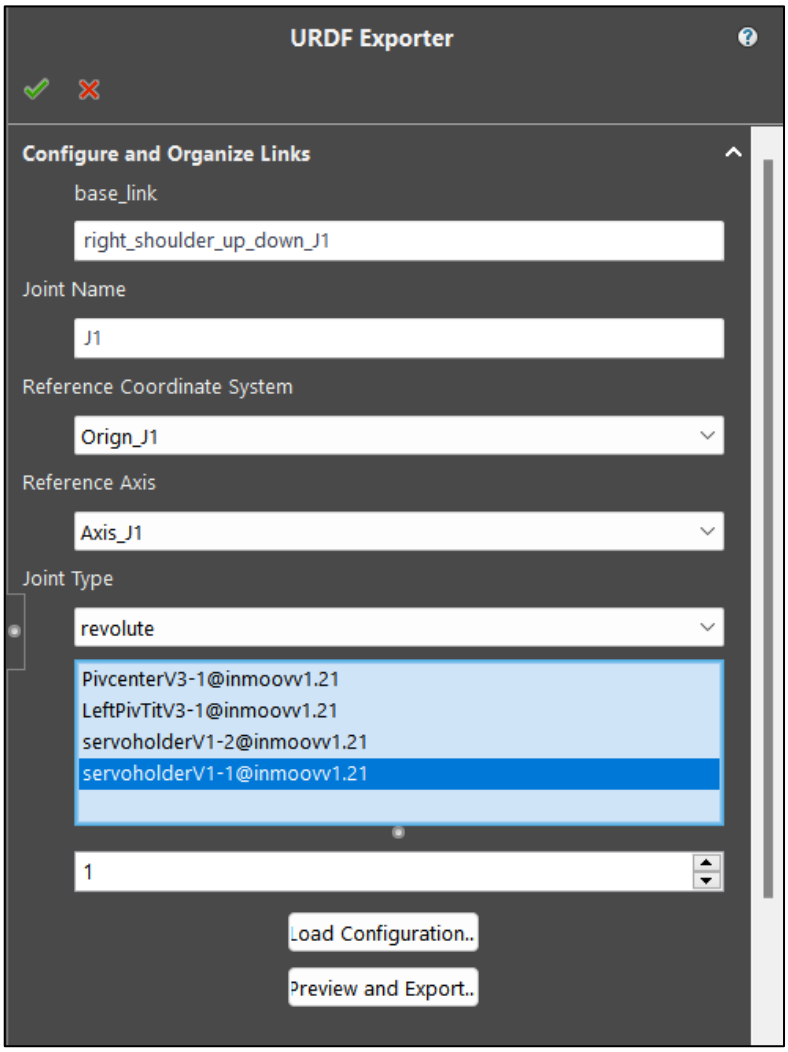


Figure 3-2 Configure and Organize Links Using URDF Exporter Plugin

To seamlessly integrate robot assemblies with ROS, the generated model must adhere to the URDF format. SolidWorks offers an efficient solution by enabling the export of URDF files. URDF is a standardized XML-based format that describes the structure and properties of a robot. SolidWorks provides an export mechanism that converts the assembled robot model into a URDF file, capturing essential information such as joint types, joint limits, visual representations, and collision geometries. This export functionality simplifies the integration process with ROS, allowing for accurate simulation, control, and visualization of the robot within the ROS ecosystem.

The utilization of SolidWorks in converting and assembling robot parts for ROS integration brings several advantages. Firstly, the parametric modeling capabilities empower designers to iteratively refine and optimize their designs, ensuring optimal performance and functionality. Secondly, the assembly features of SolidWorks facilitate the creation of complex robot structures, promoting efficient and accurate assembly workflows. Lastly, the ability to export URDF files directly from SolidWorks streamlines the integration process with ROS, enabling seamless simulation and control of the robot within the ROS environment.

SolidWorks stands as a powerful parametric modeler that revolutionizes the design and assembly of robot parts. Its ability to define dimensions, parameters, and relationships empowers designers to create and modify 3D shapes with ease. By leveraging SolidWorks' capabilities, designers can convert and assemble robot parts efficiently, culminating in the generation of URDF files that seamlessly integrate with ROS. The use of SolidWorks in robotics enhances the design process, promotes accurate assembly, and enables seamless integration with ROS, ultimately driving advancements in the field of robotics and automation.

3.1.2 ROS Modeling (Gazebo)

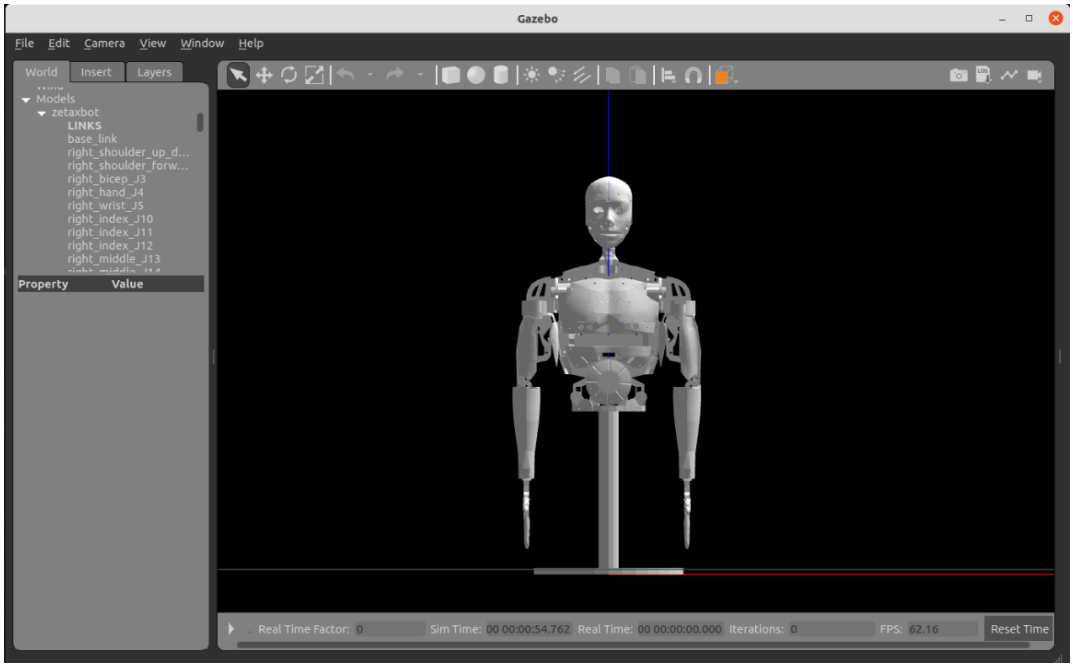


Figure 3-3 Gazebo

Gazebo is a simulation environment that enables testing of complex systems. It has various applications, including testing the dynamics of control systems before their actual implementation [7]. Moreover, Gazebo allows easy creation of robots, actuators, sensors, and objects [8]. The Gazebo environment consists of a division between the server and the client, provided by two executable programs: “gzserver” for simulating physics, rendering, and sensors, and “gzclient” for the graphical interface to visualize and interact with the simulation [9]. The first step in modeling within ROS requires creating a 3D environment model. The 3D model was generated using Solidworks. Once the model is complete, it was exported in STL format meshes to be integrated into the ROS environment [4]. This integration is made possible with the help of robot model packages that include URDF (Unified Robot Description Format) parser files. The URDF file is used to record all information about the virtual robot. It is common to create a robot model using Computer Aided Design (CAD) tools such as Solidworks, Pro-engineer, or Blender [10]. The URDF file contains descriptions of the model and the robot to be imported into ROS. These descriptions include the object's name, network structure, and additional information about the visual components of an object, such as color and texture [4]. The tag contains the robot's name, which will be displayed across all ROS subsystems. The tag is a child tag of the robot, which describes the type of movement (Fixed, Revolute, Continuous, Prismatic, Planar, or Floating). The concept of links and joints' structure can be seen in Figure 3-4.

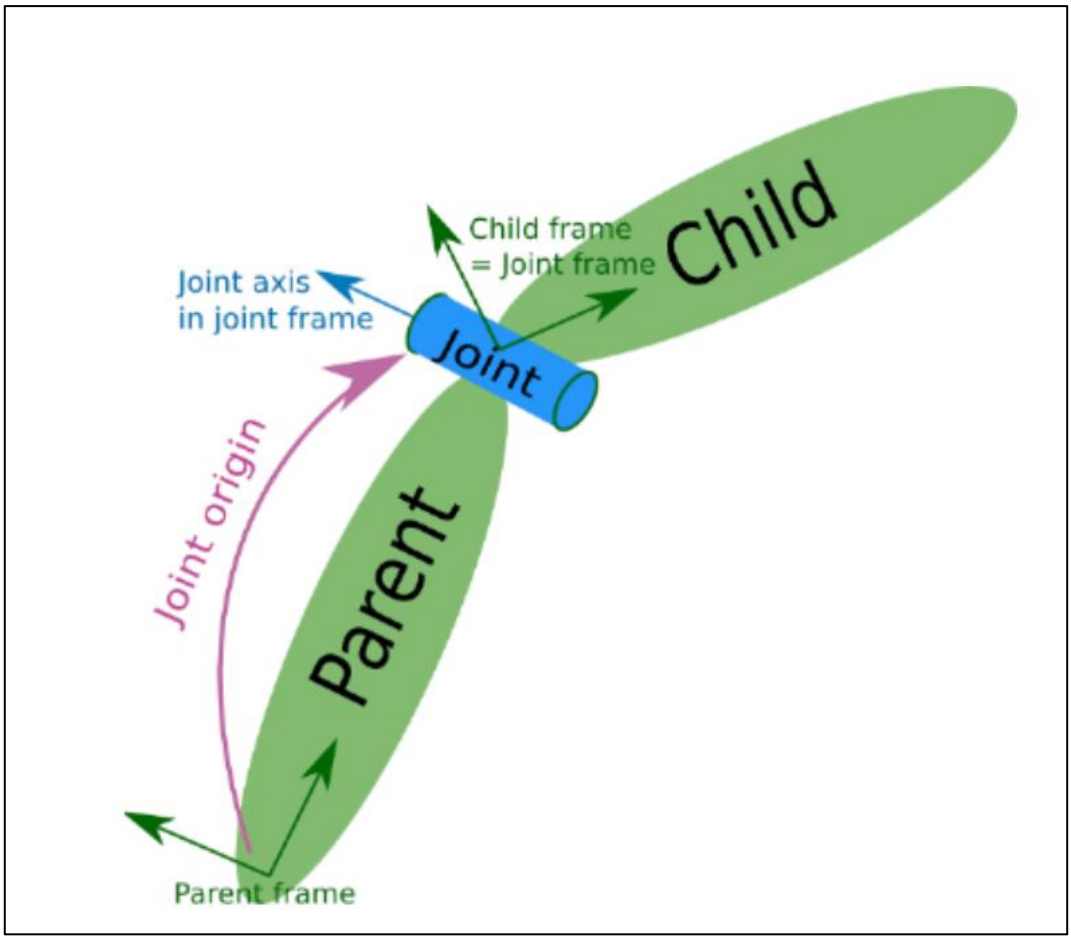


Figure 3-4 The structure of links and joints

3.1.3 Virtual robot

To facilitate the research and creation of a 3D robot design, we used Solidworks software. The process of converting 3D printed parts files into CAD files for simulation and prototyping involves several steps. The first step is to import the 3D printed file into Solidworks software, clean up the file, and convert it into a CAD format. Once the CAD file is created, it can be exported as a URDF file, which is compatible with ROS, by using the URDF Exporter plugin.

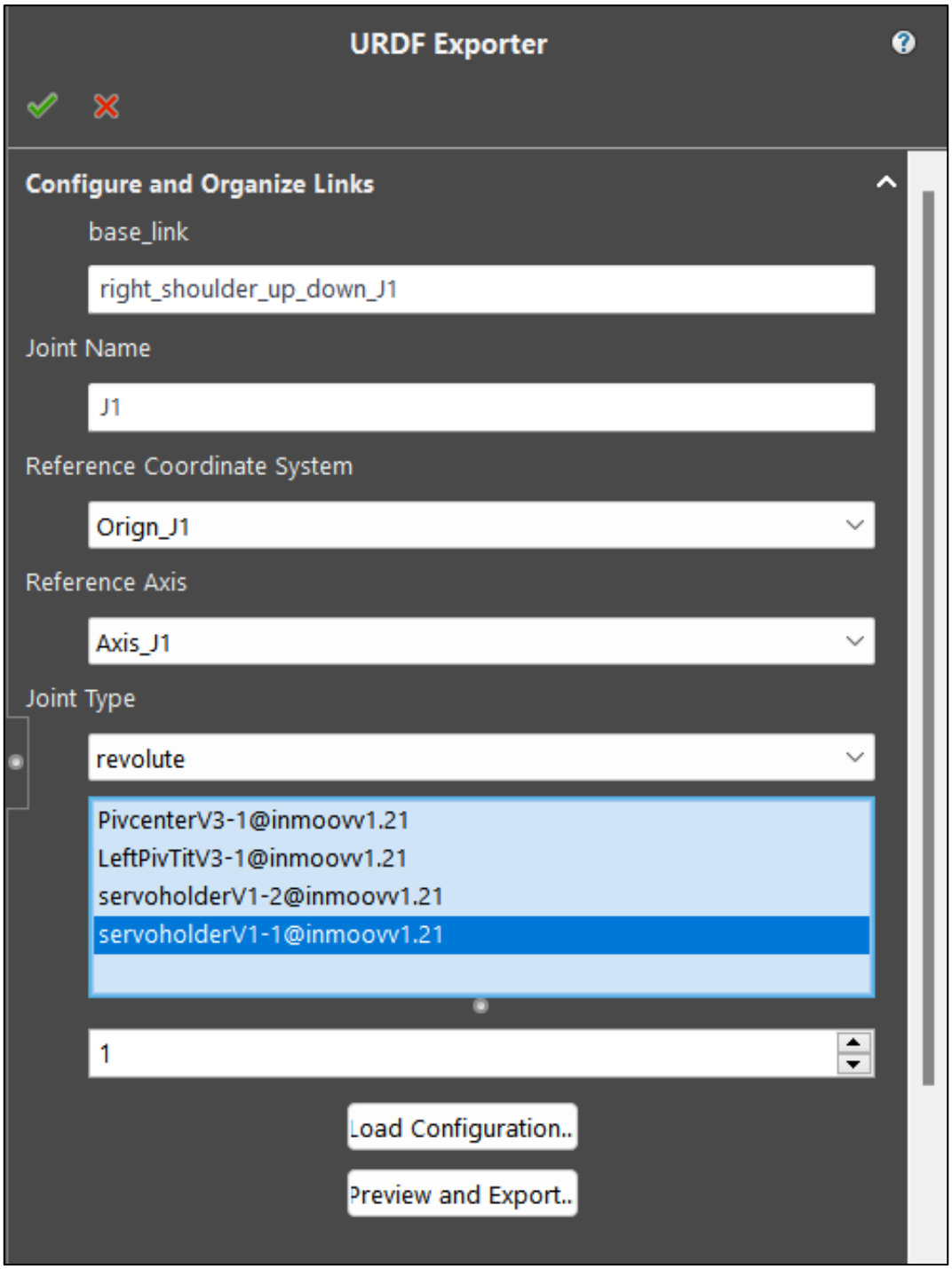


Figure 3-5 Configure and Organize Links Using URDF Exporter Plugin

To export the CAD file as a URDF, a new URDF file needs to be created, and the joints and links of the robot model need to be defined. This can include adding sensors and other components to the model. The URDF file must contain accurate and properly defined joint limits, joint types, and joint positions for the robot model to move correctly. The collision models in URDF files must also be properly defined to ensure that the robot model interacts correctly with the environment.

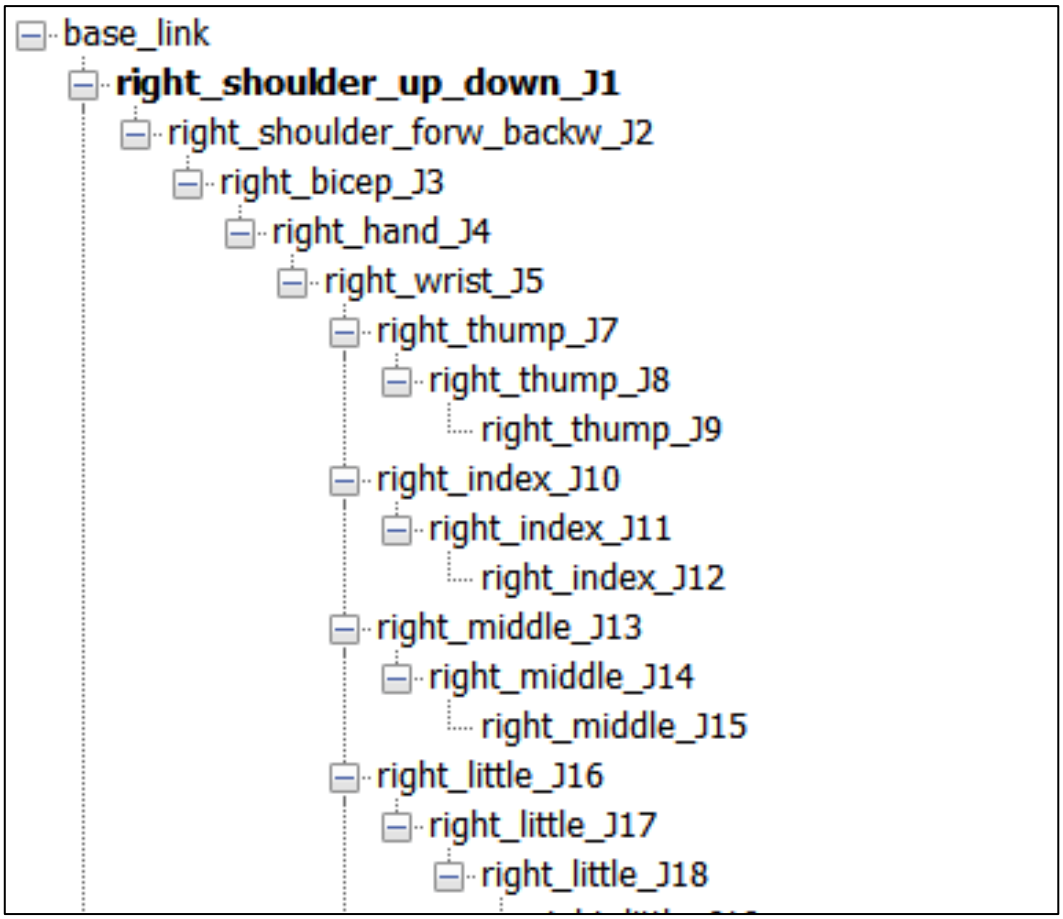


Figure 3-6 The joint structure of the ZETA robot

In the robot's joint structure, each joint connects a parent link to a child link, defining the relationship between different parts of the robot. For this project, the base of the robot utilizes the robot's body structure. The structure of links and joints for the ZETA robot, which was recently created, can be seen in Figure 3-5 & Figure 3-6.

Once the URDF file is exported, it can be launched in ROS for simulation and testing using the ROS launch command. This allows us to test the robot model's motion, control, and performance in a virtual environment, making it possible to identify any issues or areas for improvement before moving on to physical prototyping and testing. It is important to check all file paths and joint definitions and to test the robot model in a variety of scenarios to ensure that it is functioning correctly.

In Gazebo simulator, it has a specific hierarchy that allows seamless integration between programs to work effectively. Simulations are executed by calling a launch file. The launch file itself contains an XML program that configures the controller and robot description [11]. In the virtual robot, a configuration is made according to the ROS protocol to enable the virtual robot to run in the Gazebo simulator [12]-[13].

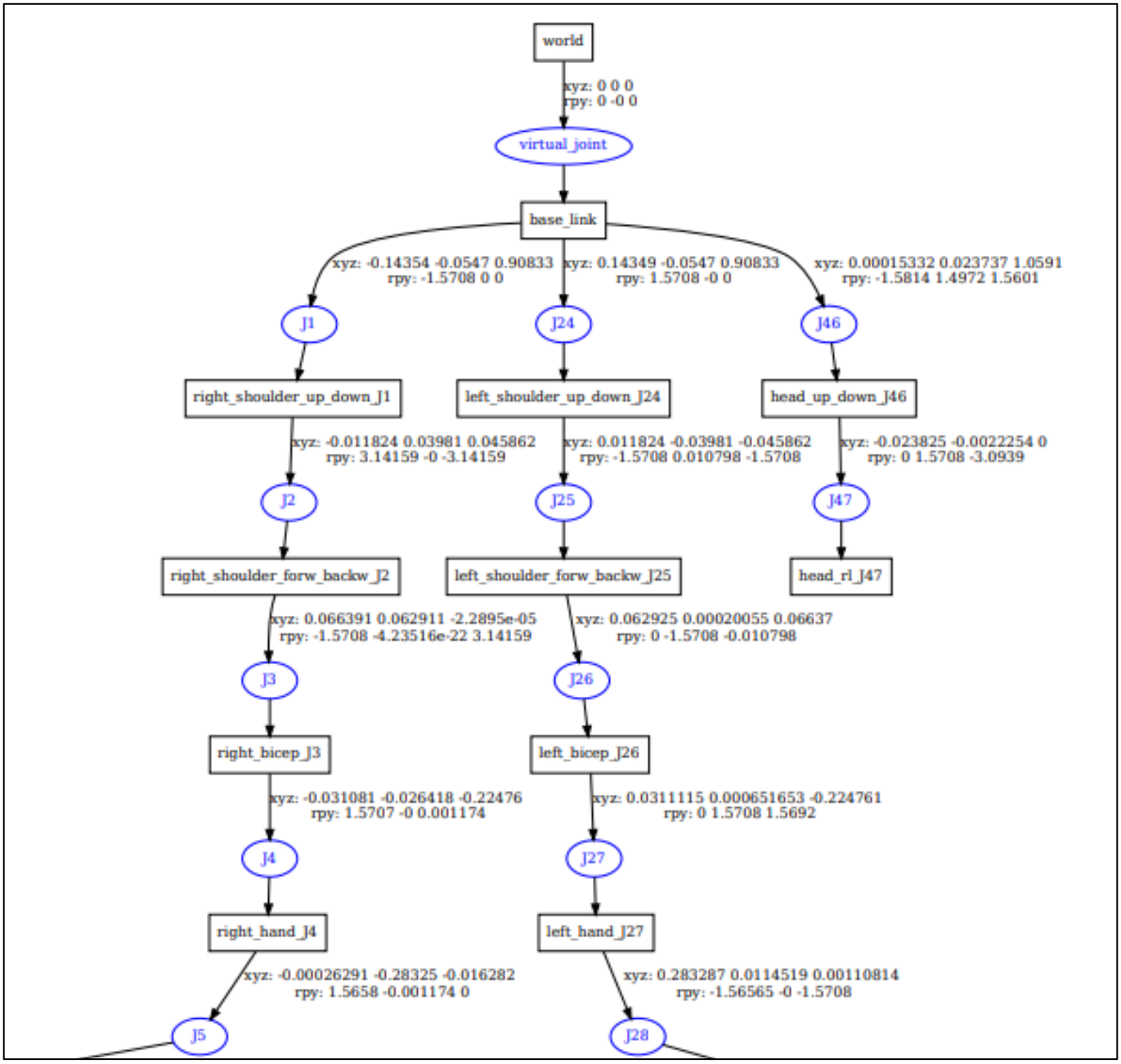


Figure 3-7 Links and Joints Structure Flow Chart 1 of 3

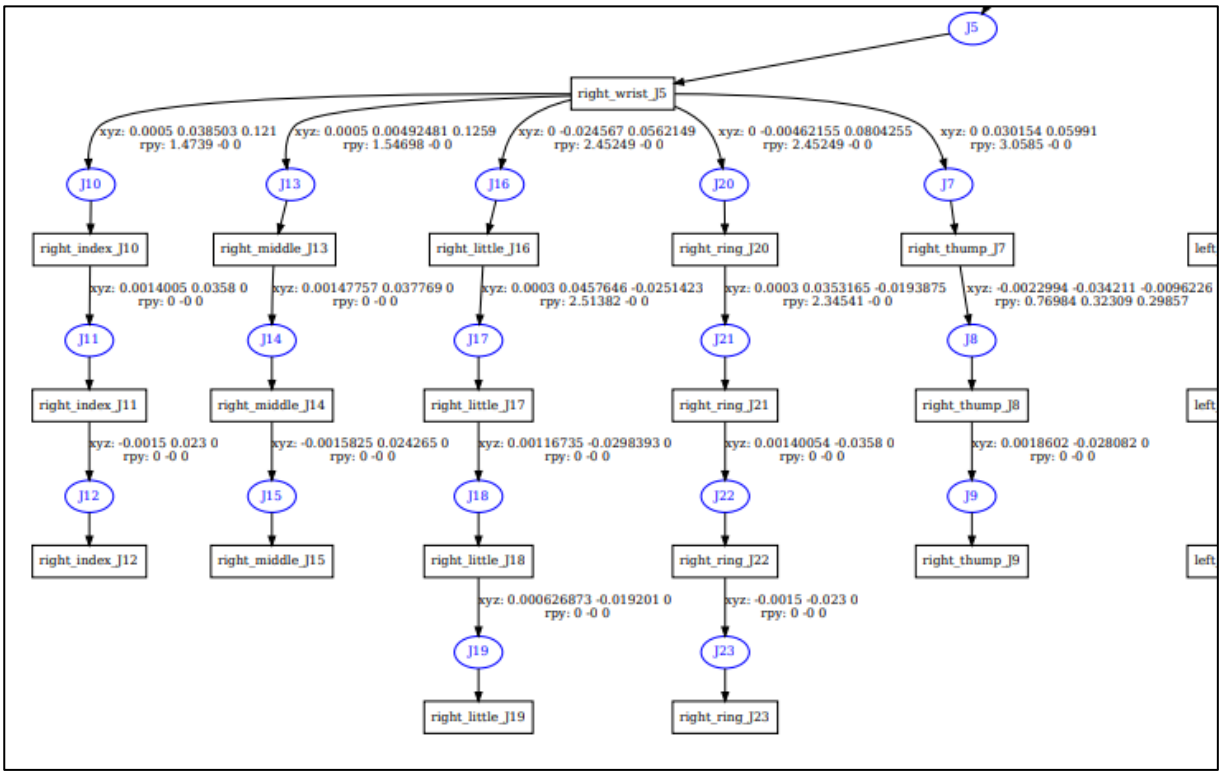


Figure 3-8 Links and Joints Structure Flow Chart 2 of 3

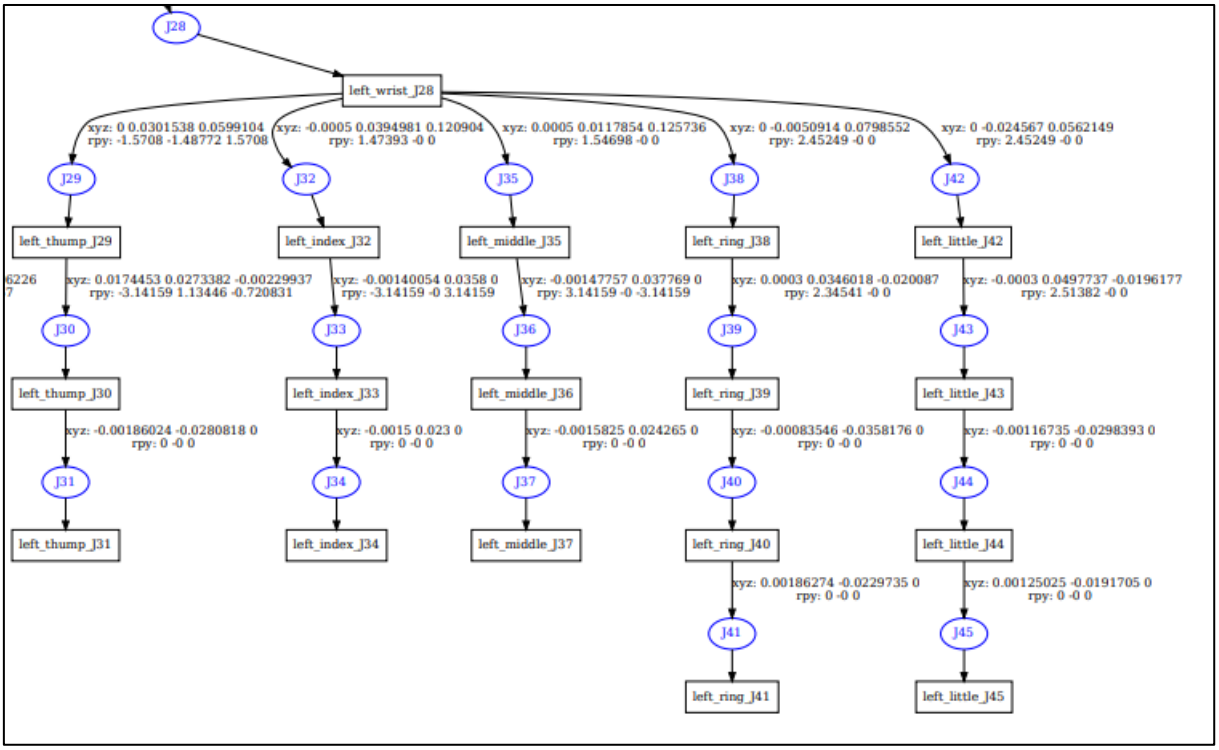


Figure 3-9 Links and Joints Structure Flow Chart 3 of 3

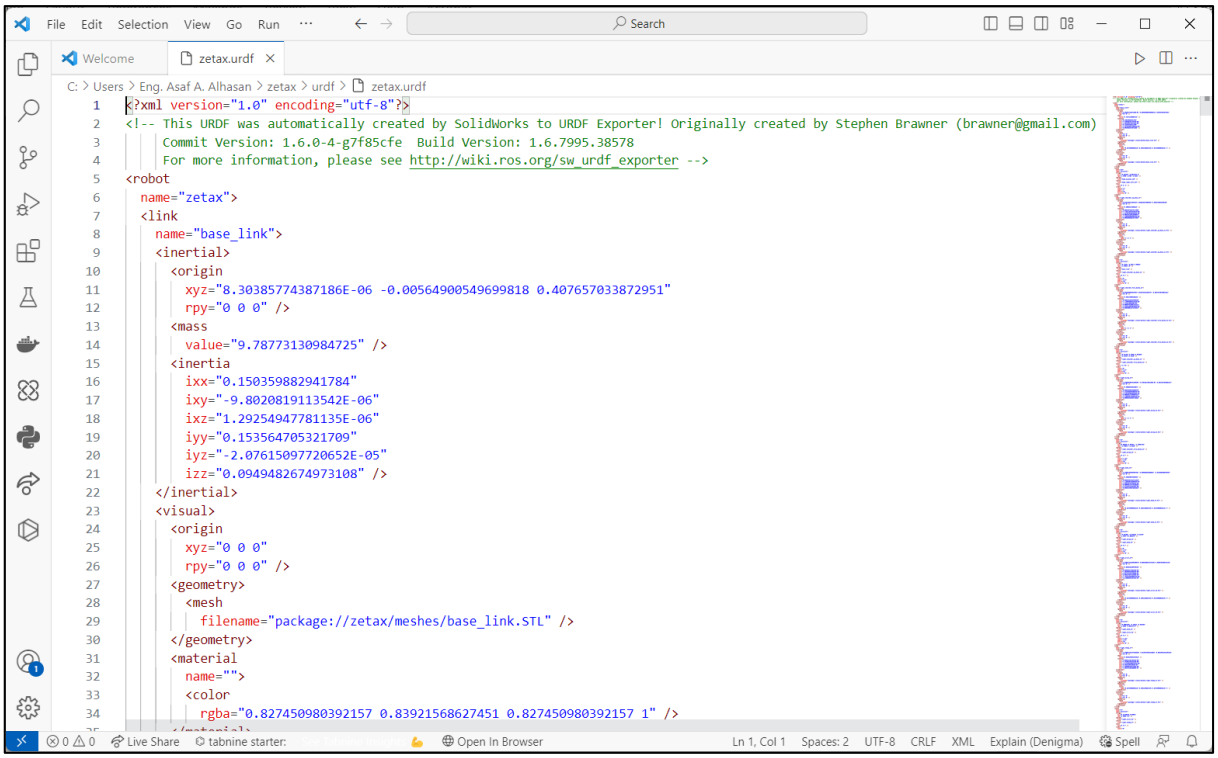


Figure 3-10 Original Exported URDF

Figure 3-11 Modified URDF

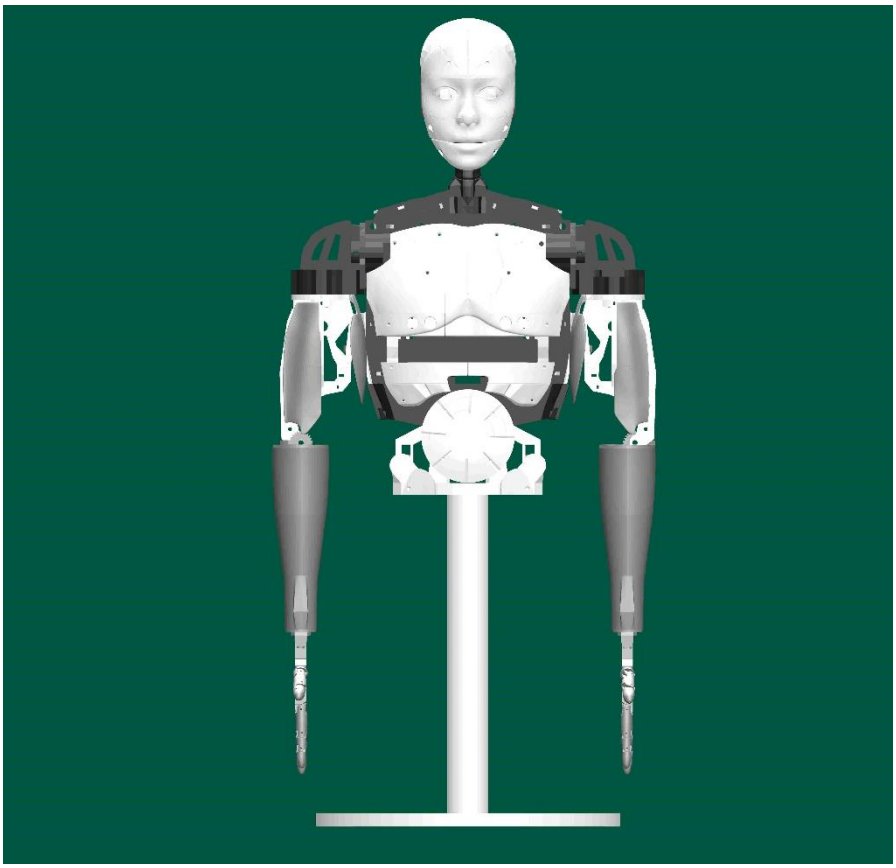


Figure 3-12 ZETA Robot Full Assembled in SolidWorks

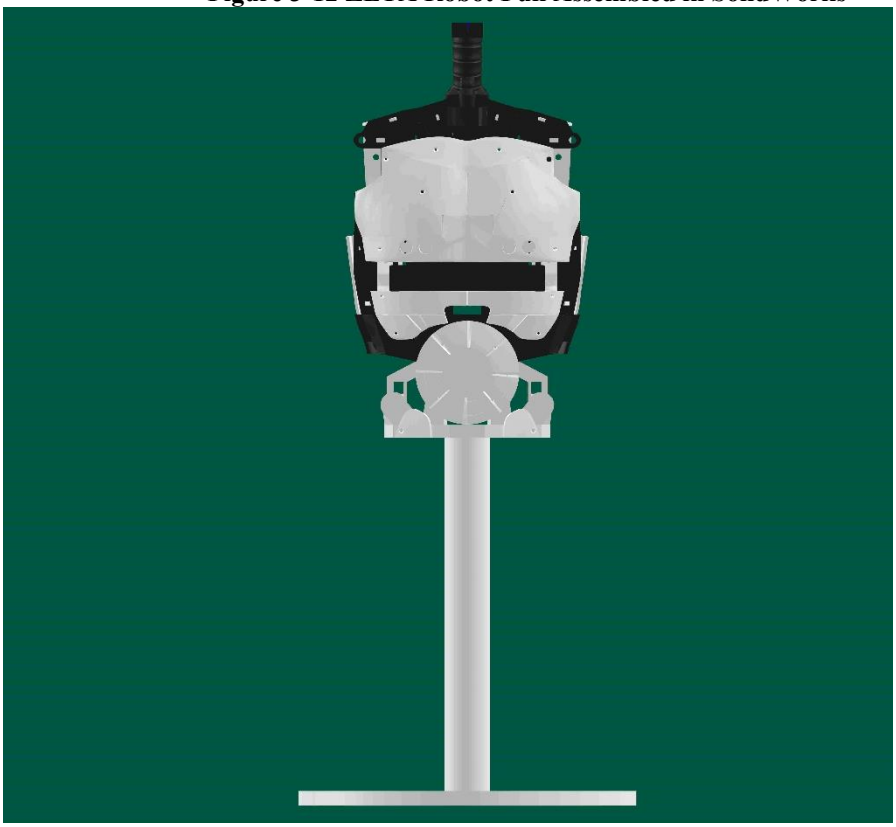


Figure 3-13 Base Link assembly

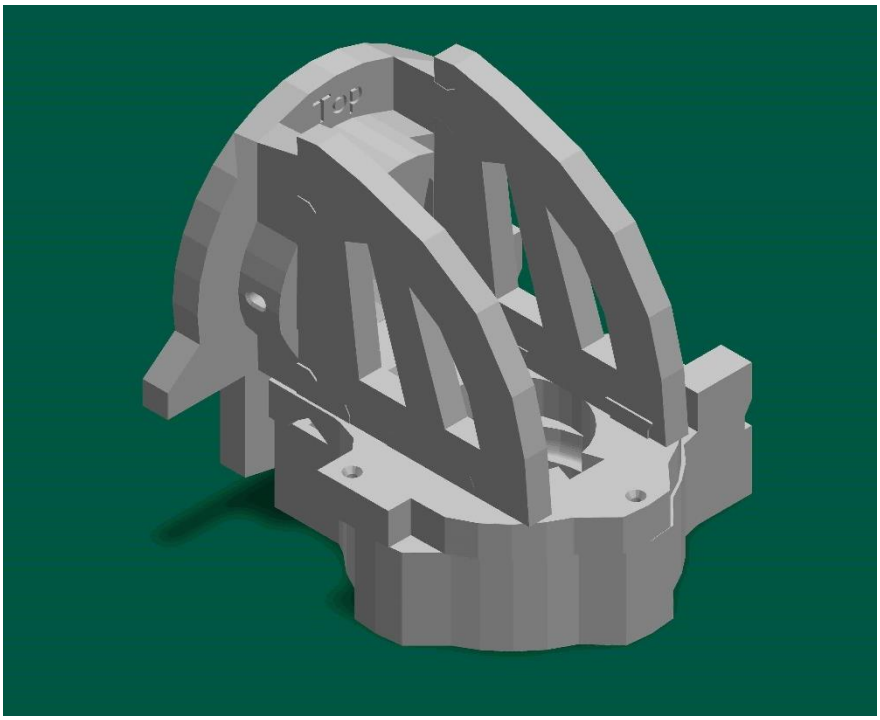


Figure 3-15 2x. Shoulder assembly

Figure 3-14 2x. Biceps assembly

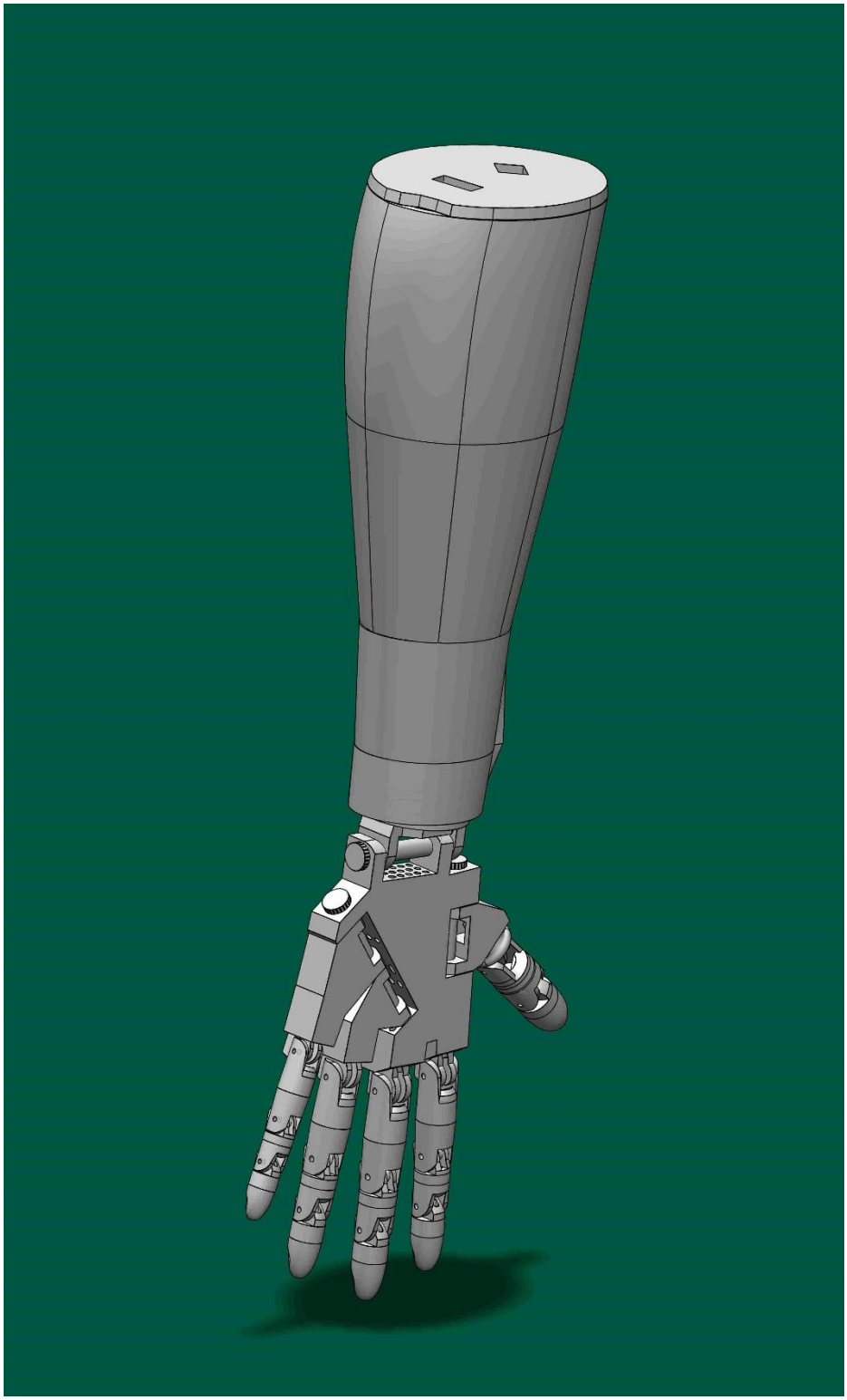


Figure 3-16 2x. Arm assembly

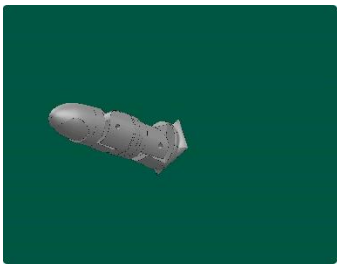


Figure 3-17 2x. Thump Finger assembly

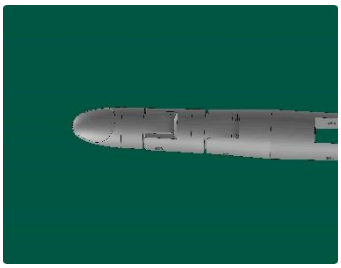


Figure 3-18 2x. Middle Finger assembly

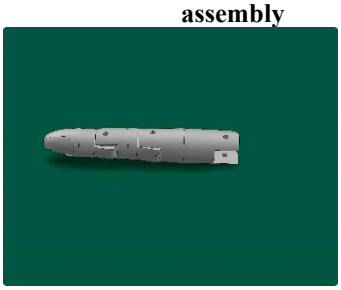


Figure 3-19 4x. Index Finger assembly

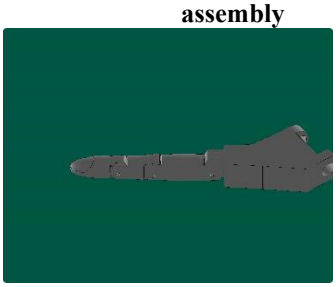


Figure 3-20 2x. Little Finger assembly

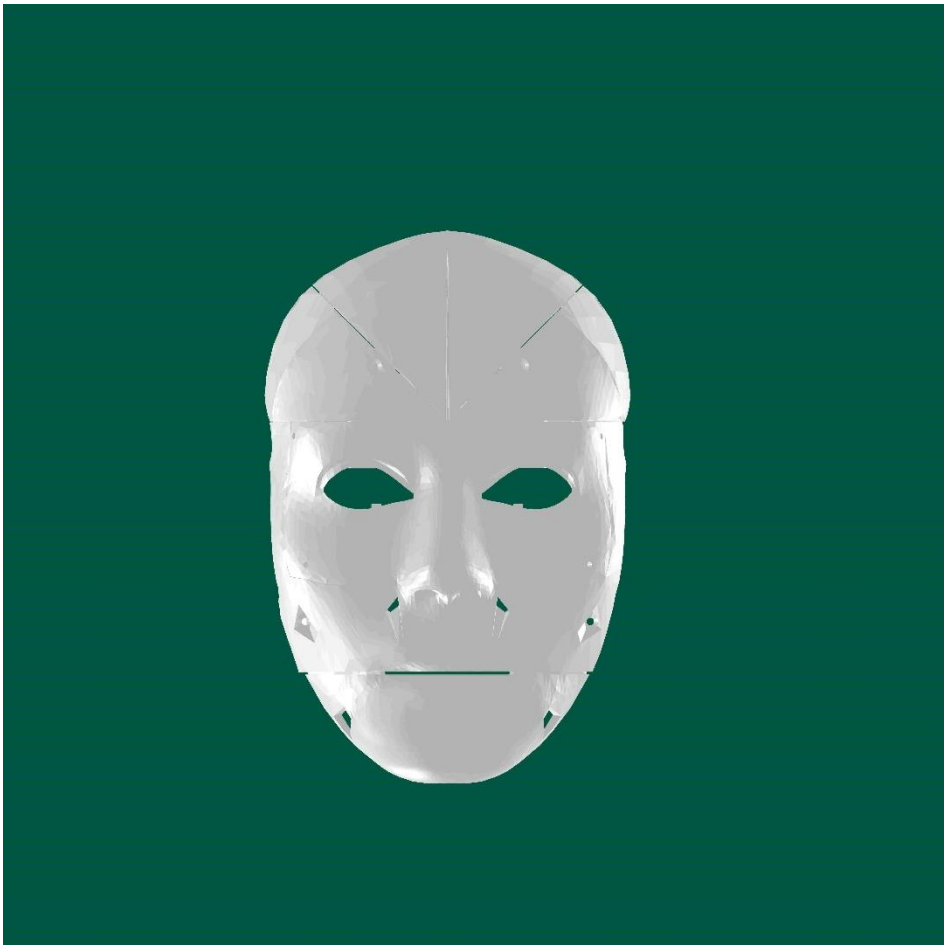


Figure 3-21 Face and Head assembly

3.1.4 Integration of Gazebo, MoveIt, and Real Robot in ROS using ESP

The software architecture serves as the central nervous system of the humanoid robot, enabling coordinated control and interaction. ROS, with its modular framework, is the cornerstone of this architecture. The integration of ROS nodes and packages fosters seamless communication between sensors, actuators, and high-level control components.

The architecture consists of interconnected layers, each responsible for specific functionalities. Sensory data is acquired through ROS-enabled drivers and processed through perception nodes. These nodes implement advanced algorithms for object detection, facial recognition, gesture analysis, and speech recognition. The motion planning layer generates dynamic and stable planning trajectories, ensuring balance and agility during walking and other actions.

In this section, we discuss the integration of Gazebo, MoveIt, and a real robot in ROS using ESP as a part of the Zeta Robot graduation project. This integration plays a crucial role in developing and deploying a comprehensive robotic system. By combining the capabilities of Gazebo for simulation, MoveIt for motion planning, and ESP for communication, we create a powerful platform for testing, planning, and controlling the Zeta Robot.

- **Gazebo Simulation Setup**

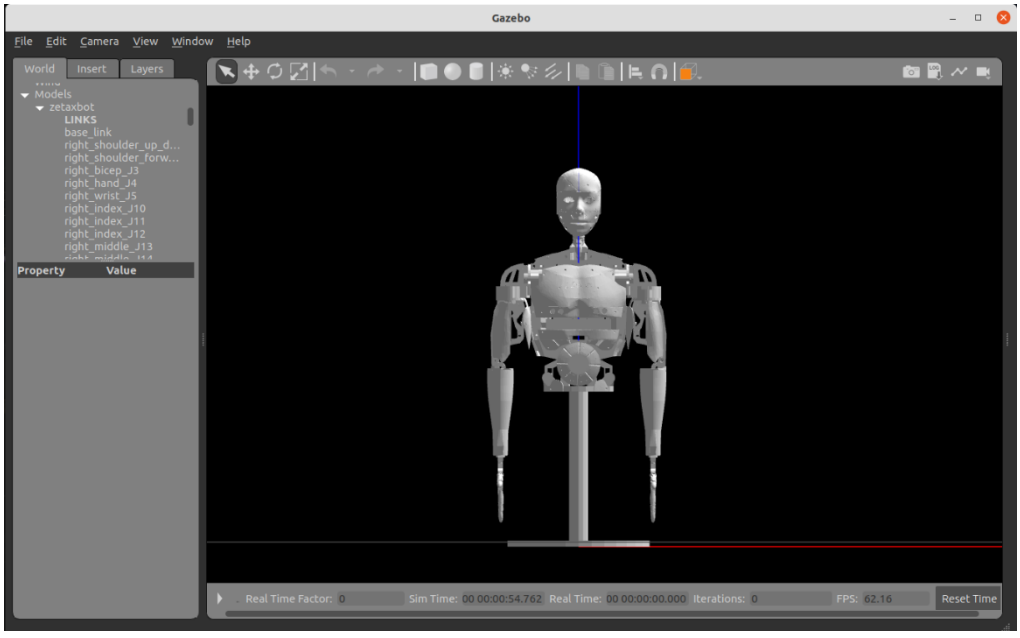


Figure 3-22 Zeta Robot visualized on Gazebo

The first step in the integration process involved setting up the Gazebo simulation environment. Gazebo provided a realistic simulation platform to visualize and test the behavior of the Zeta Robot. We created a virtual representation of the robot within Gazebo, defining its physical properties, such

as links, joints, sensors, and actuators. Through accurate modeling of the robot's dynamics, Gazebo enabled us to perform realistic simulations for development and testing purposes.

- **MoveIt Configuration**

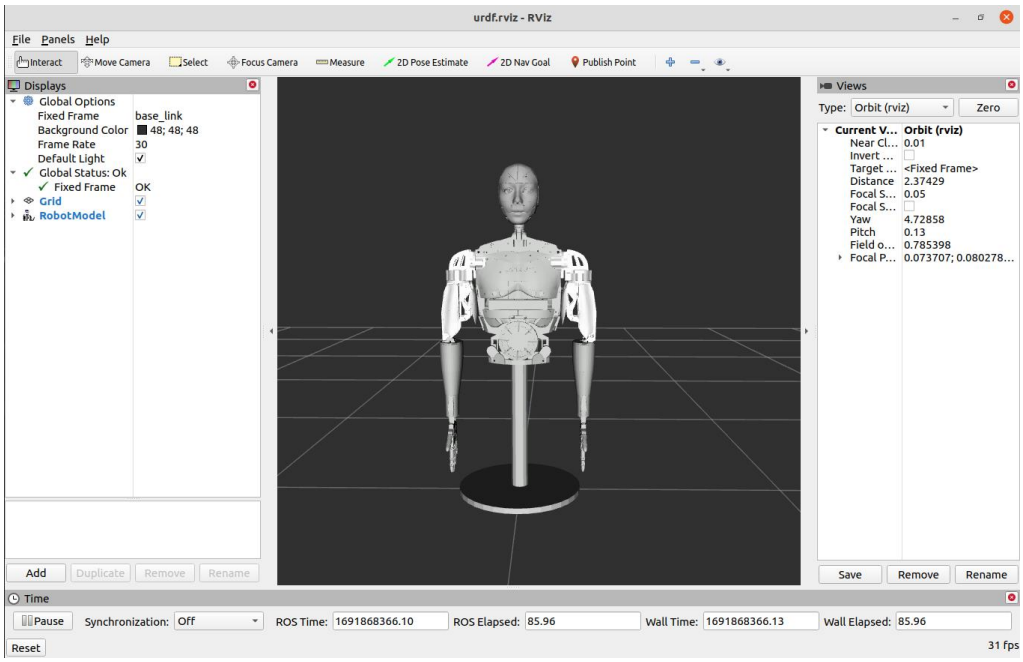


Figure 3-23 Zeta Robot visualized on Rviz

MoveIt, a motion planning framework in ROS, was integrated to enhance the capabilities of the Zeta Robot. We configured MoveIt to work seamlessly with the Gazebo simulation. This involved defining the kinematic and dynamic properties of the robot within MoveIt configuration files. We specified the robot's URDF model, joint limits, and configured planning groups and end effectors. MoveIt provided advanced motion planning and manipulation capabilities, allowing us to efficiently plan collision-free paths and control the Zeta Robot within the Gazebo simulation environment.

- **Upload the URDF file to Moveit Setup Assistant**

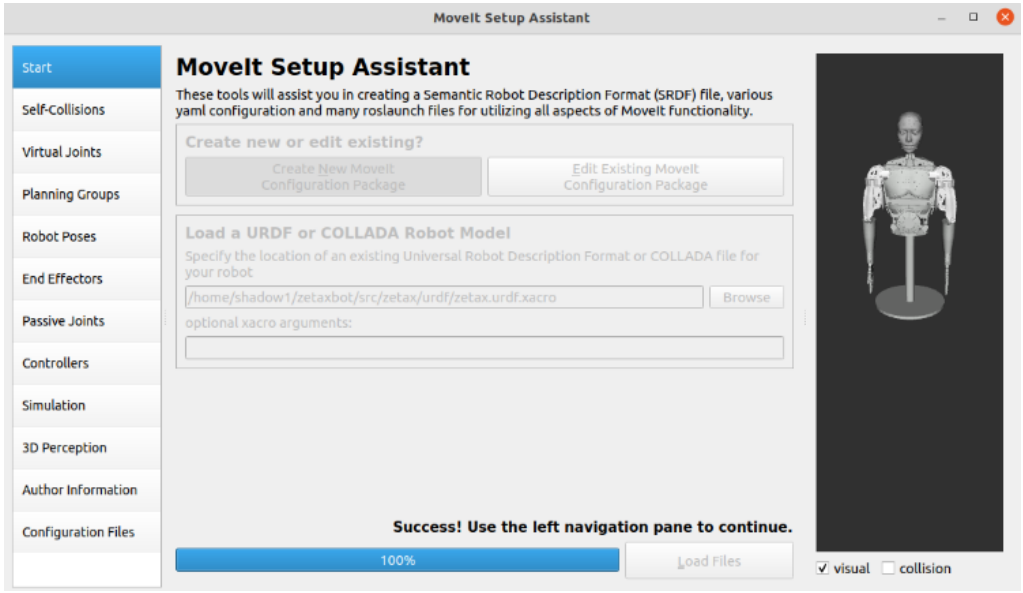


Figure 3-24 Upload the URDF file to moveit setup assistant

- **Configuring the rest of steps listed on the left side**

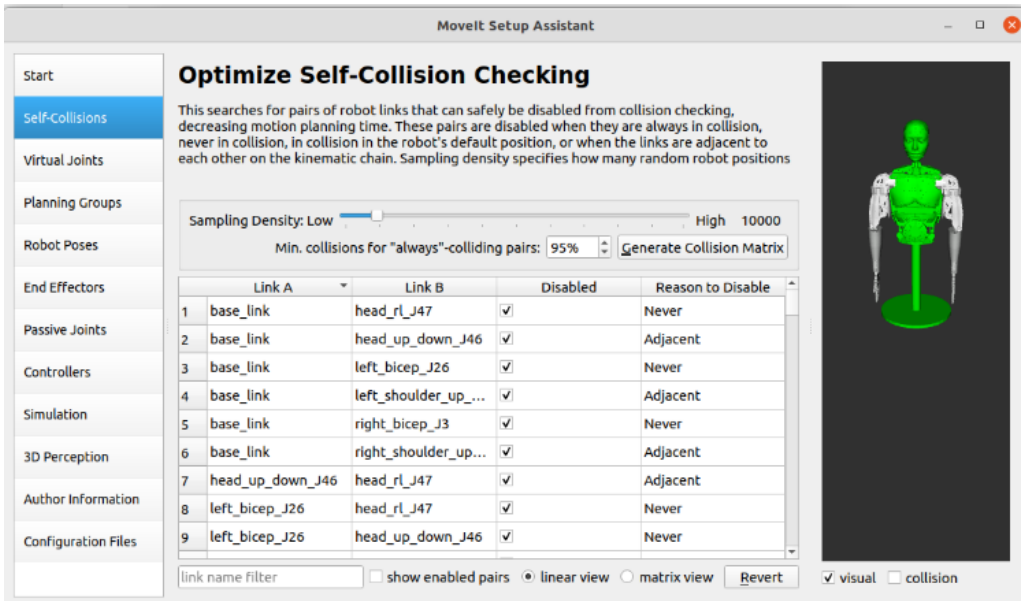


Figure 3-25 Configuring self-collisions

- **ESP Integration**

  ESP, the Embedded Systems Platform, served as the communication bridge between the virtual and real robot. It facilitated the exchange of data and commands between Gazebo, MoveIt, and the physical robot. ESP provided a framework for seamless communication and synchronization. By establishing this connection, we were able to send commands generated by MoveIt in the simulation to the real robot and receive sensor data from the physical robot back into the simulation environment.

- **ESP Configuration**

  To integrate ESP into the system, we configured the communication interfaces and protocols between Gazebo, MoveIt, and the real robot. This involved setting up network connections, defining message formats, and establishing synchronization mechanisms. ESP provided a flexible framework that allowed us to customize the integration based on the specific requirements of the Zeta Robot and the project setup.

- **Testing and Deployment**

  Thorough testing of the integrated system was conducted before deployment. We validated the motion planning capabilities of the Zeta Robot within the Gazebo simulation, ensuring proper synchronization and communication with the real robot. The testing phase helped us identify any issues or discrepancies between the virtual and real environments and allowed us to fine-tune the system parameters for optimal performance.

In summary, the integration of Gazebo, MoveIt, and the real robot using ESP in the Zeta Robot provided a powerful platform for developing and deploying a comprehensive robotic system. The combination of simulation, motion planning, and communication capabilities enabled us to test and refine the behavior of the Zeta Robot in a virtual environment, plan efficient and collision-free motion paths, and control the real robot seamlessly. This integration played a vital role in the successful development and deployment of the Zeta Robot, contributing to its overall functionality and performance.

3.1.5 Integrating moveit with external nodes

The software architecture of the humanoid robot is intricately woven using ROS. Nodes and packages are created to encapsulate specific functionalities, fostering a modular and scalable system. The implementation leverages ROS's messaging framework to facilitate seamless communication between nodes, enabling real-time data exchange.

ROS nodes dedicated to perception tasks, such as object detection and facial recognition, incorporate state-of-the-art algorithms. Deep learning models are trained on extensive datasets to recognize objects and faces, while image processing techniques analyze gestures for sign language translation. These nodes subscribe to sensory data from cameras and sensors, processing the information to generate actionable insights.

In ROS the actionlib package provides a framework for implementing and using asynchronous actions. Actions in ROS allow you to define high-level tasks or goals that can be executed by a robot or a system. The actionlib package consists of an action server and an action client, which work together to enable the execution and monitoring of these actions.

- **Action Server in Zeta Robot**

  In the context of Zeta Robot, an action server can be implemented to handle high-level tasks or actions that the robot needs to perform. For example, the action server could be responsible for executing actions such as grasping an object, navigating to a specific location, or performing a specific manipulation task. The action server receives action requests from the action clients and executes the necessary actions to fulfill those requests. It can provide feedback on the progress of the action and send a result once the action is completed.

- **Action Client in Zeta Robot**

  The action client in Zeta Robot is used to initiate action requests and monitor the progress of those actions. For instance, a high-level controller or a user interface can act as an action client and send action goals to the Zeta Robot. The action client can receive feedback from the action server, allowing it to provide updates to the user interface or take necessary actions based on the feedback. It can also receive the final result from the action server once the action is finished.

3.1.6 actionlib package benefits

The actionlib package can be beneficial for Zeta Robot in several ways:

- **Asynchronous Execution**

  Actions in actionlib are designed for asynchronous execution, allowing the robot to perform multiple tasks concurrently. This is particularly useful when Zeta Robot needs to handle complex actions that may take time to complete, such as performing a sequence of motions or interacting with the environment.

- **Feedback and Progress Monitoring**

  The action server can send periodic feedback messages to the action client, enabling real-time progress monitoring. This feedback mechanism allows Zeta

Robot to provide updates on the current state of the action, which can be useful for visualizing progress, making decisions, or providing feedback to the user.

- **Cancellation and Preemption**

  The action client can request the cancellation or preemption of an ongoing action if necessary. This feature allows Zeta Robot to adapt to changing circumstances or user requests by interrupting or stopping an ongoing action and initiating a new one.

- **Reusability and Modularity**

  The actionlib package promotes modularity and reusability of code. By encapsulating actions within action servers, different components of Zeta Robot can interact with the same action server interface, enabling code reuse and enhancing the overall flexibility and maintainability of the system.

Overall, the actionlib package and its components, the action server and action client, can provide a structured and efficient way to implement and manage complex actions in the Zeta Robot, enabling asynchronous execution, progress monitoring, and flexibility in action execution and control.

3.1.7 Perception and Sensing Node

The perception modules are a fundamental aspect of the robot's interaction with its environment. These modules employ state-of-the-art computer vision techniques and machine learning algorithms to interpret sensory data. Cameras capture visual information through Kinect v2 sensor and facilitate obstacle detection and mapping.

- **libfreenect2**

  libfreenect2 is an open source cross-platform driver for Kinect for Windows v2 devices. This documentation is designed for application developers who want to extract and use depth and color images from Kinect v2 for further processing.

  This documentation may require some understanding on camera calibration and 3-D geometry.

  i. Features

  - Color image processing
  - IR and depth image processing
  - Registration of color and depth images
  - Multiple GPU and hardware acceleration implementations for image processing

- **kinect2_bridge**

  kinect2_bridge is a ROS package that provides a bridge between the Kinect One (Kinect v2) and ROS [15]. It contains a library for depth registration with OpenCL support, a calibration tool for calibrating the IR sensor of the Kinect One to the RGB sensor and the depth measurements, and a bridge between libfreenect2 and ROS [15-16].

  The kinect2_bridge package is used to receive data from the Kinect-2 sensor in a way that is useful for robotics [15]. It is designed to work with point-clouds, depth-clouds, or images (computer vision) to do useful things with the data [15].

  To configure kinect2_bridge, you can follow the instructions provided in this tutorial2. The tutorial provides step-by-step instructions on how to install and configure kinect2_bridge on Ubuntu 16.04 and ROS Kinetic.

ROS nodes are developed to process sensory inputs and generate relevant information for decision-making. For instance, object detection nodes utilize deep learning models to identify and locate objects within the robot's field of view. Facial recognition modules employ neural networks to recognize and classify human faces. Gesture analysis nodes leverage real-time image processing to interpret sign language gestures.

3.2 Hardware components

3.2.1 Mechanical parts

3.2.1.1 3D-printing

Three-dimensional (3D) printing, also known as additive manufacturing, is a transformative technology that enables the creation of physical objects layer by layer from a digital model. It has revolutionized various industries by offering unprecedented capabilities in prototyping, customization, and small-scale production. 3D printing encompasses a range of techniques, materials, and processes, each with its unique advantages and applications.

One widely used 3D printing technique is Fused Deposition Modeling (FDM). FDM operates by extruding a thermoplastic filament through a heated nozzle, which forms thin layers of molten material that quickly solidify upon deposition. The nozzle moves in a controlled manner, following the desired path, and successive layers are added to build the final object. FDM offers several notable advantages, including affordability, ease of use, and a wide range of material options.

FDM 3D printing has found extensive use in various domains, including prototyping, product development, and small-scale manufacturing. Its ability to produce functional parts with relatively low cost and quick turnaround time has made it popular among hobbyists, designers, and engineers. The layer-by-layer deposition process allows for intricate geometries and complex internal structures, enabling the creation of objects with high levels of detail.

However, FDM does have some limitations. The surface finish of FDM-printed objects may exhibit visible layer lines, requiring post-processing steps for a smoother appearance. Additionally, the mechanical properties of FDM parts can be influenced by factors such as layer adhesion, orientation, and infill density. Optimization of printing parameters and material selection is crucial to achieve desired strength and durability.

Despite these limitations, FDM 3D printing continues to evolve, with advancements in printer technology, materials, and software. Innovations such as dual extrusion systems and soluble support materials have expanded the capabilities of FDM, allowing for the creation of more complex and functional objects. As 3D printing technology progresses, FDM is expected to further contribute to fields such as rapid prototyping, customized manufacturing, and even biomedical applications through the use of biocompatible materials.

In summary, 3D printing, including FDM, has revolutionized the way objects are designed, prototyped, and manufactured. FDM offers an accessible and versatile approach to 3D printing, making it an attractive option for various applications. As the technology continues to advance, it holds great potential for further innovation and impact in diverse industries.

3.2.1.2 Selection of 3D Printed Filament Types

In this section, we will compare two different types of 3D printed filaments used in the production of Zeta Robots. The filaments under consideration are PLA (Polylactic Acid) and PETG (Polyethylene Terephthalate Glycol). Each filament has its own unique properties and characteristics, which we will explore in detail.

Figure 3-26 PETG filament

Each filament type, PLA and PETG, has its own advantages and considerations. PLA is easy to use, biodegradable, and suitable for low-temperature applications. PETG offers a balance between strength, ease of printing, and temperature resistance, making it a versatile choice for various applications.

Intiaily the PETG filament was selected to print the Zeta Robot parts, considering factors such as desired mechanical properties, temperature resistance, printability, and environmental impact. But according to the existance of the materials in market, we had no choich unless use PLA to print the rest of the parts.

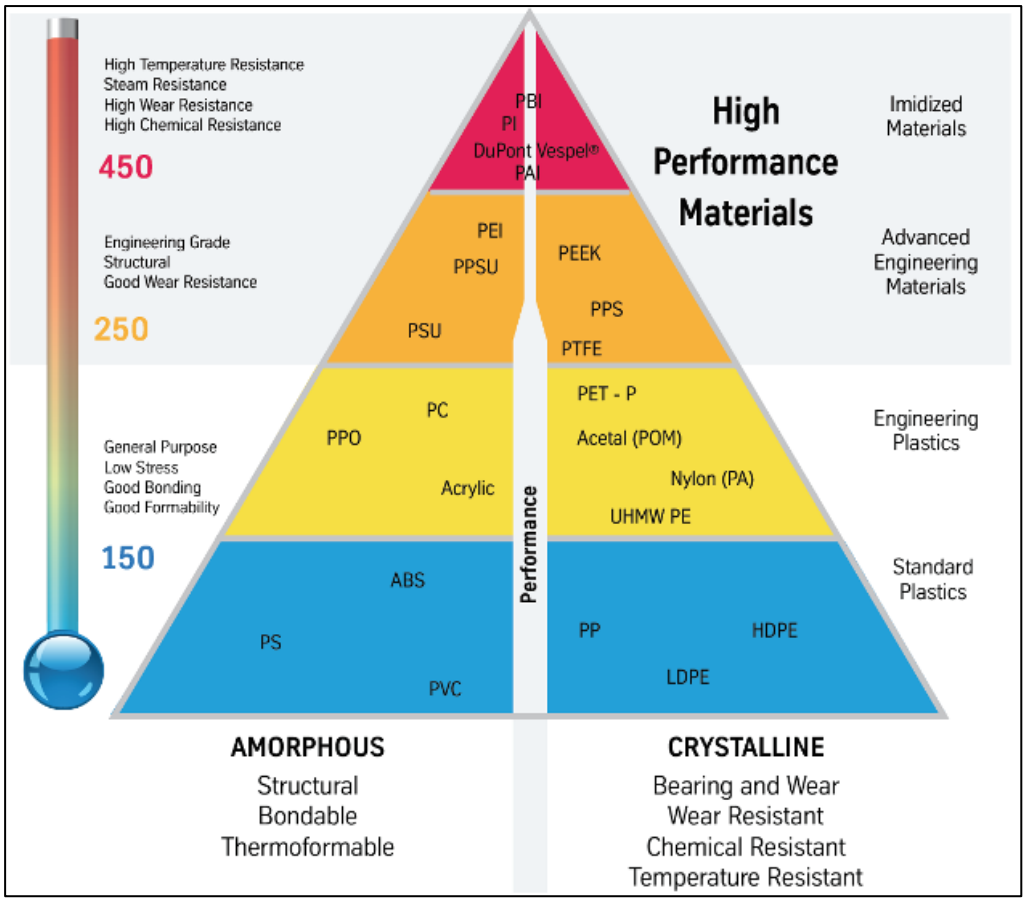


Figure 3-27 Comparison between different types of filament

3.2.1.3 *Selection of motors*

Figure 3-28 MG996R Servo motor

Servo motors are a really important part of the Zeta robot. The robot it's made up of 20 servos. All of them gives the robot its full ability of movement. When successfully assembled and working, all of these servos will provide Zeta robot with a human-like movements.

A modelling servo, usually just called servo is an actuator device with the ability to locate itself in any position within the operating range, and to stay stable at that exact same position. A typical servo consists of a direct current engine, a train of reduction gears and a control circuit. Its working range is usually less than a 360 degree turn. This kind of servos are usually found in radio controlled vehicles and robotics, although its use is not exclusively limited to those.

- **MG996R**

This High-Torque MG996R Digital Servo features metal gearing resulting in extra high 10kg stalling torque in a tiny package. The MG996R is essentially an upgraded version of the famous MG995 servo, and features upgraded shock-proofing and a redesigned PCB and IC control system that make it much more accurate than its predecessor. The gearing and motor have also been upgraded to improve dead bandwith and centering. The unit comes complete with 30cm wire and 3 pin 'S' type female header connector that fits most receivers, including

Futaba, JR, GWS, Cirrus, Blue Bird, Blue Arrow, Corona, Berg, Spektrum and Hitec.

This high-torque standard servo can rotate approximately 180 degrees (90 in each direction). You can use any servo code, hardware or library to control these servos, so it's great for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. The MG996R Metal Gear Servo also comes with a selection of arms and hardware to get you set up nice and fast!

- **Specifications**

Table 3-1 Specifications MG996R Servo motor

Name	Value
Weight	55 g
Dimension	40.7 x 19.7 x 42.9 mm approx.
Stall torque	9.4 kgf·cm (4.8 V), 11 kgf·cm (6 V)
Operating speed	0.17 s/60º (4.8 V), 0.14 s/60º (6 V)
Operating voltage	4.8 V a 7.2 V
Running Current	500 mA-900 mA (6V)
Stall Current	2.5 A (6V)
Dead band width	5 µs
Temperature range	0 ºC – 55 ºC
Stable and shock proof double ball bearing design double ball bearing design	

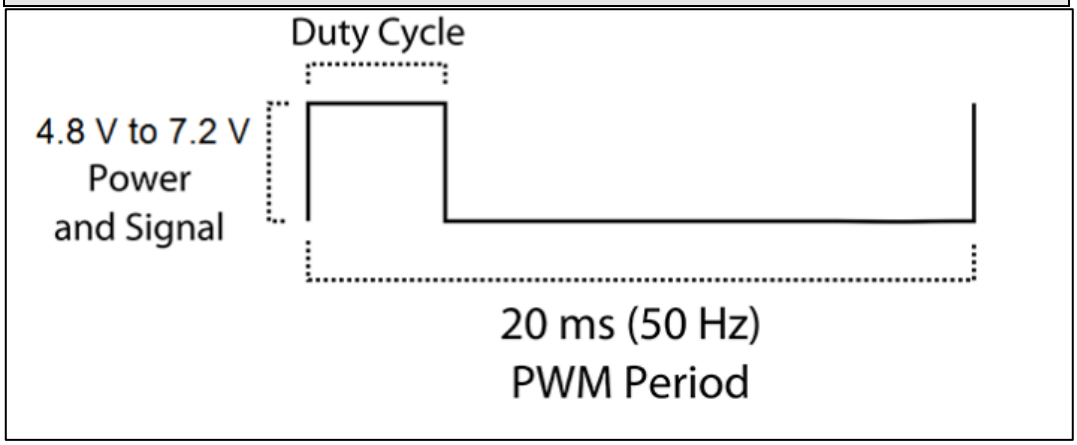


Figure 3-29 PWM and Duty cycle of MG996R Servo motor

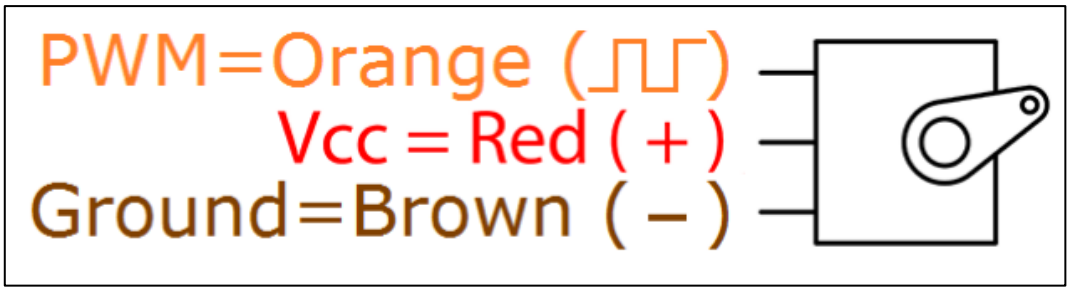


Figure 3-30 Hardware connection of MG996R Servo motor

- **Internal Structure and how it Works**

  The main component of a servo is a direct current engine, which acts as an actuator of the device. When applying a voltage between its two terminals, the engine will spin in a direction at high speed, but producing a low amount of torque. To maximize this torque it has group of reduction gears, with reduce the speed and amplify the torque.

  The device utilises a control circuit to ensure that the engine stays at the desired position with the help of a potentiometer. The reference point, or setpoint is the value of the desired position. This point is indicated by a square control signal. The pulse width of the signal indicates the angle position. A wider pulse means a higher angle.

  Initially, an error amplifier calculates the value of the position error, which is the difference between the setpoint and the current position of the engine. A bigger position error will mean that the offset between the desired position and the current position is bigger, therefore making the engine spin faster in order to achieve that position. A smaller position error will mean the engine is closer to the desired position, making the engine spin slower.Once the engine reaches the desired position, this position error will be zero, making the engine to stop.

  For the error amplifier to be able to calculate the position error it must subtract two analogic voltage values. The setpoint control signal is converted then in an analogic voltage value using a converter with reads the width pulse as a voltage. The value of the engine position is obtained using a potentiometer mechanically coupled to the reduction gears of the engine. When the engine will spin, so will the potentiometer, varying the value of the voltage that the error amplifier receives. Once obtained this position error, it is amplified and sent to the engine terminals.

3.2.2 Electrical parts

3.2.2.1 PCA9685

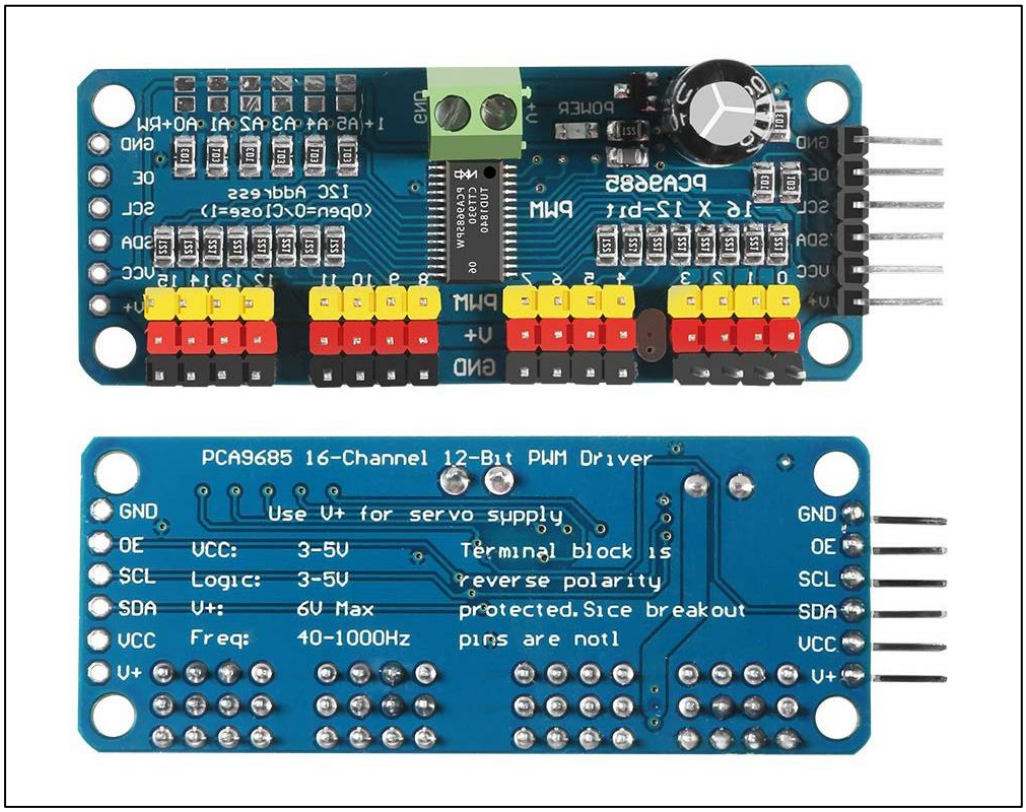


Figure 3-31 PCA9685 Driver

The PCA9685 is an I2C-bus controlled 16-channel LED controller optimized for Red/Green/Blue/Amber (RGBA) color backlighting applications. Each LED output has its own 12-bit resolution (4096 steps) fixed frequency individual PWM controller that operates at a programmable frequency from a typical of 24 Hz to 1526 Hz with a duty cycle that is adjustable from 0 % to 100 % to allow the LED to be set to a specific brightness value. All outputs are set to the same PWM frequency.

3.2.2.2 *8A 180KHz 40V Buck DC to DC Converter*

Figure 3-32 8A 180KHz 40V Buck DC to DC Converter

The XL4016 is a 180 KHz fixed frequency PWM buck (step-down) DC/DC converter, capable of driving a 8A load with high efficiency, low ripple and excellent line and load regulation. Requiring a minimum number of external components, the regulator is simple to use and include internal frequency compensation and a fixed-frequency oscillator. The PWM control circuit is able to adjust the duty ratio linearly from 0 to 100%. An over current protection function is built inside. When short protection function happens, the operation frequency will be reduced from 180KHz to 48KHz. An internal compensation block is built in to minimize external component count.

3.2.2.3 *Vertical Linear Rotary Trimmer Potentiometer Variable Resistor Varistor*

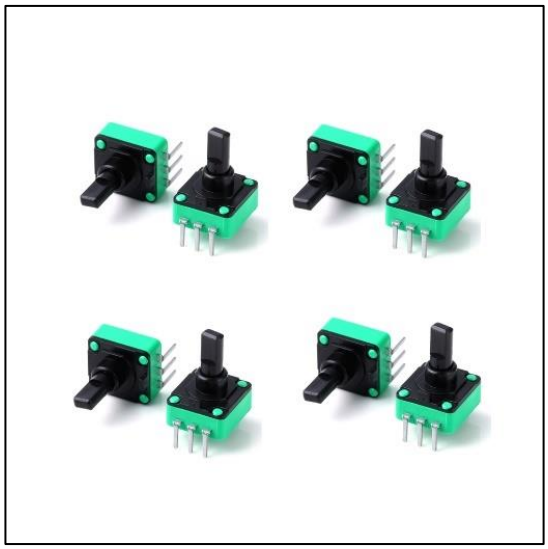


Figure 3-33 Vertical Linear Rotary Trimmer Potentiometer

- **Feature**

  1.Type: Through Hole, Vertical
  2.Color: Green
  3.Taper Law: Linear
  4.Resistance: 100Ω~100KΩ
  5.Tolerance: ±10%

- **Electrical**

  1.Rated Voltage: <50V
  2.Rated Power: 0.5W
  3.Sliding Noise: 100mV

- **Mechanical**

  1.Rotational Torque: 5~15gf
  2.Total Rotational Angle: 260±10º
  3.Rotational Stop Strength: ≥3.0kgf
  4.Operating Life: ≥100,000Cycles

3.2.2.4 *ESP32*

Figure 3-34 ESP 32

ESP32 is a powerful and versatile microcontroller-based system-on-chip (SoC) that has gained popularity in the field of embedded systems and Internet of Things (IoT) applications. Developed by Espressif Systems, the ESP32 offers a wide range of features and capabilities, making it suitable for a variety of projects.

The ESP32 SoC combines a dual-core processor, Wi-Fi connectivity, Bluetooth Low Energy (BLE), and a rich set of peripherals, all integrated into a single chip. This integration provides developers with a compact and cost-effective solution for building IoT devices and applications.

- **Key features of the ESP32 include:**

  - Dual-Core Processor: The ESP32 features two powerful Tensilica Xtensa LX6 processors, which allow for parallel task execution and improved performance. This enables developers to run multiple tasks simultaneously or distribute processing load across the cores for optimal efficiency.
  - Wi-Fi and Bluetooth Connectivity: The ESP32 supports both Wi-Fi 802.11b/g/n and Bluetooth 4.2/BLE, allowing seamless wireless communication with other devices and networks. This connectivity capability makes the ESP32 well-suited for IoT projects that require wireless data transfer, remote control, or network connectivity.

- Rich Peripheral Set: The ESP32 offers a variety of peripherals, including GPIO (General Purpose Input/Output) pins, UART (Universal Asynchronous Receiver-Transmitter) interfaces, I2C (Inter-Integrated Circuit) and SPI (Serial Peripheral Interface) buses, and analog-to-digital converters (ADCs). These peripherals enable interfacing with sensors, actuators, displays, and other external devices.
- Low Power Consumption: The ESP32 is designed to operate efficiently in low-power scenarios. It includes sleep modes and power management features that help conserve energy, making it suitable for battery-powered and energy-conscious applications.
- Development Ecosystem: The ESP32 is supported by a comprehensive development ecosystem, including the Arduino IDE, Espressif IDF (IoT Development Framework), and various libraries and tools. These resources simplify the development process, provide extensive documentation, and offer a rich set of examples and community support.

The versatility and capabilities of the ESP32 have made it a popular choice for a wide range of IoT projects and applications. It is used in projects such as home automation, smart agriculture, industrial monitoring, wearable devices, and more. With its powerful processing capabilities, wireless connectivity, and rich peripheral set, the ESP32 offers a flexible platform for building connected and intelligent devices.

3.2.2.5 *Raspberry Pi*

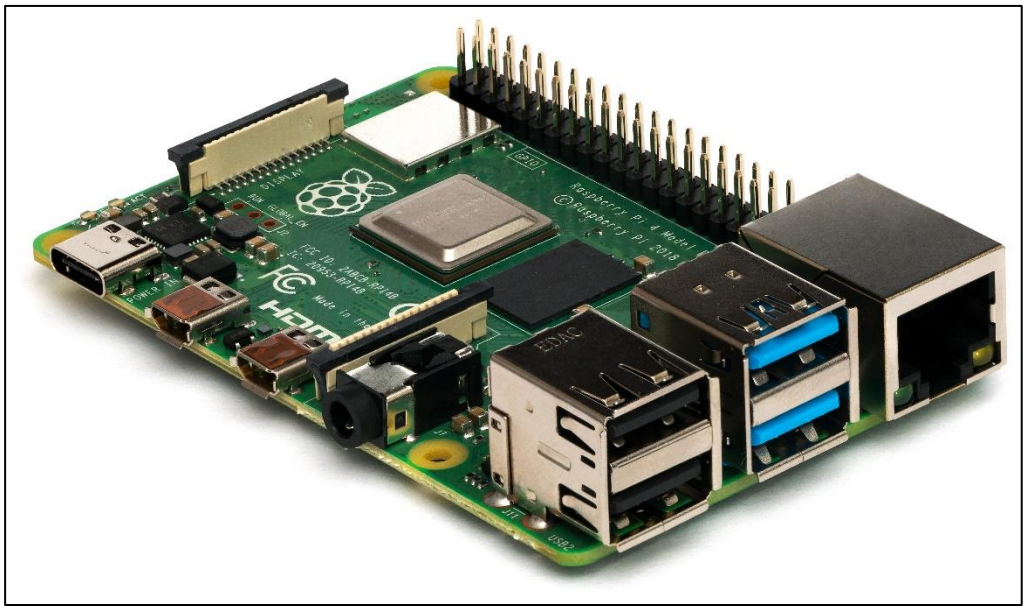


Figure 3-35 Raspberry Pi

Raspberry Pi is a series of single-board computers (SBCs) developed by the Raspberry Pi Foundation. These compact and affordable computers are designed to promote computer science education and enable users to explore programming, electronics, and various projects.

- **Key features of Raspberry Pi include:**

  - Hardware: Raspberry Pi boards feature a Broadcom system-on-chip (SoC), which includes a central processing unit (CPU), graphics processing unit (GPU), and other components. They also have various ports for connectivity, such as USB, HDMI, Ethernet, audio, and GPIO pins for interfacing with external devices.
  - Operating System: Raspberry Pi supports different operating systems, including the official Raspberry Pi OS (formerly known as Raspbian), which is a Debian-based Linux distribution optimized for the Raspberry Pi. Other operating systems like Ubuntu, Windows 10 IoT Core, and various Linux distributions can also be installed on Raspberry Pi.
  - Programming and Software: Raspberry Pi supports a wide range of programming languages, including Python, C/C++, Java, and Scratch. It allows users to develop software applications, control hardware components, and interface with sensors and actuators. It also supports popular software frameworks and libraries, making it versatile for various projects.
  - Connectivity and Networking: Raspberry Pi boards come with built-in Ethernet and Wi-Fi capabilities, allowing them to connect to local networks

and the internet. This enables users to create devices for IoT applications, web servers, media centers, and more.

- Expansion and Customization: Raspberry Pi boards have GPIO pins, which allow for easy connection and control of external components and circuits. This flexibility makes it suitable for electronic projects, robotics, home automation, and prototyping.
- Community and Resources: Raspberry Pi has a large and active community of users, developers, and enthusiasts. This community provides extensive resources, documentation, tutorials, and projects, making it easy for beginners to get started and for experienced users to explore advanced applications.

Raspberry Pi has been widely adopted for various projects, including home automation, robotics, media streaming, retro gaming consoles, weather stations, and educational tools. Its affordability, versatility, and support from a strong community have made it a popular choice for hobbyists, students, and professionals alike, encouraging innovation and learning in the field of computer science and technology.

3.2.2.6 *Kinect*

Figure 3-36 Kinect V2

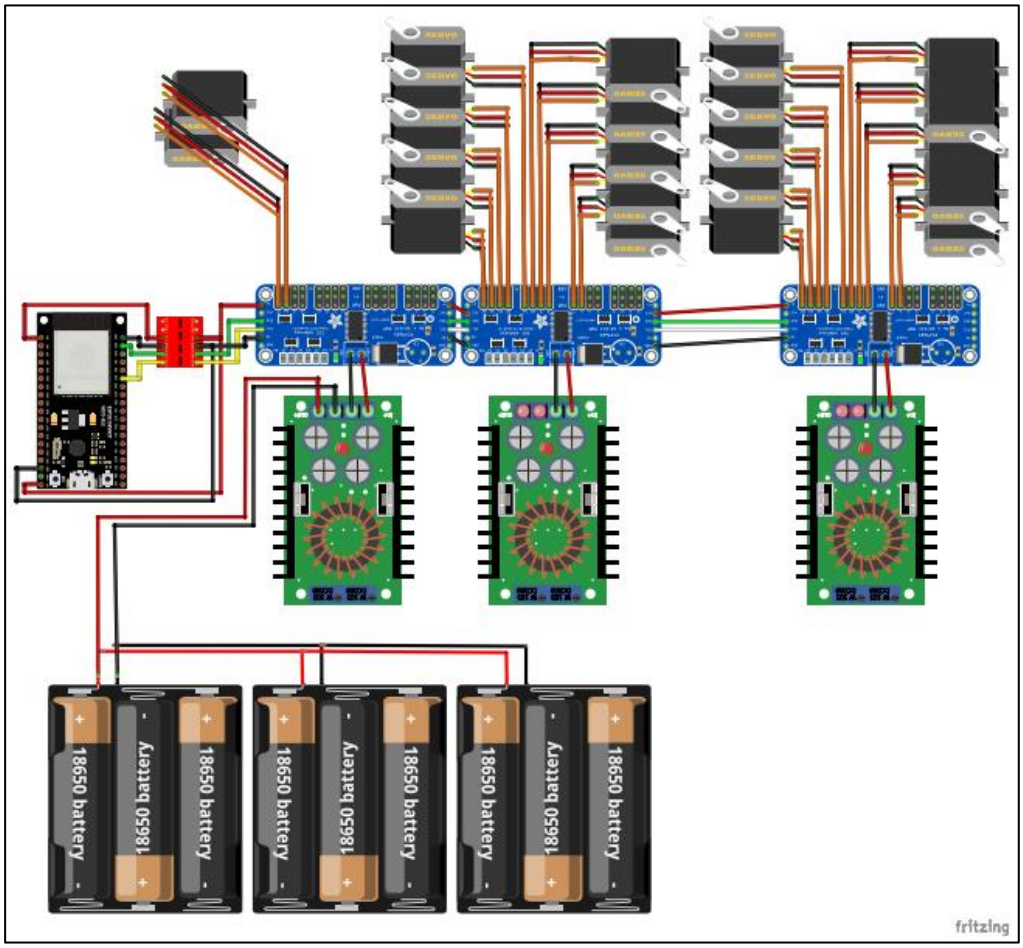
Kinect is a series of motion-sensing devices made by Microsoft, introduced in 2010. These devices have cameras, infrared projectors, and detectors that measure depth using structured light or time of flight calculations. This depth data can be used for things like recognizing gestures and detecting body movement in real-time. Kinect devices also come with microphones for speech recognition and voice control.

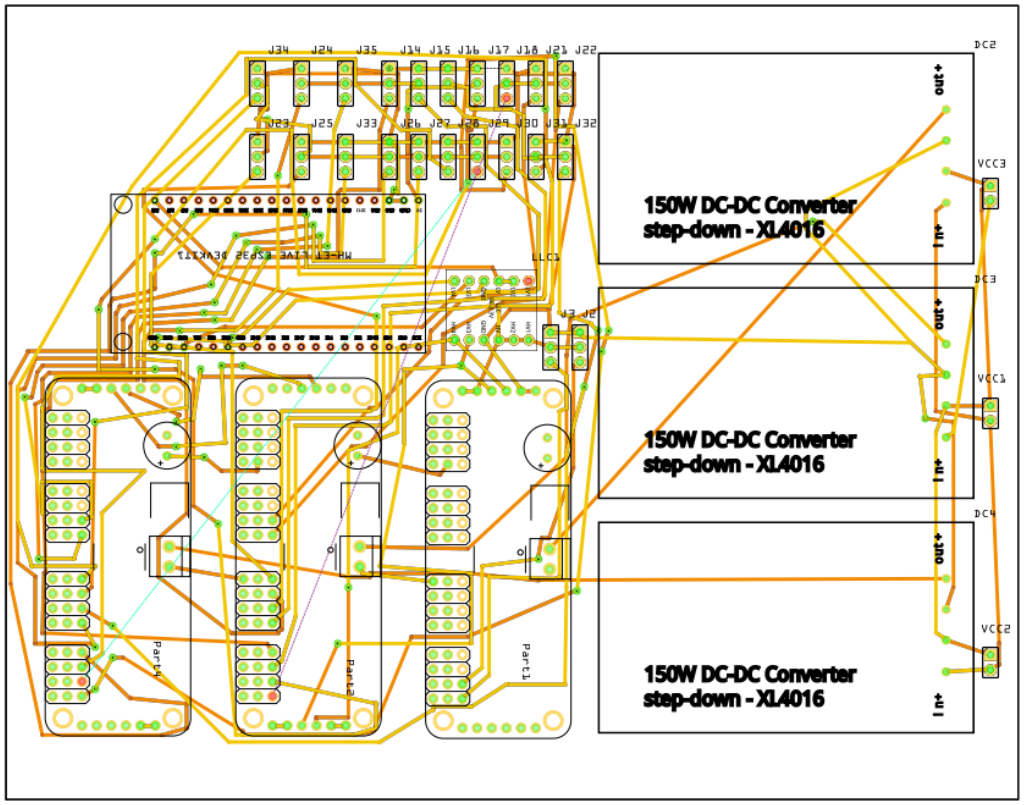
Initially designed as a motion controller for Xbox video game consoles, Kinect set itself apart by not needing physical controllers like its competitors, the Wii Remote and PlayStation Move. The first version of Kinect, based on technology from PrimeSense, was announced in 2009 and released in 2010. It sold eight million units in its first two months. Most Kinect games were family-oriented, appealing to new Xbox 360 users, but not the existing userbase.

With the launch of the Xbox One in 2013, a new version of Kinect was introduced with improved tracking. Initially, Microsoft made Kinect a required part of the console, which sparked privacy concerns. They later changed this requirement. Kinect was bundled with Xbox One consoles initially, but the market for Kinect-based games didn't take off. Microsoft eventually offered Xbox One bundles without Kinect and stopped making Kinect for Xbox One in 2017.

Kinect found uses beyond gaming, especially in academic and commercial settings. It became a popular choice for depth sensing due to its affordability and

reliability. Microsoft released software development kits (SDKs) for making Windows applications that used Kinect. In 2020, they introduced Azure Kinect as part of their cloud computing platform, and elements of Kinect technology were used in Microsoft's Hololens project.





3.2.3 Hardware Construction and Assembly

The physical realization of the humanoid robot is a culmination of meticulous design and precise assembly. Each component, from the articulated joints to the sensory modules, is carefully integrated to create a functional and aesthetically appealing robot. Zeta robot structure is composed of lightweight yet durable materials, ensuring both stability and agility.

Mechanical joints, designed to replicate human articulation, undergo precise calibration to enable natural and fluid movements. Actuators, selected for their torque capabilities and efficiency, are meticulously attached to mimic muscles' actions.

3.2.3.1. Parts printing

To get started with the robot building, first you will need to have all the parts printed. Regardless of the printer that you are using, any that has more than 12x12x12 centimetres of printing surface will work for this purpose.

Figure 3-37 Fresh printed parts

There are some parameters though that you can specify in every software of a 3D printer, and that you need to be aware of in order to print the pieces correctly, and be sure that they won't break down while the robot is functioning. The following specifications are the ones that we have used to print all the pieces of the robot except for the parts called piston. These parts require more infill (75-100%) to be resistant enough. Also, the parts called worm will need support activated in order to be printed correctly. For the rest of the parts the specifications are as it follows:

- Infill: 30%
- Nº of shells: 3
- Layer thickness: 0.3 mm
- Raft: None
- Support: None

Following we've completed a table with the estimated printing times of all the pieces designed so far, so it's easily seen how long this project can take. Just the printing alone adds up to an estimated combined total printing time of 464.4 hours. In our case, we've had at the disposal of the project two different printers, which helped to reduce the amount of time needed to print everything substantially.

Table 3-2 3D Printed Parts and there time estimated for print

Part	Divergent folder that contains file	File name	Time to print
Hand		Auriculaire3.stl	36
Hand		Bolt_entretoise5.stl	72
Hand		cableholder1.stl	9
Hand		cableholderwrist1	9
Hand		coverfinger1.stl	9
Hand		Index3.stl	36

Hand		Majeure3.stl	36
Hand		ringfinger3.stl	36
Hand		robcap3V1.stl	126
Hand		robpart2V2.stl	432
Hand		robpart3V3.stl	432
Hand		robpart4v3.stl	432
Hand		robpart5V2.stl	540
Hand		rotawrist1V2	270
Hand		rotawrist2	216
Hand		rotawrist3	54
Hand		stand1	27
Hand		stand2	54
Hand		thumb5.stl	36
Hand		topsurface4.stl	360
Hand		Wristgears3	36
Hand		WristlargeV3.stl	324
Hand		WristsmallV3.stl	162
Hand Left	Hand	Auriculaire3.stl	36
Hand Left		Bolt_entretoise4.stl	72
Hand Left		cableholder1.stl	9
Hand Left	Hand	cableholderwrist1	9
Hand Left	Hand	Index3.stl	36
Hand Left		leftarduinosupport.stl	36
Hand Left		leftcoverfinger1.stl	9
Hand Left		leftrobcap3V1.stl	126
Hand Left		leftrobpart1.stl	432
Hand Left		leftrobpart2V2.stl	432
Hand Left	Hand	Index3.stl	36
Hand Left	Hand	Majeure3.stl	36
Hand Left	Hand	rotawrist3	54
Hand Left	Hand	Wristgears3	36
Upper arm		armtopcover1.stl	360
Upper arm		armtopcover2.stl	288
Upper arm		armtopcover3.stl	432
Upper arm		elbowshaftgearV1.stl	108
Upper arm		gearholderV1.stl	36
Upper arm		gearpotentioV1.stl	18
Upper arm		higharmsideV1.stl	180
Upper arm		lowarmsideV1.stl	90
Upper arm		PistonanticlockV1.stl	180
Upper arm		PistonbaseantiV1.stl	180
Upper arm		reinforcerV1.stl	108
Upper arm		RotcenterV2.stl	540
Upper arm		RotGearV3.stl	234
Upper arm		RotMitV2.stl	324
Upper arm		RotPotentioV2.stl	36
Upper arm		RotTitV2.stl	252

Upper arm		RotWormV5.stl	108
Upper arm		servobaseV1.stl	126
Upper arm		servoholderV1.stl	198
Upper arm		spacerV1.stl	36
Upper arm left	Upper arm	armtopcover1.stl	360
Upper arm left	Upper arm	armtopcover2.stl	288
Upper arm left	Upper arm	armtopcover3.stl	432
Upper arm left	Upper arm	elbowshaftgearV1.stl	108
Upper arm left	Upper arm	gearholderV1.stl	36
Upper arm left	Upper arm	gearpotentioV1.stl	18
Upper arm left	Upper arm	higharmsideV1.stl	180
Upper arm left		leftRotcenterV2	540
Upper arm left		leftRotTitV2	252
Upper arm left	Upper arm	lowarmsideV1.stl	90
Upper arm left	Upper arm	PistonanticlockV1.stl	180
Upper arm left	Upper arm	PistonbaseantiV1.stl	180
Upper arm left	Upper arm	reinforcerV1.stl	108
Upper arm left	Upper arm	RotGearV3.stl	234
Upper arm left	Upper arm	RotPotentioV2.stl	36
Upper arm left	Upper arm	RotWormV5.stl	108
Upper arm left	Upper arm	servobaseV1.stl	126
Upper arm left	Upper arm	servoholderV1.stl	198
Upper arm left	Upper arm	spacerV1.stl	36
Shoulder		ClaviBackV1.stl	324
Shoulder		ClaviFrontV1.stl	324
Shoulder		PistonbaseV4.stl	180
Shoulder		PistonClaviV2.stl	180
Shoulder		PivcenterV1.stl	540
Shoulder		PivConnectorV1.stl	108
Shoulder		PivGearV3.stl	234
Shoulder		PivMitV1.stl	324
Shoulder		PivPotentioV2.stl	18
Shoulder		PivPotholderV2.stl	90
Shoulder		PivTitV1.stl	252
Shoulder		PivWormV2.stl	108
Shoulder		servoholderV1.stl	198
Shoulder		servoHolsterV1.stl	180
Shoulder left	Shoulder	ClaviBackV1.stl	324
Shoulder left	Shoulder	ClaviFrontV1.stl	324
Shoulder left	Shoulder	PistonbaseV4.stl	180
Shoulder left	Shoulder	PistonClaviV2.stl	180
Shoulder left	Shoulder	PivConnectorV1.stl	108
Shoulder left	Shoulder	PivGearV3.stl	234
Shoulder left		leftPivcenterV1.stl	540
Shoulder left		leftPivMitV1.stl	324
Shoulder left		LeftPivPotholderV2.stl	90
Shoulder left		leftPivTitV1.stl	252

Shoulder left	Shoulder	PivPotentioV2.stl	18
Shoulder left	Shoulder	PivWormV2.stl	108
Shoulder left	Shoulder	servoholderV1.stl	198
Shoulder left	Shoulder	servoHolsterV1.stl	180
Torso		homplateback-V1.stl	198
Torso		homplateback+V1.stl	198
Torso		homplatebacklow-V1.stl	180
Torso		homplatebacklow+V1.stl	180
Torso		homplatefront-V1.stl	180
Torso		homplatefront+V1.stl	180
Torso		servoHolsterV1.stl	180
Torso		SternumV1.stl	180
Torso		ThroatLowerV1.stl	180
Neck		FaceHolderV2.stl	108
Neck		GearHolderV1.stl	216
Neck		MainGearV1.stl	126
Neck		NeckBoltsV2.stl	108
Neck		NeckHingeV1.stl	252
Neck		NeckV1.stl	540
Neck		RingV1.stl	54
Neck		ServoGearV1.stl	126
Neck		SkullServoFixV1.stl	288
Neck		TemporaryV1.stl	90
Neck		ThroatHolderV1.stl	72
Neck		ThroatHoleV2.stl	162
Neck		ThroatPistonBaseV2.stl	324
Neck		ThroatPistonV2.stl	234
Jaw		JawHingeV1.stl	72
Jaw		JawHingeV1-1.stl	63
Jaw		JawHingeV2.stl	108
Jaw		JawPistonV1.stl	378
Jaw		JawSupportV1.stl	270
Jaw		JawV3.stl	432
Skull and Face		EyeglassV2.stl	234
Skull and Face		LowbackV2.stl	432
Skull and Face		SideHearV2.stl	234
Skull and Face		TopBackskullV1.stl	468
Skull and Face		TopMouthV2.stl	360
Skull and Face		TopskullLeftV2.stl	540
Skull and Face		TopskullRightV2.stl	540
Eye mechanism		EyeCameraV3.stl	126
Eye mechanism		EyeMoverSideV3.stl	18
Eye mechanism		EyeMoverUpV3.stl	18
Eye mechanism		EyeSupportV3.stl	126
Eye mechanism		EyeToNoseV4.stl	54
Ears		EarRightV1	234
Ears		EarLeftV1	234

3.2.3.2. *Gears*

Gears are mechanical devices used to transmit torque and motion between rotating shafts.

Screw gears, also known as helical gears, have helical teeth that are angled around the gear's circumference. They are commonly used when smooth and quiet operation is required.

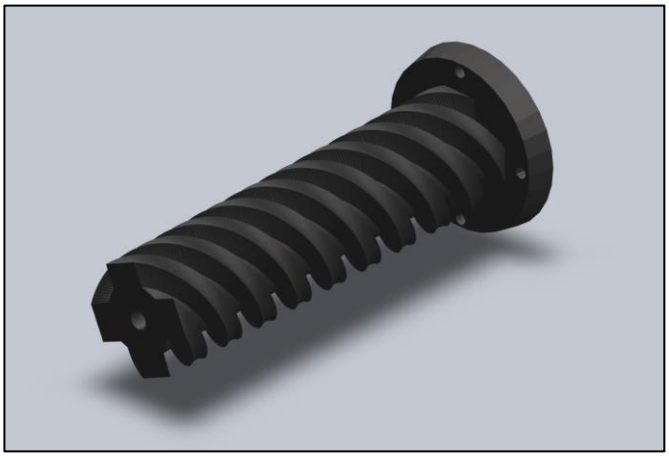


Figure 3-38 Screw gears

On the other hand, worm gears consist of a worm (a type of screw) and a mating gear called a worm wheel or worm gear. The worm has a spiral thread that meshes with the teeth of the worm wheel, which is usually perpendicular to the worm's axis. Worm gears are used when a high gear ratio is needed, and they provide a compact and efficient means of power transmission. They are commonly found in applications like lifts, conveyor systems, and automotive steering mechanisms.

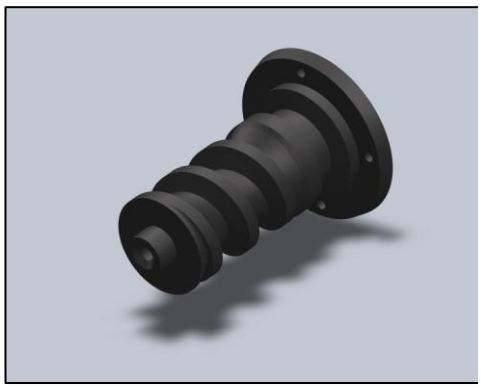


Figure 3-39 Worm gear

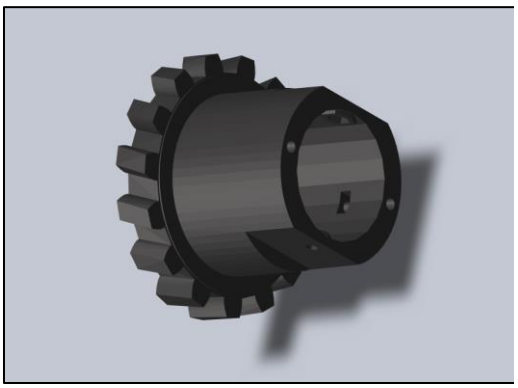


Figure 3-40 Mating gear

3.2.3.3. *Hacking Servos*

For the building of the Zeta robot, there are a lot of servos involved. Some of them will need to be hacked in order to remove the potentiometer from inside and place it in an articulation of the robot. This is done in order to send to the

servo's control board the exact position of the articulation and regulate it according to that reference.

The idea was to contribute to the assembling of it and to make it work as expected. We have been able to fully assemble both complete arms of the robot, head , face, neck, and part of the torso. Following are the instructions of all the steps that we had to make in order to ensure the correct functioning of the mentioned parts.

3.2.3.4. Assembling the Hand and Wrist

Figure 3-41 Some of the hand, wrist, and forearm parts

To begin with, we started working on the robot hands. To ensure a better grip for the robot. We have to take in consideration that trough the fingers there is going to be the tendons pulling the fingers

Bearing that in mind we should be pulling the tendons through the fingers while we are assembling them. To make the job easier, you should group the two cables and two tendons that run through each finger all together and assemble one finger at a time. Start by pulling tendons through the wrist part and the palm of the hand. Then pull carefully through each finger taking in consideration that the tendons should go one on the upper part of the finger and one on the lower part. By doing this for the rest of fingers in both hands, the hands will be assembled.

With the hand ready, we need to provide the wrist with the rotation movement. This part will be attached next to the hand, and before the forearm. This part will count with one small servo inside that will provide the force for the rotation. We will need to first assemble the wrist separately with the servo, a custom printed gear to allow the movement and a perforated plastic plate that will ensure that the cables run straight through the wrist. Once fully assembled we will attach it

to the hand. To do so we will run the tendons carefully through the wrist, and add some grease for the lubrication between gears.

3.2.3.5. Assembling the Forearm

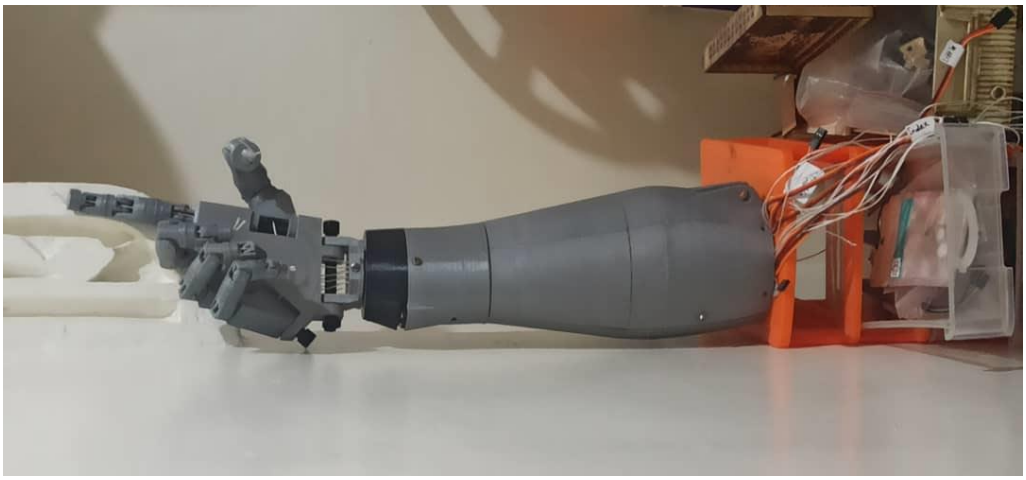


Figure 3-42 Forearm, Hand, and Wrist assembly

Once the hand is finished we are going to attach it to the forearm. To do so, first the forearm parts need to be printed and glued together. To glue plastic together you have to take in consideration which kind of plastic you are working with. In case you are working with ABS you will need acetone, and for PLA you will need epoxy. In our case we worked with PLA, but the process is the same in either case.

Now that the base of the forearm is finished the servos will need to be placed inside. This servos are the ones who will make each finger move. You first need to screw in the mount of the servos, and then screw each servo in place. Once all servos are in place we will attach custom printed gears on top of them which will help to pull the fingers correctly. Just one screw is necessary to attach these gears. After we are going to assemble to forearm part to the wrist with glue.

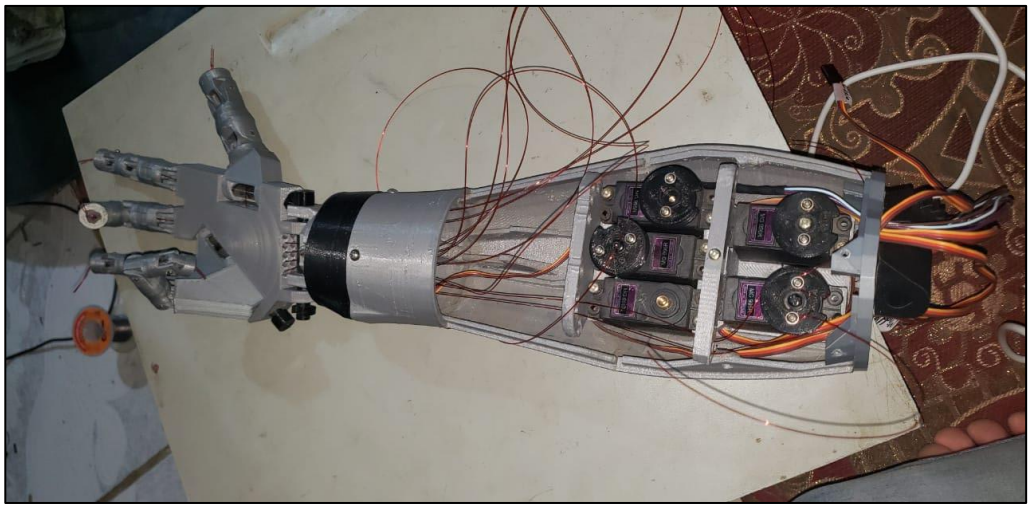


Figure 3-43 Importing the tendons step

Last and most important you are going to run the tendons through the servo gears, carefully, making sure that there's enough tension at the lines to allow the correct movement of the fingers. We attached each servo to and ESP32 board and the power supply to test each time until the correct positioning and tension was accomplished. It took several tries to achieve the perfect tension to make the fingers move smoothly in both directions (closing and opening).

You need to remember to place the servos at the default 0º position, and place the fingers straight before attaching the tendons.

3.2.3.6. Assembling the Bicep

To complete the arm, last missing part is the bicep. This part contains two servos responsible for the rotation of the whole arm, and for the movement of the elbow articulation.

For the assembly of the bicep part, we are going to begin with the upper part responsible for the rotation of the complete arm. Since this part has worm drive gear arrangement, a special effort will be needed when polishing the parts after printing.

During this build, all of the printed parts have been modified after the printing due to the printer tolerance between the model and the final printed product. Especially with this type of gear arrangement, it needs some further polishing to ensure the smoothness of the movement.

Once the gears are perfectly working, we will screw in the servo, putting the potentiometer of the servo in the middle of the worn gear. This potentiometer is usually located inside of the servo, but in this case it will need to be previously remove it from the inside and extend it using some 20 centimeters of extra cable as it is explained at section 4.2.2 Hacking Servos.

Figure 3-44 Bicep

We will then proceed to put the top to the worm drive gears after applying some grease between them. A test of the servo should be done before continuing with the assembly to assure the correct functioning of it and to stablish the neutral position (usually located around 90º).

Now all the plastic printed parts which form the structure of the bicep will be added to the previous part. Before attaching the last servo we will assemble this next to the forearm using a printed plastic screw to hold them together.

Once the arm is fully assembled we can then attach the bicep servo in its place, with its potentiometer at the elbow articulation. Finally a careful test should be conducted to find out the limits of the articulation, and therefore limit the movement of the servo. This result may vary in each case. Last only the covers of the bicep should be added to finish it. The exact same process will be used for both arms.

3.2.3.7. Assembling the Shoulders and Torso

For assembling the next part of the robot you will need 4 servo motors which also need to be hacked as the servos from the bicep part by removing the potentiometer and making the cables around 25 cm longer. Two servos are used for the shoulder movement and another two are used for the scapula movement.

The first step was to hack the servo motors. For the right side of the robot the cables of the potentiometer should the same as it was before extracting it from the servo but for the left side you will have to change the cables from the sides of the potentiometer in order to change the polarity of the servos. It will make the servo to rotate in the opposite way.

Figure 3-45 Shoulder parts

Next step is to build the shoulder, this part is similar to the upper part of the bicep. It contains the same worm gear mechanism and it needs the same actions as in the bicep part to assemble it.

The next step is to attach the servo which will create the scapular movement to the shoulder part which later will be attached to the torso.

Figure 3-46 Torso parts

Next step is to assemble the torso. First we have to mount together all the parts of the back side of torso. This part is quite easy since it's a lot like assembling a Lego, but the difficult part is that all the pieces will have to be polished with sand paper for quite some time to make everything fit. At the same time the servo holder for the servo motor which will produce the movement of the head moving up/down should be placed inside of the torso and the piston base part for the scapula movement.

Once everything is together we can put the front parts of the torso and start assembling the bottom side of torso which will hold the Kinect. Now at this point is time to assemble together the shoulder and the torso, but first we have to put a good amount of grease on the piston screws which will move the scapula, and assemble together the shoulders to the torso. We will set the potentiometer for the scapula on the back side of the shoulders connected to the torso.

3.2.3.8. Assembling the Head and Neck

Figure 3-47 Neck parts

Once the torso is done, we start assembling the neck to the right place in torso take in considertion the front and the back side of the torso. The neck and head neads both two servos. One for yaw movment and another for the pitch movment.

Figure 3-48 Neck

Next step is to add servo for the pitch movment, this part is similar to the upper part of the bicep and the shoulder. It contains the same worm gear but different ratio mechanism and it needs the same actions as in the bicep and shoulder parts to assemble it.

Figure 3-49 Neck and Torso

Next step is to add servo for yaw movment and connected to the assembled skull and face. Once everything is done assembled we started calibrating the angles in simulation to match the angles in real robot.

3.3 AI

3.3.1 Computer Vision Nodes

3.3.1.1. Overveiw

Computer vision refers to the field of study that focuses on enabling machines to interpret and understand visual information, much like how humans perceive and comprehend images and videos. It involves the development and application of algorithms and techniques to extract meaningful insights from visual data.

OpenCV is a library of programming functions mainly aimed on real time computer vision. Originally developed by Intel, it is later supported by Willow Garage .The library is a cross-platform and free to use under the open-source BSD license.

Face detection is the first step in face recognition. Dlib provides robust algorithms that can accurately locate and identify faces in images or videos. By leveraging machine learning techniques, dlib can handle challenging conditions such as variations in lighting, pose, and occlusion.

Once faces are detected, dlib allows us to perform face landmark detection. This involves identifying key facial landmarks such as the position of the eyes, nose, and mouth. These landmarks act as anchor points for subsequent analysis and enable tasks like facial expression recognition and augmented reality experiences.

The true power of dlib lies in its ability to perform face recognition. By utilizing advanced algorithms and machine learning models, dlib can compare and match faces against a database of known individuals. This technology has significant implications in security, access control systems, and personalized experiences.

Implementation using the (ROS) with dlib for face recognition, we would commonly create two nodes: one for publishing camera views and another for subscribing to the camera view and performing face recognition. Here's an overview of how these nodes would work together:

3.3.1.2. Camera Publisher Node:

This node is responsible for capturing camera images or video frames and publishing them to a specific ROS topic. It utilizes ROS libraries and camera drivers to access the camera feed and retrieve the images. The images are then converted into a ROS message format (such as sensor_msgs/Image) and published on a designated topic.

3.3.1.3. Face Recognition Node:

This node subscribes to the camera view topic and receives the published images. It uses the dlib library to perform face detection, landmark detection, feature

extraction, and face recognition on the received images. The node applies the necessary dlib functions to process the images and extract facial features. It then compares these features with a database of known faces to determine potential matches.

a. ***Steps for Face Recognition Node***

- Receive the camera view image from the subscribed topic.
- Utilize dlib's face detection algorithms to locate faces within the image.
- Apply landmark detection algorithms to identify key facial landmarks.
- Extract unique facial features based on the detected landmarks.
- Compare the extracted features with a pre-existing database of known faces.
- Publish the recognition results, such as the recognized person's identity or a confidence score, on an appropriate ROS topic.

By separating the camera publishing and face recognition functionality into two nodes, we can achieve modularity and flexibility in your ROS system. This allows you to independently modify or replace each node as needed, facilitating easier development and maintenance of the overall system.

b. ***Dlib accuracy factors:***

Dlib is known for its accuracy in face recognition due to several key factors:

- **Robust Algorithm**: Dlib utilizes a robust algorithm that combines machine learning techniques with sophisticated image processing methods. This algorithm has been trained on large datasets, allowing it to learn and recognize facial features effectively.
- **Face Detection and Landmark Detection**: Dlib's face recognition process begins with accurate face detection and landmark detection. The library employs advanced techniques to locate faces in images or videos and identify key facial landmarks such as the eyes, nose, and mouth. This precise detection helps ensure accurate recognition.

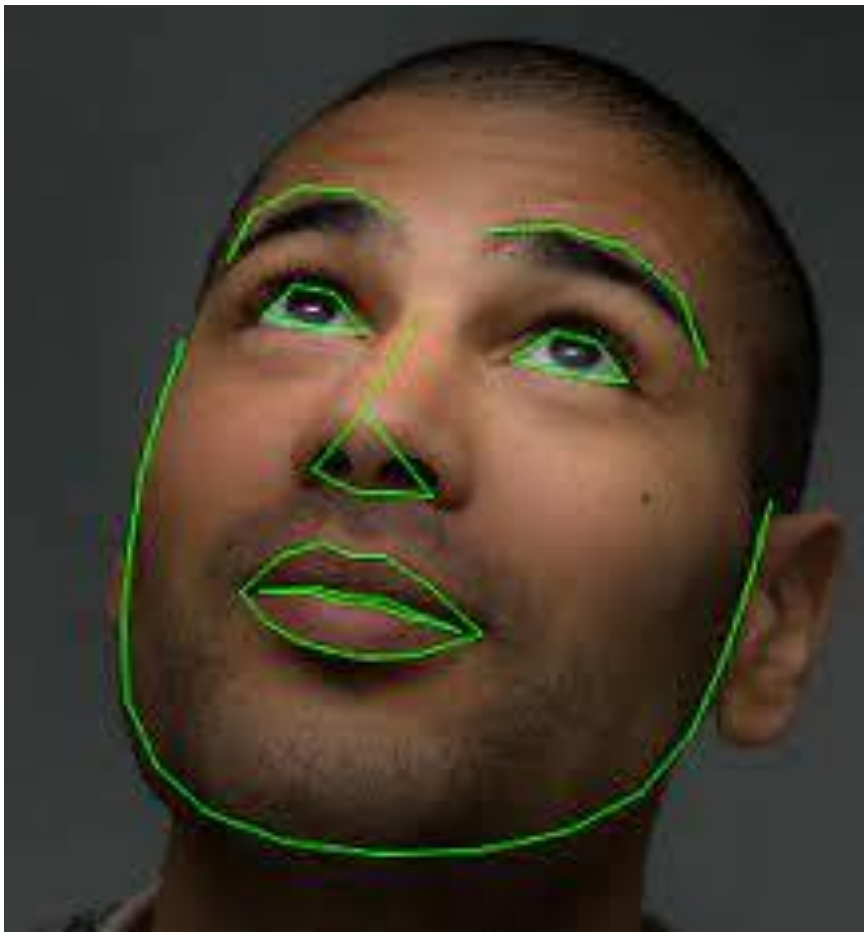


Figure 3-50 Landmark Detection

- **Feature Extraction and Encoding**: Dlib extracts unique facial features from the detected landmarks and encodes them into a numerical representation. This encoding captures the distinctive characteristics of each face, making it easier to compare and match faces accurately.

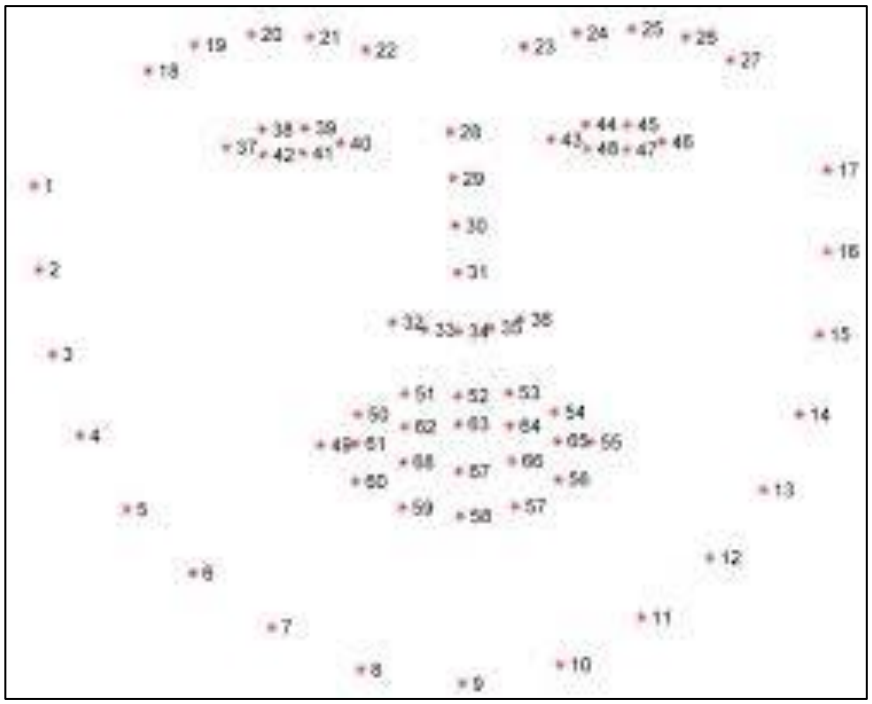


Figure 3-51 Extraction and Encoding example

- **Training on Large Datasets:** Dlib's face recognition model has been trained on large-scale datasets that contain a diverse range of faces. This extensive training enables the model to generalize well and recognize faces with higher accuracy, even in real-world scenarios with variations in lighting, pose, and expression.

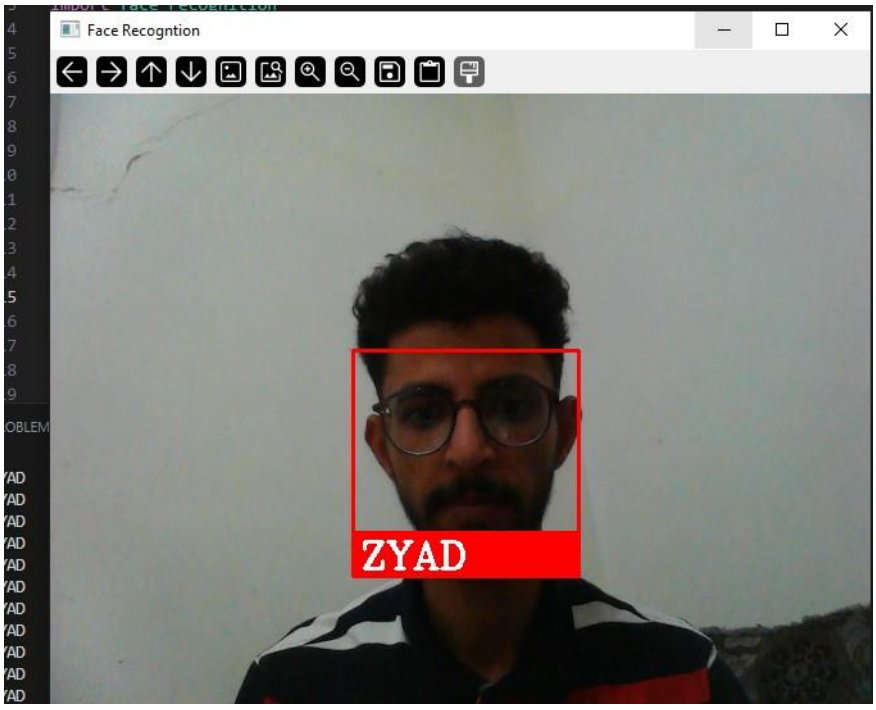


Figure 3-52 Training face recognition

- **Continual Improvements**: Dlib is an actively maintained library, and its developers continually strive to enhance its performance. Regular updates and improvements ensure that the library stays up-to-date with the latest advancements in face recognition algorithms and techniques, further enhancing its accuracy.

It's important to note that while dlib is accurate, achieving optimal performance in face recognition systems often requires fine-tuning and customization based on specific use cases and datasets. Proper preprocessing, training, and parameter optimization can contribute to further improving the accuracy of face recognition systems built with dlib.

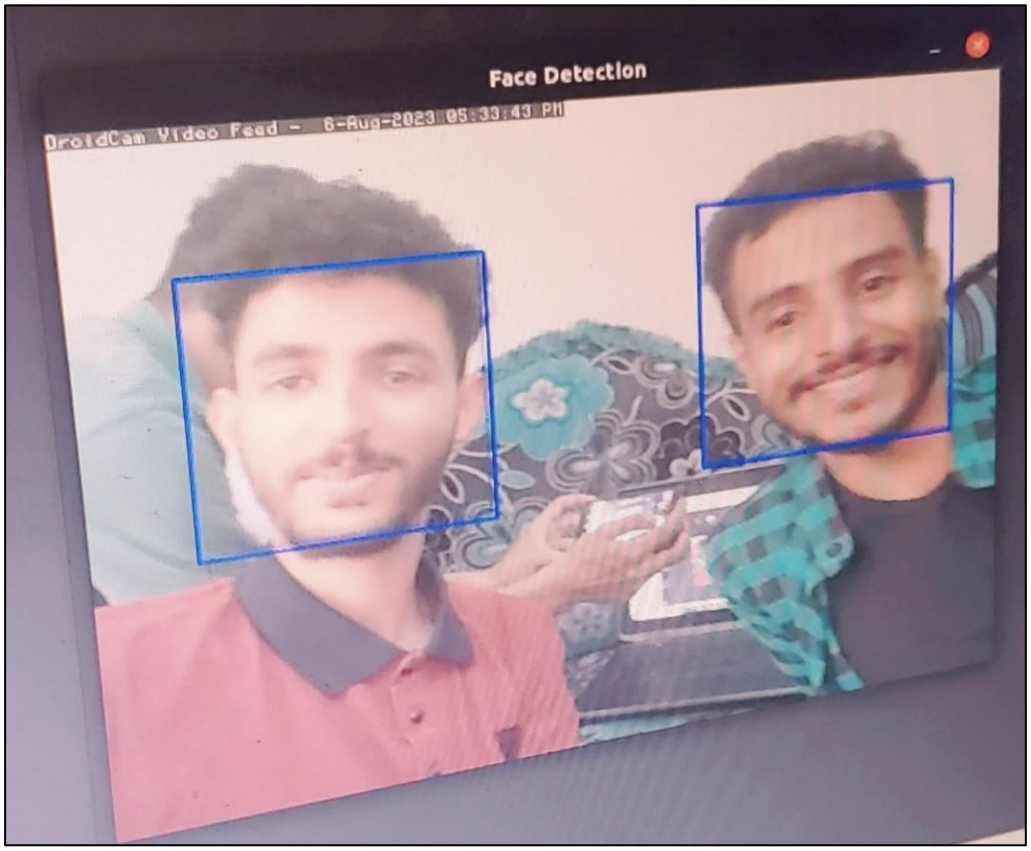


Figure 3-53 Training face detection

c. ***Advantages of Face Recognition:***

Face recognition offers several advantages over traditional identification methods. It provides a non-intrusive and convenient way to verify individuals' identities. It can enhance security measures by ensuring only authorized individuals have access to restricted areas. Additionally, face recognition systems can be seamlessly integrated into existing infrastructure, making it a scalable solution.

d. ***Challenges in Face Recognition***

Despite its advantages, face recognition also faces certain challenges. Variations in lighting, pose, and facial expressions can affect the accuracy of recognition algorithms. Additionally, the presence of occlusions, such as glasses or hats, can further complicate the recognition process. Ensuring robustness and accuracy in real-world scenarios remains an ongoing challenge.

e. ***Installing dlib Library in Ros:***

To implement face recognition with dlib, the first step is to install the library. Detailed installation instructions can be found on the official dlib website. It is recommended to set up a virtual environment to manage dependencies and ensure a smooth installation process.

f. ***Preprocessing Images for Face Recognition***

Before feeding images into a face recognition model, preprocessing steps are necessary. These steps may involve resizing the images, normalizing pixel values, and aligning faces to a standardized position. Proper preprocessing ensures consistent and accurate recognition results.

3.3.1.4. *Age and Gender Detection:*

Overview of age and gender detection and its importance across various industries. Discuss the potential applications and benefits of accurately determining age and gender from visual data.

a. ***The Role of Deep Learning***

Deep learning's pivotal role in age and gender detection. Explanation of how deep neural networks have revolutionized computer vision tasks by automatically extracting complex features from images.

b. ***Implementing Age and Gender Detection***

Step-by-step guide on implementing age and gender detection using the gande_net.caffemodel in ROS. Instructions on the required software, libraries, and code snippets to demonstrate the process.

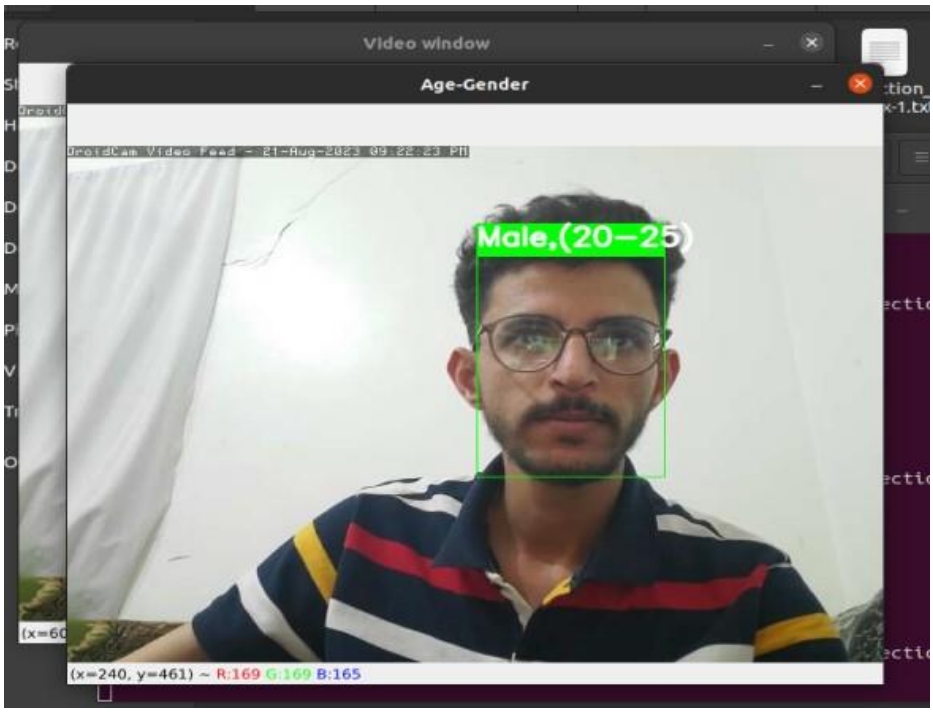


Figure 3-54 Training age and gander

c. ***Performance and Accuracy***

Evaluation techniques and performance metrics for assessing the accuracy of age and gender detection models. Discussion of the gander_net.caffemodel's strengths and limitations in terms of accuracy and computational efficiency.

d. ***Limitations and Challenges***

Addressing the limitations and challenges faced in age and gender detection using computer vision. Discussion of factors such as lighting conditions, pose variations, and expression that can affect prediction accuracy.

e. ***Future Developments***

Insights into future developments and advancements in age and gender detection. Discussion of emerging techniques, including multimodal analysis and fusion, that can enhance accuracy and robustness.

3.3.1.5. Image Acquisition Node

This node is responsible for capturing images from the robot's sensors, such as cameras or depth sensors. It subscribes to the appropriate ROS topics or services to receive image data. The acquired images are then passed to the object detection node for further processing.

3.3.1.6. Object Detection Node

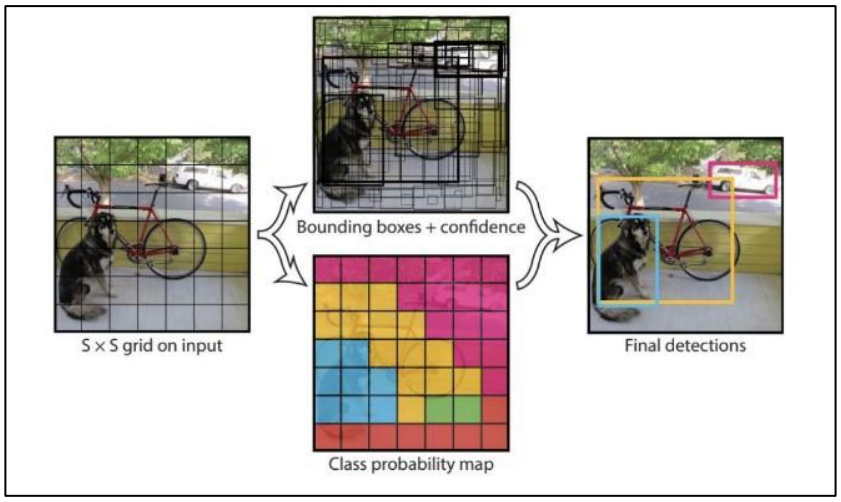


Figure 3-55 Object Detection example

This node incorporates the YOLOv8n model to perform object detection on the acquired images. It subscribes to the image topic published by the image acquisition node. The node applies the YOLOv8n model to the incoming images, detecting and classifying objects present in the scene. The detected objects are typically represented as bounding boxes with associated class labels.

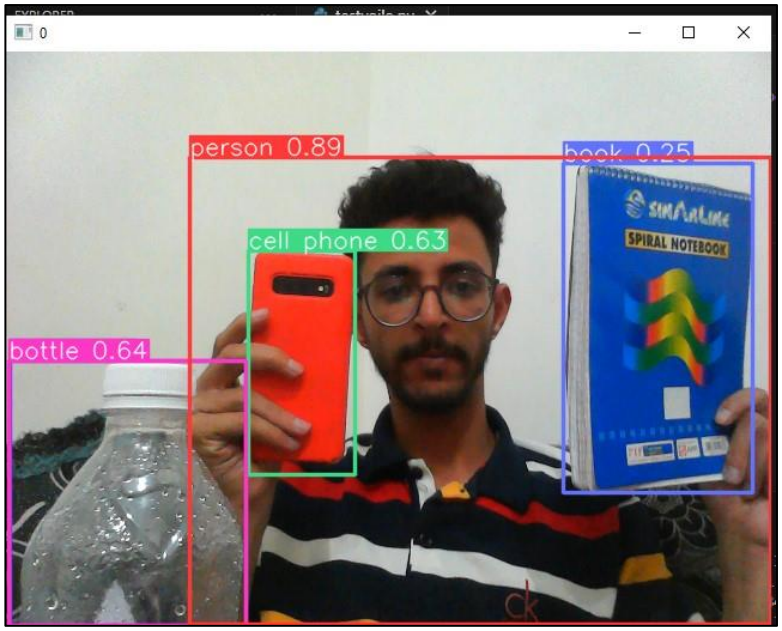


Figure 3-56 Training objects

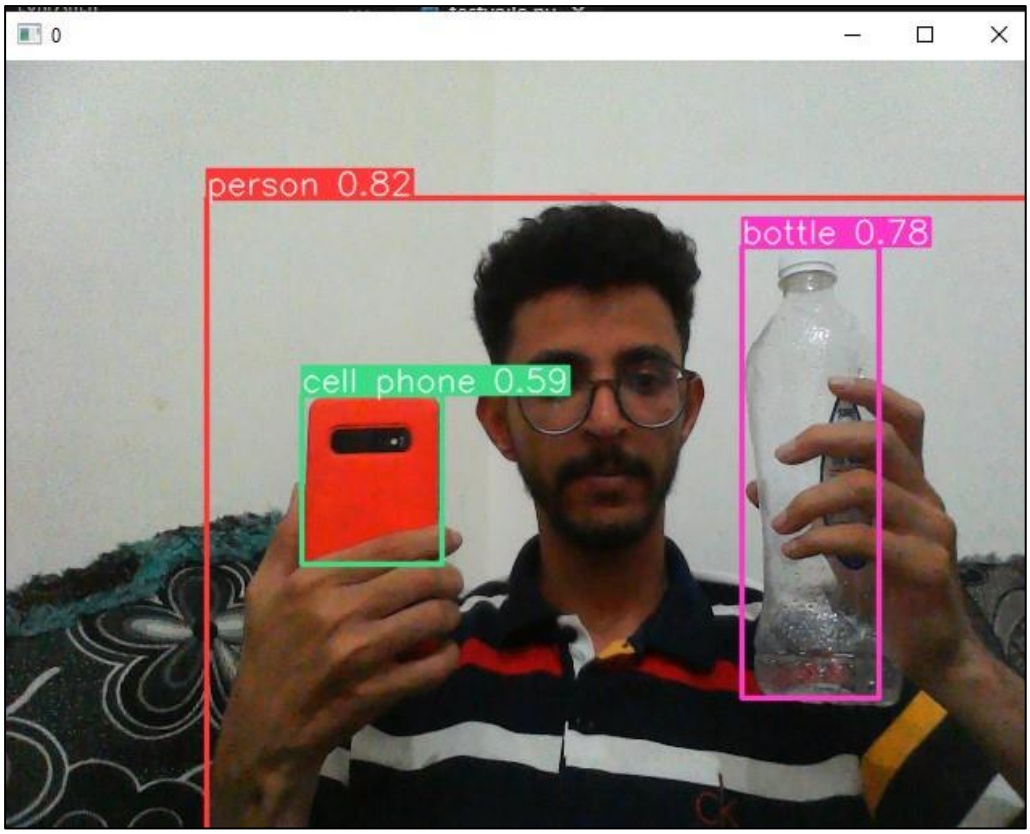


Figure 3-57 Training objects 2

- Object recognition in ROS using YOLOv8n

Object recognition in ROS (Robot Operating System) using YOLOv8n involves the integration of YOLOv8n's object detection capabilities into the ROS framework. This integration enables robots to perceive and interact with their environment by detecting and classifying objects in real-time. In ROS, object recognition using YOLOv8n typically involves the creation and coordination of multiple nodes to perform various tasks, such as image acquisition, object detection, and visualization.

3.3.1.7. Object Visualization Node

This node is responsible for visualizing the detected objects on the robot's display or user interface. It subscribes to the object detection results published by the object detection node. The node overlays bounding boxes and class labels onto the original images or visualizes them separately. The visualized objects can provide real-time feedback to the user or assist in subsequent robot actions.

3.3.1.8. Communication and Coordination

ROS provides a communication infrastructure that enables the coordination between nodes. Nodes can communicate through topics, services, or actions, passing messages to exchange data and trigger actions. The image acquisition

node publishes images to a specific topic, which the object detection node subscribes to. The object detection node publishes the detected objects to a topic, which the visualization and interaction nodes subscribe to.

3.3.1.9. Configuration and Parameter Tuning

The YOLOv8n model and ROS nodes may require configuration and parameter tuning to optimize their performance. Parameters such as image resolution, detection thresholds, and model weights can be adjusted to achieve desired results. ROS provides tools and libraries to configure and manage parameters in a modular and flexible manner.

By integrating YOLOv8n with ROS and coordinating the functionalities of different nodes, robots can effectively recognize and interact with objects in their environment. This integration enables a wide range of applications, including robotic perception, manipulation, navigation, and human-robot interaction. The principle behind object recognition in ROS using YOLOv8n is based on deep learning and computer vision techniques. YOLOv8n (You Only Look Once version 8n) is a state-of-the-art object detection model that utilizes a single neural network to perform real-time object detection and classification. The principle can be summarized as follows:

a. ***Convolutional Neural Network (CNN) Architecture***

YOLOv8n employs a deep CNN architecture, which consists of multiple convolutional layers, followed by fully connected layers. These layers are responsible for extracting features from input images and learning discriminative representations for different objects.

b. ***Training on Labeled Datasets***

YOLOv8n is trained using large annotated datasets, where images are labeled with bounding boxes and corresponding class labels for objects of interest. During the training process, the model learns to recognize patterns and features in the images that correspond to different object classes.

c. ***Object Detection***

The YOLOv8n model divides input images into a grid and predicts bounding boxes, class probabilities, and confidence scores for each grid cell. This approach allows YOLOv8n to detect multiple objects within a single pass of the neural network, resulting in real-time performance.

d. ***Non-Maximum Suppression***

To eliminate redundant and overlapping bounding box predictions, YOLOv8n applies a non-maximum suppression algorithm. This algorithm removes duplicate detections and selects the bounding box with the highest confidence score for each object.

e. ***Integration with ROS***

Integrating YOLOv8n with ROS involves creating specific ROS nodes to handle image acquisition, object detection, visualization, and interaction. These nodes communicate with each other through ROS topics, enabling the flow of data and coordination of tasks.

f. ***Real-Time Object Recognition***

By combining the power of YOLOv8n's object detection capabilities with the flexibility and modularity of ROS, robots can recognize objects in real-time. This recognition allows robots to perceive and understand their environment, enabling them to make informed decisions, perform tasks, and interact with objects.

The principle behind object recognition in ROS using YOLOv8n lies in leveraging deep learning techniques to detect and classify objects, while utilizing the ROS framework for communication, coordination, and integration within a robotic system. This combination enables robots to have a comprehensive understanding of their surroundings and enhances their capabilities in various applications.

g. ***R-CNN***

To circumvent the problem of selecting a huge number of regions, Ross Girshick et al. proposed a method where we use the selective search for extract just 2000 regions from the image and he called them region proposals. Therefore, instead of trying to classify the huge number of regions, you can just work with 2000 regions. These 2000 region proposals are generated by using the selective search algorithm which is written below

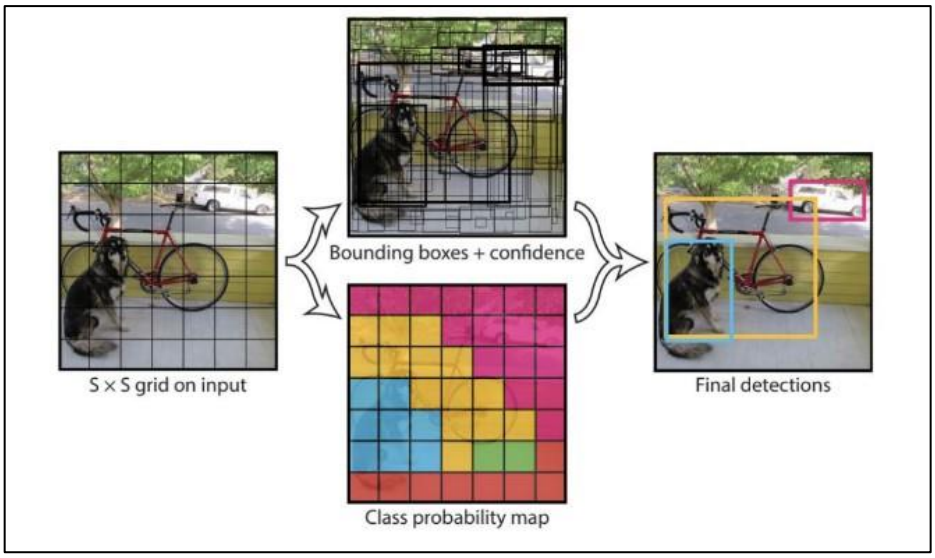


Figure 3-58 Selective search algorithm example

h. ***Selective Search***

- Generate the initial sub-segmentation, we generate many candidate regions
- Use the greedy algorithm to recursively combine similar regions into larger ones
- Use generated regions to produce the final candidate region proposals

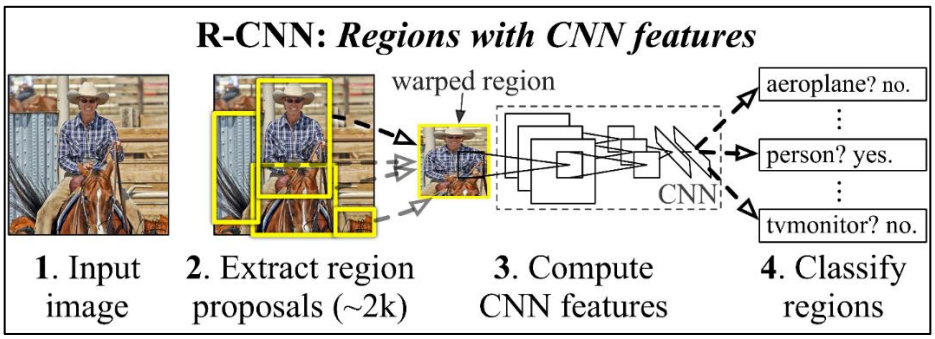
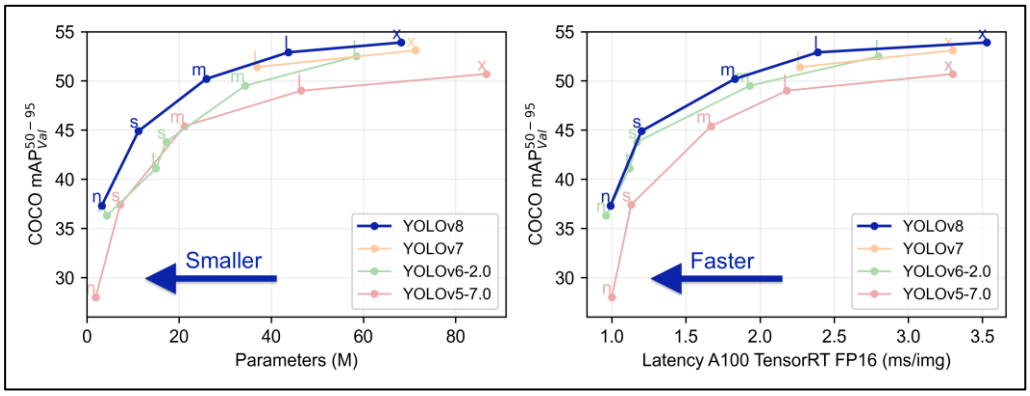


Figure 3-59 R-CNN

Ultralytics YOLOv8 is a cutting-edge, state-of-the-art (SOTA) model that builds upon the success of previous YOLO versions and introduces new features and improvements to further boost performance and flexibility. YOLOv8 is designed to be fast, accurate, and easy to use, making it an excellent choice for a wide

range of object detection and tracking, instance segmentation, image classification and pose estimation tasks.



3.3.2 Cognitive and Interaction Modules

Human-robot interaction is facilitated through cognitive modules that enable the robot to understand and respond to human commands and gestures. These modules incorporate (NLP) techniques for speech recognition and understanding. Dialogue management systems allow the robot to engage in meaningful conversations with users.

ROS nodes dedicated to cognitive functions process audio inputs, transcribe speech, and extract semantic meaning. NLP algorithms decode user commands and generate appropriate responses. The dialogue manager maintains context and generates coherent interactions, enhancing the robot's ability to comprehend and generate human-like responses.

3.3.2.1. Speech Recognition Node

This node is responsible for recognizing and converting spoken language into text. It listens to audio input from a microphone or audio source, performs speech recognition using a speech recognition API or library, and publishes the recognized text as a ROS message.

We have implemented this node to use a speech recognition service to process the audio input. The node receives audio data, sends it to the service, and receives the recognized text

- **Compare between some of the popular speech recognition libraries and services based on various factors**

Table 3-3 speech recognition libraries comparison

No.	Librariy Name	Pros	Cons
1.	**Google Cloud Speech-to-Text**	High accuracy, support for multiple languages, real-time recognition, robust API documentation, integration with other Google Cloud services.	May have associated costs depending on usage.
2.	**Microsoft Azure Speech Services**	Accurate, supports various languages and audio formats, offers customizable models, good documentation, integration with other Microsoft Azure services.	Like Google Cloud, there might be cost considerations.
3.	**IBM Watson Speech to Text**	Decent accuracy, supports multiple languages, customizable, offers various deployment options, integration with other IBM Watson services.	Can also have associated costs.
4.	**CMU Sphinx (PocketSphinx)**	Can work well on resource-constrained devices, open-source, offline capability, suitable for specific use cases.	Generally less accurate compared to cloud-based solutions, may require more effort to set up and configure.
5.	**Kaldi**	Highly customizable, used extensively in research, can achieve state-of-the-art accuracy with proper tuning, open-source.	Steeper learning curve, may require significant effort to set up and optimize, not as user-friendly as cloud-based APIs.
6.	**Mozilla DeepSpeech**	Open-source, deep learning-based, suitable for various applications, continually improving, supports some languages.	May not be as accurate as commercial solutions, limited language support compared to larger cloud services.

7.	**Wit.ai**	Part of Facebook's suite of tools, supports speech recognition along with other NLP tasks, relatively easy integration, customizable.	Limited to the capabilities offered by Wit.ai, may not have the same accuracy as specialized ASR systems.
8.	**SpeechRecognition (Python library)**	Easy to use, supports multiple engines including Google Web Speech API and CMU Sphinx, suitable for simple applications and prototyping.	Limited customization, accuracy might not match specialized ASR systems.

a. ***SpeechRecognition (Python library)***

- Ease of Use

The SpeechRecognition library is easy to use and provides a straightforward API for integrating speech recognition into your Python applications.

- Supported Engines

It supports various speech recognition engines, including Google Web Speech API (which is part of Google Cloud Speech-to-Text), CMU Sphinx, and more.

- Customization

While it provides some level of customization, the extent of customization might be more limited compared to more specialized services.

- Offline Capability

Depending on the engine you use, it might support offline speech recognition (e.g., CMU Sphinx), which can be beneficial in scenarios with limited or no internet connectivity.

- Accuracy

The accuracy can vary based on the engine used. Google Web Speech API, which is integrated via the library, is known for good accuracy.

- Language Support

It depends on the engine you choose. Google Web Speech API, for example, supports a wide range of languages, including Arabic.

- Cost

The library itself is open-source and free to use. However, if you use a cloud-based engine like Google Web Speech API, there might be associated costs based on your usage.

b. ***Google Cloud Speech-to-Text***

- **Accuracy**: Google Cloud Speech-to-Text is known for its high accuracy due to advanced machine learning models used by Google.
- **Language Support**: It supports a wide range of languages, including Arabic, with robust models and recognition accuracy.
- **Real-time Recognition**: Google Cloud Speech-to-Text offers real-time recognition capabilities, making it suitable for applications that require immediate responses.
- **Integration**: It's easily integrated into applications using the Google Cloud SDK and APIs.
- **Customization**: Google Cloud Speech-to-Text provides some customization options, allowing you to fine-tune models to your specific needs.
- **Cost**: While there is a free tier, usage beyond that comes with costs based on the number of requests and audio duration. Cost considerations might be important if you have high-volume usage.

c. ***Offline Arabic Speech Recognition Libraries***

Table 3-4 Offline Arabic Speech Recognition Libraries

No.	Librariy Name	Pros	Cons
1.	**PocketSphinx (CMU Sphinx)**	Open-source, offline capability, supports multiple languages including Arabic, can work on resource-constrained devices, suitable for specific use cases.	Might have lower accuracy compared to cloud-based solutions, requires appropriate language model files, might require some configuration.
2.	**Julius**	Open-source, supports Arabic and other languages, offers large vocabulary continuous speech recognition (LVCSR), adaptable for various applications.	Might require more expertise to set up and optimize, may not provide the same level of accuracy as cloud solutions.
3.	**Kaldi**	Highly customizable, can be adapted for Arabic speech recognition, powerful toolkit for speech processing, research-oriented.	More complex to set up, might require substantial effort for customization and tuning, suitable for those with technical expertise.
4.	**CMU Sphinx (PocketSphinx)**	Can work well on resource-constrained devices, open-	Generally less accurate compared to cloud-based

		source, offline capability, suitable for specific use cases.	solutions, may require more effort to set up and configure.
5.	**Kaldi**	Highly customizable, used extensively in research, can achieve state-of-the-art accuracy with proper tuning, open-source.	Steeper learning curve, may require significant effort to set up and optimize, not as user-friendly as cloud-based APIs.

- **Considerations for Offline Arabic Speech Recognition:**

  - Accuracy

While offline solutions have their advantages, cloud-based services tend to offer higher accuracy due to their access to large datasets and machine learning advancements.

  - Language Models

Offline speech recognition relies heavily on language models. These models need to be carefully trained or obtained for accurate recognition of Arabic speech.

  - Resource Requirements

Offline solutions like PocketSphinx can work on resource-constrained devices, making them suitable for scenarios with limited processing power or connectivity.

  - Customization

Some offline libraries, like Kaldi, offer extensive customization options, allowing you to tailor the system to your specific needs.

  - Development Effort

Setting up and configuring offline solutions can require more effort compared to cloud-based APIs, especially in terms of obtaining or generating appropriate language models.

  - Use Cases

Offline solutions are particularly valuable when internet connectivity is unreliable or unavailable. They can be useful for applications in remote areas or embedded systems.

  - Technical Expertise

Utilizing offline libraries might require more technical knowledge compared to using cloud-based APIs, which are often designed for ease of use.

  - Updates

Ensure that the library you choose is actively maintained and can accommodate any updates or changes in the future.

3.3.2.2. GPT Node

This node utilizes the GPT (Generative Pre-trained Transformer) model to generate human-like text responses based on user input. It takes the recognized text from the speech recognition node as input and uses the GPT model to

produce a response. The GPT model has been trained on a vast amount of text data and is capable of generating coherent and contextually relevant text.

After generating the response, the GPT node publishes the text response as a ROS message. This text can then be converted into speech using text-to-speech (TTS) libraries or services to make the robot's response audible to the user.

Models for generating human-like text responses

Table 3-5 Models for generating human-like text responses

No.	Librariy Name	Pros	Cons
1.	**GPT-3**	Can generate coherent and contextually relevant text across a wide range of topics, highly creative, user-friendly interface, handles prompts well, large model size.	Expensive to use, might produce incorrect or nonsensical responses, potential bias in generated content.
2.	**BERT**	Strong understanding of bidirectional context, good for tasks requiring context comprehension, fine-tuning capabilities, effective for sentiment analysis, question answering.	Not inherently designed for text generation, can be computationally expensive.
3.	**T5**	Versatile due to text-to-text framework, good performance on various tasks, including summarization, translation, and more.	Large model size, resource-intensive.
4.	**XLNet**	Captures bidirectional context, maintains coherence, strong performance across different tasks, offers flexibility.	Might be computationally demanding, potential complexity in implementation.
5.	**CTRL**	Allows controlled text generation, suitable for style-specific content, creativity in generating text.	Limited use cases compared to more general models.
6.	**DALL-E and CLIP**	Combines vision and language, generates images and text, innovative applications.	Primarily focused on images, might not be

			suitable for pure text generation tasks.
7.	**UniLM**	Handles various NLP tasks, supports both autoregressive and bidirectional approaches.	Might not excel in specific tasks as much as specialized models.
8.	**Pointer-Generator Networks**	Effective for abstractive text summarization, combines extraction and generation.	Specific to summarization tasks, might not generalize well to other text generation tasks.

3.3.2.3. Classification Node

This node is dedicated to classifying user input into different categories or classes. It helps the robot understand the intent or type of user input, such as commands, greetings, questions, etc. To perform this classification, the node likely employs machine learning techniques, such as using a BERT model to classify the input based on pre-defined categories.

The classification node processes the recognized speech from the speech recognition node and uses the classification model to determine the category of the input. It then publishes the classification result as a ROS message, which can be used by other nodes to determine appropriate actions.

3.3.2.4. Voice Node

The voice node is responsible for handling the audio output of the robot's responses. It receives the text responses generated by the GPT node or other nodes and converts them into audible speech. This conversion can be achieved using text-to-speech (TTS) libraries or services.

Once the response text is converted to speech, the voice node plays the speech through the robot's speakers or audio output device. This allows the robot to communicate its responses to the user in a human-like manner.

some popular TTS libraries and services that were well-regarded:

a. ***Google Text-to-Speech API***

Google's TTS API offers high-quality and natural-sounding voices. It's user-friendly and provides a simple way to integrate TTS functionality into applications.

Table 3-6 Properties of Google Text-to-Speech API

Strengths	Offers high-quality and natural-sounding voices, easy integration with Google Cloud services, supports multiple languages, and provides a straightforward API.
Considerations	Might have limitations in terms of customization compared to some other options.
Arabic Support	Google's TTS API offers support for Arabic language, allowing you to synthesize text into Arabic speech.
Quality	Google's TTS API generally provides good quality Arabic speech synthesis with natural-sounding voices.

b. ***Amazon Polly***

Amazon Polly is a cloud-based TTS service offered by Amazon Web Services (AWS). It supports multiple languages and provides a wide range of customizable voices.

Table 3-7 Properties of Amazon Polly

Strengths	Offers a wide variety of voices and languages, supports SSML for detailed customization, allows real-time synthesis and batch processing, and integrates well with other AWS services.
Considerations	Pricing can vary depending on usage, and more advanced customization might require some technical knowledge.
Arabic Support	Amazon Polly supports Arabic language and offers multiple Arabic voices.
Quality	Amazon Polly's Arabic voices can provide decent quality, though voice quality can vary depending on the selected voice.

c. ***Microsoft Azure Cognitive Services Text-to-Speech***

Microsoft's Azure Cognitive Services includes a TTS API that offers diverse voices and customizable speech synthesis capabilities.

Table 3-8 Properties of Microsoft Azure Cognitive Services

Strengths	Provides various voices and languages, supports SSML for customization, can be easily integrated into Microsoft Azure applications, and offers user-friendly API documentation.
Considerations	Pricing can be a consideration, especially for larger usage.
Arabic Support	Microsoft's TTS service supports Arabic language and provides Arabic voices.
Quality	The quality of Azure TTS's Arabic voices can be considered acceptable.

d. ***IBM Watson Text-to-Speech***

IBM's Watson TTS service provides a variety of voices and allows customization through SSML (Speech Synthesis Markup Language).

Table 3-9 Properties of IBM Watson Text-to-Speech

Strengths	Offers a range of voices, supports SSML for customization, integrates with other IBM Watson services, and provides flexibility in terms of deployment options.
Considerations	Pricing can be a factor, and the voice selection might be more limited compared to other services.
Arabic Support	IBM Watson TTS supports Arabic language and offers Arabic voices.
Quality	The quality of Arabic voices might vary.

e. ***Open-source options:***

eSpeak: A compact and open-source TTS engine that supports a variety of languages and voices. Lightweight and efficient, suitable for embedded systems or resource-constrained environments. However, voice quality might not be as advanced as cloud-based services. Supports Arabic language, but the quality

might not be as high as commercial solutions. Voice quality might be more robotic.

Festival: An open-source TTS system that offers customization options but might require more setup. Offers customization and flexibility due to its open-source nature, but might require more effort to set up and integrate. With proper configuration and voice training, Festival can be used for Arabic TTS, but achieving good quality might require more effort.

f. ***Mozilla TTS (Tacotron 2)***

An open-source TTS engine developed by Mozilla, based on the Tacotron 2 architecture. It allows users to train their own TTS models for specific voices.

Table 3-10 Properties of Mozilla TTS (Tacotron 2)

Strengths	Provides an open-source solution with the potential to create custom voices, allowing for more control over the generated speech quality.
Considerations	Requires more technical expertise to set up, train models, and integrate into applications.
Arabic Support	Mozilla TTS can be trained to support Arabic language, but it might require training a custom model specifically for Arabic speech synthesis.
Quality	The quality of Arabic speech generated by Mozilla TTS would depend on the quality of the trained model.

g. ***Natural Language Processing (NLP)***

It is a field of AI that focuses on the interaction between computers and human language. It encompasses the development of algorithms and models that enable computers to understand, interpret, generate, and manipulate human language in a way that is both meaningful and valuable. NLP plays a crucial role in bridging the gap between human communication and computer processing

NLP involves a wide range of tasks and techniques, which can be categorized into several types:

- Tokenization

Tokenization is the process of breaking down a text into individual tokens, such as words or subwords. It's a fundamental step in NLP as it helps the computer understand the structure of the text.

- Part-of-Speech Tagging

This involves tagging each word in a text with its part of speech (noun, verb, adjective, etc.). It's important for understanding the grammatical structure and meaning of a sentence.

- Named Entity Recognition (NER)

NER is the identification of named entities like names of people, organizations, dates, and locations in a text. It's crucial for extracting structured information from unstructured text.

- Sentiment Analysis

Sentiment analysis determines the sentiment or emotion expressed in a piece of text. It's used to understand public opinion, customer feedback, and social media sentiments.

- Language Modeling

Language models learn the probability distribution over sequences of words in a language. They are the basis for various NLP tasks including machine translation, text generation, and speech recognition.

- Machine Translation

NLP is used to automatically translate text from one language to another. Neural machine translation models have significantly improved translation quality.

- Text Generation

NLP models can generate coherent and contextually relevant text, which has applications in chatbots, creative writing, and content creation.

- Question Answering

NLP models can be trained to understand questions and provide accurate answers. This is used in search engines and virtual assistants.

- Topic Modeling

This involves identifying topics within a collection of text documents. It's useful for organizing and summarizing large amounts of textual data.

- Speech Recognition

NLP is used to convert spoken language into written text, making voice-based interactions with computers possible.

- Text Summarization

NLP techniques are used to automatically generate concise summaries of long documents, which is useful for quickly extracting important information.

- Coreference Resolution

This involves determining when different words in a text refer to the same entity, allowing for better understanding of context.

- Dependency Parsing

This is the analysis of grammatical structure by determining the relationships between words in a sentence.

- Conversational AI

NLP models are used to create chatbots and virtual assistants that can understand and respond to human conversations.

Natural Language Processing (NLP) aspects used in each of the nodes of voice chat system.

- **Classification Node**

In this node, NLP is used for text classification. The input user text is processed using a transformer-based model (BERT) for classification into different categories. Here's how NLP is utilized:

- **Tokenization:** The input text is tokenized into subwords using the BERT tokenizer. This process converts text into a format suitable for the model's input.
- **Model Inference:** The tokenized input is then passed through a pretrained BERT model. The model outputs a representation of the input text.
- **Prediction:** The output representation is used to predict the category of the input text. This is achieved by applying a classification layer on top of the BERT model.

The BERT model is employed as a feature extractor, capturing contextual information from the input text, which helps in better classification.

- **GPT Node**

This node focuses on generating human-like responses using the GPT-3 model, powered by OpenAI. NLP plays a significant role in response generation:

- **Prompt Crafting:** The user's input and the classification result are combined into a prompt for the GPT-3 model. The prompt sets the context for generating a relevant response.

- **Model Generation:** The GPT-3 model uses the provided prompt to generate a continuation, effectively creating a response to the user's input.
- **Text Generation:** The model generates text by predicting the next words based on the context provided. It employs a language model that has been trained on a vast amount of text data.

NLP is instrumental in ensuring that the responses generated are coherent, contextually relevant, and linguistically accurate.

- **Speech Recognition Node**

NLP is used in this node to convert spoken language (audio) into written text. Here's how it works:

- **Speech Recognition:** The node uses the *speech_recognition* library to capture audio from a microphone and recognizes the spoken words in the form of text.
- **Google API:** The Google Speech Recognition API is employed to convert audio data into text. This API uses NLP techniques to process the audio and transcribe it into text.
- **Audio Preprocessing:** Behind the scenes, the library performs NLP operations on the audio data, such as identifying phonemes, words, and sentences, and then converting them into a coherent textual form.

In summary, NLP is used throughout voice chat system to process, classify, generate, and understand both user input and system responses. It enables the system to understand spoken or typed language, classify it, generate contextually relevant responses, and facilitate seamless communication with users.

3.4 Human-Robot Interaction Features

ROS plays a pivotal role in enabling seamless human-robot interaction. Speech recognition modules transcribe spoken words into text, which are then processed by natural language understanding nodes. Dialogue management systems utilize context to generate meaningful responses, allowing the robot to engage in coherent conversations.

Furthermore, the sign language translation module capitalizes on ROS's communication framework to interpret hand gestures. Computer vision algorithms analyze hand movements, recognizing sign language symbols and converting them into text or speech. This integration empowers the robot to comprehend and respond effectively to sign language inputs.

3.5 Integration of Perception and Action

The essence of the humanoid robot's capabilities lies in the integration of perception and action. ROS facilitates the seamless fusion of sensory information with motor commands, enabling the robot to perform tasks based on environmental cues. Perception nodes feed data into the control architecture.

For instance, facial recognition data influences the robot's behavior, allowing it to identify and interact with specific individuals. The integration of perception and action is a testament to ROS's role in orchestrating complex, real-time robotic tasks.

3.6 Control and Actuation

Control algorithms drive the motion of the humanoid robot, ensuring coordinated and dynamic movement. ROS's control architecture provides a framework for joint-level and task-level control, manipulation, and interaction. Inverse kinematics and dynamic modeling techniques are employed to calculate joint angles and torques for achieving desired movements.

ROS-based control nodes orchestrate the synchronization of joint movements, enabling smooth and coordinated actions. Feedback loops are established to ensure stability and robustness in manipulation tasks. Actuators, driven by control signals, generate forces and torques necessary for executing complex movements, such as turning, and grasping.

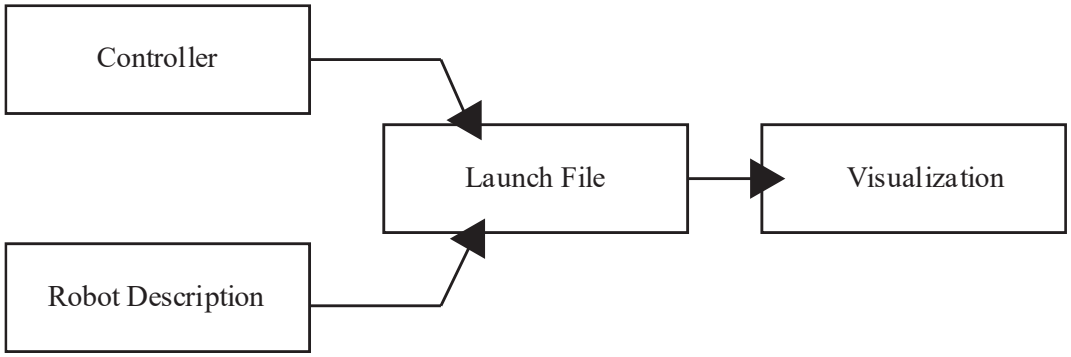


Figure 3-60 Virtual Robot Flow

- **Launching all nodes related to real time simulation**

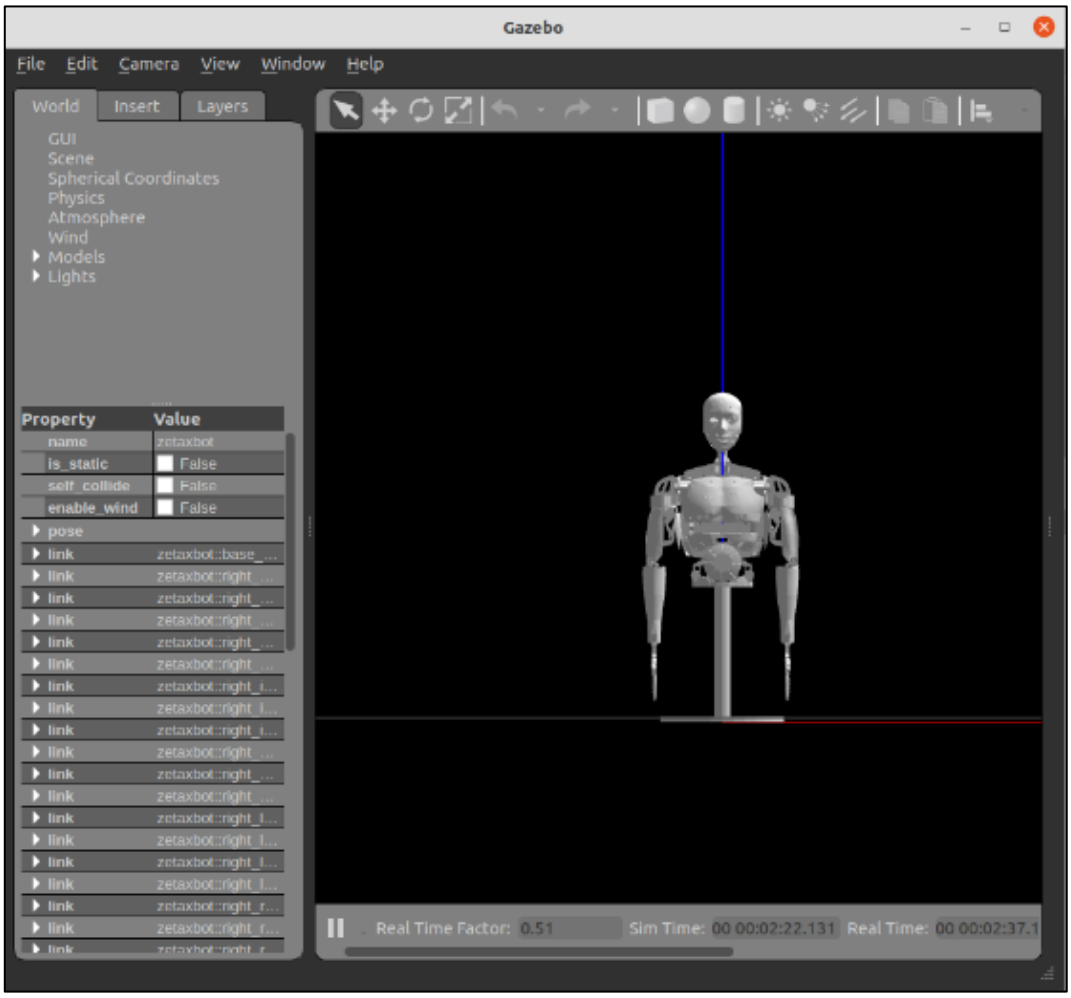


Figure 3-61 Launched node one (Gazebo)

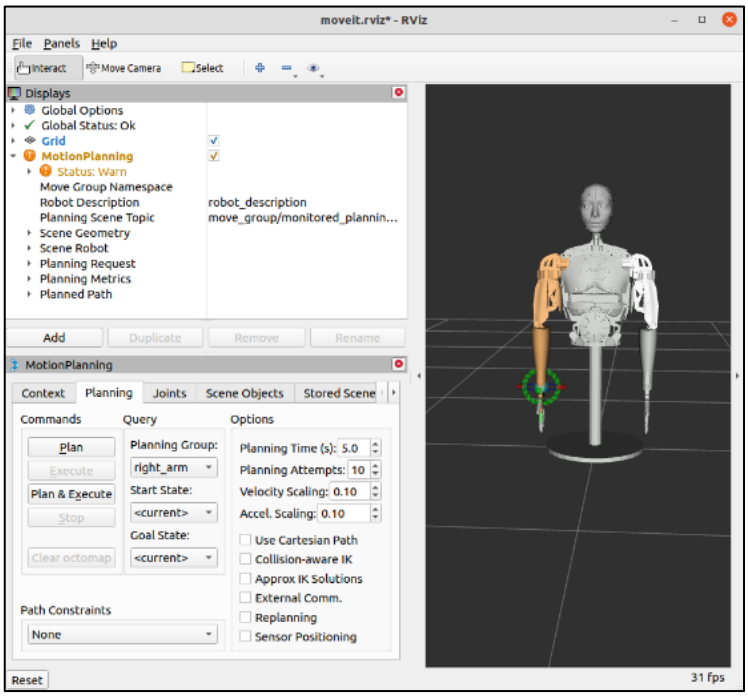


Figure 3-62 Launched node two (Rviz)

- **Planning and excuting the movments**

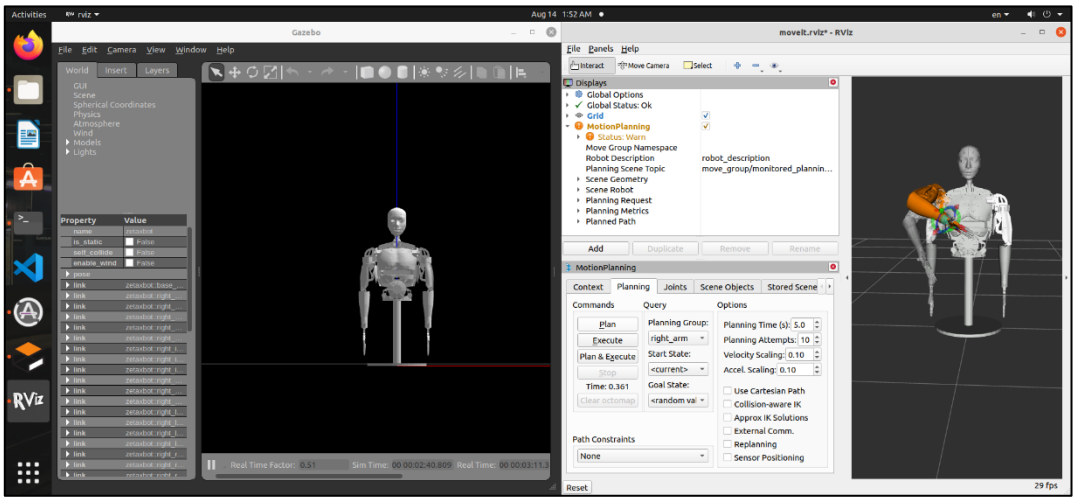


Figure 3-63 Planning the right arm

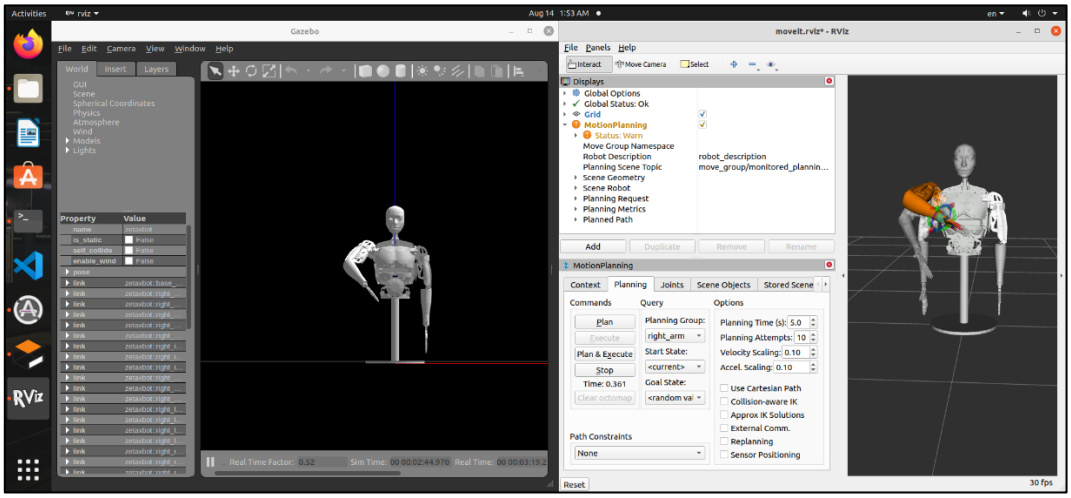


Figure 3-64 Executing the planned movement on Gazebo

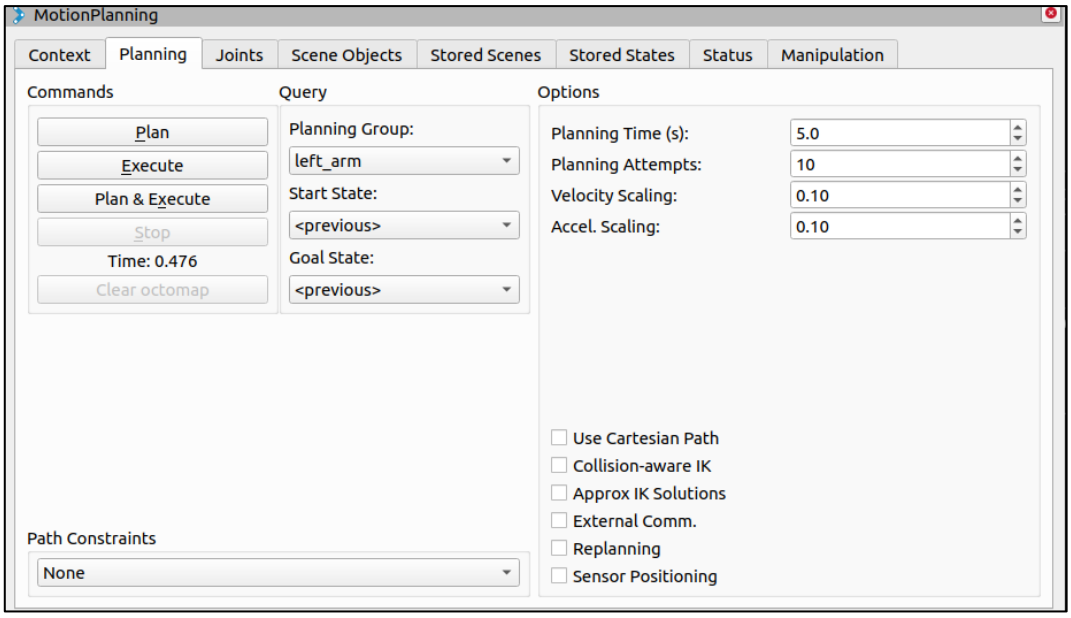


Figure 3-65 Motion planning panel on Rviz

The methodology employed in this project encapsulates a holistic approach, integrating mechanical design, software architecture, sensor integration, control algorithms, and ethical considerations. By leveraging the modularity and adaptability of ROS, the development process embraces a systematic and iterative strategy, ultimately leading to the creation of a humanoid robot that embodies advanced perceptual, cognitive, and locomotion capabilities.

Chapter 4
Results and Discussion

Chapter 4 Results and Discussion

4.1 Perception and Interaction Achievements

The implemented perception modules exhibit commendable performance in object detection and facial recognition. Deep learning algorithms accurately identify and classify objects, contributing to the robot's environmental awareness and interaction capabilities. Facial recognition enables the robot to recognize individuals and adapt its behavior based on pre-defined profiles, enhancing personalized interactions.

Sign language translation capabilities are a notable achievement, enabling the robot to interpret hand gestures and convert them into text or speech. The integration of computer vision algorithms with ROS facilitates real-time gesture analysis, bridging communication gaps between the robot and users proficient in sign language.

4.2 Human-Robot Interaction and User Experience

User studies have provided valuable insights into the effectiveness of human-robot interactions. Participants engaging in conversations with the robot report positive experiences, highlighting the system's responsiveness and naturalness of dialogue. Speech recognition accuracy and language understanding contribute to meaningful interactions, allowing users to issue commands and receive relevant responses.

Sign language translation garners significant attention, particularly from individuals proficient in sign language. Users express satisfaction with the robot's ability to comprehend and respond to sign language gestures, making interactions more inclusive and accessible. The integration of sign language translation with ROS has a tangible impact on promoting effective communication.

4.3 Challenges and Future Enhancements

While the project has achieved notable milestones, several challenges and areas for improvement have been identified. Fine-tuning perception algorithms for complex environments and enhancing object detection accuracy in varying lighting conditions remain ongoing endeavors. Locomotion algorithms could benefit from further optimization for energy efficiency and adaptive walking.

Future enhancements may involve the incorporation of advanced cognitive capabilities, such as emotion recognition and context-aware dialogue management. Expanding the robot's vocabulary and linguistic understanding could further enhance its conversational abilities. Additionally, exploring real-time learning and adaptation mechanisms could enable the robot to dynamically adjust its behavior based on user interactions.

Chapter 5 provides a comprehensive overview of the results obtained from rigorous testing and validation, highlighting the achievements, challenges, and future directions of the humanoid robot development project using ROS. The integration of advanced locomotion, perception, and interaction capabilities within the ROS ecosystem underscores the project's significance in advancing the frontiers of robotics and human-robot collaboration.

Chapter 5
Future Directions

Chapter 5 Future Directions

5.1 Advanced Cognitive Capabilities

The future development of the humanoid robot holds immense potential in enhancing its cognitive abilities. Integration of advanced machine learning techniques can enable the robot to recognize emotions, interpret contextual cues, and exhibit more human-like decision-making. This would elevate the robot's interactions to a level where it can respond empathetically and adapt its behavior based on user sentiment.

Expanding the robot's linguistic capabilities through continued natural language processing research could lead to more nuanced and contextually aware conversations. Real-time learning mechanisms can enable the robot to adapt its language understanding and dialogue responses dynamically, creating more fluid and intuitive interactions.

5.2 Multimodal Perception and Fusion

Enhancing the robot's perception capabilities by incorporating multiple modalities can unlock new realms of interaction and understanding. Integrating additional sensors, such as depth cameras and tactile sensors, can provide richer environmental data. Fusion of data from various sensors can lead to more robust object recognition, obstacle avoidance, and gesture analysis.

Multimodal perception can also enable the robot to interpret and respond to non-verbal cues more effectively, such as interpreting gestures, facial expressions, and body language. The seamless integration of these cues with ROS-driven control can create a more immersive and intuitive human-robot interaction.

5.3 Autonomy and Learning

The future of humanoid robotics lies in the realm of autonomy and learning. Empowering the robot to perform tasks autonomously, such as environment exploration or object manipulation, can expand its utility in real-world applications. Developing reinforcement learning algorithms within the ROS framework can enable the robot to acquire new skills through trial and error.

Long-term autonomy involves enabling the robot to learn from continuous interactions with its environment and users. Implementing memory mechanisms and experience replay systems can facilitate learning and adaptation over time. This would allow the robot to refine its capabilities, making it more adept at understanding user preferences and delivering personalized interactions.

5.4 Collaborative and Swarm Robotics

Extending the capabilities of the humanoid robot can involve exploring collaborative scenarios with other robots or entities. Integrating swarm robotics concepts within the ROS ecosystem can enable coordinated actions among multiple robots to accomplish complex tasks. Collaborative efforts can range from collaborative object manipulation to distributed environmental sensing.

Swarm intelligence algorithms can facilitate efficient task allocation and resource sharing among a group of robots, enhancing their collective capabilities. The interoperability of ROS can serve as a foundation for orchestrating communication and coordination among heterogeneous robotic agents.

5.5 Applications in Healthcare and Assistance

The future development of the humanoid robot has profound implications for healthcare and assistive applications. Expanding the robot's capabilities to include remote health monitoring, medication reminders, and assistance with activities of daily living can provide significant support to elderly individuals and those with disabilities.

ROS-enabled teleoperation and remote control mechanisms can enable healthcare professionals to utilize the robot as a surrogate in remote environments. Integrating medical sensors and diagnostic tools can transform the humanoid robot into a versatile medical assistant, bridging the gap between healthcare providers and patients.

Chapter 6
Conclusion

Chapter 6 Conclusion

In conclusion, Humanoid robots integrated with ROS (Robot Operating System) can greatly benefit from the fusion of computer vision, voice conversation, and natural language processing techniques. By combining these technologies, humanoid robots can perceive and understand the environment, interact with humans through speech, and comprehend and respond to spoken language.

Computer vision enables robots to recognize and track objects, detect faces, understand gestures, and navigate their surroundings. It enables robots to interpret visual information and make informed decisions based on the perceived environment.

Voice conversation capabilities allow robots to communicate with humans through speech. Natural language processing techniques enable them to understand and process spoken language, extract meaning from conversations, and generate appropriate responses. This facilitates more natural and intuitive interactions between humans and robots.

The integration of these technologies in humanoid robots with ROS provides several advantages:

1. Enhanced Perception: Computer vision allows robots to perceive the visual world, enabling them to recognize and interact with objects, people, and their surroundings.
2. Improved Interaction: Voice conversation capabilities enable robots to engage in meaningful conversations with humans, understand their commands, and respond accordingly. This enhances the user experience and promotes more seamless human-robot interaction.
3. Intelligent Decision-Making: With computer vision and voice conversation capabilities, humanoid robots can analyze visual and auditory information, process natural language, and make intelligent decisions based on the context. This enables them to adapt to dynamic environments and perform complex tasks.
4. Task Automation: By combining computer vision with voice conversation, humanoid robots can automate various tasks that require visual perception and verbal communication. This includes tasks such as object recognition, navigation, human-robot collaboration, and assistance.

ROS provides a flexible and modular framework for integrating these capabilities into humanoid robots. It allows developers to leverage existing computer vision libraries, natural language processing tools, and speech recognition systems, and seamlessly integrate them into the robot's software architecture.

Overall, the integration of computer vision, voice conversation, and natural language processing techniques in humanoid robots with ROS empowers them with advanced perception, communication, and decision-making abilities. This paves the way for more intelligent, interactive, and capable robots that can effectively collaborate with humans in various domains and applications.

Appendix .A
References

References

1. Koenig, N., & Howard, A. (2004). Design and use paradigms for gazebo, an open-source multi-robot simulator. In 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (Vol. 3, pp. 2149-2154). IEEE.
2. Bethune, James D., 1941. Engineering design and graphics with SolidWorks 2016 / James D. Bethune. First edition. | Boston : Pearson, [2017]. LCCN 2016017519| ISBN 9780134507699 | ISBN 013450769X
3. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., ... & Wheeler, R. (2009). ROS: an open-source Robot Operating System. ICRA workshop on open source software, 5.
4. T. Linner, A. Shrikathiresan, M. Vetrenko, B. Ellmann, T. Bock, "Modeling and operating robotic environments using Gazebo/ROS," Proc. 28th Int. Symp. Autom. Robot. Constr. ISARC 2011, pp. 957– 962, 2011, https://doi.org/10.22260/ISARC2011/0177.
5. K. Takaya, T. Asai, V. Kroumov, F. Smarandache, "Simulation environment for mobile robots testing using ROS and Gazebo," 2016 20th Int. Conf. Syst. Theory, Control Comput. ICSTCC 2016 - Jt. Conf. SINTES 20, SACCS 16, SIMSIS 20 - Proc., pp. 96–101, 2016, https://doi.org/10.1109/ICSTCC.2016.7790647.
6. A. Dobrokvashina, R. Lavrenov, E. A. Martinez-Garcia, Y. Bai, "Improving model of crawler robot Servosila "Engineer" for simulation in ROS/Gazebo," 2020 13th International Conference on Developments in eSystems Engineering (DeSE), pp. 212-217, 2020, https://doi.org/10.1109/DeSE51703.2020.9450233.
7. N. Koening, A. Howard, "Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator," J. Am. Med. Assoc., vol. 285, no. 3, pp. 290–292, 2001, https://doi.org/10.1001/jama.285.3.290.
8. Z. B. Rivera, M. C. De Simone, D. Guida, "Unmanned ground vehicle modelling in Gazebo/ROS-based environments," Machines, vol. 7, no. 2, pp. 1–21, 2019, https://doi.org/10.3390/machines7020042.
9. W. Qian et al., "Manipulation task simulation using ROS and Gazebo," 2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014), pp. 2594-2598, 2014, https://doi.org/10.1109/ROBIO.2014.7090732.
10. D. Chikurtev, "Mobile Robot Simulation and Navigation in ROS and Gazebo," 2020 International Conference Automatics and Informatics (ICAI), pp. 1-6, 2020, https://doi.org/10.1109/ICAI50593.2020.9311330.
11. T. Wright, A. West, M. Licata, N. Hawes, B. Lennox, "Simulating Ionising Radiation in Gazebo for Robotic Nuclear Inspection Challenges," Robotics, vol. 10, no. 3, p. 86, 2021, http://dx.doi.org/10.3390/robotics10030086.
12. M. Sánchez, J. Morales, J. L. Martínez, J. J. Fernández-Lozano, A. García-Cerezo, "Automatically Annotated Dataset of a Ground Mobile Robot in Natural Environments via Gazebo Simulations," Sensors, vol. 22, no. 15, p. 5599, 2022, https://doi.org/10.3390/s22155599.
13. K. K. P. Gayashani, U. U. S. Rajapaksha and C. Jayawardena, "Moving a Robot In Unknown Areas Without Collision Using Robot Operating System," 2022 2nd

International Conference on Advanced Research in Computing (ICARC), pp. 84-89, 2022, https://doi.org/10.1109/ICARC54489.2022.9754138.
14. https://bostondynamics.com/atlas/
15. https://github.com/code-iai/iai_kinect2
16. https://www.ybliu.com/2021/03/config-kinect-v2.html
17. https://asimo.honda.com/
18. https://twistedsifter.com/2013/11/the-writer-automaton-pierre-jaquet-droz-modern-computer-ancestor/
19. https://www.johnogroat-journal.co.uk/news/discovering-the-wonders-of-automatons-116096/
20. https://global.honda/innovation/robotics/ASIMO.html
21. https://en.cs.ustc.edu.cn/2016/0420/c23602a93710/page.html
22. https://economictimes.indiatimes.com/magazines/panache/worlds-first-robot-citizen-attends-conference-in-india-makers-reveal-sophia-can-draw-now/articleshow/71633205.cms
23. https://global.toyota/en/detail/19666346
24. https://www.facebook.com/agilityrobotics/photos
25. https://global.honda/innovation/robotics/robot-development-history.html
26. https://global.honda/innovation/robotics/ASIMO/history.html