# BALL COLLECTION ROBOT


TARINI MAGOTRA – 2022MEB1358
VAISHNAVI GOWRISHANKAR – 2022MEB1362
SHREYAS MANAK – 2022MEB1354
SURENDRA MEENA – 2022MEB1354
SUGLI AKASH NAIK – 2022MEB1350

# COMPONENTS USED

1.Arduino Uno

2.Motor Driver Module L289 for controlling the motors

3.Motors

4.Wheels and Chassis for robot’s physical structure

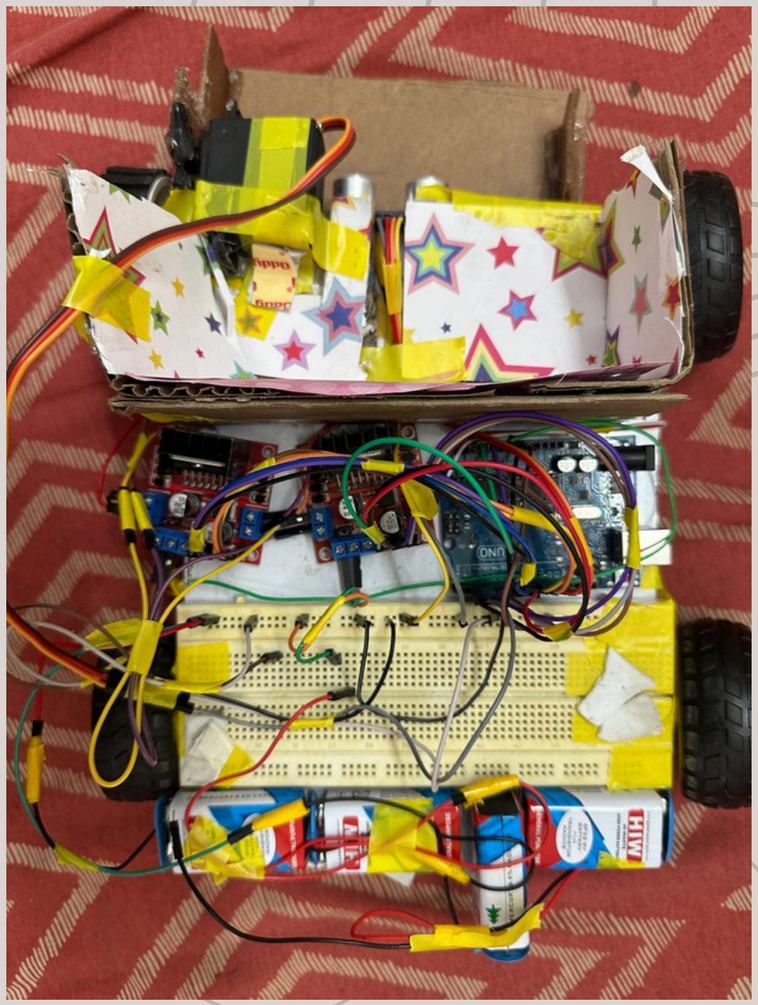
5.Ultrasonic sensor

6.Servo Motor for picking up balls

7.Power Supply 9V Batteries

# WHAT IS A BALL COLLECTING ROBOT?

The ball-collecting robot employs a range of sensors to identify the presence of balls, then utilizes a front-mounted scooping mechanism akin to a car. It employs an ultrasonic sensor to measure distances, enabling it to approach and gather the balls efficiently. The collection process involves rotating a board attached to the front via servo motors, facilitating the seamless collection of the detected balls.



# 1.Arduino Uno

The Arduino Uno is a microcontroller board comprising a microcontroller (ATmega328P), digital and analog input/output pins, power connection, USB interface for programming, and a reset button. Its working involves writing code on a computer, uploading it via USB to the Uno, which then executes the code. The Uno interprets and executes instructions, interacting with sensors, motors, and other components based on the programmed logic.

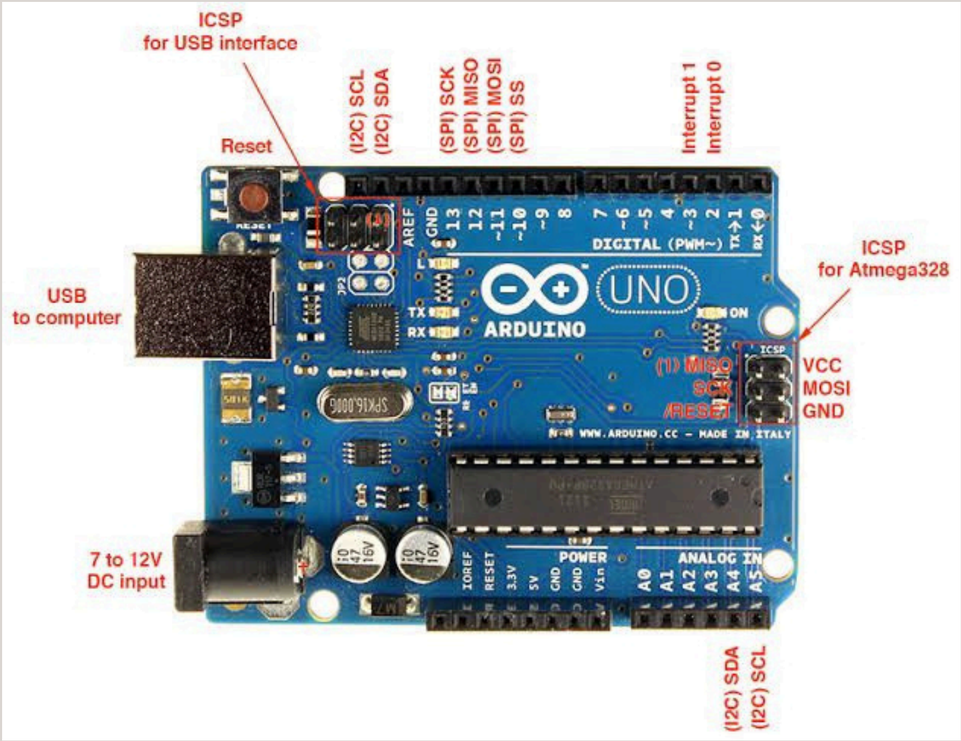
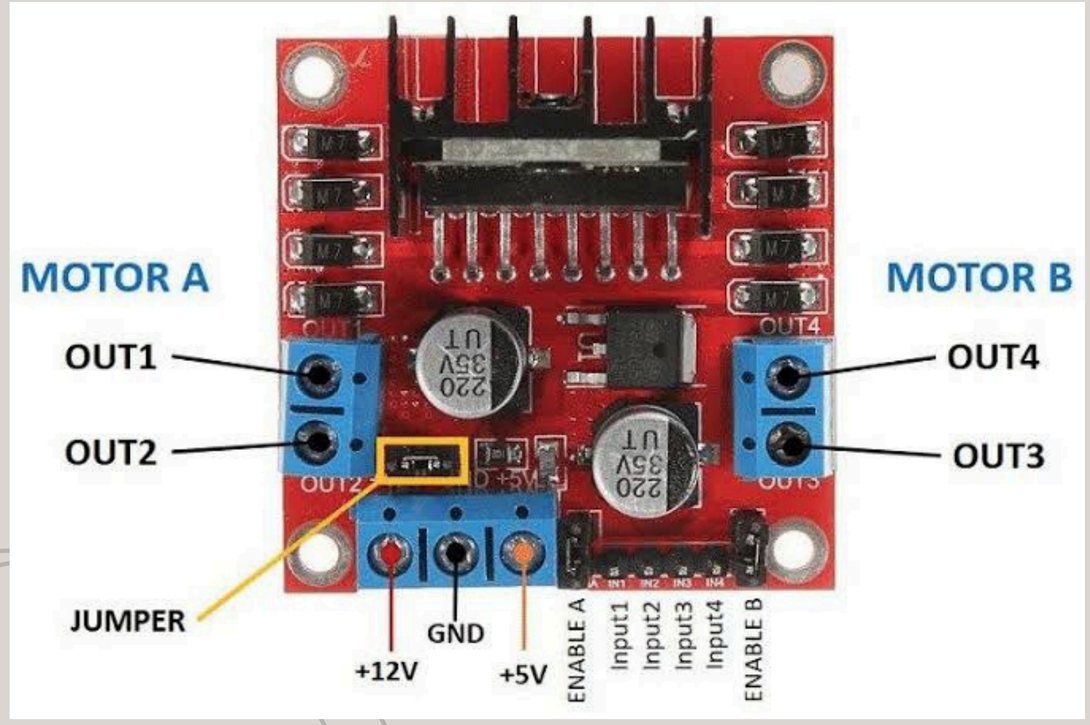


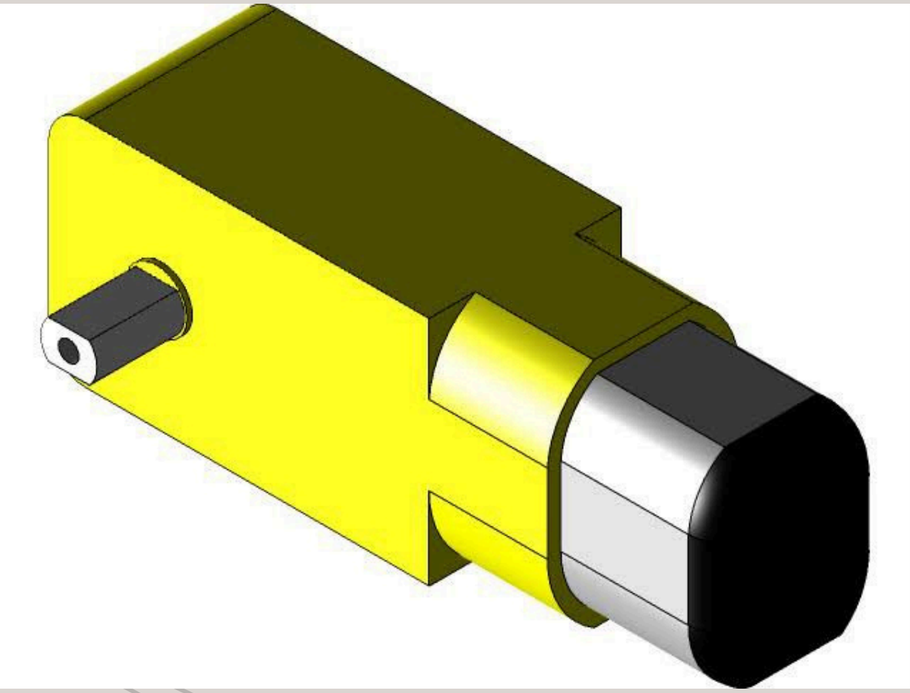

Fig 1 :Arduino Uno

# 2.Motor Driver Module L289




The L289N module acts like a traffic cop for two DC motors. It takes low-power control signals (0 or 5V) and uses them to control the direction (forward/reverse) and speed (via pulse-width modulation) of each motor. It has separate high-power connections for the motors themselves. Think of it as a translator between your microcontroller's brain and the muscle of the DC motors.

# 3.DC MOTORS

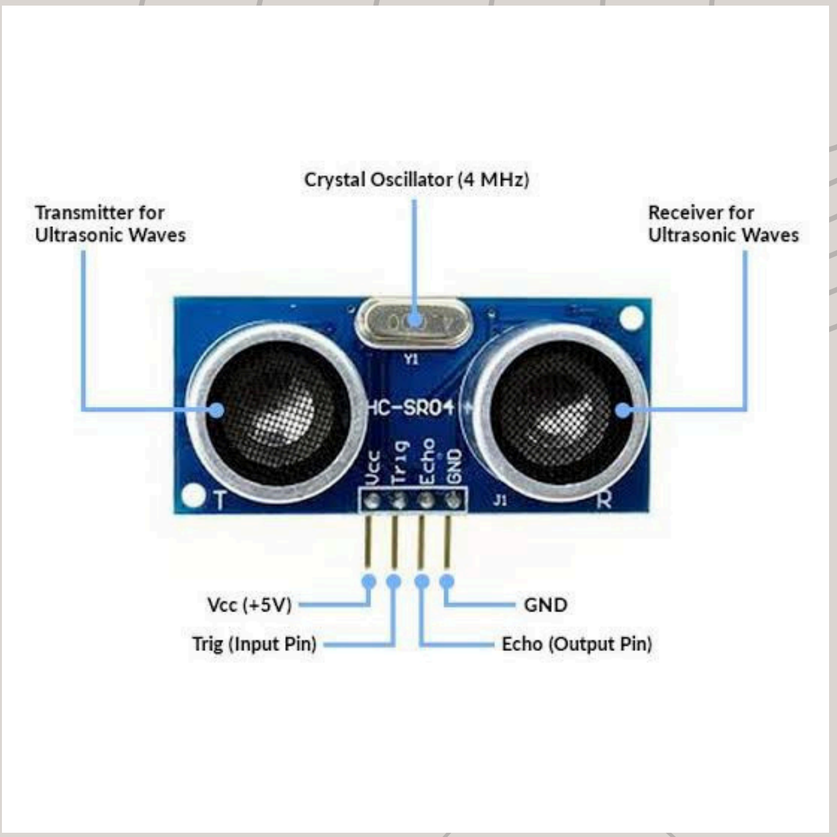


The car's movement will be powered by 4 direct current (DC) motors. These motors provide rotational torque when supplied with voltage. The direction of rotation (clockwise or counter-clockwise) depends on the polarity of the applied voltage. By varying the voltage using Pulse Width Modulation (PWM) signals from the Arduino Uno, the speed of each motor can be controlled. This precise control over speed and direction allows the Arduino to maneuver the car effectively.

# 4.Ultrasonic sensor

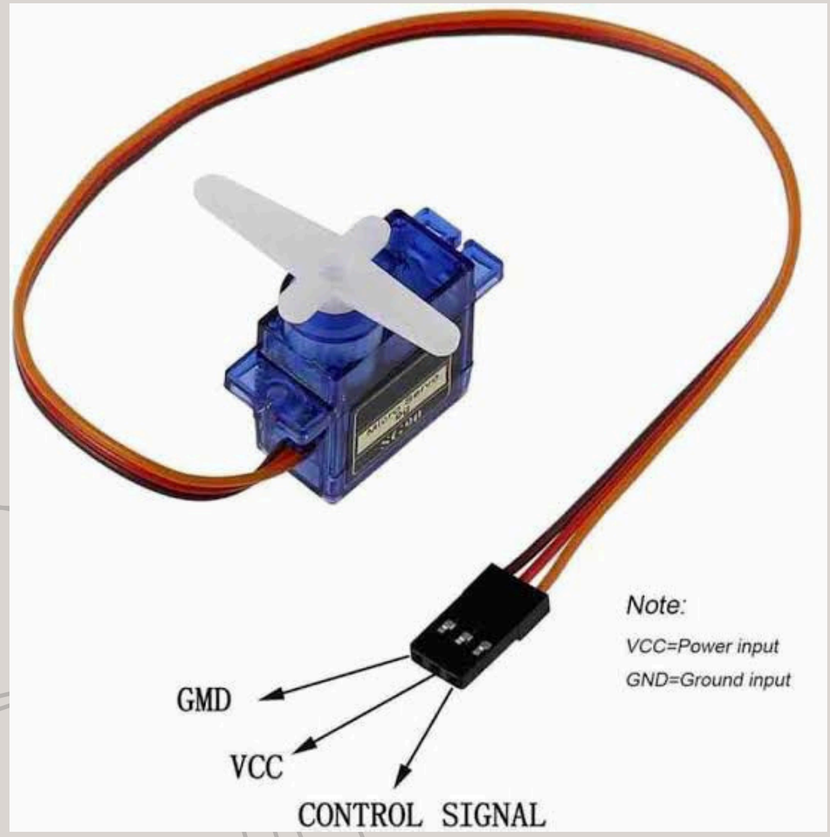
Here's how an ultrasonic sensor functions for Arduino IDE:

1. Trigger Pulse: The Arduino sends a short (microsecond-level) high pulse to the sensor's trigger pin. This tells the sensor to emit a burst of ultrasound.
2. Ultrasonic Emission: The sensor generates a high-frequency sound wave (typically above 20 kHz, inaudible to humans). This wave travels through the air.



- Echo Detection: If an object is within the sensor's range, the sound wave bounces off the object and returns to the sensor.
- Echo Reception: The sensor picks up the reflected sound wave (echo) with its receiver.
- Time Measurement: The Arduino measures the time it takes for the sound wave to travel to the object and back.
- Distance Calculation: Knowing the speed of sound (approximately 343 meters per second at room temperature), the Arduino can calculate the distance to the object by dividing the round-trip time by 2 (since the sound traveled the distance twice).

# 5.Servo Motors




A servo motor functions as a controlled rotary actuator. It integrates a DC motor with feedback mechanisms and control circuitry. Unlike a standard DC motor, a servo motor precisely positions its shaft at specific angles. The Arduino Uno transmits a pulse width modulation (PWM) signal to the servo's control pin. This pulse width dictates the servo's shaft position.

# 6.Chasis and Wheels

Chassis
The chassis is the robot's skeleton, a sturdy base that holds all the components together. like a car's frame - it supports the wheels, and other parts / sensors, allowing them to function as a unit.

Wheels
The wheels are the robot's feet. They make contact with the ground, enabling the robot to move around similar to the functioning of a car's tires - they provide traction and control the direction of movement.

# CODE

```
#include <Servo.h>

#define TRIGGER_PIN  12
#define ECHO_PIN     13
// Motor A connections
int bl2=9;
int bl1=2;
int bls=16;
// Motor B connections
int br2=4;
int br1=5;
int brs=6;

int fr2=10;
int fr1=11;
int frs=17;
// Motor B connections
int fl1=7;
int fl2=8;
int fls=15;

Servo armServo; // Servo motor for the arm


}
```

```
void setup() {
  pinMode(TRIGGER_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);

  pinMode(br1, OUTPUT);
  pinMode(br2, OUTPUT);

  pinMode(bl1, OUTPUT);
  pinMode(bl1, OUTPUT);

  pinMode(fl1, OUTPUT);
  pinMode(fl2, OUTPUT);

  pinMode(fr1, OUTPUT);
  pinMode(fr2, OUTPUT);

  pinMode(brs, OUTPUT);
  pinMode(bls, OUTPUT);
  pinMode(frs, OUTPUT);
  pinMode(fls, OUTPUT);

  armServo.attach(3); // Attach the servo to pin 3
  Serial.begin(9600);
}
```

# CODE

```
void loop() {
  armServo.write(150);
  long duration, distance;

  // Send pulse to trigger pin
  digitalWrite(TRIGGER_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIGGER_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIGGER_PIN, LOW);

  // Read the pulse on echo pin
  duration = pulseIn(ECHO_PIN, HIGH);

  // Calculate distance in cm
  distance = duration * 0.034 / 2;

  // Print distance to serial monitor
  Serial.print("Distance: ");
  Serial.println(distance);
```

```
if (distance < 100 && distance>10) { //
Adjust the threshold as needed
    moveTowardsBall();
    Serial.println("moving");
    delay(200); // Adjust delay according
to your robot's speed and response
time
  }
  else if (distance<10){
    stopMoving();
    collectBall();
    Serial.println("collect ball");
    }
  else {
    // If the object is not within range,
stop
    findBall();
    Serial.println("finding");
    delay(200);
  }
}
```

# CODE

```
void stopMoving(){

  digitalWrite(br1, LOW);
  digitalWrite(br2, LOW);

  digitalWrite(bl1, LOW);
  digitalWrite(bl2, LOW);

  digitalWrite(fl1, LOW);
  digitalWrite(fl2, LOW);

  digitalWrite(fr1, LOW);
  digitalWrite(fr2, LOW);

}
```

```
void moveTowardsBall() {

  digitalWrite(br1, LOW);
  digitalWrite(br2, HIGH);

  digitalWrite(bl1, LOW);
  digitalWrite(bl2, HIGH);

  digitalWrite(fl1, LOW);
  digitalWrite(fl2, HIGH);

  digitalWrite(fr1, LOW);
  digitalWrite(fr2, HIGH);

  analogWrite(brs,255);
  analogWrite(bls,255);
  analogWrite(frs,255);
  analogWrite(fls,255);

  delay(1000);

  stopMoving();

  delay(500);
}
```

```
void findBall() {

  stopMoving();

  delay(500);

  digitalWrite(br1, LOW);
  digitalWrite(br2, HIGH);

  digitalWrite(bl1, HIGH);
  digitalWrite(bl2, LOW);

  digitalWrite(fl1, HIGH);
  digitalWrite(fl2, LOW);

  digitalWrite(fr1, LOW);
  digitalWrite(fr2, HIGH);

  analogWrite(brs,255);
  analogWrite(fls,255);
  analogWrite(bls,255);
  analogWrite(frs,255);

  delay(500);

  stopMoving();

  delay(500);

}
```

```
void collectBall() {
  // Move the arm to
collect the ball
  stopMoving();

  armServo.write(150);
  delay(2000);
    armServo.write(30);
 delay(2000);
}
```

# Thank You