PriMera Scientific Engineering
Volume 5 Issue 4 October 2024
DOI: 10.56831/PSEN-05-156
ISSN: 2834-2550

Design and Implementation of Humanoid Robot with Artificial Intelligence

Type: E-Book
Received: August 22, 2024
Published: September 30, 2024

Citation:
Israa S Al-Furati., et al. "Design and Implementation of Humanoid Robot with Artificial Intelligence". PriMera Scientific Engineering 5.4 (2024): 27-77.

Israa S Al-Furati* and Mahmoud K Hashem

Electrical Engineering Department, University of Basrah, Basrah, Iraq

***Corresponding Author:** Israa S Al-Furati, Assistant Professor/University of Basrah, Basrah, Iraq.

Abstract

(The presented research represents a humanoid robot that was developed and implemented by ourself. The robot encompasses sensory capabilities similar to those found in humans, such as vision, touch, and hearing. Through comprehensive research, design, and implementation, the robot was able to accurately mimic human senses. The success of this project sheds light on the potential of humanoid robots in enhancing communication between humans and machines. The development of a robot with human-like sensory abilities opens up new horizons for applications in fields such as healthcare, assistance, and entertainment. Human beings have been forced to directly interact with infectious disease patients and undertake challenging tasks that consume time and effort. Therefore, we created a fully implemented humanoid robot equipped with human-like senses to assist humans in these tasks.

Our project involves a humanoid robot that we have developed entirely, which includes senses similar to humans. For example: it has the ability to recognize people by their faces and call them by their names, interacting with them accordingly. It can also recognize objects and emotions, enabling it to identify various things in our real-life and communicate with humans based on their feelings. It possesses the sense of hearing, allowing it to listen to people and speak to them in the Arabic language, without the need for internet connectivity. It can answer their questions or perform specific actions, such as fetching items for them. It has the ability to move its eyes, head, and all its joints, and it can face people, speak with them, and interact with its hands just like ordinary humans. Because of these capabilities, we can utilize it in medical, educational, military, and industrial fields. For example, we can use it in the field of education as a teacher who delivers lectures, answers students' questions, and creates a complete lecture atmosphere.

We used a Raspberry Pi controller, which serves as a small computer in the robot's head, and programmed it using the Python language to operate with artificial intelligence similar to that found in humans. With the presence of large servo motors, it is capable of moving each joint just like humans. This research serves as a foundation for future advancements in the field of robotics, with the aim of creating robots capable of interacting and functioning alongside humans naturally and intuitively).

Keywords: Exploration Robotics; Humanoid Robot; Python language; Raspberry Pi Controller; Robotics

Chapter 1: Introduction of robotics

Introduction

- Robotics has witnessed astonishing advancements in recent decades, enabling humans to design and develop humanoid mechanisms capable of executing various tasks independently. This type of robot is known as an android and represents a significant achievement in the fields of artificial intelligence and technology. This introduction aims to explore the concept of androids, shed light on their unique features, highlight their significant importance, and discuss the diverse applications of this innovative technology [1].
- Interest in androids dates back to a long period in human history, where ideas and concepts about the existence of mechanical beings resembling humans sparked imagination and curiosity. In literature and popular culture, humanoid robots have been depicted in numerous stories and films, stimulating analysis and scientific research into the possibility of realizing this reality [2].
- However, the true leap in android development occurred in recent decades with tremendous progress in the fields of artificial intelligence and robotics. One pivotal milestone in the history of androids can be traced back to 1997 when computers managed to defeat the reigning world chess champion at that time, Gary Kasparov, in a historic match. This event demonstrated the capability of artificial intelligence to excel in areas considered challenging by humans [3].

Figure (1-1): The Event Demonstrated the Capability of Artificial Intelligence to Excel in Areas Considered Challenging by Humans.

Since then, with continuous advancements in robotics and artificial intelligence technology, practical applications of androids have started to emerge. These applications encompass fields such as medicine, industry, logistics services, space exploration, and many others. The android is considered an innovative and versatile tool that can be utilized across a wide range of domains and industries. In the field of medicine, precision surgical robots can assist doctors in performing intricate and complex operations with high accuracy, reducing risks and improving surgical outcomes. In the industrial sector, robots can be employed in production lines to enhance efficiency, precision, and minimize human errors.

Furthermore, androids can serve as personal assistants in people's daily lives. For instance, home robots can perform routine household tasks such as cleaning, cooking, and shopping, saving time and effort for individuals and families. Additionally, androids can be utilized in the realms of education and training, providing interactive and tailored educational lessons to meet the specific needs of each individual [4].

The advancements in artificial intelligence have contributed to the development of the creative capabilities of humanoid robots. For example, robots can be trained to sense and interact with art and music, enabling them to produce unique artistic and musical works. Humanoid robots can also be utilized in research and exploration, such as space exploration or accessing harsh environments that are difficult for humans to reach.

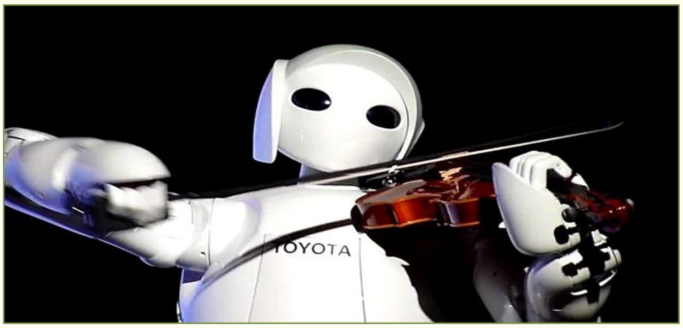


Figure (1-2): The Advancements in Artificial Intelligence Have Contributed to the Development of the Creative Capabilities of Humanoid Robots.

The evolution of humanoid robots extends beyond the realms of technology and science; it also has significant impacts on social and ethical domains. Humanoid robots raise complex ethical challenges and questions: What are their rights and responsibilities? Are they considered part of social life in the same way humans are? Can humanoid robots interact and communicate with humans naturally? On the other hand, humanoid robots also pose economic and social challenges. Despite the numerous benefits of using robots and humanoid AI in various fields, they may also lead to job displacement for humans. We need to consider how to steer technological development in a way that benefits everyone and does not negatively affect the human workforce [5]. Furthermore, we must address the security challenges associated with humanoid robots. With increased technology integration in our daily lives, security vulnerabilities may arise, putting sensitive data and privacy at risk. It is crucial to ensure that humanoid robots have robust security systems and are used in ways that protect user rights and maintain their safety.

We anticipate further progress in humanoid AI technology in the future, where humanoid robots are expected to become more advanced and sophisticated. Humanoid robots may be developed to efficiently and accurately perform human tasks, thereby improving human lives and advancements in various fields.

The humanoid robot represents a remarkable advancement in the field of robotics and artificial intelligence, combining technology and computer science to create humanoid beings capable of performing complex tasks. Humanoid robots possess unique features and advanced learning capabilities, making them powerful tools in various fields. The evolution in this field is expected to continue, opening up new and exciting possibilities for humanoid robots in the future [6].

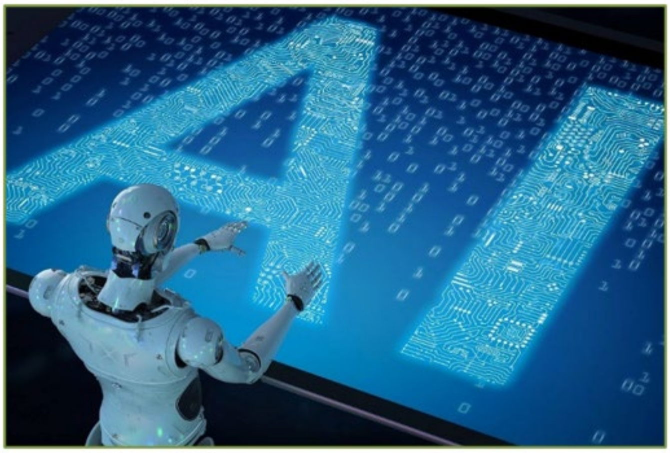


Figure (1-3): The Evolution in This Field is Expected to Continue, Opening Up New and Exciting Possibilities for Humanoid Robots in the Future.

Concept of Humanoid Robots

In this work, we will explore the concept of humanoid robots in more detail. The humanoid robot represents a remarkable advancement in technology, aiming to create an artificial being that possesses human-like qualities and capabilities. We will discuss the definition, influential factors, technical and ethical challenges, and potential impacts of this emerging concept.

Definition of Humanoid Robots

The concept of humanoid robots refers to the creation of an artificial being that emulates humans in their abilities and characteristics. Humanoid robots aim to embody artificial intelligence, enhance interaction with the environment, engage in social interaction, and continuously learn. Achieving humanoid robots requires advanced technologies such as robotics, artificial intelligence, and machine learning [7].

Influential Factors in Humanoid Robot Design

Designing humanoid robots involves several influential factors. Humanoid robots need to have the ability to learn, adapt to changes, and face various challenges. This requires the use of advanced techniques in areas such as artificial intelligence, data analysis, and sensors. Humanoid robots should also possess the ability to express emotions and understand social contexts [8].

Technical and Ethical Challenges

The technology surrounding humanoid robots faces multiple challenges. From a technical perspective, designing humanoid robots requires the utilization of advanced techniques in areas such as artificial intelligence, machine learning, and data interpretation. Humanoid robots should also have the ability to interact naturally with humans, comprehend and collaborate with them. From an ethical standpoint, we need to consider issues such as data privacy, ethical control, the impact of humanoid robot utilization on human jobs, and addressing the ethical challenges associated with the development of this technology [9].

Potential Impacts of Humanoid Robots

Humanoid robots can have diverse impacts on society and individuals. On one hand, humanoid robots may contribute to improving people's daily lives by providing assistance in routine tasks, healthcare services, and social care. On the other hand, humanoid robots may raise concerns about job displacement for humans and their impact on the econoour and human identity [10].

Applications of Humanoid Robots

Humanoid robots can be utilized in various fields to perform diverse and complex tasks. Humanoid robots excel in their ability to work for long periods without needing breaks and to execute tasks with precision and speed. In this paper, we will explore some areas where humanoid robots can be effectively used and compare them to humans in terms of functionality.

Manufacturing

Humanoid robots can be employed in manufacturing and production sectors to enhance efficiency and improve quality. Robots can carry out hazardous or repetitive tasks with high accuracy and without the need for breaks, reducing the risk of errors and increasing productivity. In contrast, humans require regular breaks and may make mistakes due to fatigue or emotional fluctuations [11].

Figure (1-4): The Humanoid Robots can be Employed in Manufacturing and Production Sectors to Enhance Efficiency and Improve Quality.

Healthcare

Humanoid robots can be used in healthcare to assist in surgeries and delicate procedures. Robots can perform surgical procedures with high precision and reduce the risk of errors. In comparison, human doctors and surgeons may be affected by fatigue, stress, and psychological pressure, which can impact their performance and accuracy. Additionally, they can provide psychological and emotional support to patients, execute therapeutic exercises, and monitor their health status [12].

Figure (1-5): The Robots Can Perform Surgical Procedures with High Precision and Reduce the Risk of Errors.

Exploration and Mining

Humanoid robots can be utilized in exploration and mining tasks in unsafe or unsuitable environments for humans. Robots can explore outer space, deep ocean areas, and hazardous locations such as mines. Humanoid robots have the advantage of not being affected by dangerous conditions or environmental pressures, while humans require special protection and equipment to deal with such conditions [13].

Figure (1-6): The Robots Can Explore Outer Space, Deep Ocean Areas, and Hazardous Locations Such as Mines.

Logistics and Delivery Services

Humanoid robots can contribute to improving logistics and delivery services. Robots and self-driving vehicles can deliver parcels and goods quickly and accurately without the need for breaks or delays. Humanoid robots can also be efficient in sorting, packing, and warehouse management operations [14].

Figure (1-7): The Robots and Self-driving Vehicles Can Deliver Parcels and Goods Quickly and Accurately Without the Need for Breaks or Delays.

Education and Training

Humanoid robots can be used in the educational field to provide interactive lessons and exercises for students. Robots can establish connections and interact with students, delivering clear explanations and support in the learning process [15].

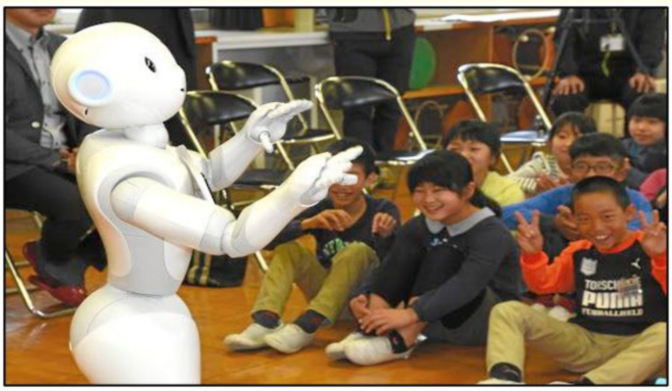


Figure (1-8): The Robots Can Establish Connections and Interact with Students.

Elderly Assistance

Humanoid robots can assist the elderly in their daily lives, such as providing mobility support, reminding them to take medication, and accompanying them in social activities. Humanoid robots can also play a role in alleviating loneliness and providing psychological well-being for the elderly [16].

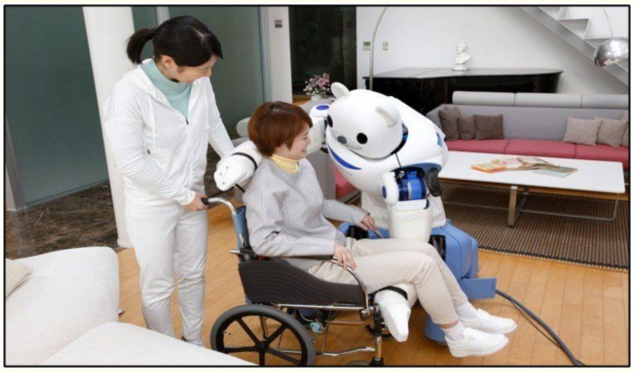


Figure (1-9): The Humanoid Robots Can Also Play a Role in Alleviating Loneliness and Providing Psychological Well-Being for the Elderly.

Social Interactions

Humanoid robots can be used to enhance social interactions when humans are unable to be physically present. For example, humanoid robots can represent individuals in conferences or remote meetings, allowing them to participate and interact with others [17].

Figure (1-10): The Humanoid Robots can Represent Individuals in Conferences or Remote Meetings.

Entertainment and Performing Arts

Humanoid robots can be employed in artistic and entertainment fields, such as shows and theatrical performances, where robots can assume roles and interact with the audience [18].

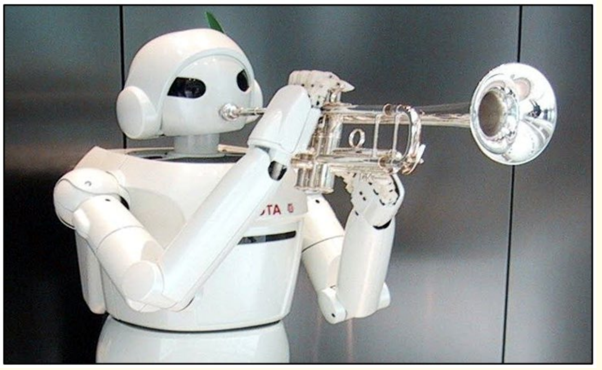


Figure (1-11): The Robots Can Assume Roles and Interact with the Audience.

These are just a few examples, and there are many other applications of humanoid robots in society. It is important to note that while this technology has significant benefits, it also requires considering ethical issues and ensuring the safety and security of its utilization in society.

Comparison between Humanoid Robots and Humans in Practical Matters

Efficiency

Humanoid robots excel over humans in efficiency, as they can work for long periods without the need for breaks and with high precision. On the other hand, humans require rest periods and can be affected by fatigue or emotional pressure, which can impact their performance and ability to work efficiently [19].

Accuracy

Humanoid robots are considered more accurate in task execution compared to humans. Robots can perform tasks with extreme precision and avoid human errors. In contrast, humans may make mistakes due to fatigue, pressure, or other human errors.

Flexibility

Humans may be more flexible in adapting to changing conditions and diverse tasks. While humanoid robots can be limited by programming and specific capabilities they are designed for, humans can learn and adapt to changes more extensively and quickly adjust to new tasks [20].

Social Interaction

Humans excel in social interaction and communication with each other. Humans possess the ability to interpret emotions, show empathy, and engage in social interaction, making them more capable of effectively interacting with others. In contrast, humanoid robots are programmed and mechanical, often lacking deep social interaction abilities [21]. These are some of the aspects where humanoid robots and humans differ in practical matters. While humanoid robots have advantages in certain areas such as efficiency and accuracy, humans possess unique qualities such as adaptability and advanced social interaction capabilities. The comparison highlights the complementary nature of humans and humanoid robots, where both can contribute their strengths to different tasks and contexts [22].

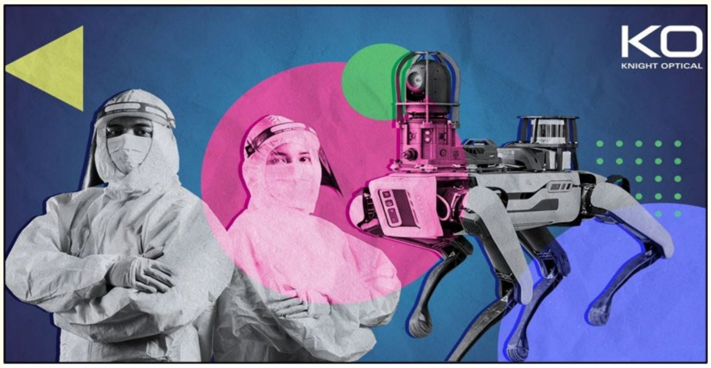


Figure (1-12): The Robots Have been Utilized in Several Areas to Enhance Disinfection and Sanitization Efforts.

During the COVID-19 pandemic, routine disinfection and sanitization procedures have become crucial in reducing the spread of the virus. Robots have been utilized in several areas to enhance disinfection and sanitization efforts. Here are some examples:

1. *Hospital Sanitization*: Robots have been used to sanitize rooms and public areas in hospitals. These robots employ techniques such as ultraviolet (UV) radiation or chemical disinfection to kill germs and viruses present on surfaces.
2. *Autonomous Delivery*: Robots have been utilized to deliver food, medication, and essential supplies to COVID-19 patients in hospitals or home isolation. These robots are programmed to navigate safely and deliver items without exposure to risks.
3. *Awareness and Education*: Robots have been used to spread awareness and provide information about the coronavirus and preventive measures. Robots can interact with people and offer advice and guidance on how to maintain safety and practice social distancing.
4. *Medical Screening*: Robots have been employed in some cases to assist in conducting non-contact medical screenings, such as temperature checks, heart rate measurement, and oxygen saturation levels, reducing direct interaction between individuals and healthcare workers.

The role of robots in these scenarios is to minimize human exposure to risks and enhance efficiency and accuracy in necessary disinfection and healthcare processes during the COVID-19 pandemic [23].

Chapter 2: Related Works

Inmoved Robot

The Inmoved Robot is an open-source humanoid robot developed by Gaël Langevin. It is one of the most popular open-source humanoid robots that can be built and customized by users. Inmoved is designed to be a bipedal and bimanual robot that can be controlled through human-like movements and equipped with a variety of sensors. The robot's motion and interaction are achieved using intelligent control and sensing techniques [24].

Development and Usage History

The inmoved Robot was first developed in 2012 when Gaël Langevin released his initial design online as an open-source project. Since then, inmoved has gained significant popularity and became available for users to download and 3D print the parts for self-assembly. inmoved has been used in various applications and research fields, including educational and interactive robotics, scientific research, and human-robot interface development [25].

Figure (2-1): Person who first designed the Inmoved robot.

Features and Capabilities of inmoved

Inmoved boasts a wide range of features and capabilities that make it intriguing for many applications. The robot's motion is achieved through a combination of controllable motors and joints. Users can control the robot's movements either through motion sensors or external control interfaces such as a smartphone or a keyboard. inmoved can interact with its surrounding environment by sensing objects and responding to voice inputs. Additionally, the shape and appearance of inmoved can be customized according to user needs [26].

Future Applications and Challenges

The flexible design of InMoov enables its usage in a wide range of future applications. It can be utilized in education and scientific research, where it can be used to develop and test new models in robotics and control. It can also find applications in medical fields, such as the development of advanced prosthetics and physical therapy exercises. Furthermore, InMoov can be employed in artistic and entertainment domains, where it can be customized to act as an assistant in performances or artistic displays.

With these exciting future applications, inmoved also faces some challenges. Among these challenges, there may be difficulty in properly building and assembling the robot, requiring expertise in engineering and electronics. Additionally, users may encounter difficulties in programming and controlling the robot, especially if they lack the necessary experience in robot programming.

In conclusion, the inmoved Robot is a remarkable creation in the world of open-source robotics. It provides users with an opportunity to explore the capabilities of advanced humanoid robots, customize them, and utilize them in various fields. With the advancement of technology and the open community around inmoved, we can expect further developments and exciting applications in the future [27].

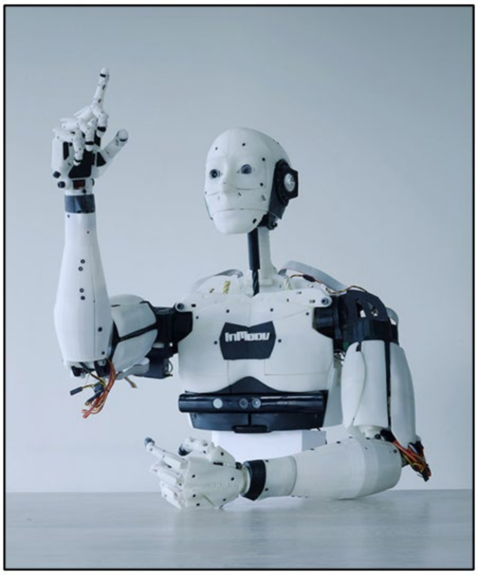


Figure (2-2): Inmove robot moves its arm.

Pepper Robot

Pepper is a social robot developed by SoftBank Robotics. It was first unveiled in 2014. Pepper is one of the humanoid robots programmed with artificial intelligence designed to interact with humans and provide social support.

Pepper is equipped with a range of sensors and technologies that enable it to recognize faces, analyze facial expressions, and engage in voice communication with humans. It relies on machine learning and natural language processing techniques to understand human commands and respond appropriately [28].

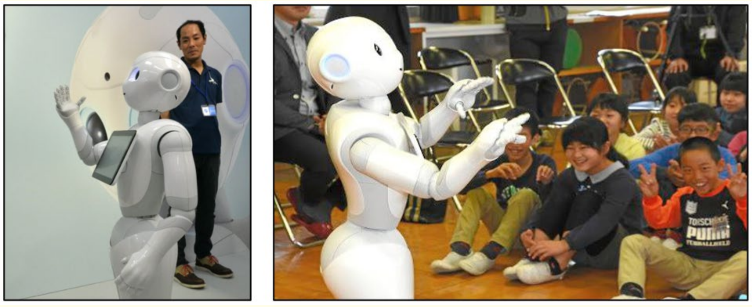


Figure (2-3): Pepper Robot.

Development and Usage

Pepper is used in various fields. In the commercial sector, Pepper can be seen in stores and restaurants where it is used to greet customers and provide information about products and services. It can also guide customers to the appropriate sections or direct them to points of sale. Pepper can also provide interactive experiences for customers through games or entertainment activities.

In addition, Pepper can be used in educational settings, healthcare, museums, exhibitions, and social events. This robot can provide interactive and educational experiences for people in these contexts, such as assisting students or providing historical information in museums.

Pepper is programmed by developers using programming languages such as Python and Java, and it has its own development platform. Developers can create customized applications and skills for Pepper to suit the specific needs of the context in which it is used.

In summary, Pepper is a social robot developed by SoftBank Robotics in 2014. It is used in various fields such as commerce, education, healthcare, museums, and exhibitions. It is programmed by developers using different programming languages, and customized applications and skills can be developed for it [28].

Chapter 3: Structure industry

The inmoved robot chassis consists of several moving parts and different components that integrate together to form the complete robot chassis. The skeleton includes arms, feet, head, and fingers, and is characterized by its ability to achieve movements and actions similar to those of a human. The parts are carefully designed to achieve the highest degree of flexibility and precision in movement.

Inmoved robot is characterized by providing maximum flexibility to users, as they can customize and modify the robot according to their specific needs. Allowing them to create a custom robot model that matches their vision and requirements.

The Head makin

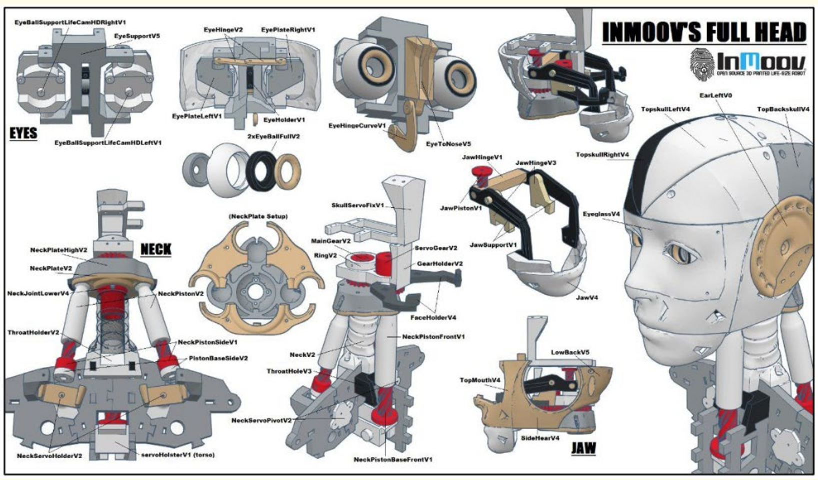


Figure (3-1): The Head component.

The eyes

We printed the eyes according to the design in the picture and added two servo motors to enable them to move right, left, up and down. We also added a Raspberry Pi camera to the inside of the eye.

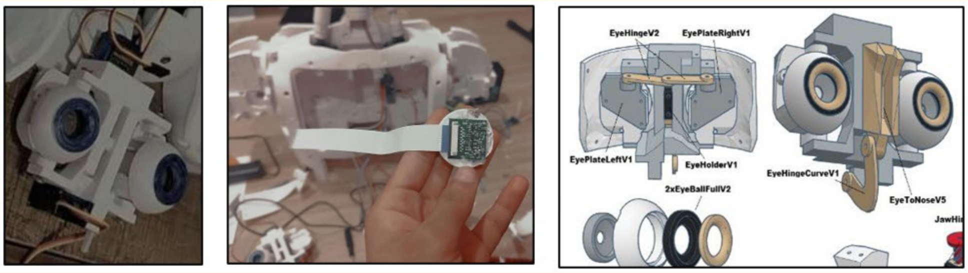


Figure (3-2): Real pictures of the eye.

The Mouth

We printed the parts that you see in the picture, so that the mouth is mounted on supports and a servo motor is installed in it, which rotates the part called the gas pipe to open and close the mouth as you see in the picture.

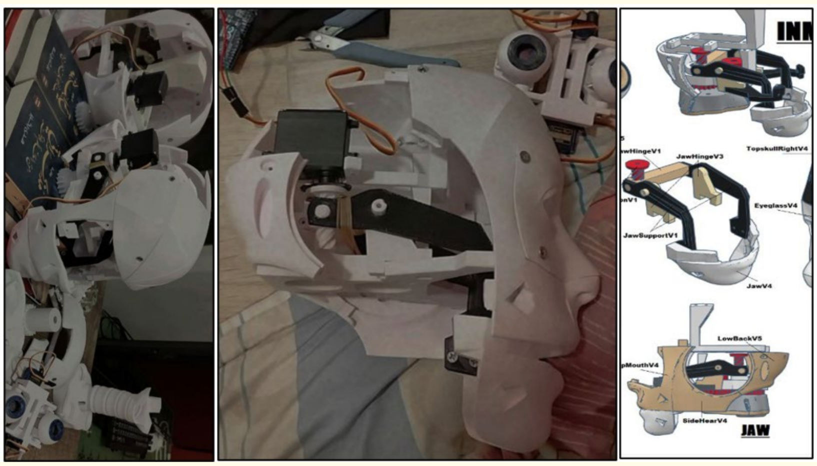


Figure (3-3): Real pictures of the Mouth.

The Neck

We printed the parts you see in the picture. The three parts at the bottom of the neck, called piston, we connected to servo motors to move the neck in three dimensions so that it moves exactly like a human neck. The trachea-like structure is hollow so that we can pass wires through it later. As for the part that rotates the head right and left, it is located above the neck, which is called the main gear. The part called the servo gear is connected to a servo motor to rotate the main gear to turn the head.

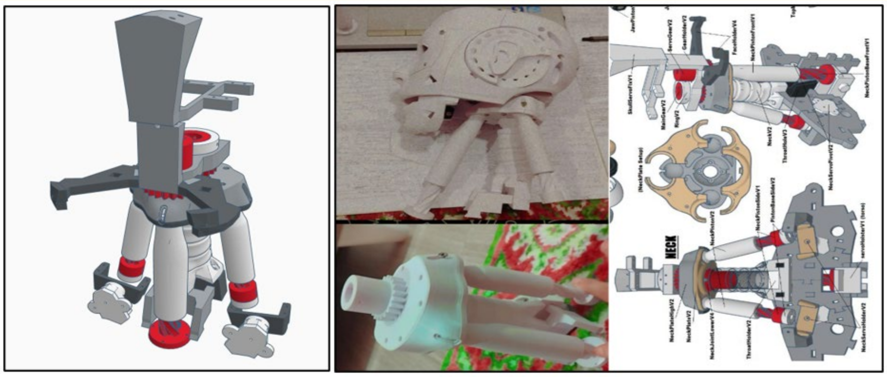


Figure (3-4): Real pictures of the neck.

The Chest / Back making

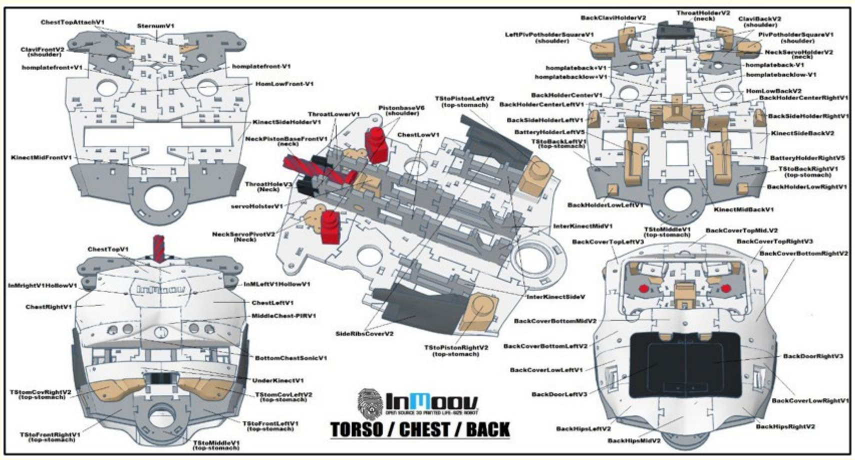


Figure (3-5): The component of Chest / Back.

The Chest

We printed the chest, its internal and external parts, as you can see in the picture below. It contains many places for me to install additions to hold a specific sensor or to hold other parts of the body, such as the arms. As for the structure between the chest and the back, it contains the part called the neck piston and a designated place. It has a large servo motor to raise and lower the neck.

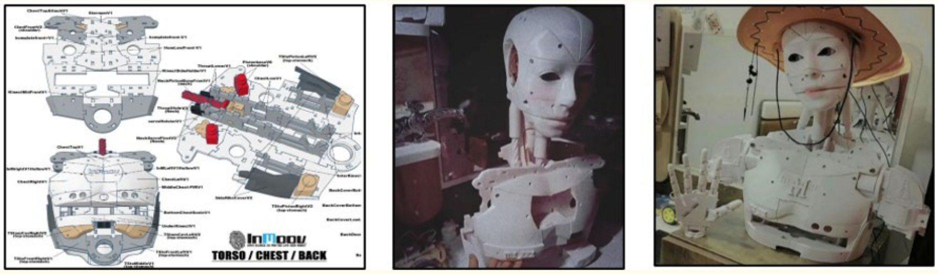


Figure (3-6): Real pictures of the chest.

The Back

We printed the back as in the picture below. Here in the back of the robot, all electrical and electronic parts are installed, such as motor controllers and wires. We changed the design a lot here because the robot was designed for different controllers than the controllers we used, and we also changed the batteries and all the back design.

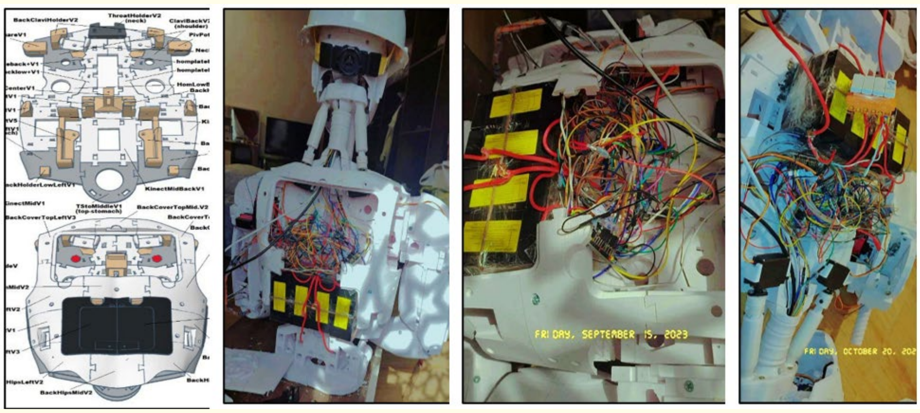


Figure (3-7): Real pictures of the back.

The Shoulder and Hand makin

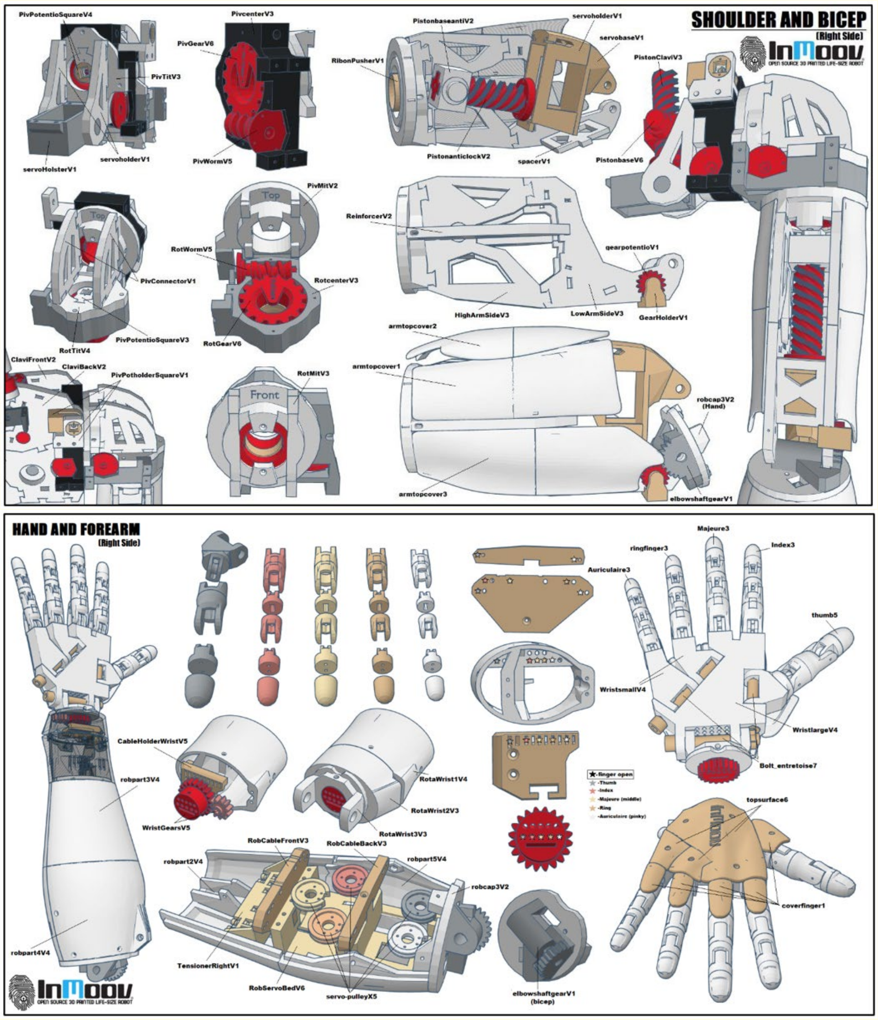


Figure (3-8): The component of Shoulder and Hand.

The Shoulder

The shoulder contains many details, which you can see in the picture below. The shoulder contains three large servo motors that allow it to move in all axes, but the piston here requires rotation by much more than 360 degrees, i.e. more than the angle of rotation of the servo motors, so we modified the servo motors to rotate. To what degree do I want it, even if it is, for example, 9000 degrees.

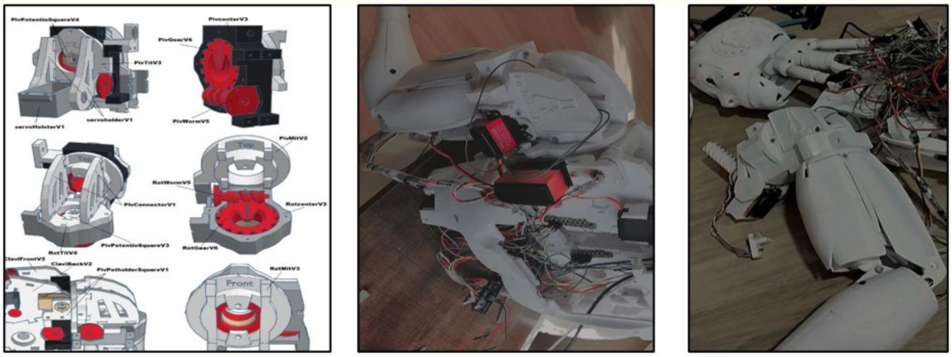


Figure (3-9): Real pictures of the Shoulder.

The Bicep

We printed this part as you see in the figure below. It is simply supports to hold the rest of the hand, and it also contains a large servo motor that moves the arm up and down. Here we also modified the motor to rotate more than 360 degrees.

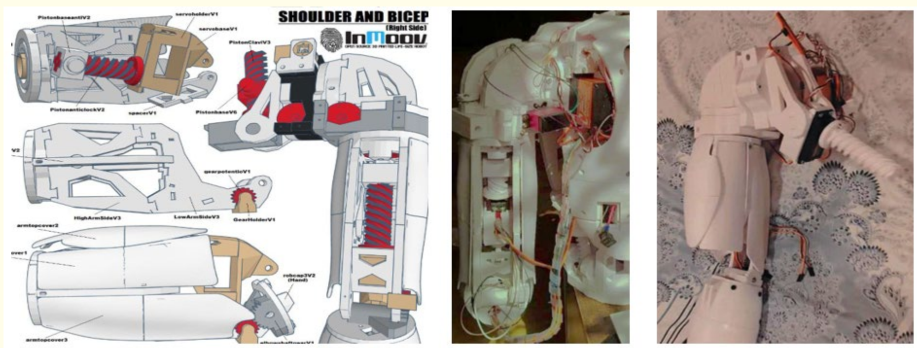


Figure (3-10): Real pictures of the Bicep.

The Hand

Here we printed the hand as in the picture below. The hand consists of many parts, the outer casing that holds the motors, The part that holds the motors of the fingers and the small disc that holds the strings that the motor pulls to close and open the fingers. And also, the parts through which the strings and gears pass, which rotate the wrist Finally, the palm and fingers. Here, each joint can be closed and opened like a human hand, and there are openings for the strings to pass through.

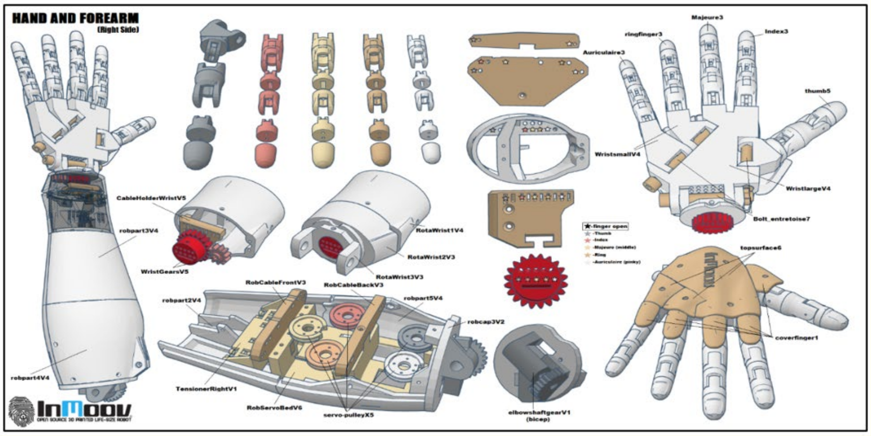


Figure (3-11): The component of Shoulder and Hand.

Figure (3-12): Real pictures of the hand.

Printing the 3D models

We bought an Ender 3 S1 Pro 3D printer to print the models we needed. The printer has many features that we will mention.

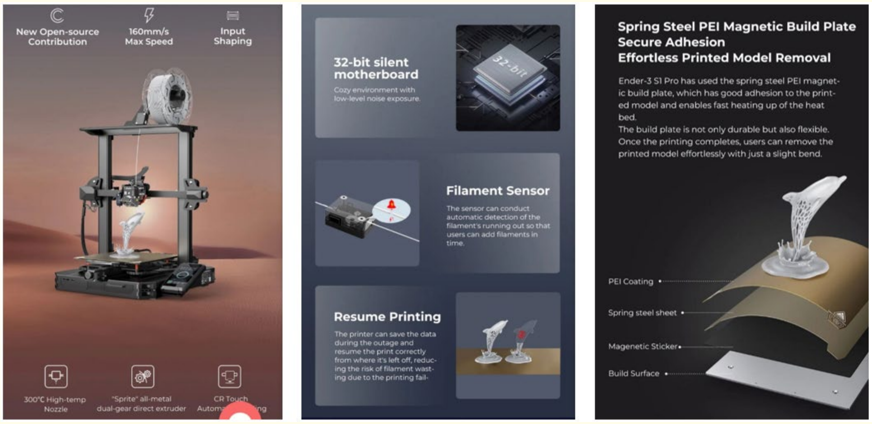


Figure (3-13): 3D printer.

Figure (3-14): 3D printer, Real photos.

The Hardware components

In this paragraph, we will talk about the hardware components of the robot, how to connect them and how they work, and how I made modifications to them, but first we will talk in an introductory paragraph about all the materials and the electrical circuit in general.

raspberry pi

Today, we talking about Raspberry Pi, a small-sized computer that runs on the Linux operating system. Raspberry Pi was designed to be a versatile device that can be used in a wide range of projects and applications. Raspberry Pi stands out for its small size and low cost, making it accessible and suitable for both hobbyists and professionals. It features a range of connectivity ports such as HDMI, USB, Ethernet, and GPIO, allowing it to be connected to various devices and electronic components. Raspberry Pi can be used in many fields such as robotics development, smart home automation systems, building sensors, networked storage devices, Internet of Things (IoT) applications, and more. It can be programmed using different programming languages like Python and C++, enabling users to develop advanced and customized applications to meet their individual needs [25].

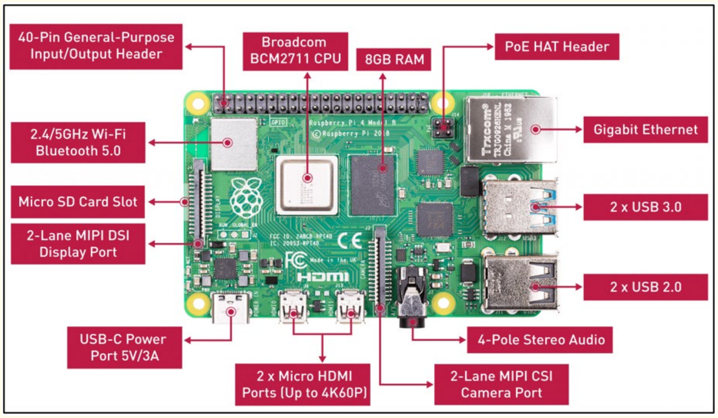


Figure (3-15): raspberry pi.

raspberry pi camera

Raspberry Pi Camera is a camera specifically designed for use with Raspberry Pi computers. It is one of the popular and useful additions to the Raspberry Pi developer community, The camera features the following specifications:

- *High Resolution*: It boasts a high resolution of up to 8 megapixels (original version) and 12.3 megapixels (second version). This high resolution allows for capturing high-quality photos and recording high-definition videos.
- *Compact Design*: The camera comes in a small and lightweight design, making it easy to use and install with Raspberry Pi computers.
- *CSI Interface*: The camera utilizes a Camera Serial Interface (CSI) to directly and quickly connect to a Raspberry Pi computer. This enables fast and efficient data transfer from the camera to the computer.
- *Video Support*: The camera supports video recording with resolutions up to 1080p (original version) and 4K (second version). It can be used for applications such as photography, video recording, visual monitoring, object recognition, and more.

Wide Range of Software and Tools: Raspberry Pi Camera offers a wide range of software and tools that facilitate its usage and the development of various applications. It can be used with libraries like OpenCV and other available libraries for image and video analysis, object recognition applications, Raspberry Pi Camera is used in a variety of applications, including amateur photography, video recording, surveillance and security, face and object recognition, robotics, and various electronic projects [26].

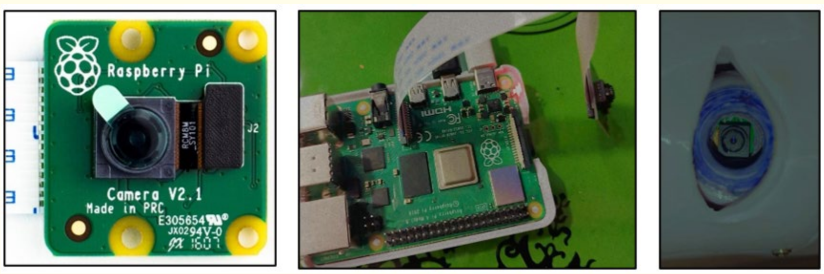


Figure (3-16): raspberry pi camera.

PCA9685

PCM9685 is a driver and control interface for serial signals used to control small motors and other devices. It is a widely adopted module in the developer and hobbyist community working with the Raspberry Pi platform and other embedded systems. PCM9685 features the following specifications and features: Motor Control Capability: PCM9685 provides multiple channels for controlling small motors and devices that rely on Pulse Width Modulation (PWM) signals. It can be used to operate small motors, fans, lights, indicators, and other devices that respond to PWM signals. I2C Interface: PCM9685 relies on the I2C (Inter-Integrated Circuit) interface to communicate with the Raspberry Pi or other devices. This interface enables fast and reliable communication between the driver and the host device. High-Precision PWM Timing: PCM9685 offers high precision in generating PWM signals through accurate timing techniques. Duty cycle and pulse timing can be finely adjusted for precise control of motors and devices. Multiple Channels: PCM9685 is available in different versions that provide multiple channels for motor and device control. It can be used to connect and control up to 16 devices or motors [27].

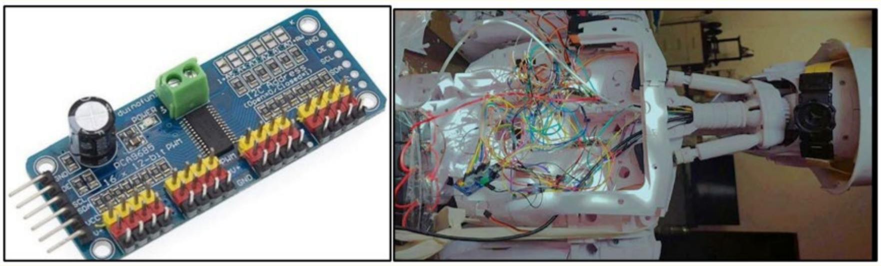


Figure (3-17): PCA9685.

Servo Motor

A servo motor is a type of electric motor commonly used in control systems, robotics, and electronic projects. The servo motor is characterized by its ability to precisely move its shaft under direct control from a control signal. A servo motor consists of several key components:

- *Motor*: The servo motor contains an electric motor that converts electrical power into rotational motion.
- *Gear set*: A gear set is used inside the servo motor to convert the rotational motion from the motor into precise and controlled movement of the motor's shaft.
- *Potentiometer*: The potentiometer contains a sensor that measures the rotational angle of the motor's shaft. This sensor is used to determine the position of the shaft and guide it based on the control signal.
- *Control circuit*: The control circuit includes electronics that analyze the control signal, move the motor, and respond to received commands.

A key feature of a servo motor is its ability to accurately determine and convert position. The position is determined by the control signal sent to the control circuit. The position is expressed in terms of the rotational angle of the shaft, and the shaft is moved precisely based on the desired value. Servo motors are used in various applications such as:

- *Robotics*: Servo motors are used in the joints of robots to precisely move arms, legs, grippers, and other mechanical parts under specific control.
- *Control systems*: Servo motors are used in automated control systems and robotics to move mechanical parts with precision according to control signals [28].

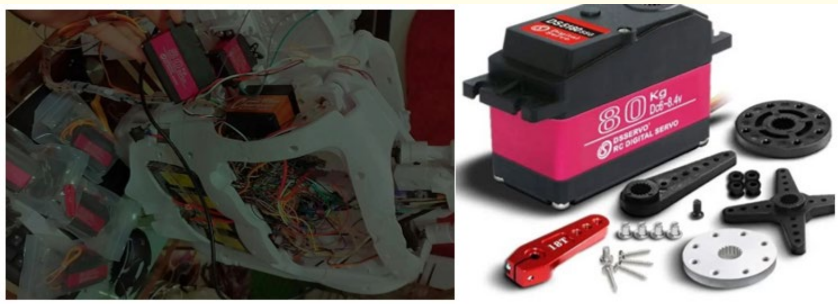


Figure (3-18): Servo motor.

Power and charging department

In the beginning, we used the circuit I made, shown in the picture below, with a current of 2 amps and 6 volts, to operate the robot's motors when we was examining it and experimenting with it to measure angles and so on.

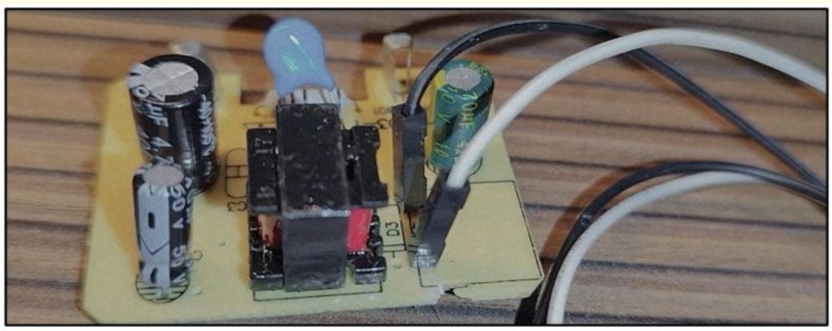


Figure (3-19): Charging circuit.

After that, we used four batteries and a power source, as in the pictures below.

Figure (3-20): Battery and power supply.

We used four 6-volt batteries in parallel with a current of 4 amps per hour, so that the total voltage was 6 volts and the total current was 16 amps per hour, as in the picture.

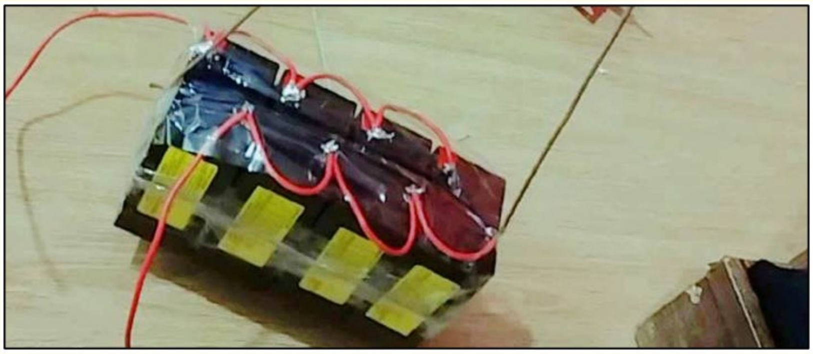


Figure (3-21): Battery circuit.

But the problem was that when we tried to charge the batteries with a 12-volt power source, we needed a voltage reduction circuit, i.e. a converter. we brought a converter and reduced the voltage to 6 volts, but the current was very high. The rated current of the converter was 2 amps, while the batteries were 16 amps, so they were this method is unsuccessful, meaning we have to disconnect all the batteries to connect each two batteries in parallel and then charge them.

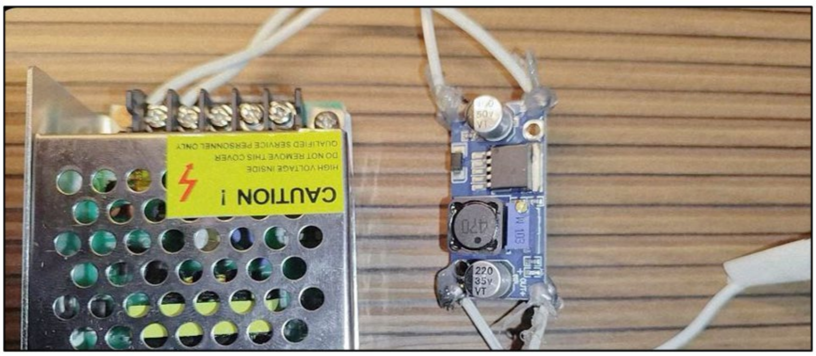


Figure (3-22): Charging circuit.

So we resorted to a method Charging circuit: - which is to connect each two batteries in parallel and connect the two groups in series with each other, so the voltage becomes 12 volts and the current becomes 8 amps, so the power supply we have can charge them. The robot's power circuit: This involves connecting all the batteries in parallel, as in the previous case, to supply the robot's motors with high current. we created these two states using Switch, when you press it, it changes from one state to another, as in the picture below.

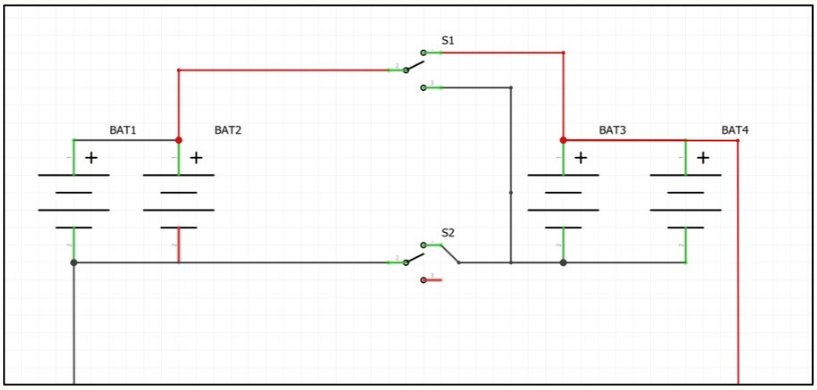


Figure (3-23): Charging circuit.

This picture represents how to connect batteries to achieve the conditions of series and parallel by pressing a button (S1 and S2 pressed together) Here are the first and second switches. we glued them together so that they can be pressed together so that the operating state consisting of four batteries connected in parallel is transferred to the charging state consisting of two parallel sets in series.

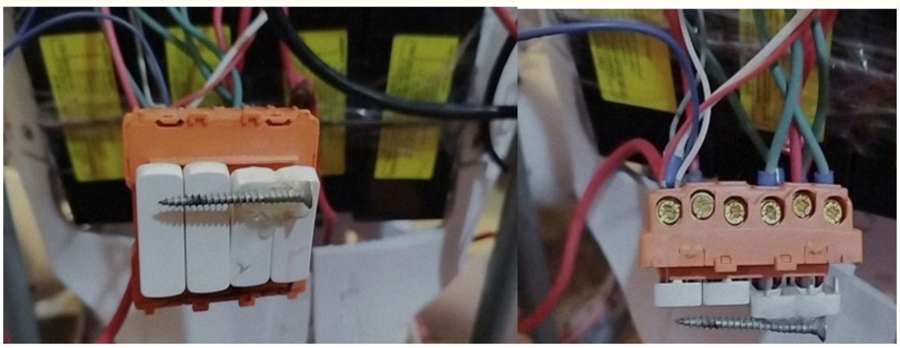


Figure (3-24): Charging circuit.

We passed the screw to the end of the switch. The first switch turns on the robot, so when we turn on the charging circuit, the screw turns off the robot circuit to avoid putting 12 volts on the servo motors, because these voltages damage them.

Manufacturing Method

In this paragraph, we will talk about making a robot from the components that we talked about previously.

The Head

First, we installed the eyes whose printing we explained previously and added two servo motors to move them right, left, up and down. Then we attached the eyes to the face with screws and attached the pieces for the mouth and head that we talked about previously. Here we used two servo motors, the first with a torque of ten kilograms per centimeter to open and close the mouth, and the second with a torque of thirty kilograms per centimeter to rotate the neck right and left. These motors will link their movement to the software code that will move them intelligently.

Figure (3-25): Eyes head mechanics.

Here we changed the design of the head a little, as I removed the piece that carries the weight of the head from the top to place the Raspberry Pi at the top of its head, because here we will be using the Raspberry Pi and our own circuits, not the electronic circuits and controllers assigned to this robot. At the same time, we passed the camera from the Raspberry Pi to the robot's eye and connected it. The driver for the Raspberry servo motors and the wires and wires to operate and display the Raspberry Pi were passed through the neck as in the pictures.

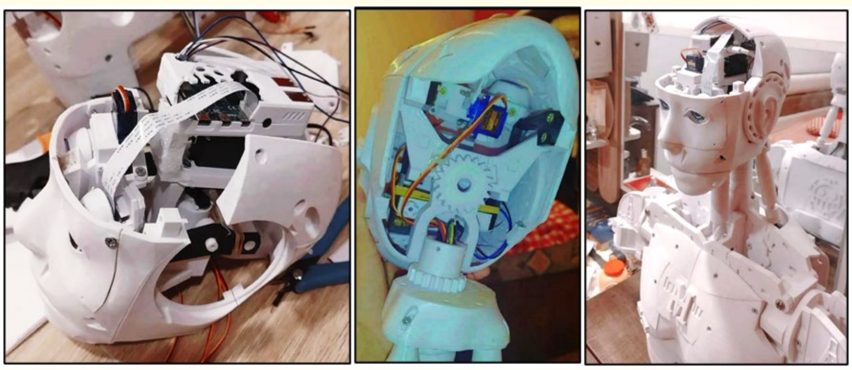


Figure (3-26): Head mechanics and components.

We wanted to print a smart helmet for the robot that could open and close itself according to what the robot wanted and had other properties, but it was not done due to lack of support.

Figure (3-27): Robot helmet.

We printed the neck as we explained previously, where three pistons will be placed in it along with the neck joint to move it in all possible directions. Here we used two servo motors with a torque of ten kilograms per centimeter for the pistons on the edges, and a motor with a torque of eighty kilograms per centimeter to raise and lower the entire head. This motor is installed in the middle of the chest. Modified motor, here I started to get to the modified motors, so let us explain what we mean by modified motor and how to make the motor rotate more than 1000 degrees to reach the location we want.

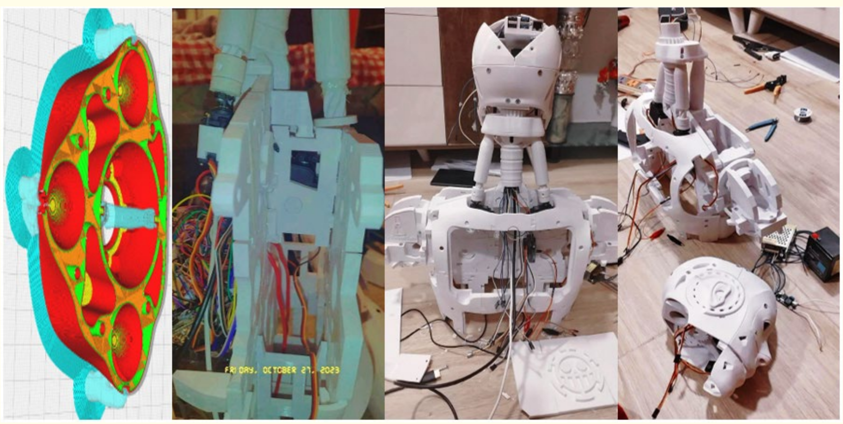


Figure (3-28): Components and composition of the neck and chest.

Modified servo motors

In many places of the robot that contain gears and pistons,we needed to rotate the motor many revolutions, taking into account that it stops when it reached its position. The robot is designed to allow the presence of rectangular places for the motors, such as the structure of the servo, and it is not possible to modify it to add a stepper motor, so we modified the servo itself to rotate more. From 360 degrees only. Here there are two ways to do it.

- *The first method*: Which is to modify the wave width of the pulse width modulation process, and this is a long and tiring method.
- *The second method*: Which is to take out the location sensor represented by a variable resistance and place it in a mechanical design, i.e. When the motors rotate more than one revolution, the gear rotates half a revolution, for example, and here we read the angle, not from the motors, but from the gear that is placed in the place where it rotates. An example of this is the image below.

First, we opened the gear box to remove the mechanical lock. we cut the variable resistance and lengthened the wires to put them in gears to take the angle from these gears and not from the motor itself.

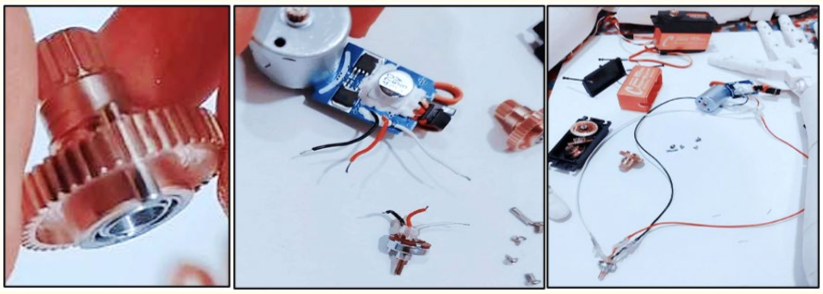


Figure (3-29): Modification of servo motor.

We can put this resistance in a gear box. The benefit of this method is: Example: We put the resistance in a gear box. When this gear rotates 20°, the motor rotates ten revolutions, and so on.

Figure (3-30): Modification of servo motor.

The Arm

We start with the hand. I connected five motors to the fingers (which we talked about printing previously), and I connected strings to the fingers. Each finger has two strings (a string is a plastic thread), one string to close and another to open the finger. As for the wrist, we connected a servo motor to rotate it 180 degrees, as in a human hand.

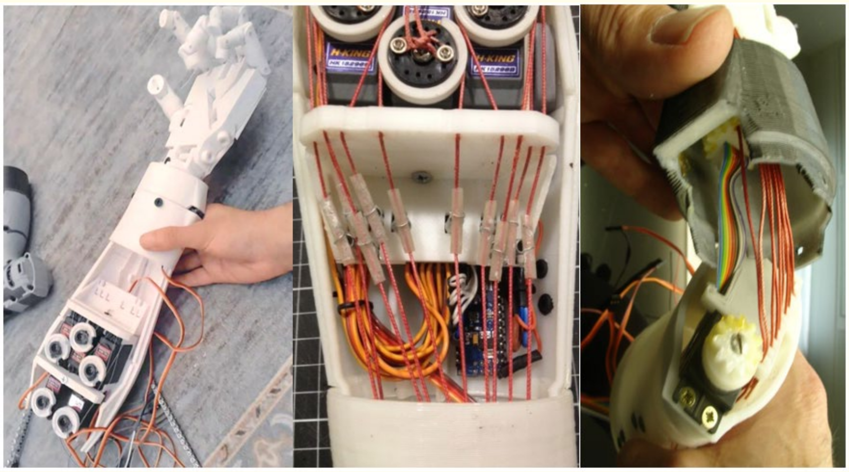


Figure (3-31): Installing the arm and fingers.

Then I installed the joint with a screw and glue and passed the wires through it. Here I connected a motor to raise the lower part and lower it, a modified motor with a torque of 80 kilograms per centimeter via a gear. while the blue arrow indicates the gear that reads the angle of the modified motor.

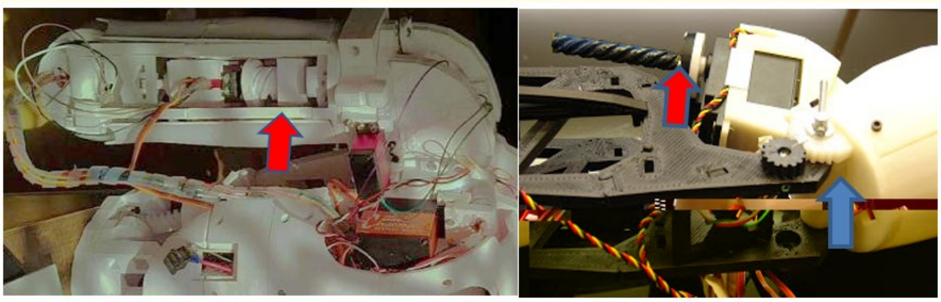


Figure (3-32): Installation of muscles and gears.

As for the shoulder, we made a structure as in the picture, and we placed small balls and fat under the large gears to make it rotate smoothly. I used three large modified servo motors with a torque of 80 kilograms per centimeter to rotate the shoulder in three different directions, as in the picture.

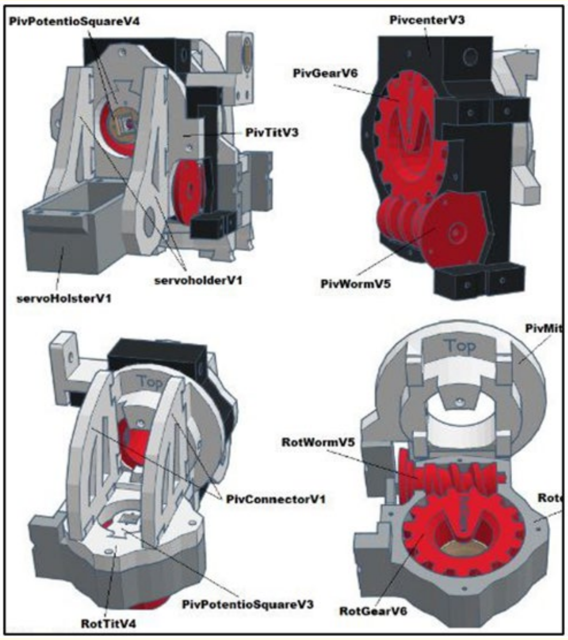


Figure (3-33): Shoulder gear.

The Back

Here we made a radical change from the original design because most of the electronic parts are different from what is usually used in this robot. We installed two metal strips to connect the wires to them and they are connected to the battery system. One represents the positive electrode, the indicator in red, and the other represents the negative electrode, which is the indicator in black, as in the picture below. As for the special signal wires with the servo motors, we connected them all to the PCA9685 piece marked in green.

(*Note*: we did not connect the three wires for each servo motor to the PCA9685 piece, because the current of all the motors may reach 24 amps, which this piece cannot handle, so we sent the corners only through this piece. As for the power on the battery system), installed the battery system and the outer casing, and placed the switches that convert the battery system from the charging circuit to the operating circuit.

The Software

The controller we used is a Raspberry Pi 4 using the Raspbian operating system. I programmed the robot using the Python language with several algorithms. We will discuss everything now.

Operating system and requirements

Instead, we downloaded the 64-bit Raspbian system. The problems we faced was our inability to install some of the main libraries in the project, including openCv, midiapipe, and keras, and others.

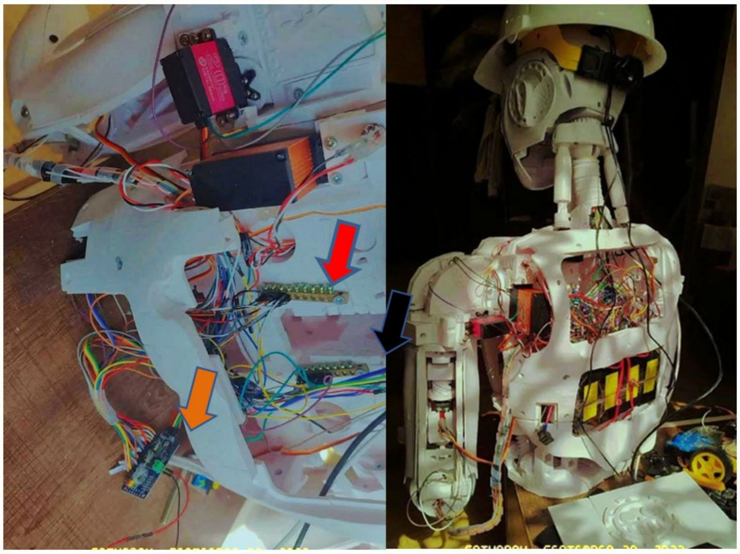


Figure (3-34): Installation of back parts and wires.

It took me a month of trying and we did not find any solution in YouTube, Google, or Gethub, so we deleted the system and downloaded the system (Raspbian Lite), Here we modified several settings from the terminal to be able to use the camera and some libraries. The most notable settings we modified are: As in the picture below, we changed the settings marked in red to make them match our requirements.

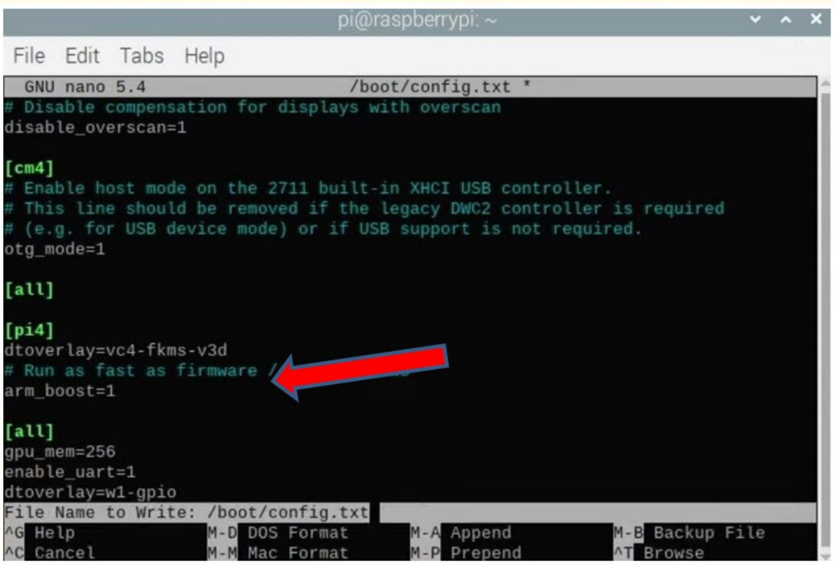


Figure (3-35): Fix a system problem related to the camera.

The Input of robot

The robot has many inputs, like humans, including the sense of hearing and sight, and in the future, it will have the sense of touch and smell, which we will discuss and explain in some detail.

Sense of hearing

It took me many months to sense our hearing and we designed it in very many ways. In the beginning, we designed it in a normal way, such as what is explained on YouTube, which is to create a voice assistant using two libraries pyttsx3 and speech recognition, but it was a simple voice assistant and programmed in a way that was not intelligent and relied on an Internet connection. This is a simple and quick example of it.

```
import speech_recognition as sr
import pyttsx3

engine = pyttsx3.init()
recognizer = sr.Recognizer()
mic = sr.Microphone()

def assistant_speech():
    with mic as source:
        print("قل شيء تريده.")
        audio = recognizer.listen(source)
        try:
            print("جارٍ التعرف على الكلام...")
            text = recognizer.recognize_google(audio, language="ar")
            print("تم التعرف على الكلام: " + text)
            assistant_response(text)

        except sr.UnknownValueError:
            print("لم يتم التعرف على الكلام.")
        except sr.RequestError as e:
            print("حدث خطأ أثناء تحويل الصوت إلى نص: " + str(e))

def assistant_response(text):
    print("الرد: " + text)
    if "السلام عليكم" in text:
        response = "عليكم السلام"
    elif "اين تسكن" in text:
        response = "أنا أعيش في البصرة"
    elif "انت شنو" in text:
        response = "أنا روبوت بشري مثلكم"
    elif "هل انت ذكي" in text:
```

```
        response = "نعم، أنا ذكي نوعًا ما"
    else:
        response = "عذرًا، لم أفهم ما قلته."
    print("الرد: " + response)
    engine.say(response)
    engine.runAndWait()

assistant_speech()
```

Figure (3.4.2.A): Online voice assistant.

Then we used Librasa to train a voice assistant called the natural language processing method. It has courses on YouTube, but this method needs a powerful computer and needs enormous data and takes a long time, so it took another method.

It is that when a person speaks, there is a wave imprint for every word, meaning that sound waves are arranged in a certain way in each word.

Arabic				
vosk-model-ar-mgb2-0.4	318M	16.40 (MGB-2 dev set)	Repackaged Arabic model trained on MGB2 dataset from Kaldi	Apache 2.0
vosk-model-ar-0.22-linto-1.1.0	1.3G	52.87 (cv test) 28.50 (MBG-2 dev set) 1.0xRT	Big model from LINTO project	AGPL

Figure (3-36): Offline Arabic model.

Using this dictionary and the method we explained above, we can create code without the Internet and in the Arabic language, Below we will put the code for the voice assistant that I created.

```
import vosk
import pyttsx3
import datetime
import pyaudio
import json
import os
import time
import playsound
from playsound import playsound
import msvcrt as m

def wait():
    m.getch()

# Initialize the text-to-speech engine
PTNK_AI_assistant = pyttsx3.init()

# Set the voice to Arabic
voice = PTNK_AI_assistant.getProperty('voices')
assistant_voice_id =
'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS_MS_AR-
EG_ABDULRAHMAN_11.0'
PTNK_AI_assistant.setProperty('voice', assistant_voice_id)

# Function to speak text in Arabic
def speak(audio):
    print('مساعد zxz: ' + audio)
    PTNK_AI_assistant.say(audio)
    PTNK_AI_assistant.runAndWait()

# Function to announce the current time
def time():
    Time = datetime.datetime.now().strftime('%I:%M: %p')
    speak('الوقت الآن')
    speak(Time)

# Function to announce the current date
def date():
    today = datetime.datetime.now().strftime(' اليوم هو%d %m %Y ، يرجى ملاحظة أن
التنسيق هو اليوم، الشهر، السنة')
    speak(today)

# Function to greet the user
def welcome():
    hour = datetime.datetime.now().hour
    if hour >= 3 and hour < 12:
        speak('صباح الخير يا سيدي')
    elif hour >= 12 and hour < 18:
```

```
        speak('مساء الخير يا سيدي')
    elif hour >= 18 and hour < 21:
        speak('مساء الخير يا سيدي')
    elif hour >= 21 and hour < 24:
        speak('تصبح على خير يا سيدي!')
    elif hour >= 0 and hour < 3:
        speak('لقد تأخر الوقت يا سيدي ، دعنا نأخذ قسطًا من الراحة')
    speak('كيف يمكنني مساعدتك الآن؟')
    print('')
    print('أستمع ...')
    print('')

# Play the welcome sound
playsound('assets/PTNK-on.mp3')

# Load the language model
model = vosk.Model("assets/model-light-with-graph")

# Clear the console
os.system('cls')

# Welcome the user
welcome()

# Initialize the speech recognizer
rec = vosk.KaldiRecognizer(model, 16000)

# Initialize the audio stream
cap = pyaudio.PyAudio()
stream = cap.open(format=pyaudio.paInt16, channels=1, rate=16000,
input=True, frames_per_buffer=8192)

# Start the stream
stream.start_stream()

# Set a counter to track the number of times the loop has run
a = 0

# Main loop
while True:
    # Read audio data from the microphone
    data = stream.read(4000, exception_on_overflow=False)

    # Check if the data is empty
    if len(data) == 0:
        break

    # If the recognizer accepts the data, process it
```

```
    if rec.AcceptWaveform(data):
        # Get the recognized text
        result = rec.Result()
        result = json.loads(result)

        # Print the recognized text
        print('المدير: ' + result['text'])

        # Respond to specific commands
        if "مرحبا" in result['text']:
            os.system('cls')
            stream.stop_stream()
            print('المدير: ' + result['text'])
            speak('مرحبا يا سيدي ، كيف يمكنني مساعدتك؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
        elif "من أنت" in result['text']:
            os.system('cls')
            stream.stop_stream()
            print('المدير: ' + result['text'])
            speak('أنا مساعدك الافتراضي. كيف يمكنني مساعدتك الآن يا سيدي ؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
        elif "من صنعك" in result['text']:
            os.system('cls')
            stream.stop_stream()
            print('المدير: ' + result['text'])
            speak('السيد محمود هو من صنعني. هل هناك شيء آخر تحتاجني لمساعدتك فيه؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
        elif "منو سواك" in result['text']:
            os.system('cls')
            stream.stop_stream()
            print('المدير: ' + result['text'])
            speak('السيد محمود هو من صنعني. هل هناك شيء آخر تحتاجني لمساعدتك فيه؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
        elif "من أنشأك" in result['text']:
            os.system('cls')
            stream.stop_stream()
```

```
            print('المدير: ' + result['text'])
            speak('السيد محمود هو من أنشأني. هل هناك شيء آخر تحتاجني لمساعدتك فيه؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
        elif "من سيدك" in result['text']:
            os.system('cls')
            stream.stop_stream()
            print('المدير: ' + result['text'])
            speak('السيد محمود هو من أنشأني لذلك هو سيدي. هل هناك شيء آخر تحتاجني لمساعدتك فيه؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
        elif "من أين أنت" in result['text']:
            os.system('cls')
            stream.stop_stream()
            print('المدير: ' + result['text'])
            speak('السيد محمود هو من أنشأني لذلك انا من البصرة. هل هناك شيء آخر تحتاجني لمساعدتك فيه؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
        elif "الوقت" in result['text']:
            os.system('cls')
            stream.stop_stream()
            print('المدير: ' + result['text'])
            time()
            speak('ماذا تريدني أن أفعل بعد ذلك يا سيدي؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
        elif "التاريخ" in result['text']:
            os.system('cls')
            stream.stop_stream()
            print('المدير: ' + result['text'])
            date()
            speak('ماذا تريدني أن أفعل بعد ذلك يا سيدي؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
        elif "موسيقى" in result['text']:
            os.system('cls')
            stream.stop_stream()
            print('المدير: ' + result['text'])
```

```
            speak('هذه بعض الموسيقى لك، أتمنى أن تستمتع بها')
            os.startfile('chill songs.mp3')
            speak('تم إيقاف المساعد مؤقتًا. يمكنك النقر علي لاحقًا والضغط على أي مفتاح على لوحة المفاتيح 
للاستئناف')
            wait()
            speak('ماذا تريدني أن أفعل بعد ذلك يا سيدي؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
        elif "ضغط" in result['text']:
            os.system('cls')
            stream.stop_stream()
            print('استاذ محمود: ' + result['text'])
            speak('هذه بعض الموسيقى لك، أتمنى أن تستمتع')
            os.startfile('chill songs.mp3')
            speak('المساعدة متوقفة. يمكنك النقر علي والضغط على أي مفتاح على لوحة المفاتيح لاستئناف')
            wait()
            speak('ماذا ترغب في أن أفعل لك، يا استاذ محمود؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
        elif "لعبة" in result['text']:
            os.system('cls')
            stream.stop_stream()
            print('استاذ محمود: ' + result['text'])
            speak('هذه لعبة صغيرة لك، أتمنى أن تستمتع')
            os.startfile('ball.exe')
            speak('المساعدة متوقفة. يمكنك النقر علي والضغط على أي مفتاح على لوحة المفاتيح لاستئناف')
            wait()
            speak('ماذا ترغب في أن أفعل لك، يا استاذ محمود؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
        elif "استرخاء" in result['text']:
            os.system('cls')
            stream.stop_stream()
            print('استاذ محمود: ' + result['text'])
            speak('هذه بعض الموسيقى لك، أتمنى أن تستمتع')
            os.startfile('chill songs.mp3')
            speak('المساعدة متوقفة. يمكنك النقر علي والضغط على أي مفتاح على لوحة المفاتيح لاستئناف')
            wait()
            speak('ماذا ترغب في أن أفعل لك، يا استاذ محمود؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
```

```
            print('')
        elif "مساعده" in result['text']:
            os.system('cls')
            stream.stop_stream()
            speak('هذا هو ملف التعليمات')
            os.startfile('instruction.docx')
            speak('المساعده متوقفة. يمكنك النقر علي والضغط على أي مفتاح على لوحة المفاتيح لاستئنافي')
            wait()
            speak('ماذا ترغب في أن أفعل لك، يا استاذ محمود؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
        elif "تعليمات" in result['text']:
            os.system('cls')
            stream.stop_stream()
            print('استاذ محمود: ' + result['text'])
            speak('هذا هو ملف التعليمات')
            os.startfile('instruction.docx')
            speak('المساعده متوقفة. يمكنك النقر علي والضغط على أي مفتاح على لوحة المفاتيح لاستئنافي')
            wait()
            speak('ماذا ترغب في أن أفعل لك، يا استاذ محمود؟')
            stream.start_stream()
            print('')
            print('أستمع ...')
            print('')
```

Figure (3-37): offline voice assistant.

The Vision (eyes)

The camera will insert an image and this image will be analyzed and see if it contains:

- Recognizing people by their faces and knowing their names.
- Recognizing their facial features and classifying their feelings through them.
- Recognize hand signals, and each signal represents a meaning that the robot understands.
- Getting to know various things from our real lives, and here we mean almost everything.

People Recognizer

we used the HOG algorithm to recognize people's faces without extensive training. we will explain how this is done.

- Before running the program, we must have a database, which consists of images.
- When you run the program, it will take the images in the database, process them, and extract certain values that we will explain later.
- The program takes the frame from the camera installed in the robot's eye.
- The code will perform some manipulations on the captured image, as it did previously on database images.
- He will then compare the results of the photos taken from the camera with those in the database to know whether the person in

the photo is himself or someone else.

What is the HOG algorithm and what are the processors mentioned above?

The HOG (Histogram of Oriented Gradients) algorithm is a feature extraction technique used in the field of computer vision to analyze and recognize objects in images. It was first introduced by Navindia and Dalala in 2005 and since then it has become a popular and useful technique in the field of object recognition.

1. *Splitting an image into cells*: The image is divided into a group of small, often overlapping, square cells. The cell size is determined based on the size of the object we wish to recognize. For example, the cell size could be 8x8 pixels.
2. *Calculating horizontal and vertical gradients*: A technique such as the Sobel technique is used to calculate the horizontal and vertical gradients in the image. This is done by calculating changes in brightness values between neighboring pixels. This results in highlighting important edges and features in the image.
3. *Creating a histogram*: A histogram is created to distribute the gradients in each cell. The range of values is divided into several classes or angles that represent the directions of the gradients. For example, the range can be divided into 9 categories representing different angles (such as 0°, 20°, 40°, etc.).
4. *Grouping of cells*: Adjacent cells are grouped together to form blocks. A block is a group of adjacent cells. This step helps deal with local gradients and minimize the effect of small changes in illumination.
5. *Feature calculation*: Features are calculated for each block based on the histograms in the shared cells. Information from different cells is combined to create a single feature that represents the block.
6. *Feature calculation*: Features are calculated for each block by merging the histograms of the cells in the block. Technologies such as artificial nerve are often used to improve performance and reduce dimensions.
7. *Object classification*: The extracted features are used for classification and recognition of objects in the image. Classifiers such as General Support Machine (SVM) or Neural Networks can be used for this purpose.

This algorithm is used in a variety of computer vision applications, including object recognition, edge detection, tracking, classification, and others. It can be applied to binary and color images.

The HOG algorithm is effective in extracting important features from images, and is characterized by its implementation simplicity and ability to handle changes in illumination and distortion. However, you may face some challenges in dealing with large changes in size, position, and diversity in target organisms.

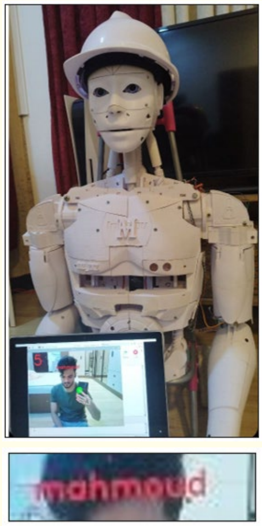


Figure (3-38): Human recognition feature.

Hand Signals Recognizer

Here we used the MediaPipe library to recognize and deal with hand gestures. The explanation below explains how this library works.

In MediaPipe library, the algorithm used for hand recognition and tracking is called "Hand Tracking." It utilizes a neural network model known as "Hand Landmark Model." The network architecture is based on deep learning and is trained on a large dataset of hand images and corresponding landmark positions. When running Hand Tracking in MediaPipe, consecutive video frames are provided to the algorithm. The algorithm analyzes these frames to identify the locations of distinctive landmarks on the hand, such as fingertips and hand joints. The hand is represented by a network of these landmark points connected in a skeletal structure. After identifying the landmark positions, they can be used for various purposes such as motion control, gesture recognition, interacting with augmented reality, or other applications that rely on hand.

Figure (3-39): Hand Signals Recognizer.

We took the points and set conditions on them. Each point has a location in the x and y axes. An example of a program code dedicated to recognizing one hand gesture only.

```python
import cv2
import mediapipe as mp

cap =  cv2.VideoCapture(0)
_, frames = cap.read()
rows, cols, _ = frames.shape

mpHands = mp.solutions.hands
hands = mpHands.Hands()
mpDraw = mp.solutions.drawing_utils
tipIds = [4, 8, 12, 16, 20]


while True:
    ret, frames = cap.read()

    if not ret:
        break

    results = hands.process(imgRGB)

    lmList = []

    if results.multi_hand_landmarks:
      for handLms in results.multi_hand_landmarks:
        #علمود نحصل النقاط على الايد
        for id, lm in enumerate(handLms.landmark):
            # ارتفاعوعرض والسنتر مالات الصورة
            h, w, c = frames.shape
            cx, cy = int(lm.x * w), int(lm.y * h)
            lmList.append([id, cx, cy])
            mpDraw.draw_landmarks(frames, handLms, mpHands.HAND_CONNECTIONS)


            if id == 8:
               cv2.circle(frames, (cx, cy), 20, (0, 255, 0), cv2.FILLED)

            if len(lmList) == 21:
               fingers = []

               if lmList[4][1] > lmList[6][1] and cccc==1 and lmList[4][1] < lmList[1][1] and lmLis
                  print("tmm")
                  kit.servo[2].angle = 90
                  cccc=2
                  os.system("mpg321 "+name+".mp3")
                  kit.servo[2].angle = 45
                  cv2.putText(frames, "okkkk", (60, 100), cv2.FONT_HERSHEY_SIMPLEX,3, (0, 0, 255),
               if lmList[tipIds[0]][1] < lmList[tipIds[0] - 2][1]:
                  fingers.append(1)
               else:
                  fingers.append(0)

               for tip in range(1, 5):
                  if lmList[tipIds[tip]][2] < lmList[tipIds[tip] - 2][2]:
                     fingers.append(1)
                  else:
                     fingers.append(0)

               totalFingers = fingers.count(1)
               print(totalFingers)
               cv2.putText(frames, f'{totalFingers}', (40, 80), cv2.FONT_HERSHEY_SIMPLEX,3, (0, 0,


    cv2.imshow('winnie', frames)

    k = cv2.waitKey(5)
    if ord('q') == k:
       break

cv2.destroyAllWindows()
cap.release()
```

Restricted Mode ⊗ 0 ⚠ 12 0 Ln 47, Col 102 Spaces: 4 UTF-8 CRLF Python

Figure (3-40): Program code of Hand Signals Recognizer.

The object recognition feature can recognize everything different in our lives through neurons programmed to classify objects in their various forms, depending on several factors.

And the code designated for object detection (the software code that classifies objects) is more than 93 thousand lines of code, so when the code passes through it, it is delayed, so I have a problem.

The duration of one cycle is approximately two seconds, and this is a problem. Therefore, when I use this part, I use Multitask Processing to use all the processors.

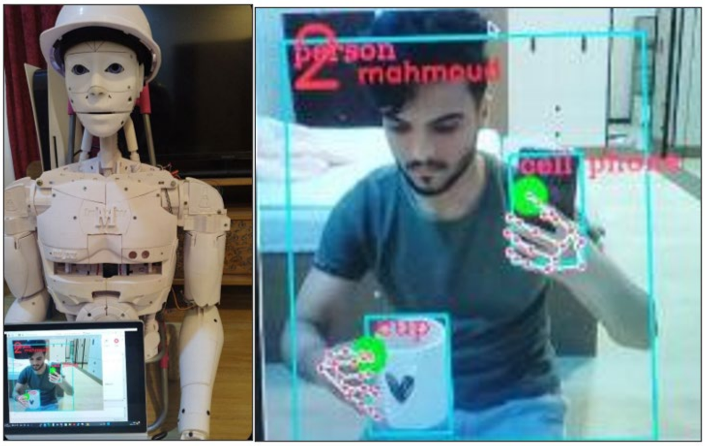


Figure (3-41): Recognizing things.

Below we will list the programming code dedicated to identifying and classifying objects. The file called Frozen Franchise Graph is a file consisting of 93,000 lines that contains the programming codes to compare different objects.

```
import cv2

thres = 0.45 # Threshold to detect object

cap = cv2.VideoCapture(1)
cap.set(3,1280)
cap.set(4,720)
cap.set(10,70)

classNames= []
classFile = 'coco.names'

with open(classFile,'rt') as f:
classNames = f.read().rstrip('n').split('n')
configPath = 'ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt'
weightsPath = 'frozen_inference_graph.pb'

net = cv2.dnn_DetectionModel(weightsPath,configPath)
net.setInputSize(320,320)
net.setInputScale(1.0/ 127.5)
net.setInputMean((127.5, 127.5, 127.5))
net.setInputSwapRB(True)
```

```
while True:
   success,img = cap.read()
   classIds, confs, bbox = net.detect(img,confThreshold=thres)
   print(classIds,bbox)
   if len(classIds) != 0:

     for classId, confidence,box in
zip(classIds.flatten(),confs.flatten(),bbox):
            cv2.rectangle(img,box,color=(0,255,0),thickness=2)
            cv2.putText(img,classNames[classId-
1].upper(),(box[0]+10,box[1]+30),
            cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)
            cv2.putText(img,str(round(confidence*100,2)),(box[0]+200,box[1]
+30),
            cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)

    cv2.imshow("Output",img)
    cv2.waitKey(1)
```

Figure (3-42): Programming code of Recognizing things.

Recognizing feelings

The feature of recognizing emotions took me a lot of time and I suffered a lot of errors with it. I learned many interesting methods and we will detail one of them. Recognizing emotions using mediapipe. This method is exactly like the one I used to recognize hand signals, and it is not a very smart and complicated method, even if it comes. A person with different features may not complete. The other method, which is using Keras and TensorFlow, here will be either relying on TensorFlow completely, and this is a difficult and lengthy method that we did and there is an explanation for it on YouTube or the other method is by using Keras, we train a neural model to detect these feelings by comparing them to a database. It already exists. We will put the code for the neurons, their training, and the code for calling them below.

```
• from tensorflow.keras.models import load_model
• from tensorflow.keras.preprocessing import image
• import numpy as np
• import cv2
• cap = cv2.VideoCapture(1)
• model = load_model('emotion_detection_final.h5') # تحميل النموذج
• emotions = ['Angry', 'Fear', 'Happy', 'Sad', 'Surprise', 'Neutral']
  #المشاعر
• while True:
•     success,img = cap.read()
•     predictions = model.predict(img) # تصنيف الصورة باستخدام النموذج
•     emotion_label = emotions[np.argmax(predictions)]
•     print('Emotion:', emotion_label)
•     cv2.rectangle(img,box,color=(0,255,0),thickness=2)
•     cv2.putText(img, emotion_label, (40, 80), (0, 0, 255), 6)
•     cv2.imshow('winnie', img)
•     k = cv2.waitKey(5)
•     if ord('q') == k:
•        break
• cv2.destroyAllWindows()
• cap.release()
```

Figure (3-43): Programming code to Summon the emotion code.

This code is intended to call the model that I trained, and I will put pictures of the model on the next page.

```
import os#'علمود تستدعي من النضام وتتعامل وياه'
from tensorflow.keras.preprocessing.image import ImageDataGenerator#منشئ الداتا مالات الصور
from tensorflow.keras.models import Sequential# علمود تسوي هواي لير
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
#علمود تحول الصوره المعقده الي بيها هواي ابعاد الى صوره بسيطة مسطحة تسوي عليها عمليات تدوير وتعديل

from tensorflow.keras.optimizers import Adam# هذا الي تعلمة بالعطلة محسن عملية التدريب مالت ادم

from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping
#وهذا يحفض افضل نموذج من عملية التدريب ويوقفها اذا شاف ماكو تحسين بالتدريب معناها انتهى التدريب

# خلي مسار الفولدر مالت البيانات
data_dir = '#المسار بالحاسبة القديمة '

#حتى تحدد حجم الصور والمعلومات
img_width, img_height = 48, 48
batch_size = 32
epochs = 50
num_classes = 7

#علمود تكون وتحدد مولد البيانات الي يولدلنا الداتا مالتنا
datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=30,
    shear_range=0.3,
    zoom_range=0.3,
    width_shift_range=0.4,
    height_shift_range=0.4,
    horizontal_flip=True,
    validation_split=0.2)

# علمود تحمل الداتا من الفولدر
train_generator = datagen.flow_from_directory(
    data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='categorical',
    subset='training')

validation_generator = datagen.flow_from_directory(
    data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='categorical',
    subset='validation')

# هنا راح ننشئ النموذج مالت الشبكة العصبية
model = Sequential()
```

```
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(img_width, img_height,
3)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(1024, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))

# وهنا راح نسوي تكوين للتدريب حتى ندربة
model.compile(optimizer=Adam(lr=0.0001, decay=1e-6),
              loss='categorical_crossentropy',
              metrics=['accuracy'])
# نحدد وضائف الردود الفعلية مالت التدريب (callbacks)
checkpoint = ModelCheckpoint('path/to/trained_model.h5',
                             monitor='val_loss',
                             mode='min',
                             save_best_only=True,
                             verbose=1)
early_stop = EarlyStopping(monitor='val_loss',
                           min_delta=0,
                           patience=3,
                           verbose=1,
                           restore_best_weights=True)
callbacks = [checkpoint, early_stop]
# هنا ندرب النموذج مالتنا هذا كود التدريب
history = model.fit(
    train_generator,
    steps_per_epoch=train_generator.samples // batch_size,
    epochs=epochs,
    callbacks=callbacks,
    validation_data=validation_generator,
    validation_steps=validation_generator.samples // batch_size)

# هذا النموذج حفضتة بهذا الاسم لازم استعملة بالكود مالتي الاصلي ابو الالف سطر
model.save('path/to/trained_model.h5')
```

Figure (3-44): Programming code of Create and train emotion code.

Chapter 4: Result

In this chapter, we will talk about the practical results of the robot, from which we will mention the outputs that we did not mention in the previous chapters.

Output for robot

The robot can do many, many things because it has inputs and processes them. All that remains is for it to do what we want it to do. We made the robot perform some outputs in response to the inputs like humans, including:

- He moves his eyes and head towards the person standing in front of him.
- He can greet the person standing in front of him.
- He can speak Arabic and answer some things.
- He may move his hand in different ways in response to input.
- It speaks to a person if he recognizes his feelings or if he recognizes a hand signal or a specific object.

Head movement towards humans

This feature is one of the output properties, not like the previous ones (the previous ones are considered inputs).

Here we benefited from the feature - People Recognizer - we took the location of the face in relation to the screen in the x & y axes and told it, when the person moves in a specific direction in the neck, head, and eyes to place the person in the middle of the screen.

Thus, the robot will always look at the person in front of it, and we can add a preference for certain people when it recognizes them, focusing on them rather than others.

```python
import cv2
from adafruit_servokit import ServoKit
import numpy as np
from cvzone.FaceDetectionModule import FaceDetec
kit = ServoKit(channels=16)
cap =  cv2.VideoCapture(0)
detector = FaceDetector()
positionX = 90
positionY = 90
kit.servo[0].angle = positionX
kit.servo[1].angle = positionY
kit.servo[3].angle = 90
_, frames = cap.read()
rows, cols, _ = frames.shape
while True:
    ret, frames = cap.read()
    if not ret:
        break
    frames = cv2.flip(frames, -1)
    frames = cv2.flip(frames, 1)
    frames = cv2.flip(frames, 1)
    frames, bboxs = detector.findFaces(frames)
    if bboxs:
        center = bboxs[0]["center"]
        x_medium = center[0]
        y_medium = center[1]
        a = x_medium // 62
        b = y_medium // 33
        print(a, b)
        if 5 < a < 20:
            positionX += 4.5
        elif 1 < a < 4:
            positionX -= 4.5
        print(positionX)
        if 5 < b < 20:
            positionY += 2.5
        elif 1 < b < 4:
            positionY -= 2.5
        print(positionY)
        if positionX < 34:
            kit.servo[3].angle = 75
        if positionX > 145:
            kit.servo[3].angle = 105
        if positionX < 15 or positionX > 165:
            positionX = 60
        if positionY < 60 or positionY > 140:
            positionY = 100
        kit.servo[0].angle = positionX
        kit.servo[1].angle = positionY
    cv2.imshow('winnie', frames)
    k = cv2.waitKey(5)
    if ord('q') == k:
        break
cv2.destroyAllWindows()
cap.release()
```

Figure (4-1): Programming code of Head movement towards humans.

Speaking

The robot can speak according to the programming code that I attached in Topic 3.4.2. we cannot put videos in a paper search, but we have many videos of the robot speaking.

We put commands in the code, the result of which is when it sees a person it knows, says hello to it and puts its name in the speech, and when He says anything that puts his name in the words to show that he is talking to him.

Recognition output

When the sense of sight recognizes feelings, for example, when it detects that Mahmoud is happy, the robot will say, “What is this beautiful smile?” Or when it sees Mahmoud sad, it will tell him a joke and tell him, “Don’t be sad, there is nothing worth being sad about,” and so on.

Figure (4-2): Programming code of Recognizing feelings.

As for sign language, it was used. For different things, and we can develop them ,we have programmed several signals, and each one will be understood by the robot and act on it. For example, one of the signals indicates that the robot should stop talking, or if you indicate to it to turn its head to the right, it will turn its head at a certain angle and look.

Figure (4-3): Recognition of Hand and gestures.

The object recognition features. The robot is able to know something and its price. For example, if it recognizes a specific pen, it will say, “This pen costs a dollar,” and so on. Another example is that we told the robot, “Give me the phone,” and it will turn its head until it finds the phone. Then it will move its arm with all its motors, and then it will pick up the phone, lift it, and turn its head again until it finds me. Then he would move the phone towards me and give it to me.

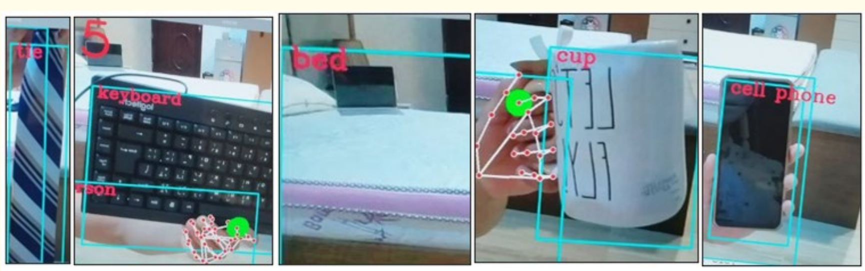
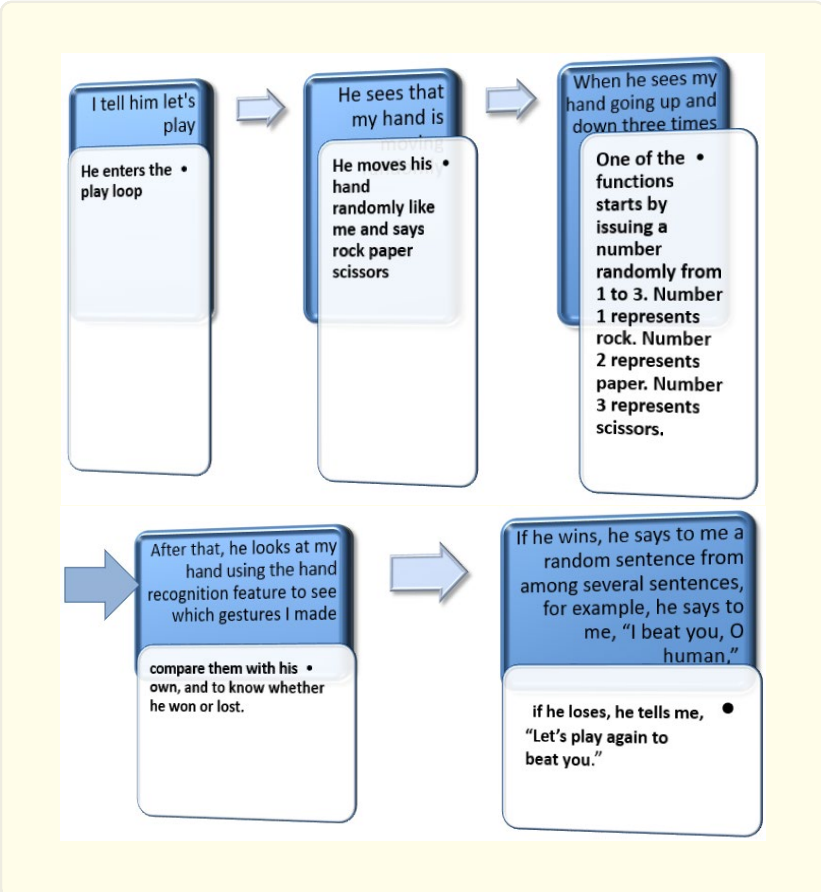


Figure (4-4): Recognition on various things Here I brought random things as you can see in the pictures and they were all recognition.

Arm response

The robot arm has joints just like human joints, and each joint has motors that move the arm just like the human arm moves. Therefore, this robot arm can do everything that a human arm does we have a video of the robot playing rock-paper-scissors with me and we will explain the code via a flowchart:



This is a simple example of what the arm can do. It can also imitate human movement and it can pick up things, but not in a professional manner. we will mention later what we mean by picking things up professionally. In the end, the robot arm can do what the human arm does.

Conclusion

Human beings have been forced to directly interact with infectious disease patients and undertake challenging tasks that consume time and effort. Therefore, we created a fully implemented humanoid robot equipped with human-like senses to assist humans in these tasks.

Our project involves a humanoid robot that we have developed entirely, which includes senses similar to humans. For example:

- It has the ability to recognize people by their faces and call them by their names, interacting with them accordingly.
- It can also recognize objects and emotions, enabling it to identify various things in our real-life and communicate with humans based on their feelings.

- It possesses the sense of hearing, allowing it to listen to people and speak to them in the Arabic language, without the need for internet connectivity. It can answer their questions or perform specific actions, such as fetching items for them.
- It has the ability to move its eyes, head, and all its joints, and it can face people, speak with them, and interact with its hands just like ordinary humans.

Because of these capabilities, we can utilize it in medical, educational, military, and industrial fields. For example, we can use it in the field of education as a teacher who delivers lectures, answers students' questions, and creates a complete lecture atmosphere. We used a Raspberry Pi controller, which serves as a small computer in the robot's head, and programmed it using the Python language to operate with artificial intelligence similar to that found in humans. With the presence of large servo motors, it is capable of moving each joint just like humans. When the material resources are available and some time is available, we will develop many of the characteristics of the robot, including: The structure of the robot if you print it from ABS. Instead of the PLA material, it will be able to carry heavy objects because we used motors for the arm with a torque of 80 kilograms per centimeter. We also want to add to it the sense of touch, which is achieved through pressure sensors placed in the hand, and I also want to add to it the feature of sweating when exposed to high temperature or when stressed, Also, we can add to it a sense of smell through various gas sensors. Also, with the passage of time, the robot will have many outputs, the most important of which is to pick up objects depending on their location, meaning that it can grasp something accurately and place it wherever it wants without affecting it.

References

1. FK Handhal and AT Rashid. "A prototype Design for Three Phase Smart Energy Meter". Second Al-Sadiq International Conference on Multidisciplinary in IT and Communication Science and Applications (2017).
2. FK Handhal and AT Rashid. "Design and building a single-phase smart energy meter using Arduino and RF communication system". Proc. of the 3rd Int. Sci. Conf., (14-15/march/2018).
3. AL-Furati I, Rashid AT and Al-Ibadi A. "IR sensors array for robots' localization using K means clustering algorithm". International Journal of Simulation Systems, Science & Technology 12.1 (2019).
4. FK Handhal and AT Rashid. "Load Balancing in Distribution System Using Heuristic Search Algorithm". International Conference on Advances in Sustainable Engineering and Applications (ICASEA), Wasit University, Kut, Iraq (2018): 48-53.
5. AT Rashid, FK Handhal and OT Rashid. "Wireless Control system for Three Phase Loads Balancing in Distribution Networks". International Journal of Computer Applications (0975 - 8887) 182.7 (2018): 38-46.
6. Al-Forati ISA and Rashid A. "Design and Implementation an Indoor Robot Localization System Using Minimum Bounded Circle Algorithm". 8th International Conference on Modeling Simulation and Applied Optimization (ICMSAO) (2019): 1-6.
7. AL-Furati IS and Rashid AT. "An Algorithm for Indoor Robot Path Planning Using Low-Cost IR Sensor Array System". In 2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE) (2020): 1-6.
8. R Panna, R Thesrumluk and C Chantrapornchai. "Development of Energy Saving Smart Home Prototype". International Journal of Smart Home 7.1 (2013): 47-66.
9. Al-Furati Israa S and Rashid AT. "Shortest Distance Orientation Algorithm for Robot Path Planning using Low-Cost IR Sensor System". In 2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE) (2020): 1-6.
10. Padmanabh K., et al. "iSense: a wireless sensor network based conference room management system". In Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings (2009): 37-42.
11. AL-Forati ISA and Rashid AT. "Robotics Path Planning Algorithms using Low-Cost IR Sensor". Iraqi Journal for Electrical and Electronic Engineering, 3RD (2020).
12. Lu J., et al. "The smart thermostat: using occupancy sensors to save energy in homes". In Proceedings of the 8th ACM conference on embedded networked sensor systems (2010): 211-224.
13. AL-Furati IS and Rashid AT. "design and implementation of an indoor robot positioning system using LED array and LDR sensor". Journal of Engineering Science and Technology 16.2 (2021).

14. W Kleiminger, F Mattern and S Santini. "Predicting household occupancy for smart heating control: A comparative performance analysis of state-of-the-art approaches". Energy and Buildings 85 (2014): 493-505.
15. Ye Z., et al. "Hardware-Based Emulator for Building Energy Cyber-Physical Control with Occupancy Sensing". In 17th International Conference on Information Technology-New Generations (ITNG 2020) (2020): 493-499. Springer, Cham.
16. S. Sabeeh and Israa S Al-Furati. "Issues and Research Fields of Medical Robotics: A Review". Iraqi Journal for Electrical and Electronic Engineering 19.2 (2023).
17. Zhang J., et al. "First-principles insights of electronic and optical properties of F-doped hexagonal boron nitride nanosheets for photo-catalytic water splitting". EPL (Europhysics Letters) 127.6 (2019): 67003.
18. Al-Furati Israa S. "Design and Implementation of a Control Water Quality System using IOT". PriMera Scientific Engineering 4.3 (2024): 13-24.
19. Mahlein J and Braun M. "A matrix converter without diode clamped over-voltage protection". In Proceedings IPEMC 2000. Third International Power Electronics and Motion Control Conference (IEEE Cat. No. 00EX435) 2 (2000): 817-822.
20. AL-Furati I, Rashid AT and Al-Ibadi A. "IR sensors array for robots' localization using K means clustering algorithm". International Journal of Simulation Systems, Science & Technology 12.1 (2019).
21. Adoghe AU and Odigwe IA. "Automatic Transfer and Controller System for Standby Power Generator". In Advanced Materials Research. Trans Tech Publications Ltd 62 (2009): 187-193.
22. S Sabeeh and Israa S Al-Furati. "Comparative Analysis of GA-PRM Algorithm Performance in Simulation and Real-World Robotics Applications". Misan Journal of Engineering Sciences 2.2 (2023).
23. Scotta IC., et al. "Over-voltage protection for Pico-Hydro generation using PV microinverters". In Ibero-American Congress of Smart Cities (2019): 25-37.
24. AL-Furati IS and Rashid AT. "Performance Comparison of Three Types of Sensor Matrices for Indoor Multi-Robot Localization". International Journal of Computer Applications 181.26 (2018): 22-29.
25. Sarah Sabeeh and Israa S. Al-Furati. "Efficient Path Planning in Medical Environments: Integrating Genetic Algorithm and Probabilistic Roadmap (GA-PRM) for Autonomous Robotics". Iraqi Journal for Electrical and Electronic Engineering.
26. Issa BA and Al-Furati IS. "High Precision Binary Coded Decimal (BCD) unit for 128-bit addition". In 2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE) (2020): 1-5.
27. Al Assfor FK, Al-Furati IS and Rashed AT. "Vedic-Based Squarers with High Performance". Indonesian Journal of Electrical Engineering and Informatics (IJEEI) 9.1 (2021): 163-172.
28. AL-Forati ISA and Rashid AT. "Practical Implementation of an Indoor Robot Localization and Identification System using an Array of Anchor Nodes". Iraqi Journal for Electrical & Electronic Engineering 16.1 (2020).