A

PROJECT REPORT

on

ROBOTIC ARM

Submitted in partial fulfilment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

Session: - Jan-June 2023

Under Guidance of
YOGENDRA SINGH SOLANKI
Professor
ECE

Submitted by
Roheetang sharma (19etcec014)
Yuvraj singh gaur (19etcec019)
Rounak bamniya (19etcec015)
Yogesh rawal (19etcec018)
Lakshdeep hiran (19etcec004)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR-313001

MAY - 2023

A

PROJECT REPORT

on

ROBOTIC ARM

Submitted in partial fulfilment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

Session: - Jan-June 2023

Under Guidance of
YOGENDRA SINGH SOLANKI
PROFESSOR
ECE

Submitted by
Roheetang Sharma (19etcec014)
Yuvraj Singh Gaur (19etcec019)
Rounak Bamniya (19etcec015)
Yogesh Rawal (19etcec018)
Lakshdeep Hiran (19etcec004)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR-313001

MAY - 2023

TECHNO INDIA NJR
INSTITUTE OF TECHNOLOGY

Department of Electronics and Communication Engineering
Techno India NJR Institute of Technology, Udaipur-313001

Certificate

This is to certify that project work titled "ROBOTIC ARM" by Roheetang sharma and team, was successfully carried out in the Department of Electronics and Communication Engineering, TINJRIT and the report is approved for submission in the partial fulfilment of the requirements for award of degree of Bachelor of Technology in Electronics and Communication.

YOGENDRA SINGH SOLANKI	Dr. VIVEK JAIN
PROFESSOR	HOD
Dept. of E.C.E TINJRIT, Udaipur	Dept. of E.C.E TINJRIT, Udaipur
Date......................	Date......................

#Certificates of candidates have been attached with their respective reports#

Department of Electronics and Communication Engineering
Techno India NJR Institute of Technology, Udaipur-313001

Examiner Certificate

This is to certify that the following student

Roheetang Sharma

Yuvraj singh gaur

Rounak bamniya

Lakshadeep hiran

Yogesh rawal

of final year B.Tech. (Electronics & Communication Engineering), was examined for the project work titled

"Robotic Arm"

during the academic year 2022 – 2023 at Techno India NJR Institute of Technology, Udaipur

Remarks:

Date:

Signature	Signature
(**Internal Examiner**)	(**External Examiner**)
Name: - ………………………	Name: - ………………………
Designation: - ………………..	Designation: - ………………..
Department: - ……………….	Department: - ……………….
Organization:- ………………	Organization:- ………………

Preface

In the vast field of robotics, one area that has captured the imagination of engineers and enthusiasts alike is the development of robotic arms. These remarkable mechanical marvels mimic the intricate movements of a human arm, providing a wide range of applications and possibilities in various industries. The project described in this documentation represents our journey into the realm of robotic arm development. It embodies countless hours of research, design, and experimentation by a team driven by the passion for exploring the frontiers of technology.

The purpose of this project is to create a functional and versatile robotic arm capable of performing precise movements and tasks. We sought to develop a system that could manipulate objects with dexterity, emulate human arm motions, and exhibit a high level of control and adaptability. Throughout the course of this project, we encountered a myriad of challenges and obstacles. From selecting the appropriate materials and components to designing the mechanical structure and implementing the control algorithms, each step demanded meticulous attention to detail and a deep understanding of robotics principles. However, it is through overcoming these hurdles that we have gained invaluable knowledge and expertise. The process of developing this robotic arm has not only expanded our technical skills but also fostered teamwork, problem-solving, and innovation.

We extend our gratitude to our mentors, advisors, and colleagues who have supported us throughout this endeavour. Their guidance, expertise, and encouragement have been instrumental in transforming our vision into reality.

With this preface, we invite you to embark on a journey through the intricacies of our robotic arm project and explore the possibilities that lie ahead.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR-313001

ACKNOWLEDGMENT

We take this opportunity to record our sincere thanks to all who helped us to successfully complete this work. Firstly, we are grateful to our **supervisor Yogendra sir** for his invaluable guidance and constant encouragement, support and most importantly for giving us the opportunity to carry out this work.

We would like to express our deepest sense of gratitude and humble regards to our,

Head of Department Dr. Vivek Jain for giving invariable encouragement in our endeavours and providing necessary facility for the same. Also, a sincere thanks to all faculty members of ECE, TINJRIT for their help in the project directly or indirectly.

Finally, I would like to thank my friends for their support and discussions that have proved very valuable for us. We are indebted to our parents for providing constant support, love and encouragement. We thank them for the sacrifices they made so that we could grow up in a learning environment. They have always stood by us in everything we have done, providing constant support, encouragement and love

Roheetang Sharma (19etcec014)
Yuvraj Singh Gaur (19etcec019)
Rounak Bamniya (19etcec015)
Lakshdeep Hiran (19etcec004)
Yogesh Rawal (19etcec018)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
TECHNO INDIA NJR INSTITUTE OF TECHNOLOGY, UDAIPUR-313001

CONTENTS

LIST OF FIGURES

List of Abbreviations

GND	Ground Pin of module
TXD	Transmit Serial data
RXD	Receive data serially
PAN	Personal Area Network
DH	Denavit-Hartenberg parameters

CHAPTER 1

INTRODUCION TO ROBOTIC ARMS

1.1 Introduction

A robotic arm is a mechanical device designed to mimic the structure and functionality of a human arm. It consists of multiple interconnected segments or links that are connected through joints, allowing it to move in a flexible and precise manner. These joints can be either rotary, enabling rotational motion, or linear, facilitating straight-line movement.

Robotic arms are commonly used in various industries, including manufacturing, healthcare, research, and space exploration. They are primarily utilized to perform repetitive, dangerous, or intricate tasks that would be challenging or unsafe for humans. The arms can be mounted on fixed structures or mobile platforms, depending on the specific application.

One of the key components of a robotic arm is the end effector, which is the device attached to the arm's terminal link. The end effector can vary depending on the task it needs to accomplish. It can be as simple as a gripper for picking up and manipulating objects, or it can be more complex, such as a welding torch, a cutting tool, or a camera for inspection purposes.

To control the movement of a robotic arm, various technologies are employed, including electric motors, hydraulic systems, or pneumatic actuators. These actuators receive signals from a computer or a control system that processes input commands and translates them into precise movements of the arm's joints.

In recent years, advancements in robotics have led to the development of more sophisticated robotic arms. Some of these arms incorporate sensors, such as force or tactile sensors, to provide feedback on the arm's interaction with the environment. This allows for improved precision and adaptability in performing tasks.

Furthermore, collaborative robotic arms, also known as cobots, have gained popularity. These arms are designed to work alongside humans in shared workspaces, enhancing productivity and safety. Cobots are equipped with advanced sensors and algorithms that enable them to detect human presence and adjust their movements accordingly, preventing collisions and ensuring collaboration.

In summary, robotic arms are versatile machines capable of performing a wide range of tasks. Their precision, flexibility, and ability to operate in hazardous environments make them indispensable in various industries, revolutionizing production processes and expanding the possibilities of automation.

1.2 Types of Robotic Arm

There are several types of robotic arms used in various industries and applications. Here are some common types:

1. Cartesian Robot: Also known as Gantry or Rectilinear robots, they move in three linear axes (X, Y, and Z) along rectangular coordinates. They are widely used in applications such as pick-and-place operations, material handling, and 3D printing.

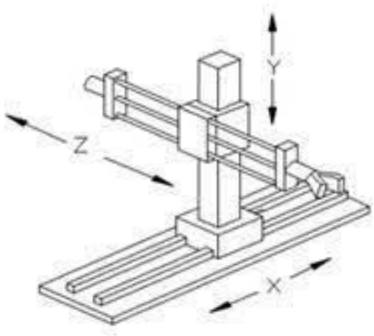

Fig.1 Cartesian Robot

2. SCARA Robot: SCARA (Selective Compliance Assembly Robot Arm) robots have four axes of motion. They are known for their fast and precise horizontal movements, making them suitable for assembly tasks, packaging, and electronics manufacturing.

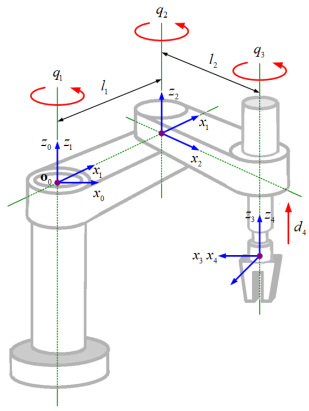

Fig.2 SCARA Robot

3. Articulated Robot: This type of robotic arm consists of multiple interconnected links and rotary joints, mimicking the structure of a human arm. It provides high flexibility and dexterity, making it ideal for tasks that require complex movements, such as welding, painting, and material handling in industrial settings.

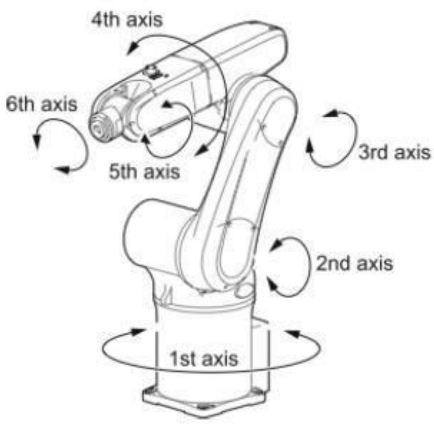

Fig.3 Articulated Robot

4. Delta Robot: Delta robots have a unique design with three arms connected to a common base. They are often used in high-speed pick-and-place operations in industries like food and beverage, electronics, and pharmaceuticals. Delta robots offer exceptional accuracy, speed, and agility.

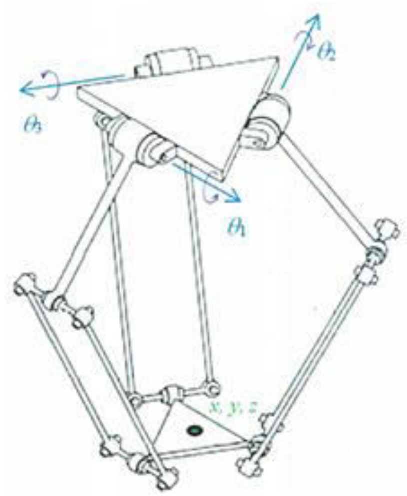

Fig.4 Delta Robot

5. Collaborative Robot (Cobot): Cobots are designed to work alongside humans, facilitating collaboration and interaction. They are equipped with sensors and safety features to ensure safe human-robot interaction. Cobots are used in various applications, including assembly, quality control, and packaging, where human assistance or cooperation is required.

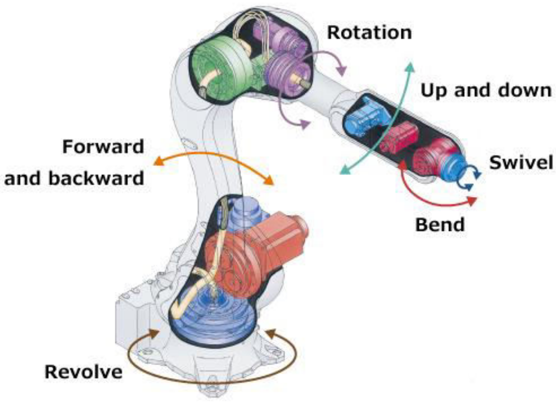

Fig.5 Collaborative Robot

6. Polar Robot: Polar robots, also known as spherical robots, consist of a rotary arm connected to a base that rotates around a fixed point. They are suitable for tasks that require a wide range of motion in a circular or spherical workspace, such as material handling, assembly, and machine tending.

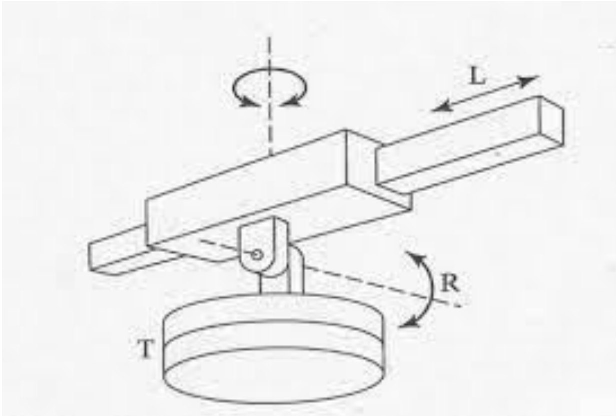

Fig.6 Polar Robot

These are just a few examples of robotic arm types, and there are many other specialized variations and configurations available, each tailored to specific tasks and industries.

1.3 Applications of Robotic Arm

Robotic arms have numerous applications across various industries. Some common applications of robotic arms include:

1. Manufacturing and Assembly: Robotic arms are extensively used in manufacturing processes, including assembly lines, to perform repetitive and precise tasks such as picking, placing, and assembling components. They can significantly increase efficiency, speed, and accuracy in industries like automotive, electronics, and consumer goods.

2. Material Handling: Robotic arms are employed for efficient material handling tasks, including loading and unloading goods, palletizing, sorting, and packaging. They can handle heavy loads, operate in confined spaces, and ensure consistent and reliable handling operations.

3. Welding and Fabrication: Robotic arms equipped with welding tools are utilized in industries such as automotive, aerospace, and construction for high-precision welding operations. They can perform complex welding tasks with consistency and repeatability, improving productivity and quality.

4. Inspection and Quality Control: Robotic arms with integrated cameras and sensors are used for quality inspection and control processes. They can

perform detailed inspections, measurements, and defect detection in products, ensuring consistent quality and reducing human error.

5. Medical and Healthcare: Robotic arms are employed in surgical procedures, providing surgeons with enhanced precision, stability, and dexterity. They are used for minimally invasive surgeries, such as laparoscopy, as well as assisting in rehabilitation and prosthetics.

6. Research and Laboratories: Robotic arms play a vital role in research and laboratory settings. They can automate experiments, handle hazardous materials, and perform repetitive tasks, freeing up researchers' time for more complex work. They are used in fields such as pharmaceuticals, biotechnology, and scientific research.

7. Space Exploration: Robotic arms have been extensively used in space exploration missions. They assist in deploying and repairing satellites, handling payloads, and performing maintenance tasks on spacecraft and space stations. Examples include the robotic arms on the International Space Station (ISS) and the Mars rovers.

8. Agriculture: Robotic arms are increasingly being utilized in agriculture for tasks such as harvesting crops, pruning, and sorting. They can enhance productivity, reduce labor costs, and enable precision agriculture techniques.

These are just a few examples of the diverse applications of robotic arms. As technology continues to advance, robotic arms are finding new and innovative uses in various industries, contributing to increased automation, efficiency, and safety.

CHAPTER 2

KINEMATICS OF ROBOTIC ARM

2.1 Forward & Inverse Kinematics in Robots

Forward and inverse kinematics are fundamental concepts in robotics that describe the relationship between a robot's joint angles and its end effector's position and orientation.

2.1.1 Forward Kinematics

Forward kinematics involves determining the position and orientation of a robot's end effector based on the known joint angles. It describes the transformation from the robot's joint space to its Cartesian space. In simpler terms, it answers the question, "Given the joint angles, where is the end effector located?"

To perform forward kinematics, the robot's geometric and kinematic parameters, such as the lengths of the robot's links and the types of joints, need to be known. By applying trigonometry and matrix transformations, the position and orientation of the end effector can be calculated using a sequence of coordinate transformations from the robot's base to the end effector.

2.1.2 Inverse Kinematics

Inverse kinematics, on the other hand, involves finding the joint angles required to position the robot's end effector at a specific location and orientation. It is the reverse process of forward kinematics. Inverse kinematics is particularly useful when the desired end effector position is specified, and the goal is to determine the joint angles needed to achieve that position.

Inverse kinematics problems can be more challenging to solve than forward kinematics, as they often involve solving complex equations and dealing with multiple possible solutions or constraints. Analytical methods, numerical methods, or optimization techniques can be employed to solve inverse kinematics problems.

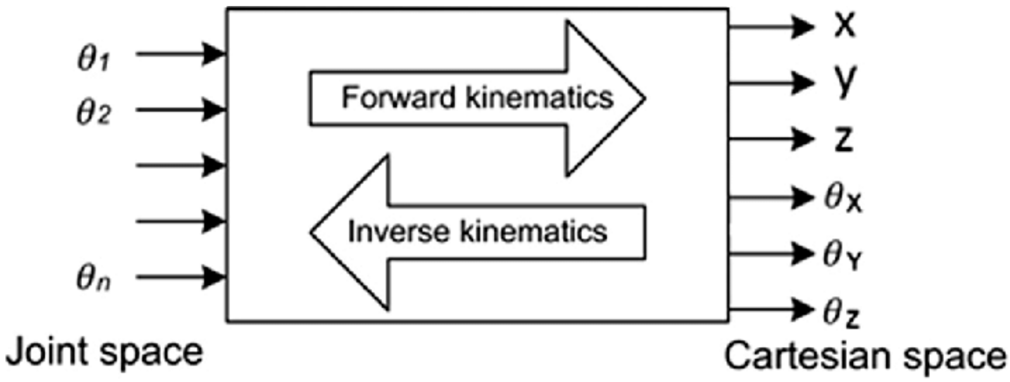

Fig.7 Forward & Inverse Kinematics in Robots

2.1.3 Applications:

Both forward and inverse kinematics are crucial in various robotic applications. Here are a few examples:

1. Path Planning: Forward kinematics helps in determining the robot's end effector position along a desired path. Inverse kinematics is used to calculate the joint angles required to follow a specified trajectory.

2. Control and Manipulation: Inverse kinematics allows for precise control of the robot's end effector. By specifying a desired position and orientation, inverse kinematics can compute the necessary joint angles for the robot to perform tasks like grasping, reaching, or manipulating objects.

3. Robot Simulation: Kinematics equations are employed in robot simulations to model and visualize the robot's motion. By applying forward kinematics, the simulated robot's movements can be accurately represented based on the specified joint angles.

4. Human-Robot Interaction: Inverse kinematics is useful in scenarios where a robot needs to mimic or respond to human movements. By tracking the position of a human's hand, for example, inverse kinematics can determine the joint angles for the robot's arm to replicate the motion.

Understanding and applying forward and inverse kinematics is vital for programming and controlling robotic arms. These concepts enable precise positioning and manipulation of the robot's end effector, allowing it to perform a wide range of tasks in various applications.

2.2 DH Parameters

DH parameters, also known as Denavit-Hartenberg parameters, are a widely used convention in robotics for modeling the kinematics of robotic arms. They provide a systematic way to describe the relationship between adjacent links and joints in a robotic arm.

The DH parameterization scheme consists of four parameters associated with each joint of the robot:

1. Link Length (a): It represents the distance between the current joint axis and the next joint axis along the common normal. It is measured along the previous joint's z-axis.

2. Link Twist (α): It denotes the angle of rotation between the previous joint's z-axis and the current joint's z-axis. It is measured about the previous joint's z-axis.

3. Link Offset (d): It signifies the distance between the previous joint's z-axis and the current joint's z-axis along the common normal. It is measured along the current joint's x-axis.

4. Joint Angle (θ): It represents the rotation angle between the previous link and the current link about the common normal. It is measured about the current joint's x-axis.

By assigning these parameters to each joint, a transformation matrix can be constructed for the link connecting two adjacent joints, expressing the spatial relationship between them. Multiplying these transformation matrices together yields the complete transformation from the robot's base to the end effector.

The DH parameters can vary depending on the convention and specific robotic arm configuration. Different conventions may have different notations or variations in the parameter order. It is essential to establish a consistent DH parameterization scheme based on the robot's geometry and joint configurations.

DH parameters provide a compact representation of a robotic arm's kinematics and are widely used in forward and inverse kinematics calculations, robot modelling, and simulation. They enable the efficient computation of joint angles and the transformation between different frames in the robot's kinematic chain, facilitating control, planning, and analysis of robot motion.

2.3 Joint Control & Trajectory Planning of a Robotic Arm

Joint control and trajectory planning are important aspects of controlling the motion of a robotic arm. Let's explore these concepts in more detail:

2.3.1 Joint Control

Joint control involves controlling the individual joints of a robotic arm to achieve desired positions or movements. It focuses on regulating the joint angles or velocities to achieve specific goals. There are several approaches to joint control, including:

1. Position Control: In position control, the goal is to accurately control the joint angles of the robotic arm to reach a desired position. This can be achieved using feedback control techniques such as Proportional-Integral-Derivative (PID) control or more advanced control algorithms.

2. Velocity Control: Velocity control involves regulating the joint velocities to achieve a desired speed or rate of movement. This type of control is useful for applications that require precise control of the arm's speed or when executing specific motion profiles.

3. Torque Control: Torque control focuses on controlling the joint torques or forces exerted by the robotic arm. It is essential for tasks that involve interacting with the environment or applying specific force profiles during manipulation tasks.

2.3.2 Trajectory Planning

Trajectory planning deals with generating a path or trajectory for the robotic arm to follow from its initial position to a desired target position. It involves determining the joint angles or velocities over time to achieve smooth and efficient motion. Trajectory planning can be approached in different ways, including:

1. Point-to-Point Trajectories: This type of trajectory planning involves planning a direct path from the start position to the target position. The trajectory can be generated by interpolating the joint angles or velocities between the two points.

2. Cartesian Space Trajectories: In Cartesian space trajectory planning, the desired path is defined in the task space (e.g., XYZ coordinates). The trajectory planner calculates the corresponding joint angles or velocities that will guide the end effector along the desired path.

3. Trajectory Generation Algorithms: Various algorithms can be employed to generate trajectories, such as cubic splines, quintic splines, or polynomial interpolation. These algorithms ensure smooth and continuous motion profiles by controlling the velocity, acceleration, and jerk along the trajectory.

4. Dynamic Constraints: Trajectory planning may need to consider dynamic constraints, such as joint limits, velocity limits, and torque limits, to ensure safe and feasible motion of the robotic arm.

By combining joint control and trajectory planning techniques, robotic arms can achieve accurate and controlled motion, enabling them to perform tasks with precision and efficiency. These concepts are foundational for programming robotic arms to execute complex movements, manipulate objects, or follow predefined paths in various applications.

CHAPTER 3

COMPONENTS USED

3.1 Arduino UNO

3.1.1 Introduction

The Arduino Uno is a popular microcontroller board based on the ATmega328P microcontroller. It is widely used for prototyping and DIY electronics projects due to its simplicity, versatility, and extensive community support. The Arduino Uno is a versatile and beginner-friendly microcontroller board that is widely used for a wide range of electronic projects, from simple LED blinking to complex robotics and home automation systems. Its ease of use, affordability, and large community make it a popular choice for hobbyists, students, and professionals alike.

3.1.2 Architecture of Arduino UNO

The Arduino Uno is a popular microcontroller board based on the ATmega328P microcontroller. It is widely used for prototyping and DIY electronics projects due to its simplicity, versatility, and extensive community support. Here are some key features and characteristics of the Arduino Uno:

1. Microcontroller: The Arduino Uno is powered by the ATmega328P microcontroller, which operates at 16 MHz and has 32KB of flash memory for program storage, 2KB of SRAM, and 1KB of EEPROM.

2. Digital and Analog I/O: The board has 14 digital input/output pins (of which 6 can be used as PWM outputs) and 6 analog input pins. These pins can be used to connect various sensors, actuators, and other electronic components.

3. Communication: The Arduino Uno supports serial communication through a USB connection, allowing it to communicate with a computer for programming

and serial data exchange. It also has I2C and SPI interfaces for communication with other devices.

4. Programming: The Arduino Uno is programmed using the Arduino Software (IDE), which provides a simplified programming environment based on C/C++. It has a user-friendly interface for writing, compiling, and uploading code to the board.

5. Power Supply: The Arduino Uno can be powered via USB from a computer or an external power source (7-12V DC). It has a built-in voltage regulator that provides a stable 5V output for powering external components.

6. Shields: The Arduino Uno is compatible with various expansion boards called "shields." These shields can be plugged onto the Arduino Uno to add additional functionality such as Wi-Fi, Bluetooth, motor control, LCD displays, and more.

7. Community and Resources: The Arduino Uno has a large and active community of users, which provides extensive documentation, tutorials, and examples for beginners and experienced users alike. This community support makes it easy to find help and guidance for your projects.

The Arduino Uno is a versatile and beginner-friendly microcontroller board that is widely used for a wide range of electronic projects, from simple LED blinking to complex robotics and home automation systems. Its ease of use, affordability, and large community make it a popular choice for hobbyists, students, and professionals alike.

3.1.3 Pinout diagram

Pinout diagram of Arduino UNO is given as follows:

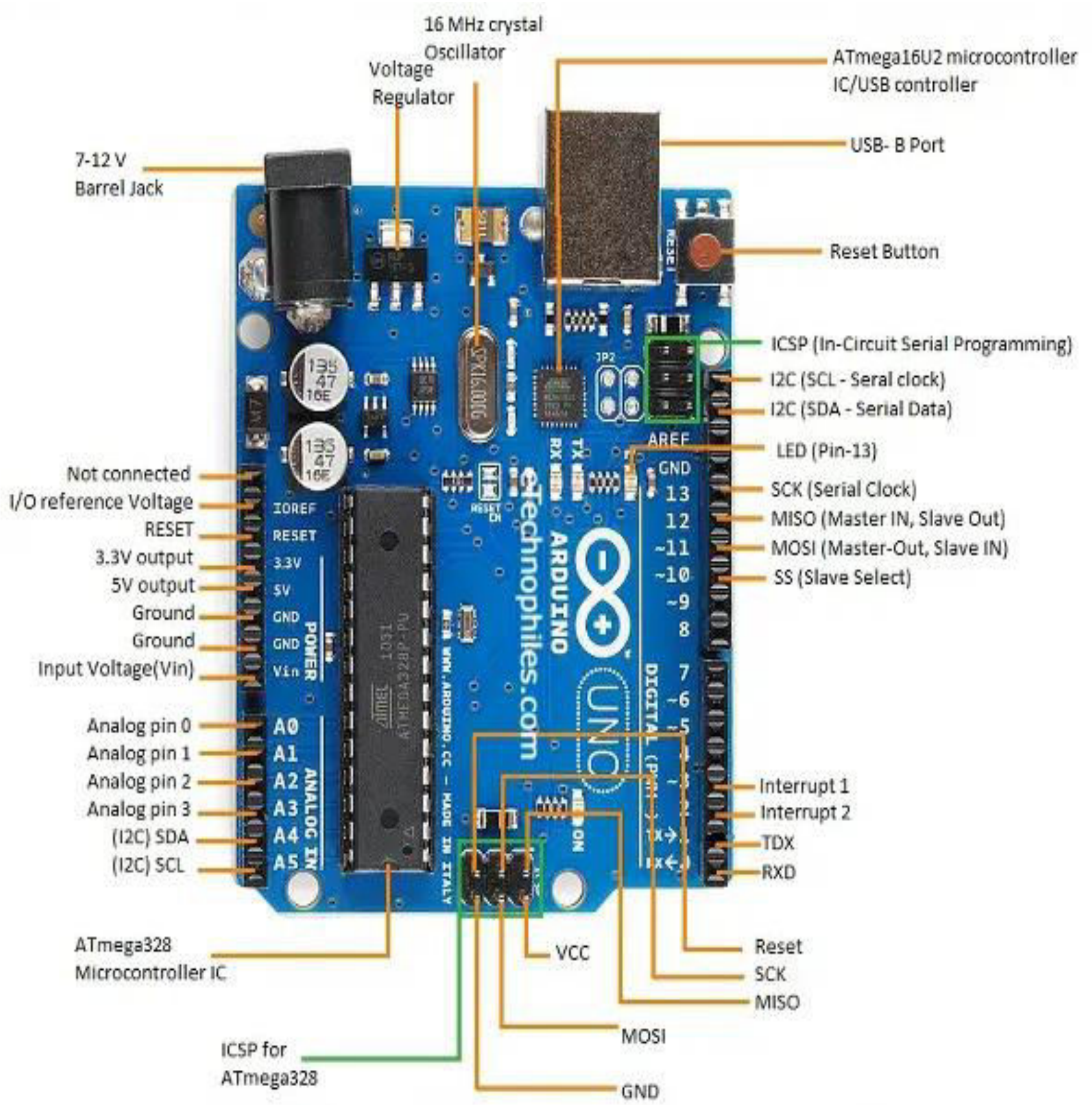

Fig.8 Pinout diagram of Arduino UNO

3.1.4 programming structure of Arduino

The structure of Arduino programming follows a specific format and consists of various essential elements. Here's an overview of the typical structure of Arduino programming:

1. Setup Function:

The Arduino program starts with a setup function. This function is executed only once when the Arduino board is powered on or reset. It is used to initialize variables, set pin modes, configure communication interfaces, and perform any necessary setup tasks.

```
void setup() {
  // Initialization code here
}
```

2. Loop Function:

After the setup function, the Arduino program enters the loop function. This function is continuously executed in a loop after the setup function completes. It forms the main body of the program and contains the code that defines the desired behavior of the Arduino.

```
void loop() {
  // Main code here
}
```

3. Variable Declaration:

Within the setup and loop functions, you can declare variables to store data and perform calculations. Variables can be of different types, such as integers (int), floating-point numbers (float), characters (char), and more. Declare variables based on your program's requirements.

```
int ledPin = 13; // Example variable declaration
```

4. Pin Mode Configuration:

If your program involves interacting with digital pins, you may need to set the pin modes in the setup function. The pin mode determines whether a specific pin is used as an input or output.

```
pinMode(ledPin, OUTPUT); // Example pin mode configuration
```

5. Reading Inputs and Writing Outputs:

Arduino programs often involve reading sensor inputs or controlling actuators. You can use digitalRead() and analogRead() functions to read the state of digital and analog pins, respectively. Similarly, you can use digitalWrite() and analogWrite() functions to set the output state of pins.

```
int sensorValue = analogRead(A0); // Example reading from an analog pin
digitalWrite(ledPin, HIGH); // Example writing to a digital pin
```

6. Control Structures:

Arduino programming supports control structures, such as if-else statements and loops, to perform conditional operations and repeated tasks. These structures allow you to make decisions and iterate through code blocks based on specific conditions.

```
if (sensorValue > 500) {
  // Code to execute if the condition is true
} else {
  // Code to execute if the condition is false
}

for (int i = 0; i < 10; i++) {
  // Code to repeat 10 times
}
```

7. Function Definitions:

You can define your own functions to encapsulate a specific task or a sequence of instructions. Function definitions can be placed outside the setup and loop functions, and they can be called from within these functions or from other functions.

```
void myFunction() {
  // Code for the function
}
```

These are the basic elements that make up the structure of Arduino programming. By combining these elements, you can create Arduino programs to control and interact with various components and devices. Remember to follow the specific syntax and conventions of the Arduino programming language (based on C/C++) to ensure proper execution of your code.

3.2 Servo Motors

A servo motor is a type of motor that can rotate with great precision. Normally this type of motor consists of a control circuit that provides feedback on the current position of the motor shaft, this feedback allows the servo motors to rotate with great precision. If you want to rotate an object at some specific angles or distance, then you use a servo motor. It is just made up of a simple motor which runs through a servo mechanism. If motor is powered by a DC power supply then it is called DC servo motor, and if it is AC-powered motor then it is called AC servo motor. For this tutorial, we will be discussing only about the DC servo motor working. Apart from these major classifications, there are many other types of servo motors based on the type of gear arrangement and operating characteristics. A servo motor usually comes with a gear arrangement that allows us to get a very high torque servo motor in small and lightweight packages. Due to these features, they are being used in many applications like toy car, RC helicopters and planes, Robotics, etc.

Some basic concepts of servo motors is as follows:

1. Servo Mechanism: Servo motors operate based on a closed-loop control system known as a servo mechanism. The mechanism includes a position sensor (such as a potentiometer or encoder), a control circuitry, and a feedback loop. The position sensor provides feedback on the motor's current position, which is compared to the desired position. The control circuitry adjusts the motor's position based on this feedback to minimize any position error.

2. Pulse Width Modulation (PWM): Servo motors are controlled using PWM signals. A PWM signal consists of a series of pulses where the width (duration) of the pulses determines the desired position of the servo motor. By adjusting the pulse width, the motor's angular position can be controlled within its operating range.

3.2.1 Servo Motor Working Mechanism

It consists of three parts:

1. Controlled device
2. Output sensor
3. Feedback system

It is a closed-loop system where it uses a positive feedback system to control motion and the final position of the shaft. Here the device is controlled by a feedback signal generated by comparing output signal and reference input signal.

Here reference input signal is compared to the reference output signal and the third signal is produced by the feedback system. And this third signal acts as an input signal to the control the device. This signal is present as long as the feedback signal is generated or there is a difference between the reference input signal and reference output signal. So the main task of servomechanism is to maintain the output of a system at the desired value at presence of noises.

3.2.2 **Servo Motor Working Principle**

A servo consists of a Motor (DC or AC), a potentiometer, gear assembly, and a controlling circuit. First of all, we use gear assembly to reduce RPM and to increase torque of the motor. Say at initial position of servo motor shaft, the position of the potentiometer knob is such that there is no electrical signal generated at the output port of the potentiometer. Now an electrical signal is given to another input terminal of the error detector amplifier. Now the difference between these two signals, one comes from the potentiometer and another comes from other sources, will be processed in a feedback mechanism and output will be provided in terms of error signal. This error signal acts as the input for motor and motor starts rotating. Now motor shaft is connected with the potentiometer and as the motor rotates so the potentiometer and it will generate a signal. So as the potentiometer’s angular position changes, its output feedback signal changes. After sometime the position of potentiometer reaches at a position that the output of potentiometer is same as external signal provided. At this condition, there will be no output signal from the amplifier to the motor input as there is no difference between external applied signal and the signal generated at potentiometer, and in this situation motor stops rotating.

Interfacing Servo Motors with Microcontrollers:

Interfacing hobby Servo motors like s90 servo motor with MCU is very easy. Servos have three wires coming out of them. Out of which two will be used for Supply (positive and negative) and one will be used for the signal that is to be sent from the MCU. An MG995 Metal Gear Servo Motor which is most commonly used for RC cars humanoid bots etc.

3.2.3 Applications of servo motor

1. Robotics: Servo motors play a crucial role in robotics, particularly in robot arms and grippers. They enable precise control of joint angles, allowing robots to perform tasks such as picking and placing objects, assembly operations, and intricate movements.

2. RC Vehicles: Servo motors are extensively used in remote-controlled (RC) vehicles, including cars, boats, airplanes, and drones. They control steering mechanisms, throttle, and other actuations, providing precise control and maneuverability.

3. Camera Stabilization: Servo motors are utilized in camera stabilization systems, such as camera gimbals. They help maintain the camera's stability and counteract any unwanted movements, ensuring smooth and steady footage.

4. Automation: Servo motors are employed in various automation systems, including conveyor belts, CNC machines, and industrial machinery. They facilitate precise positioning and control of moving parts, enhancing efficiency and accuracy in manufacturing processes.

5. Robotics Education and Hobbyist Projects: Servo motors are widely used in educational settings and hobbyist projects to introduce basic concepts of robotics and motion control. They are accessible and easy to use, making them ideal for learning about programming, control systems, and mechanical linkages.

6. Domotics (Home Automation): Servo motors find applications in home automation systems for tasks such as opening and closing doors, controlling blinds or curtains, and adjusting robotic furniture. They enable automated and controlled movement in various household devices.

7. Animatronics: Servo motors are utilized in animatronic systems for creating lifelike movements in animatronic characters, puppets, and humanoid robots. They enable precise control of facial expressions, gestures, and body movements, enhancing the realism of these creations.

These are just a few examples of the vast range of applications where servo motors are employed. Their ability to provide accurate and controlled motion makes them an essential component in various fields, contributing to the advancement of robotics, automation, entertainment, and many other industries.

3.3 Bluetooth Technology

Bluetooth communication is a wireless communication technology that allows devices to exchange data over short distances. It uses radio waves in the frequency range of 2.4 to 2.485 GHz to establish a connection between devices.

Bluetooth is commonly used for connecting devices such as smartphones, tablets, laptops, headphones, speakers, smartwatches, and other peripherals. It enables the transfer of data, audio, and control signals between these devices without the need for physical cables.

3.3.1 Bluetooth Module HC-05

Introduction

- It is used for many applications like wireless headset, game controllers, wireless mouse, wireless keyboard, and many more consumer applications.
- It has range up to <100m which depends upon transmitter and receiver, atmosphere, geographic & urban conditions.

- It is IEEE 802.15.1 standardized protocol, through which one can build wireless Personal Area Network (PAN). It uses frequency-hopping spread spectrum (FHSS) radio technology to send data over air.
- It uses serial communication to communicate with devices. It communicates with microcontroller using serial port (USART).

HC-05 Bluetooth Module

- HC-05 is a Bluetooth module which is designed for wireless communication. This module can be used in a master or slave configuration.

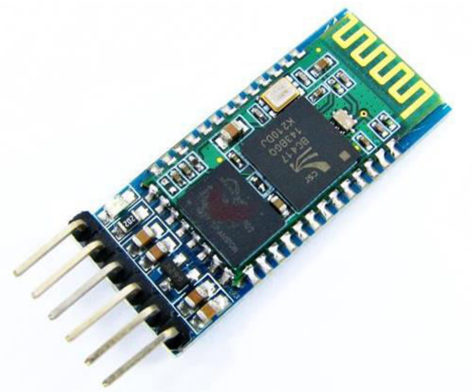

Fig.9 HC-05 Bluetooth Module Fig.10 HC-05 Bluetooth Module Pin Diagram

Bluetooth serial modules allow all serial enabled devices to communicate with each other using Bluetooth.

It has 6 pins,

1. Key/EN: It is used to bring Bluetooth module in AT commands mode. If Key/EN pin is set to high, then this module will work in command mode. Otherwise by default it is in data mode. The default baud rate of HC-05 in command mode is 38400bps and 9600 in data mode.

HC-05 module has two modes,

1. Data mode: Exchange of data between devices.

2. Command mode: It uses AT commands which are used to change setting of HC-05. To send these commands to module serial (USART) port is used.

2. VCC: Connect 5 V or 3.3 V to this Pin.

3. GND: Ground Pin of module.

4. TXD: Transmit Serial data (wirelessly received data by Bluetooth module transmitted out serially on TXD pin)

5. RXD: Receive data serially (received data will be transmitted wirelessly by Bluetooth module).

6. State: It tells whether module is connected or not.

Specification of HC-05 Bluetooth Module

- Bluetooth version: 2.0 + EDR (Enhanced Data Rate)
- Frequency: 2.4 GHz ISM band
- Modulation: GFSK (Gaussian Frequency Shift Keying)
- Transmit power: Class 2 (up to 4 dBm)
- Sensitivity: -80 dBm typical
- Range: approximately 10 meters (or 33 feet) in open air
- Operating voltage: 3.3V to 5V DC
- Operating current: less than 50mA
- Standby current: less than 2.5mA
- Sleep current: less than 1mA
- Interface: UART (Universal Asynchronous Receiver/Transmitter)
- Baud rates: 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, and 460800

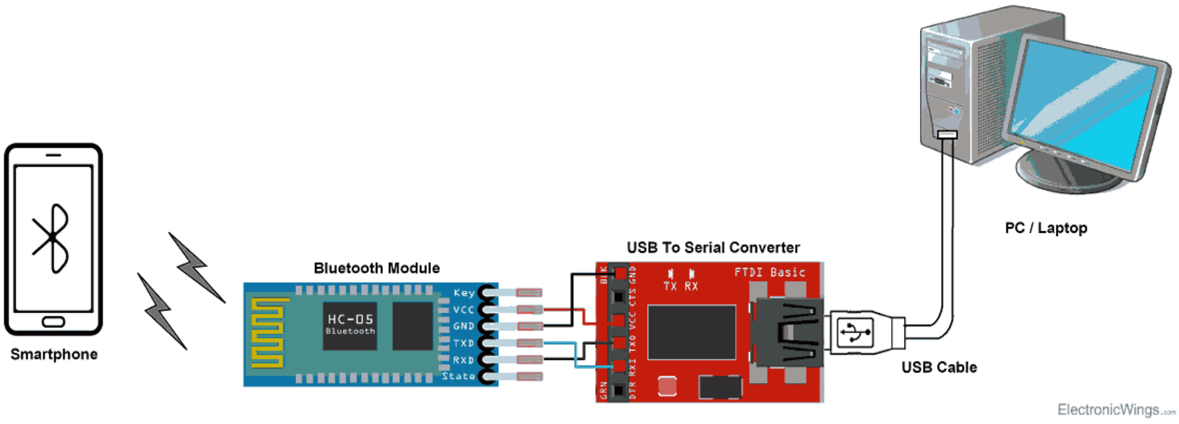

Fig.11 Bluetooth communication between Devices

CHAPTER 4

CIRCUIT DIAGRAM AND CODING

4.1 Circuit Diagram

The circuit diagram of the robotic arm setup is given as follows:

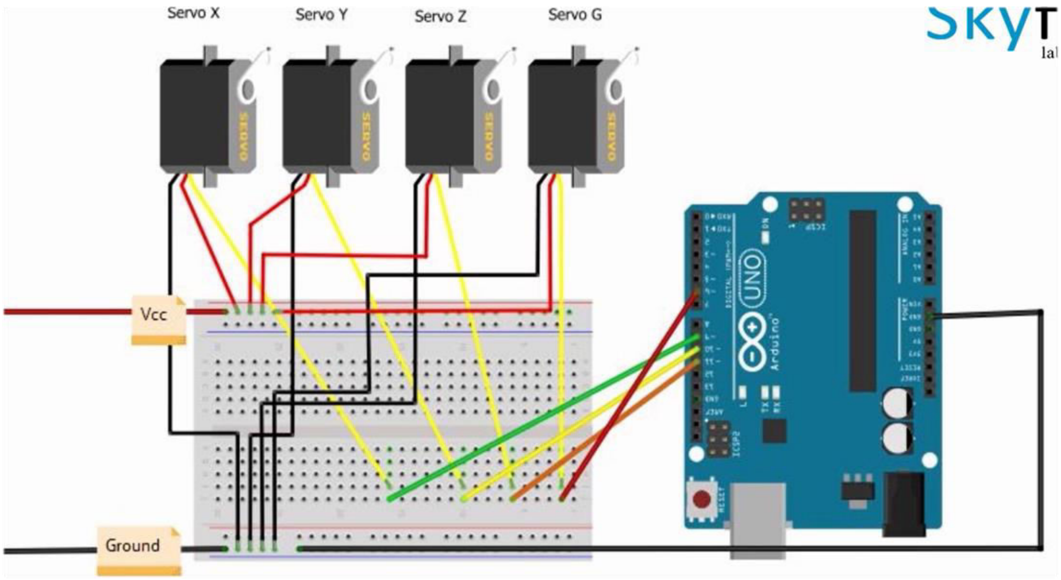

Fig.12 Circuit connection of Arduino and servo motors

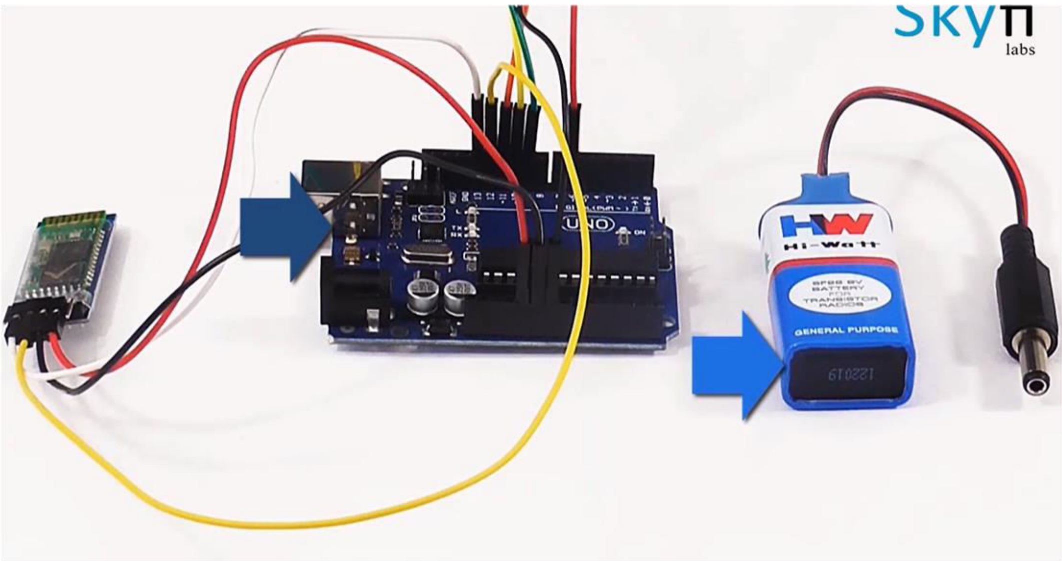

Fig.13 Circuit connection of Arduino and Bluetooth module

4.2 Code

Coding of Arduino uno of robotic arm is done in the Arduino IDE application. The code is written using C/C++ language. The programming code is given below:

```
#include<Servo.h>

#include<SoftwareSerial.h>


SoftwareSerial bluetooth(12,13);

Servo servox;

Servo servoy;

Servo servoz;

Servo servog;


void arm() {


 servox.write(90);
```

```
 servoy.write(90);

 servoz.write (40);

 servog.write(0);

 delay(1000);

}


void setup(){ // put your setup code here, to run once:


 servox.attach (9);

 servoy.attach (10);

 servoz.attach (11);

 servog.attach (6);

 arm();

Serial.begin(9600);

bluetooth.begin(9600);

}


void loop() {


if(bluetooth.available()>=2)

{

 servox.detach();

 servoy.detach();

 servoz.detach();

 servog.detach();
```

```
unsigned int servopos = bluetooth.read();

unsigned int servopos1 = bluetooth.read();

unsigned int realservo = (servopos1 *256) +servopos;


Serial.println(realservo);

int servo1 = realservo;

Servo currservo;


if (realservo>=1000 && realservo<1180) {

servox.attach (9);

servo1 = map (servo1,1000,1180,0,180);

currservo = servox;

}


if (realservo>=2000 && realservo<2180) {

servoy.attach (10);

servo1 = map(servo1,2000,2180,0,180);

currservo = servoy;

}


if (realservo>=3000 && realservo<3180) {

servoz.attach (11);

servo1 = map(servo1,3000,3180,0,180);

currservo = servoz;

}
```

```
if (realservo>=4000 && realservo<4180){
  servog.attach (6);
  servo1 = map (servo1,4000,4180,0,180);
  currservo = servog;
}

 currservo.write(servo1);
}
}
```

RESULT

The project ran successfully, with a lot of efforts from the team members and support of seniors and the staff.

The robotic arm was able to move and pick/replace objects upto a certain weight with the help of mobile application.

The application has 4 sliders which represent the 4 servo motors of the arm. As we slide the indicator, the respective servo motor moved in the desired direction and as a result we were easily able to move/place any object.