AUTOMATIC SEED SOWING AGRIBOT

A course project report submitted in partial fulfilment of the requirement

of

SMART SYSTEM DESIGN

By

N.HARSHITHA	**(2205A21026)**
A.PRAHARSHITHA	**(2205A21001)**
B.SUMANTH	**(2203A51551)**
CH.AJAY KUMAR	**(2203A51554)**

Under the guidance of

Mr.Y.Srikanth

Asst. Professor, Department of ECE

Mrs.K.Manasa

Research scholar,Department of ECE

SR UNIVERSITY

ABSTRACT

Automatic seed sowing and weeding is a robotic car, which works according to the commands of humans through the Bluetooth Module. Because this robotic car has an HC-05 Bluetooth module which connects to any Android device and performs the same functions given by the farmer as command such as to start the robot, control weed with help of the cutter ,etc . Many devices have been used in this robotic car, out of them Arduino Uno chip is more important and special, so we have named our project it, "Arduino based Automatic Seed Sowing and Weeding Agriculture Robot". As we mentioned, many devices have been used in this, each device has its own specialty. Those complete their work accordingly and this project as well Arduino based Automatic Seed Sowing and Weeding Agriculture Robot is much useful in farming system like irrigation, weeding, etc. This robotic car is very useful for those who is physically handicapped.

CONTENTS

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION

In our country, we do not have sufficient machinery factors in the agriculture sector and it increases the load of labor on our farmers. So, it's time to automate the sector to overcome this problem. In India,70% of people depend on agriculture. So we need to study agriculture. Innovative idea of our project is to automate the process of irrigation and inspection of the soil nutrients periodically to yield nutritious crops. The farming system like irrigation, weeding, etc. is the different process. All the processes are advance to modifying the mechanism in farming which works automatically without the man power requirement. Manually irrigation method suffers from various problems. The tendency of man work is going on reducing. The man power shortage is one of the biggest problems faced continuously to all farmers. Due to labor shortage the plantation cost should be increased. So, it is not economically beneficial for all farmers. Now a day's instrumentation and control system play an important role. So, we develop a "Agribot" using micro controller which is very economical and beneficial. Due to automation the work become easiest, error less and it saves also. Our system is nothing but the three-tyre vehicle which is driven by geared DC motor. According to the micro controller program.

1.2 OBJECTIVES

The objective of this project is to implement a low cost, reliable, and scalable automatic seed sowing and spraying agriculture robot to establish communication between farmer and Agribot via Android App for starting the robot and to control weed with the help of cutter.

CHAPTER 2

PROJECT DESCRIPTION

2.1 BLOCK DIAGRAM OF THE PROJECT

The block diagram of the project is shown in Fig. 2.1. The block diagram mainly consists of an Arduino, 2 DC motors, Motor Driver IC(L293D), a Bluetooth module, a battery(12v), 1 servo motor. It works according to the commands of Bluetooth.And weeding is also done by commands.

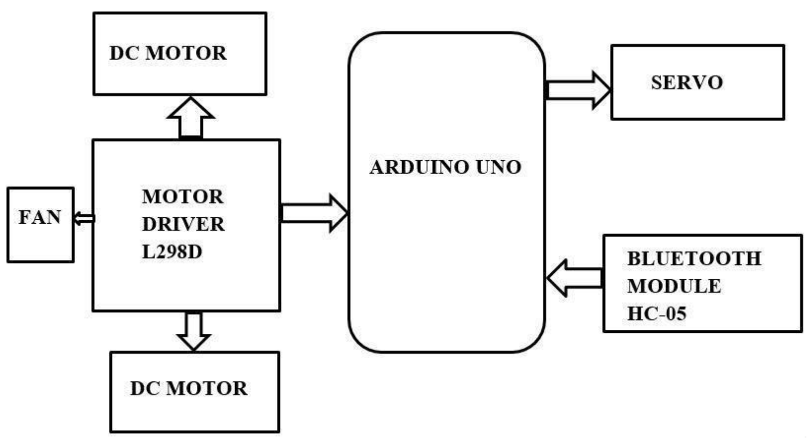

Fig.2.1 Block Diagram

2.2 HARDWARE DESCRIPTION

2.2.1 Arduino UNO

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet) is shown in Fig.2.2. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter. Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode. Revision 3 of the board has the following new features: 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes. Stronger RESET circuit. Atmega 16U2 replace the 8U2. "Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)

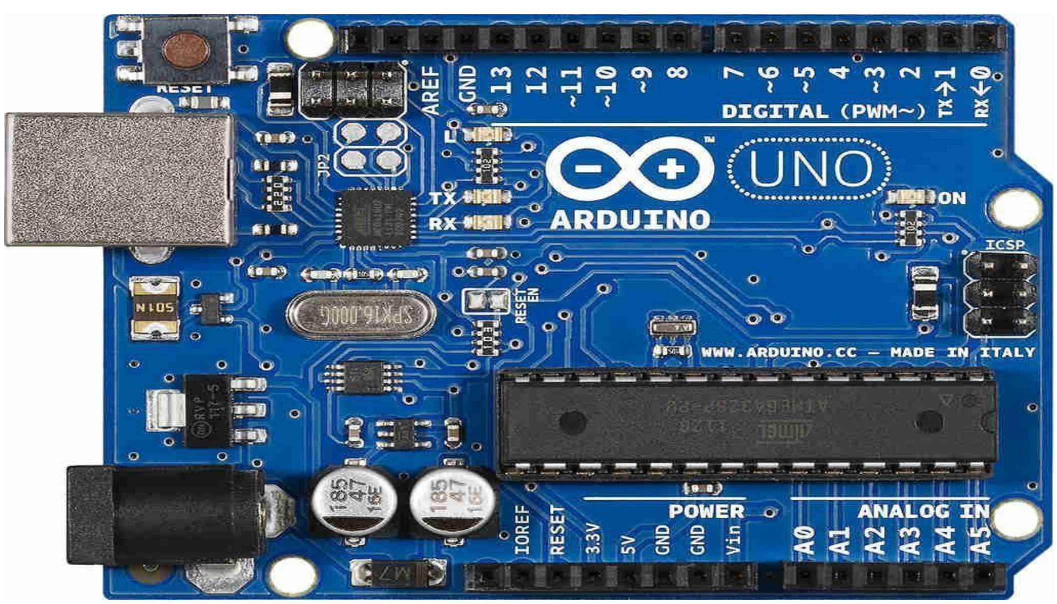

Fig. 2.2 Arduino Uno

Applications:

- Xoscillo, an open-source oscilloscope
- Arduinome, a MIDI controller device that mimics the Monome
- OBDuino, a trip computer that uses the on-board diagnostics interface found in most modern cars
- Gameduino, an Arduino shield to create retro 2D video games
- ArduinoPhone, a do-it-yourself cellphone
- Water quality testing platform
- Automatic titration system based on Arduino and stepper motor
- Low cost data glove for virtual reality applications
- Impedance sensor system to detect bovine milk adulteration
- Homemade CNC using Arduino and DC motors with close loop control by Homofaciens
- DC motor control using Arduino and H-Bridge

2.2.2 Servo Motor

A standard servo motor as shown in fig.2.3 typically has three main pins:

1. **Power/VCC Pin:** This pin is connected to a power source, usually around 4.8V to 6V. It supplies power to the servo motor for its operation.
2. **Ground Pin:** This pin is connected to the ground (GND) or negative terminal of the power source. It completes the circuit and provides the reference voltage for the servo motor.
3. **Control Signal Pin:** This pin is connected to a microcontroller or other control device. It receives control signals that determine the position or angle at which the servo motor should move.

Fig. 2.3 Servo Motor

The function of a servo motor is to rotate to a specific position or angle based on the control signal it receives. It is commonly used in robotics, automation, and various other applications where precise control of angular position is required.

Servo motors work based on a feedback control mechanism. The control signal provided to the servo motor represents the desired position or angle. The motor's internal control circuitry compares this desired position with the motor's current position (feedback mechanism) and adjusts the motor's rotation accordingly to minimize the error between the desired and actual positions.

The control signal for a servo motor is usually a pulse-width modulation (PWM) signal. The pulse width (ON time) of the signal determines the desired position or angle. A typical servo motor expects a PWM signal with a frequency of around 50 Hz, and the pulse width ranges from 1 ms to 2 ms. A pulse width of 1 ms corresponds to one extreme position, a pulse width of 1.5 ms corresponds to the center position, and a pulse width of 2 ms corresponds to the opposite extreme position.

By varying the pulse width of the control signal, you can control the servo motor to rotate to different angles within its specified range.

2.2.3 Motor Driver (L298D)

This L298N Motor Driver Module is a high power motor driver module for driving DC and Stepper Motors which is shown in Fig 2.4. This module consists of an L298 motor driver IC and a 78M05 5V regulator. L298N Module can control up to 4 DC motors, or 2 DC motors with directional and speed control.

Fig. 2.4 L298D

- Ground Pins-4
- Input Pins-4
- Output Pins-4
- Enable Pins- 2
- Voltage Pins – 2

Motor Driver ICs are primarily used in autonomous robotics only. Also most microprocessors operate at low voltages and require a small amount of current to operate while the motors require a relatively higher voltages and current . Thus current cannot be supplied to the motors from the microprocessor. This is the primary need for the motor driver IC.

The L298D IC receives signals from the microprocessor and transmits the relative signal to the motors. It has two voltage pins, one of which is used to draw current for the working of the L298D and the other is used to apply voltage to the motors. The L298D switches it output signal according to the input received from the microprocessor.

Technical Specifications:

Motor output voltage	4.5V – 36V
Logic input voltage	5V
Output Current per channel	600mA
Peak Output Current per Channel	1.2A

Control Pins: **Direction**

IN1	IN2	Spinning Direction
Low(0)	Low(0)	Motor OFF
High(1)	Low(0)	Forward
Low(0)	High(1)	Backward
High(1)	High(1)	Motor OFF

2.2.4 HC-05

HC-05 is a Bluetooth module which is designed for wireless communication is shown in below Fig 2.5. This module can be used in a master or slave configuration. 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 uses serial communication to communicate with devices. It communicates with microcontroller using serial port (USART).

HC-05 module Information

HC-05 has red LED which indicates connection status, whether the Bluetooth is connected or not. Before connecting to HC-05 module this red LED blinks continuously in a periodic manner. When it gets connected to any other Bluetooth device, its blinking slows down to two seconds. This module works on 3.3V. We can connect 5V supply voltage as well since the module has on board 5 to 3.3V regulator.

As HC-05 Bluetooth module has 3.3V level for RX/TX and microcontroller can detect 3.3 V level, so, no need to shift transmit level of HC-05 module. But we need to shift the transmit voltage level from microcontroller to RX of HC-05 module.

Fig 2.5 HC-05

Pin Functions:

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,

➢ **Data mode:** Exchange of data between devices.

➢ **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.

2.2.5 DC Motor Description:

Dual Shaft DC motor with gear box which gives good torque and rpm at lower voltages. This motor can run at approximately 200rpm when driven by a Dual Li-Ion cell battery at 6 V and approximately at 300 rpm when driven by a 9V Li-Ion cell. It is most suitable for light weight robot running on small voltage. Out of its two shafts one shaft can be connected to wheel, other can be connected to the position encoder.

Fig. 2.6 Dual Shaft DC Motor

Features :

a. Working voltage: 3V to 9V
b. 30gm weight
c. Ability to operate with minimum or no lubrication, due to inherent lubricity.
d. 1.9Kgf.cm torque

e. No-load current = 60mA, Stall current = 700mA

2.3 SOFTWARE DESCRIPTION

The software used here is ARDUINO SOFTWARE:

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

Writing Sketches:

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

NB:

Versions of the Arduino Software (IDE) prior to 1.0 saved sketches with the extension pde. It is possible to open these files with version 1.0, you will be prompted to save the sketch with the ino extension on save.

Verify

Checks your code for errors compiling it.

Upload

Compiles your code and uploads it to the configured board. See uploading below for details.

Note: If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer"

New

Creates a new sketch.

Open

Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.

Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the File | Sketchbook menu instead.

Save

Saves your sketch.

Serial Monitor

Opens the serial monitor.

Additional commands are found within the five menus: File, Edit, Sketch, Tools and help.

Programming on Arduino Uno

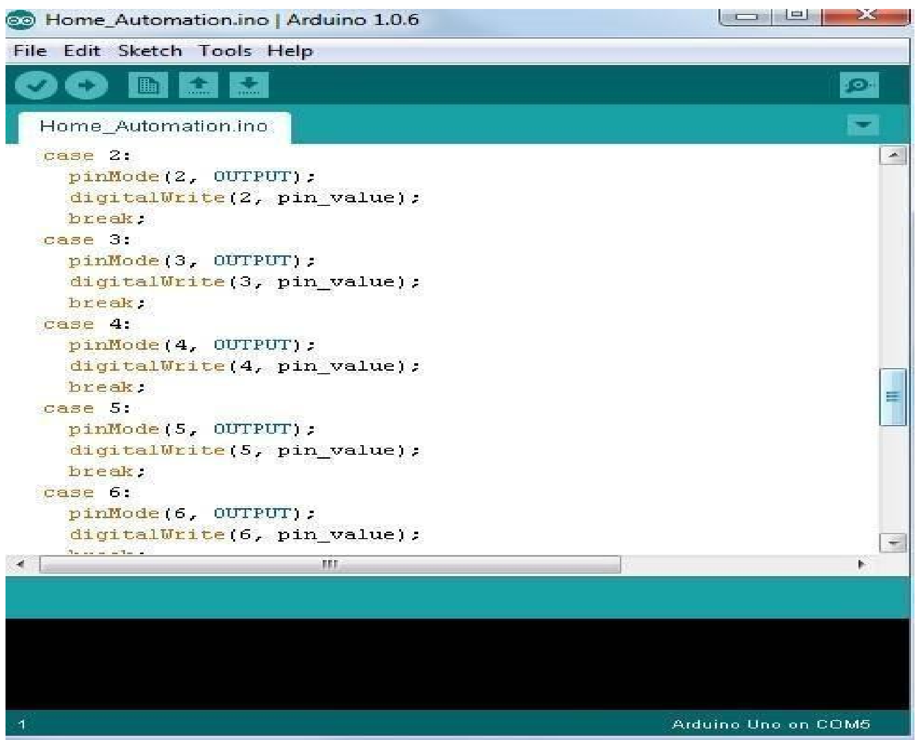

Fig.2.7 Software IDE

In order for the Arduino-Uno board to be able to interact with the application used in this project certain program (code) needs to be uploaded to the Arduino-Uno. Arduino Company provides user friendly software which allows writing any code for any function wanted to be performed by the Arduino-Uno and upload it to the board.Refer to appendix A for the full source code of the Arduino-Uno board.

CHAPTER 3

CIRCUIT DIAGRAM AND DESCRIPTION

3.1 Working

The sensor and Arduino is connected to the battery. Motors are soldered with the wires positive and negative. It is built with the 2DC motors they run with the speed provided in Arduino IDE code. Whenever the soil moisture sensor detects the less moisture in the soil it sends the information to the Arduino. The Arduino receives it automatically sprays the water with the help of pump motor. The robot is equipped with a seed container that holds a supply of seeds and it drops the seeds on to the soil with the help of servo motors. The robot's onboard controller ensure accurate seed spacing and depth, optimizing the sowing process. The robot distinguish between crops and the weeds applying target the weeds are removed with the help of fan. The code fed to the Arduino runs continuously and the cycle repeats in regular intervals. As the working of seed sowing and spraying using Arduino is as shown in fig 3.1

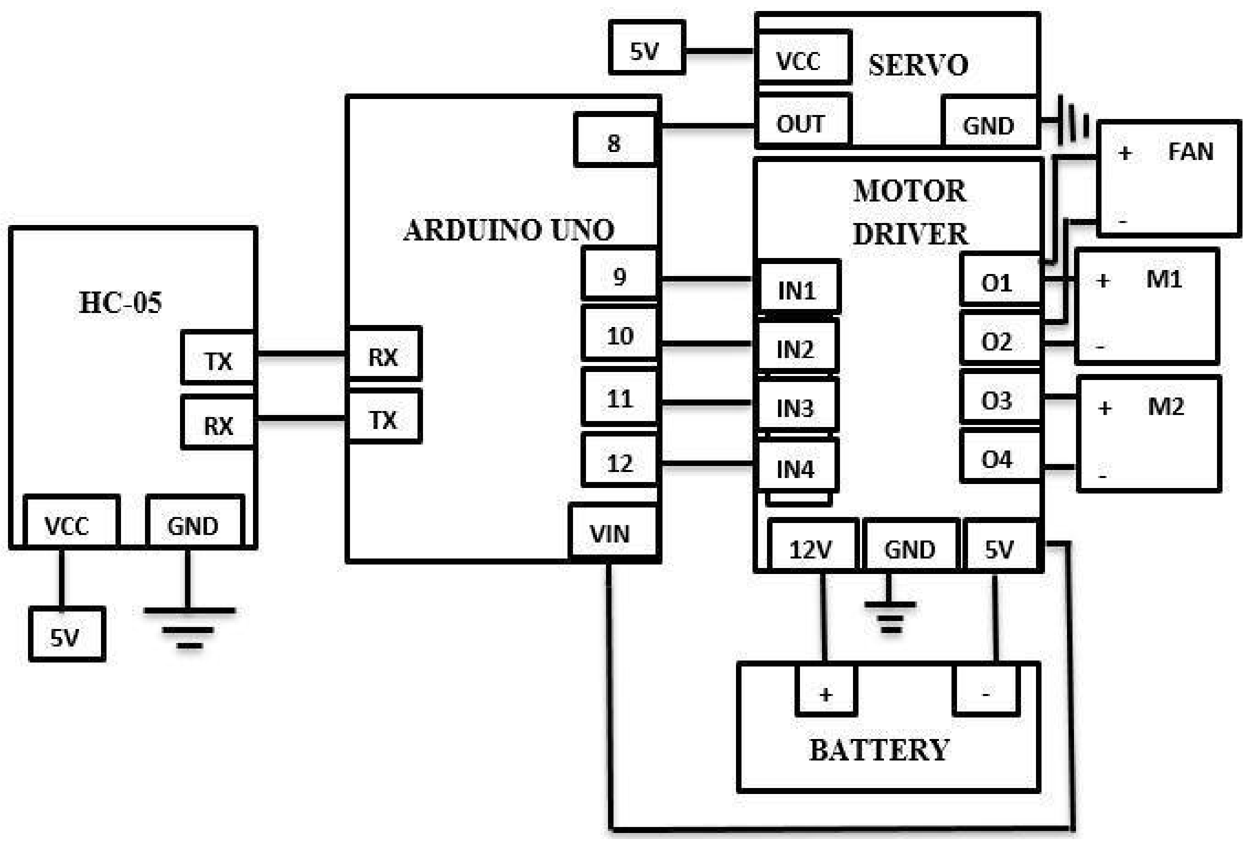

Fig.3.1 Circuit diagram

3.2 RESULTS

The experimental result is as shown in below fig. 3.2

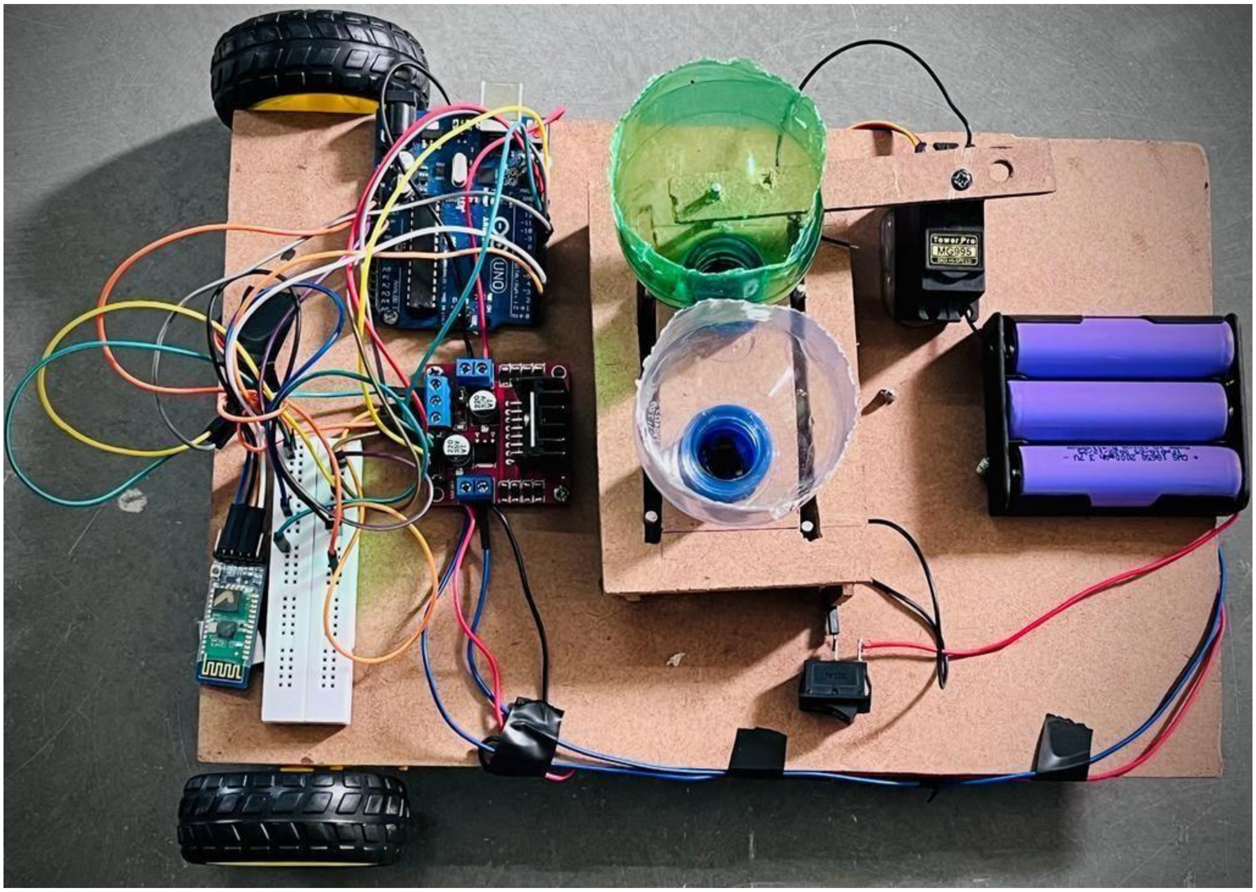

Fig.3.2 Experimental result

AUTOMATIC SEED SOWING AND WEEDING AGRICULTURE ROBOT USING ARDUNIO performs the operations according to the commands given in the Bluetooth. User friendly as in anyone can use just a click of a button on mobile and everything works. It has potential to enhance agricultural processes, increase productivity and improve overall efficiency. And it is cost effective as in it will cost exactly as the project requires (optimum price).

3.3 ADVANTAGES

- Increases efficiency and productivity compared to the manual labour
- Labour savings.
- Precision and accuracy on planting.
- Contributes to the economy.
- Optimal resource utilization.
- Improved crop quality.
- Reduced environmental impact.

3.4 DISADVANTAGES

- Equipment and installation cost.
- Limited adaptability.
- Human errors.
- Reliability.
- System compatibility.

CHAPTER 4

CONCLUSION

4.1 CONCLUSION

In this project Automatic seed sowing and weeding agriculture robot has been implemented. It works on a predefined code inserted in Arduino UNO. The project helps to sow seeds and weeding. Ultimately, agrirobots represent a significant opportunity to optimise productivity, reduce labour requirements, and promote sustainability in the agriculture sector. With continued advancements and responsible deployment, agrirobots can contribute to the development of more efficient, profitable, and environmentally friendly farming systems. This is simple and cost effective. However this agrirobot helps the farmer to reduce the work.

4.2 FUTURE SCOPE

The future scope of agrirobots is vast and holds immense potential for transforming the agriculture industry. As days pass labours are decreasing. Agrirobot don’t need any human work it works only by switching on and charging the batteries is up to us. Overall, the future of agrirobots is exciting and holds tremendous potential for revolutionizing agriculture.

BIBLIOGRAPHY

[1]. Kise, M., Ohno, S., Yoshida, K., & Tsujimoto, K. (2018). Agrirobot: Towards a fully autonomous robot for precision agriculture. Journal of Field Robotics, 35(1), 86-108.

[2]. Zhang, Q., Zhou, Y., & Wei, S. (2020). Recent advances in agri-robotics: A comprehensive review. Biosystems Engineering, 191, 37-54.

[3]. Mantri, P., Mourya, R., & Tiwari, R. (2020). Agricultural robots for crop management: A review. Computers and Electronics in Agriculture, 174, 105507.

[4]. Barghini, A., Kondak, K., Boley, D., & Valente, J. (2019). A review on agricultural robots for field operations. Applied Sciences, 9(14), 2834.

[5]. Zhang, J., Chen, Y., Zhu, Y., & Tang, H. (2021). Agricultural robot for autonomous harvesting: A review. Agricultural Engineering International: CIGR Journal, 23(2), 31-43.

[6]. Giacomelli, F., Demarinis, A., Ferrigno, G., & Antonucci, F. (2018). Robotic systems for agricultural applications: A review. Computers and Electronics in Agriculture, 153, 69-81.

[7]. Rasheed, A., Mahmood, A., & Shah, M. (2021). A comprehensive review on agricultural robots: Research advancements, challenges, and future scope. Robotics, 10(1), 7.

[8]. Nayak, R., Pandey, S., Tiwari, S., & Mishra, S. (2020). Precision agriculture using agricultural robots: A review. IOP Conference Series: Earth and Environmental Science, 459(1), 012049.

[9]. Mishra, S., Nayak, R., & Shukla, A. (2018). Smart farming system using agricultural robots: A review. International Journal of Engineering and Technology, 7(4.25), 206-210.

APPENDIX

```
#include <Servo.h>
Servo myservo;
bool runServo = false;
int motorPin1 = 9; // Motor 1 direction pin
int motorPin2 = 10; // Motor 1 PWM pin
int motorPin3 = 11; // Motor 2 direction pin
int motorPin4 = 12; // Motor 2 PWM pin
void setup() {
 Serial.begin(9600);
 myservo.attach(8);
 pinMode(motorPin1, OUTPUT);
 pinMode(motorPin2, OUTPUT);
 pinMode(motorPin3, OUTPUT);
 pinMode(motorPin4, OUTPUT);
 myservo.write(0);
}
void loop() {
 if    (Serial.available())
   {      char      c      =
   Serial.read();
   Serial.println(c);
     if (c == 'F') {          // Move forward
     moveForward();
   } else if (c == 'B') {    // Move backward
     moveBackward();
   } else if (c == 'L') {    // Turn left
     turnLeft();
   } else if (c == 'R') {    // Turn right
     turnRight();
   } else if (c == 'S') {    // Stop
```

```
stopMotors();
```

```
} else if (c == 'A') {    // Run servo
    rServo();
   } else if (c == 'X') {    // Stop servo
    sServo();
   }
  }
 }
 void moveForward()
  { digitalWrite(motorPin1,
  HIGH); digitalWrite(motorPin2,
  LOW); digitalWrite(motorPin3,
  HIGH); digitalWrite(motorPin4,
  LOW);
 }
 void moveBackward()
  { digitalWrite(motorPin1,
  LOW); digitalWrite(motorPin2,
  HIGH); digitalWrite(motorPin3,
  LOW); digitalWrite(motorPin4,
  HIGH);
 }
 void turnLeft()
  { digitalWrite(motorPin1,
  LOW); digitalWrite(motorPin2,
  HIGH); digitalWrite(motorPin3,
  HIGH); digitalWrite(motorPin4,
  HIGH);
 }
 void turnRight()
  { digitalWrite(motorPin1,
  HIGH); digitalWrite(motorPin2,
  HIGH); digitalWrite(motorPin3,
```

```
LOW); digitalWrite(motorPin4,
HIGH);
}
```

```
void stopMotors()
  { digitalWrite(motorPin1,
  LOW); digitalWrite(motorPin2,
  LOW); digitalWrite(motorPin3,
  LOW); digitalWrite(motorPin4,
  LOW);
}
void rServo()
    { while(1)
    {
        for (int i = 0; i <= 90; i++)
          { myservo.write(i);
          delay(10);
        }
        for (int i = 90; i >= 0; i--)
          { myservo.write(i);
          delay(10);
        }
      if (Serial.available())
      {
        char c = Serial.read();
        if (c == 'X')
        {
          sServo();
          break;
        }
      }
    }
}
void sServo()
      { myservo.write(0);
}
```