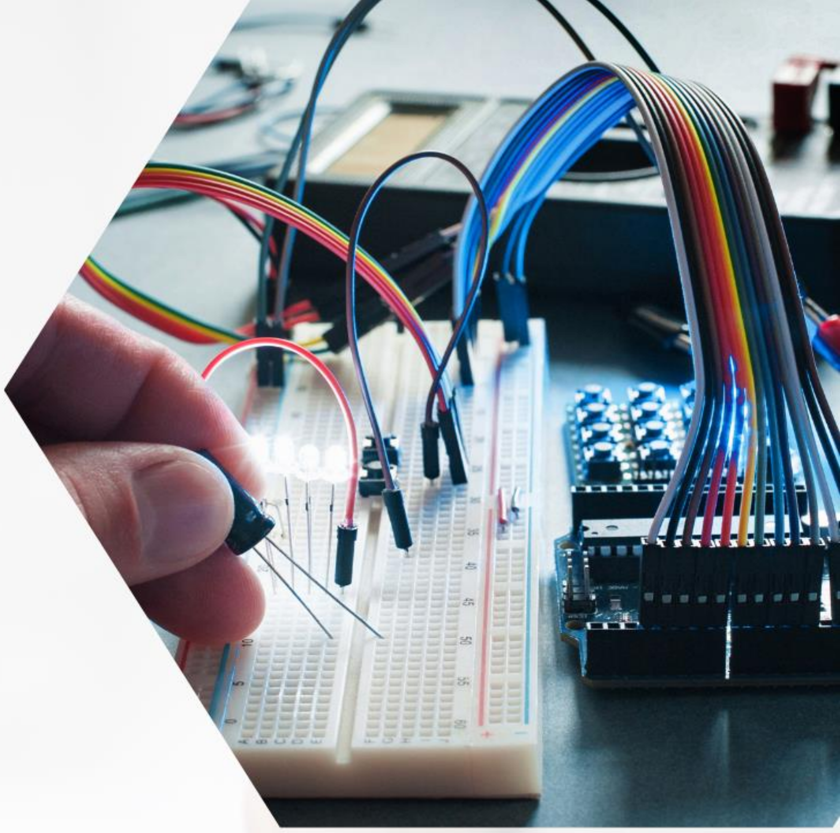
SOUL

Design Report

Robotivia Hackathon 1.0 Competition

8-2-2025



PREPARED BY: -

MOATAZ RAFEQ HAMDY MOUSA

ADHAM ABD EL-FATTAH AHMED

HAZEM ABD EL-FATTAH AHMED

OMAR AYMAN MOHAMED SAED

Table of Contents

1.Introduction

We are "SOUL" Team, a group of three passionate and dedicated individuals with a shared interest in robotics and engineering. Our team consists of:

- **Omar Ayman Mohamed:** A second-year student at Zagazig University, Faculty of Engineering, Department of Communications and Electronics, with expertise in circuit design and programming.
- **Moataz Rafik Hamdy:** A second-year student at Zagazig University, Faculty of Engineering, Department of Communications and Electronics, with a strong interest in control systems and signal processing.
- **Adham Abd El-Fattah:** A second-year student at the Egyptian Russian University, Faculty of Engineering, Department of Mechatronics, with experience in mechanical design and automated control systems.
- **Hazem Abd El-Fattah:** A second-year student at Egypt-Japan University of Science and Technology, Faculty of Engineering, Department of Energy Resources Engineering, with experience in mechanical system design, and energy analysis and sustainability.

Driven by our desire to apply the theoretical knowledge we have gained during our academic studies to a practical and grueling   design, we're agitated to share in the Robotiva Hackathon 1.0. Our thing is to design and make a completely integrated robot able of tackling the three main challenges Line Following, Obstacle Avoidance, and Pick & Place.

Our robot is designed to be a model of effectiveness and invention. It features a robust mechanical system, a precise electrical setup, and advanced software algorithms that enable it to perform the needed tasks with delicacy and speed.

Through this design, we aim to strike a balance between high performance and strict adherence to the competition's rules.

The primary ideal of this design is to apply the engineering generalities we have learned in a practical and competitive terrain, while also developing our cooperation and design operation chops. We believe that this design will not only be a significant corner in our academic and professional peregrinations but also an occasion to contribute innovative results in the field of robotics.

2. Mechanical Design

2.1 Overview

The mechanical design of our robot is grounded on a four- wheeled mobile platform, designed to perform multiple tasks, including line following, obstacle avoidance, remote- controlled navigation, and pick- and- place operations. The chassis is structured with two acrylic layers, icing continuity while maintaining a featherlight form factor. All factors are securely mounted using screws, barring the need for soldering or bonds, which facilitates easy conservation and variations. The robot's modular design allows for royal upgrades and reconfigurations, making it adaptable for various operations.

2.2 Chassis Structure and Materials

The chassis consists of two laser- cut acrylic wastes, which are chosen for their featherlight parcels, high continuity, and ease of revision. Acrylic provides a balance between severity and flexibility, icing a sturdy foundation for the robot while keeping the overall weight low. The chassis is assembled using high-perfection screws and spacers, icing an establishment and vibration- resistant structure.

The multi-layer design allows for optimized weight distribution, with the lower subcaste accommodating the motors, wheels, and electronic boards, while the upper subcaste houses the robotic arm and detectors. This layout ensures minimum hindrance between factors and maximizes space application.

Chassis Load Calculation

- Total weight estimation:
    - Motors: 4 × 150g = **600g**
    - Wheels: 4 × 50g = **200g**
    - Electronics (Arduino, sensors, etc.): **500g**
    - Battery pack: **800g**
    - Robotic arm: **1.5kg**
    - Frame (acrylic sheets + fasteners): **800g.**
    - Total estimated weight: **4.5 kg**
- Weight distribution:
    - Seventy percent of the weight is concentrated on the motor-driven wheels.
    - The remaining 30% is distributed across the chassis and robotic arm.

2.3 Locomotion System

The robot employs four independent DC motors, each driving a standard rubber wheel for superior traction and maneuverability on different surfaces. These motors are differentially driven, allowing the robot to perform both direct and rotational movements efficiently. The motors are powered and controlled via an L298N H- Bridge motor driver, which ensures proper power distribution and directional control.

Motor Torque Calculation

- Motor Specifications:
    - Rated voltage: **12V.**
    - Stall current: **1.5A**
    - Stall torque: **1.2 Nm.**
    - No-load speed: **300 RPM**
- Torque required per wheel:
    - Assuming a **4.5 kg** total weight and **four wheels**,
    - Load per wheel: 4.5 kg ÷ 4 = **1.125 kg**
    - Rolling resistance coefficient (~0.015 for rubber on smooth surfaces):
        - Required force per wheel: 1.125 kg × 9.81m/s^2 × 0.015 **= 0.16554 N**
        - Required torque per wheel: Torque = Force × Wheel Radius

        - With a wheel radius of **5 cm (0.05m)**: **0.**16554N × 0.05m = **0.00828Nm**
    - Since the motor provides **1.2Nm** stall torque, it is more than sufficient for smooth operation.

2.4 Pick & Place Arm Mechanism

A 5 Degrees of Freedom (DOF) robotic arm is mounted on the chassis, enabling the robot to perform pick-and-place operations. This arm is powered by six servo motors, which provide precise movement and control. The servo configuration includes:

- Three SG-90 micro servos for fine adjustments and lightweight movements.
- Three MG-996R high-torque servos for handling heavier loads and ensuring robust operation.

Arm Torque Calculation

- Payload capacity: **~200g**
- Required torque per joint (assuming max reach of 25cm):
    - Torque = Force × Distance
    - Force = Mass × Gravity = 0.2kg × 9.81m/s^2 = **1.96N**
    - Torque at max reach: 1.96N × 0.25m = **0.49Nm**
    - The MG-996R provides **10kg.cm (≈1Nm)** per motor, which is sufficient for stable operation.

2.5 Dimensions and Design Illustrations

To ensure a clear representation of the robot's structure and proportions, the following dimensions are defined:

- Chassis dimensions: **24cm (L)** × **15cm (W)** × **8cm (H)** (including wheels)
- Wheel diameter: **10cm**
- Robotic arm dimensions:
    - Base to shoulder: **6 cm**
    - Shoulder to elbow: **8 cm**
    - Elbow to wrist: **7 cm**
    - Wrist to gripper: **4 cm**

- Overall robot height (including arm at maximum extension): **35 cm.**

2.6 Integrated CAD Schematics for Mechanical Design

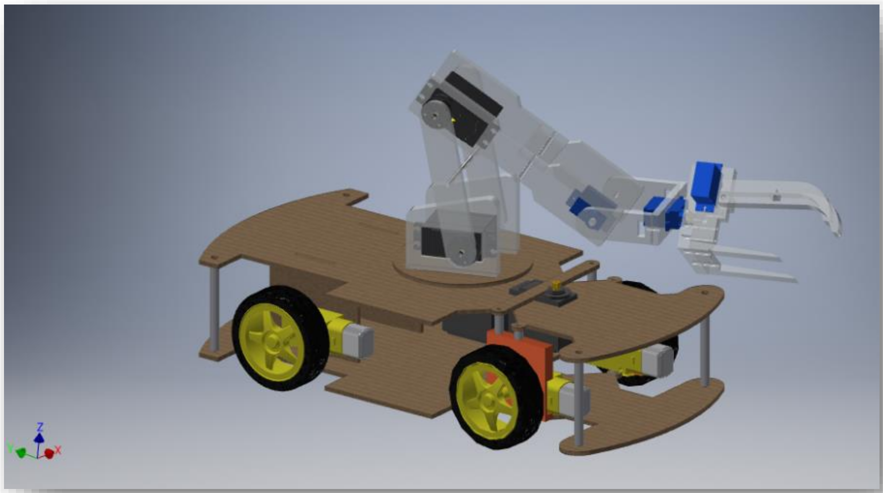
1.Robotic chassis

These images depict a 3D CAD model of a mobile robotic platform with a multi-layered lattice and a wheeled drive system. The platform features a structurally rigid base designed for stability and mobility, with strategically placed support rudiments to insure balance and continuity. The wheel configuration suggests a nimble movement capability, making it suitable for independent navigation, robotic competitions, or advanced control operations.

..

Top view

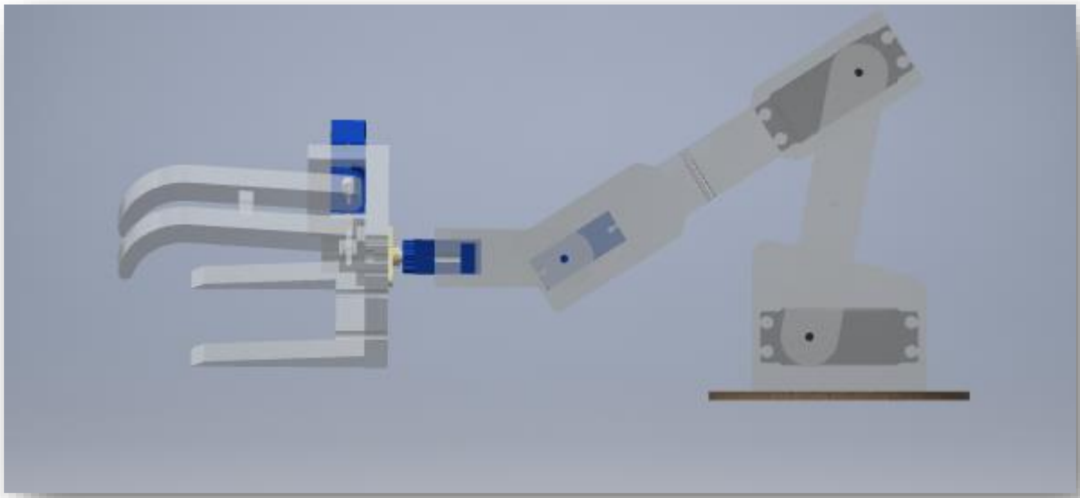
Side view



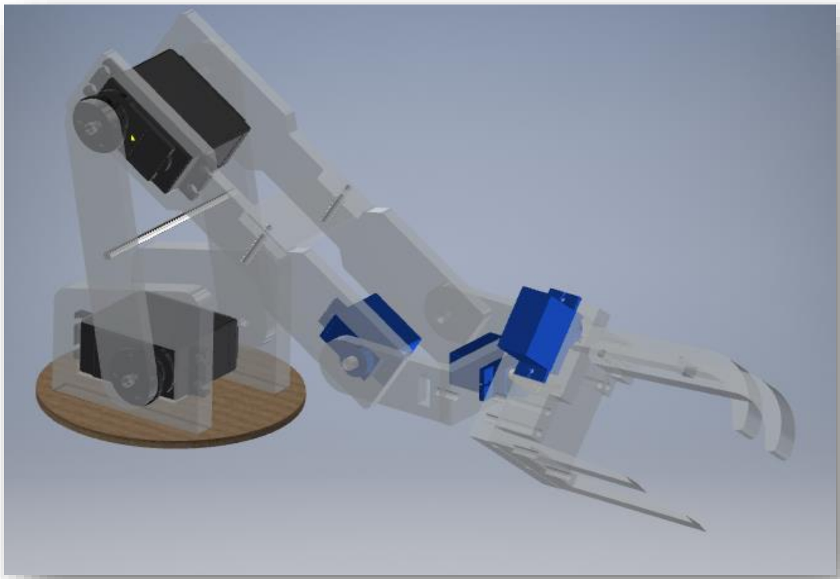
Front view

2.Robotic arm

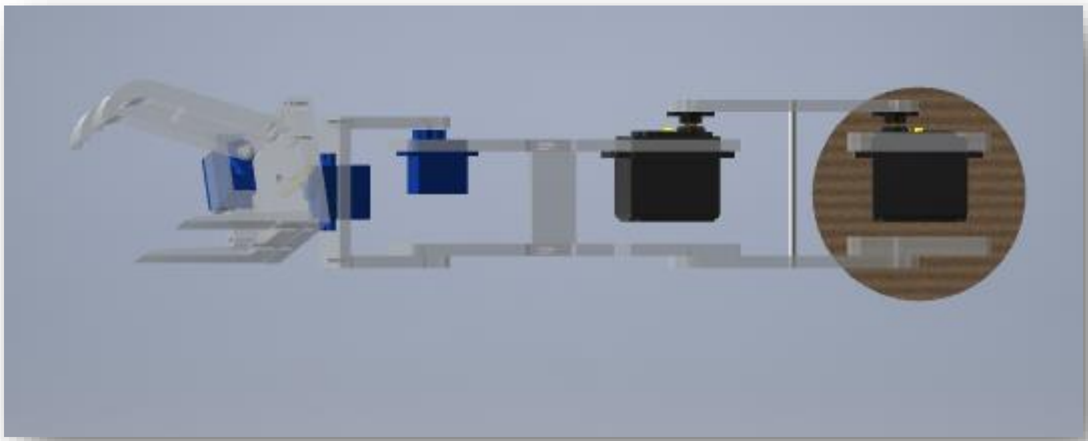
The robotic arm features a multi-jointed design for enhanced inflexibility and perfection. It is equipped with a gripper medium for object running and manipulation. The structure is optimized for stability and effective movement, making it suitable for robotization and robotic operations.



Side view



Front view


Top view

3. Electrical & Electronic System

3.1 Overview

The electrical design of the robot is developed with a modular approach to ensure reliability, ease of integration, and optimal performance. The design integrates power distribution, motor control, sensor interfacing, wireless communication, and a complex robotic arm while maintaining strict insulation between subsystems.

3.2. System Overview

The robot is built on a chassis made of two precision-cut acrylic pieces fastened together using screws (thus avoiding soldered joints for easy maintenance). The overall electrical system is powered by a single battery source that supplies energy to:

- Four DC motors for propulsion.
- Two Arduino microcontrollers operating in distinct roles.
- An array of sensors (IR for line following, ultrasonic for obstacle detection).
- An HC-05 Bluetooth module for remote control.
- A 5 DOF robotic arm equipped with six servos (three SG-90 and three MG-996R) for pick-and-place tasks.
- H-Bridge for controlling and distributing the voltage over DC motors and protecting them from high voltage.

The design emphasizes proper current distribution, voltage regulation, and signal isolation to avoid interference between the two Arduinos and between the various subsystems.

3.3. Power Distribution & Battery Calculations

Battery Selection and Capacity:

- Motors:
  Each of the four DC motors draws approximately **1 A** under normal load. Under stall conditions, a motor might reach **1.5 A**.
  - Normal operation: 4 motors × 1 A = **4 A**
  - Peak load: 4 motors × 1.5 A = **6 A**
- Robotic Arm Servos:
  - SG-90 servos: ~0.5 A each under load → 3 × 0.5 A = **1.5 A**
  - MG-996R servos: ~1.2 A each under load → 3 × 1.2 A = **3.6 A**
  - Total for servos: 1.5 A + 3.6 A = **5.1 A**
- Electronics & Sensors:
  - Each Arduino: ~**50 mA (0.05 A)**
  - 5-channel IR sensor array: ~10 mA per channel → **50 mA (0.05 A)**
  - Ultrasonic sensor: ~**15 mA (0.015 A)**
  - HC-05 Bluetooth module: ~**40 mA (0.04 A)**
  - Subtotal: ~**0.16 A** (considering both Arduinos and peripherals)
- Overall Peak Current Estimate:
  ~6 A (motors) + ~5.1 A (servos) + ~0.16 A (electronics) ≈ **11.3 A**

Since not all components will peak simultaneously and for safety, we recommend a battery with a capacity of **2200–three thousand Mah** and a discharge rating of at least 20C to manage potential transient peaks reliably.

Voltage Regulation:
A set of DC-DC buck converters is used to step down the battery voltage (which could be **7.4 V, 11.1 V**, or another suitable voltage) to a stable **5 V** for the Arduino boards, sensors, and servo control circuits. Filtering capacitors and proper decoupling techniques are applied to minimize voltage ripples.

3.4. Motor and Drive System

Chassis and Propulsion:
The robot uses four standard DC motors mounted on a two-piece acrylic chassis. The motors drive wheels that provide traction and mobility.

H-Bridge Motor Driver:

- Motor control is managed by an H-bridge circuit, which allows for bidirectional control of each motor.
- Diode Isolation: To avoid crosstalk between the two Arduino controllers, four diodes (preferably Schottky type due to their low forward voltage drop of approximately 0.3 V) are connected to the outputs of the H-bridge.
    - Calculation Example:
      For an output current of 1 A, the diode dissipates:

$$\boldsymbol{P = Vdrop \times I = 0.3\,V \times 1\,A = 0.3}$$

Selecting diodes with at least a 1 W rating provides a robust safety margin.

Current Handling and Power Dissipation:
Design calculations ensure that the wiring, PCB traces, and connectors are capable of handling peak currents of up to 6 A for the motors, with additional margins for transient conditions.

3.5. Dual Arduino Architecture & Switching Mechanism

To prevent interference and ensure specialized handling of tasks:

- **Arduino 1:**
  Manages the navigation subsystem (line following and obstacle avoidance). A mode-switching algorithm in its firmware dynamically alternates between the two tasks.
- **Arduino 2:**
  Dedicated to robotic arm and Bluetooth control. This segregation ensures that high-current servo operations and wireless communications do not affect the precision needed for navigation.

A Physical and Electrical Isolation Approach:

- **Inter-Arduino Isolation:**
  Physical switches are used to isolate the two Arduinos. Additionally, four diodes (as described in Section 3.4) are installed on the outputs of the H-bridge to ensure that the signals are unidirectional and do not interfere with the other microcontroller's input channels.
- **Mode Switching:**
  In the firmware, a software-based switch toggles between the line following/obstacle avoidance and other operational codes. This design prevents simultaneous conflicting commands, ensuring system stability.

3.6. Navigation Subsystem

Line Following:

- A 5-channel IR sensor array is used to detect contrast between the line and the background.

Current Draw:

Each channel draws ~10 mA → Total = 5 × 10 mA = **50 mA**.

- Arduino 1 processes the sensor readings to adjust motor speeds and maintain trajectory along the line.

Obstacle Avoidance:

- An ultrasonic sensor continuously measures the distance to obstacles (drawing ~15 mA).
- This sensor is mounted on a servo motor (approximately 150 mA under load) to scan a wide angle.
- The distance measurements are processed in real-time, and the algorithm calculates avoidance maneuvers by adjusting motor speeds.

3.7. Bluetooth Communication Module

- The HC-05 Bluetooth module enables remote control via a smartphone.

- Operating Voltage: 3.3V to 5V logic levels.
- Current Consumption: Approximately 40 mA.
- Integrated into Arduino 2's control loop, it ensures seamless transitions between autonomous behavior and user-controlled operation.

3.8. Robotic Arm (Pick & Place) Subsystem

The robotic arm is designed for high precision and load-bearing tasks using a 5 DOF configuration and is driven by six servo motors.

Servo Details:

- **SG-90 Servos:**
  - Used for fine, precise movements.
  - Approximate current under load: 0.5 A each.
- **MG-996R Servos:**
  - Employed where higher torture is required.
  - Approximate current under load: 1.2 A each.

Calculations:

- Total Servo Current Draw:

  $$3 \times 0.5\,A + 3 \times 1.2\,A = 1.5\,A + 3.6\,A = \mathbf{5.1\,A}$$

- Power Supply Considerations:
  Due to the high peak currents, the servo power supply is designed with low impedance wiring and robust voltage regulation. Separate decoupling capacitors are used near each servo to smooth transient current spikes.

Control Method:

- Arduino 2 controls the arm via PWM signals.
- To ensure smooth motion, the control code implements gradual ramping of the PWM values.
- Positional feedback (if sensors are integrated) is used for fine-tuning the pick-and-place operations.

3.9. Wiring, Chassis Integration & Additional Calculations

Chassis and Mechanical Integration:

- The robot’s chassis is constructed from two acrylic pieces.
- All components are mounted using screws, allowing for easy assembly/disassembly and maintenance.

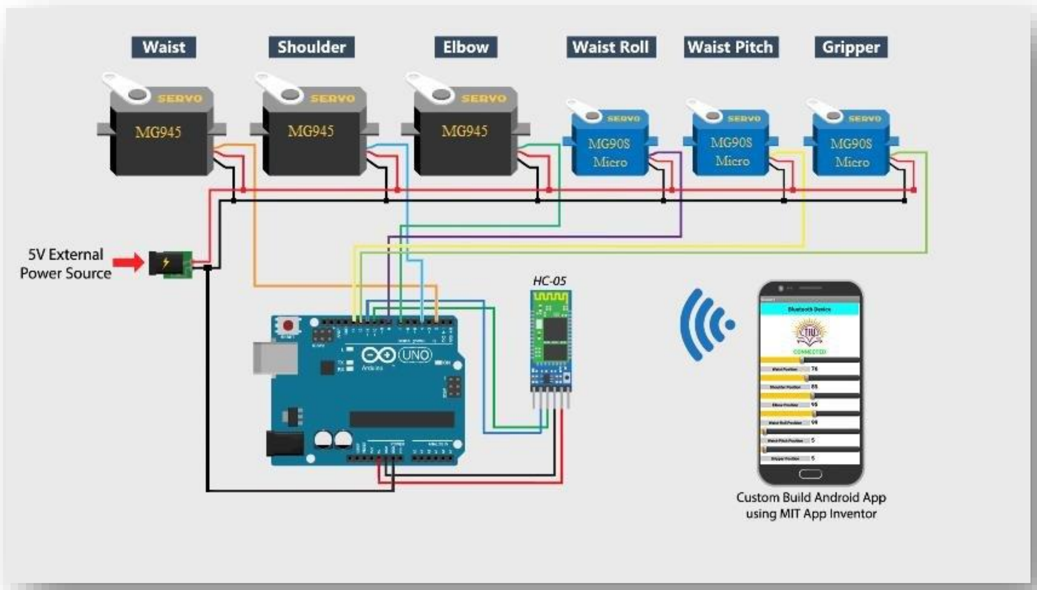
Wiring and Connector Design:

- A systematic wiring scheme minimizes noise and interference. Shielded cables are used for sensitive signals, and proper grounding practices are followed throughout the design.

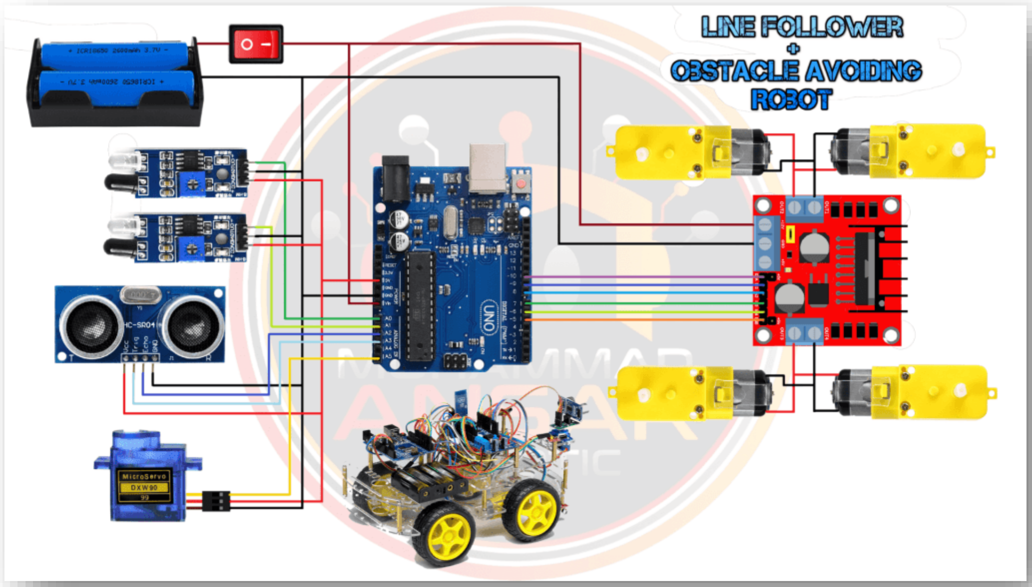
Battery Level Indicator:

- A voltage divider circuit is used to monitor battery voltage through an Arduino analog input.

3.10. Circuits Design



Arm & RC



3.11. Power Management

- Effective power distribution is essential for maintaining the robot's performance and ensuring stable operation across all factors. The power system is designed to supply harmonious voltage and current situations to the motors, sensors, and microcontrollers, while also balancing load distribution to help overheating and inefficiencies.
- **Power Distribution**: The robot utilizes a regulated power force system where each module receives the necessary voltage position. The motors operate at 12V, while the Arduino microcontrollers and detectors function at 5V, ensuring compatibility across different components.
- **Battery Management**: A rechargeable Li- ion battery pack provides the primary power source, offering a balance between energy density and weight. The system incorporates a Battery Level Indicator, allowing real-time monitoring of power levels to help unforeseen shutdowns.
- **Battery Management System (BMS)**: To enhance battery effectiveness and safety, a Battery Management System(BMS) is integrated. The BMS ensures proper charging and discharging cycles, protects against overvoltage, undervoltage, and overheating, and extends battery lifetime by balancing individual cell voltages. This prevents deep discharge scripts that could compromise battery health.

- **Load Balancing**: The motor driver(H- Bridge) distributes power to the four DC motors, ensuring each wheel receives applicable power levels for smooth locomotion. The robotic arm servos(SG- 90 and MG- 996R) are powered independently to avoid voltage oscillations affecting other systems.
- **Safety Mechanisms**: To help damage from voltage harpoons or short circuits, diodes are placed in crucial locations, particularly around the motor motorist's connections. A main power switch allows complete system shutdown when necessary.
- **3.12. Prohibited Practices:**

To ensure compliance with competition regulations and enhance system reliability, the following practices are strictly avoided:

- **No Breadboards**: Breadboards are not used in the final design to avoid loose connections and electrical instability. Instead, soldered or screw-terminal connections are used for robustness.
- **Adherence to Safety Standards**: Wiring follows industry standards for insulation and secure mounting. Electrical tape, zip ties, and dedicated mounting points are used to keep wires organized and securely fastened, minimizing risks of accidental disconnections or short circuits.

By implementing a structured power management and wiring system, the robot ensures optimal performance, longevity, and compliance with competition standards.

5. Software & Algorithms

5.1 Programming & Logic

The robot’s software is developed using C for Arduino, icing real- time control and effective resource operation. The program is structured into multiple modules managing different functionalities, including motor control, sensor processing, and communication via Bluetooth. The software follows a modular programming approach, where each function is insulated, making debugging and upgrades more effective.

 The sense is implemented using event- driven programming, where sensor data triggers specific conduct, similar as stopping the robot upon detecting an obstacle or conforming motor speeds to follow a line. Also, interrupts are used to manage real- time sensor readings, ensuring minimum detention in response to environmental changes.

5.1.1 primary code for line Follower

```
int IN1 = 3;
int IN2 = 5;
int IN3 = 6;
int IN4 = 9;

int IR1 = 11;
int IR2 = 8;
int IR3 = 7;
int IR4 = 4;
int IR5 = 2;

int sensorValues[5];

void setup() {
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);

  pinMode(IR1, INPUT);
  pinMode(IR2, INPUT);
  pinMode(IR3, INPUT);
  pinMode(IR4, INPUT);
  pinMode(IR5, INPUT);
}

void forward() {
  analogWrite(IN1, 100);
  analogWrite(IN2, 0);
  analogWrite(IN3, 100);
  analogWrite(IN4, 0);
}

void left() {
  analogWrite(IN1, 80);
  analogWrite(IN2, 0);
  analogWrite(IN3, 0);
  analogWrite(IN4, 0);
}

void turn_left() {
  analogWrite(IN1, 150);
  analogWrite(IN2, 0);
  analogWrite(IN3, 0);
  analogWrite(IN4, 0);
}

void right() {
  analogWrite(IN1, 0);
  analogWrite(IN2, 0);
  analogWrite(IN3, 80);
  analogWrite(IN4, 0);
}

void turn_right() {
  analogWrite(IN1, 0);
  analogWrite(IN2, 0);
  analogWrite(IN3, 150);
  analogWrite(IN4, 0);
}

void readSensors() {
  sensorValues[0] = digitalRead(IR1);
  sensorValues[1] = digitalRead(IR2);
  sensorValues[2] = digitalRead(IR3);
  sensorValues[3] = digitalRead(IR4);
  sensorValues[4] = digitalRead(IR5);
}

void loop() {
  readSensors();
  if (sensorValues[2] == 0) {
    forward();
  } else if (sensorValues[0] == 0) {
    turn_left();
  } else if (sensorValues[1] == 0) {
    left();
  } else if (sensorValues[4] == 0) {
    turn_right();
  } else if (sensorValues[3] == 0) {
    right();
  }
}
```

5.1.2 primary code for Obstacle Avoidance

```
<include <Servo.h#

;int IN1 = 3
```

```
;int IN2 = 5
;int IN3 = 6
;int IN4 = 9
;int trig = 2
;int echo = 4
;float time
;float distance
;float distance_left
;float distance_right
;Servo motor

} ()void setup
;pinMode(IN1, OUTPUT)
;pinMode(IN2, OUTPUT)
;pinMode(IN3, OUTPUT)
;pinMode(IN4, OUTPUT)
;pinMode(trig, OUTPUT)
;pinMode(echo, INPUT)
;(10)motor.attach
;(9600)Serial.begin
{

} ()void forward
;analogWrite(IN1, 100)
;analogWrite(IN2, 0)
;analogWrite(IN3, 100)
;analogWrite(IN4, 0)
{

} ()void backward
;analogWrite(IN1, 0)
;analogWrite(IN2, 100)
;analogWrite(IN3, 0)
;analogWrite(IN4, 100)
{

} ()void turn_right
;analogWrite(IN1, 90)
;analogWrite(IN2, 0)
;analogWrite(IN3, 0)
;analogWrite(IN4, 90)
{

} ()void turn_left
;analogWrite(IN1, 0)
;analogWrite(IN2, 90)
;analogWrite(IN3, 90)
;analogWrite(IN4, 0)
{

} ()void stop
;analogWrite(IN1, 0)
;analogWrite(IN2, 0)
;analogWrite(IN3, 0)
```

```
;analogWrite(IN4, 0)
;(500)delay
{

} ()float USS
;digitalWrite(trig, LOW)
;(2)delayMicroseconds
;digitalWrite(trig, HIGH)
;(10)delayMicroseconds
;digitalWrite(trig, LOW)

;time = pulseIn(echo, HIGH, 30000)
} if (time == 0)
;return 400
{

;distance = time * 0.034 / 2
;return distance
{

} ()void loop
;(90)motor.write
;()float current_distance = USS

} if (current_distance > 20)
;()forward
} else {
;()stop
;(1000)delay

قياس المسافة إلى اليسار //
;(0)motor.write
;(1000)delay
;()distance_left = USS

قياس المسافة إلى اليمين //
;(180)motor.write
;(1000)delay
;()distance_right = USS

;Serial.print("Distance Left: ")
;Serial.println(distance_left)
;Serial.print("Distance Right: ")
;Serial.println(distance_right)

} if (distance_left > 20 && distance_right > 20)
} if (distance_left > distance_right)
;()turn_left
;(700)delay
} else {
;()turn_right
;(700)delay
{
} (else if (distance_left > 20 {
```

```
;()turn_left
;(700)delay
} (else if (distance_right > 20 {
;()turn_right
;(700)delay
} else {
إذا كان هناك عائق في كل الاتجاهات، قم بالتراجع مع الدوران //
;()backward
;(1500)delay

;()stop
;(500)delay

اختيار اتجاه عشوائي للدوران بعد التراجع //
} if (random(2) == 0)
;()turn_left
} else {
;()turn_right
{
وقت أطول للدوران حتى يغير مساره // ;(1200)delay

;()stop
;(500)delay
{

;()forward
{
{
```

5.1.3 primary code for Remotely Controlled Arm

```
<include <Servo.h#

تعريف السيرفوهات //
;Servo motor_1, motor_2, motor_3, motor_4, motor_5, motor_6

تعريف المحركات //
define IN1 3#
define IN2 5#
define IN3 6#
define IN4 9#

زوايا البداية لكل سيرفو //
;int servo1 = 70
;int servo2 = 180
int servo3 = 40;  // Elbow position
;int servo4 = 100
;int servo5 = 60
مغلق بالكامل عند البدء //  ;int servo6 = 0

متغير استقبال بيانات البلوتوث //
;int data
```

```
;int Speed = 130

} ()void setup
;(9600)Serial.begin

ربط السيرفوهات بالأطراف //
;(2)motor_1.attach
;(4)motor_2.attach
motor_3.attach(7);  // Elbow servo
;(8)motor_4.attach
;(10)motor_5.attach
11 القماشة متوصلة على البن //  ;(11)motor_6.attach

ضبط مواضع البداية //
;motor_1.write(servo1)
;motor_2.write(servo2)
motor_3.write(servo3);  // Set initial position for Elbow
;motor_4.write(servo4)
;motor_5.write(servo5)
;motor_6.write(servo6)

تعريف أطراف المحركات //
;pinMode(IN1, OUTPUT)
;pinMode(IN2, OUTPUT)
;pinMode(IN3, OUTPUT)
;pinMode(IN4, OUTPUT)

;(1000)delay
{

} ()void loop
استقبال البيانات من البلوتوث //
} if (Serial.available() > 0)
;()data = Serial.read
;Serial.print("Received: ")
;Serial.println(data)

تحويل قيمة السلايدر إلى نطاق 0-180 والتحكم بالقماشة //
} if (data >= 20 && data <= 255)
= 180 ،مغلق بالكامل = servo6 = map(data, 20, 255, 0, 180); // 0
مفتوح بالكامل
;motor_6.write(servo6)
;Serial.print("Servo 6 Position: ")
طباعة القيمة للتأكد // ;Serial.println(servo6)
{
{

التحكم بالسيرفوهات //
if (data == 8 && servo1 < 180) { servo1++; motor_1.write(servo1); }
else if (data == 9 && servo1 > 0) { servo1--; motor_1.write(servo1); }
else if (data == 10 && servo2 > 0) { servo2--; motor_2.write(servo2); }
else if (data == 11 && servo2 < 180) { servo2++; motor_2.write(servo2);
}
```

```
else if (data == 12 && servo3 > 0) { servo3--; motor_3.write(servo3);
}  // Elbow control
else if (data == 13 && servo3 < 180) { servo3++; motor_3.write(servo3);
}  // Elbow control
else if (data == 14 && servo4 < 180) { servo4++; motor_4.write(servo4);
}
else if (data == 15 && servo4 > 0) { servo4--; motor_4.write(servo4); }
else if (data == 16 && servo5 > 90) { servo5--; motor_5.write(servo5); }
else if (data == 17 && servo5 < 150) { servo5++; motor_5.write(servo5);
}

;(30)delay

} if (Serial.available())
;()data = Serial.read
;Serial.println(data)
{
} if (data == 1)
;analogWrite(IN1, 255)
;analogWrite(IN2, 0)
;analogWrite(IN3, 255)
;analogWrite(IN4, 0)
{

} else if (data == 2)
;analogWrite(IN1, 0)
;analogWrite(IN2, 255)
;analogWrite(IN3, 0)
;analogWrite(IN4, 255)
{
} else if (data == 4)
;analogWrite(IN1, 255)
;analogWrite(IN2, 0)
;analogWrite(IN3, 0)
;analogWrite(IN4, 0)
{
} else if (data == 3)
;analogWrite(IN1, 0)
;analogWrite(IN2, 0)
;analogWrite(IN3, 255)
;analogWrite(IN4, 0)
{

} else if(data == 5)
;analogWrite(IN1, 0)
;analogWrite(IN2, 0)
;analogWrite(IN3, 0)
;analogWrite(IN4, 0)
{

{
```

Although the current system doesn't use advanced AI, it implements sensor emulsion ways to enhance the robot’s navigation. Sensor emulsion refers to combining data from multiple sensors to improve delicacy and reliability in decision- making. In our case, we integrate.

• IR sensors for line following These sensors descry variations in face colors, relating black and white paths.

• Ultrasonic sensors for obstacle avoidance The ultrasonic sensor, mounted on a servo motor, scans the surroundings to descry obstacles in real time.

The robot’s sense is grounded on threshold- grounded decision- making, where sensor readings determine motor adaptations. For example, if the IR sensor detects a divagation from the line, the software adjusts the wheel speeds to correct the line. The ultrasonic sensor also triggers immediate stopping and path recalibration if an obstacle is detected.

5.3 Control System

The control system governs the robot’s movement and manipulation tasks through a combination of structured sense and predefined movement sequences. The control strategies enforced include.

• Predefined Threshold- Grounded Line Following the IR sensor readings are continuously covered and grounded on the intensity difference between the left and right sensors, the software stoutly adjusts the motor speeds. of using PID, a commensurable correction mechanism is employed, where a divagation from the centerline results in discrimination wheel speeds to maintain the correct line.

• Finite State Machine(FSM) for Mode Switching The robot operates in multiple modes, including line following, obstacle avoidance, and Bluetooth remote control. A finite state machine(FSM) is enforced to allow flawless transitions between these modes. Each mode has specific entry and exit conditions touched off by sensor data or stoner inputs.

• Servo Position Control for Robotic Arm The 5- DOF robotic arm follows predefined movement sequences, where each servo motor moves to a set angular position for precise pick- and- place operations. The servos are controlled using PWM signals, allowing smooth and stable movement. The software includes error- checking mechanisms to help overload or exceeding safe operating angles.

5.4 Communication & Remote Control

The robot incorporates an **HC-05 Bluetooth module**, allowing remote operation via a mobile application. The software includes:

- **Serial communication protocols** for transmitting commands between the mobile device and the Arduino.
- **Command-based movement execution**, where user inputs determine robot motion and arm actions.
- **Safety overrides**, ensuring that manual control does not interfere with automated safety features such as obstacle avoidance.

6. Testing & Validation

A rigorous, multi-phase testing and confirmation strategy was enforced to ensure the robot’s performance, trustability, and safety. This process combined expansive simulation tests with real- world field trials and an iterative enhancement process. Each stage handed precious perceptivity that was used to upgrade both the hardware and software, ensuring that the robot met or exceeded its design specifications.

6.1. Simulation Tests

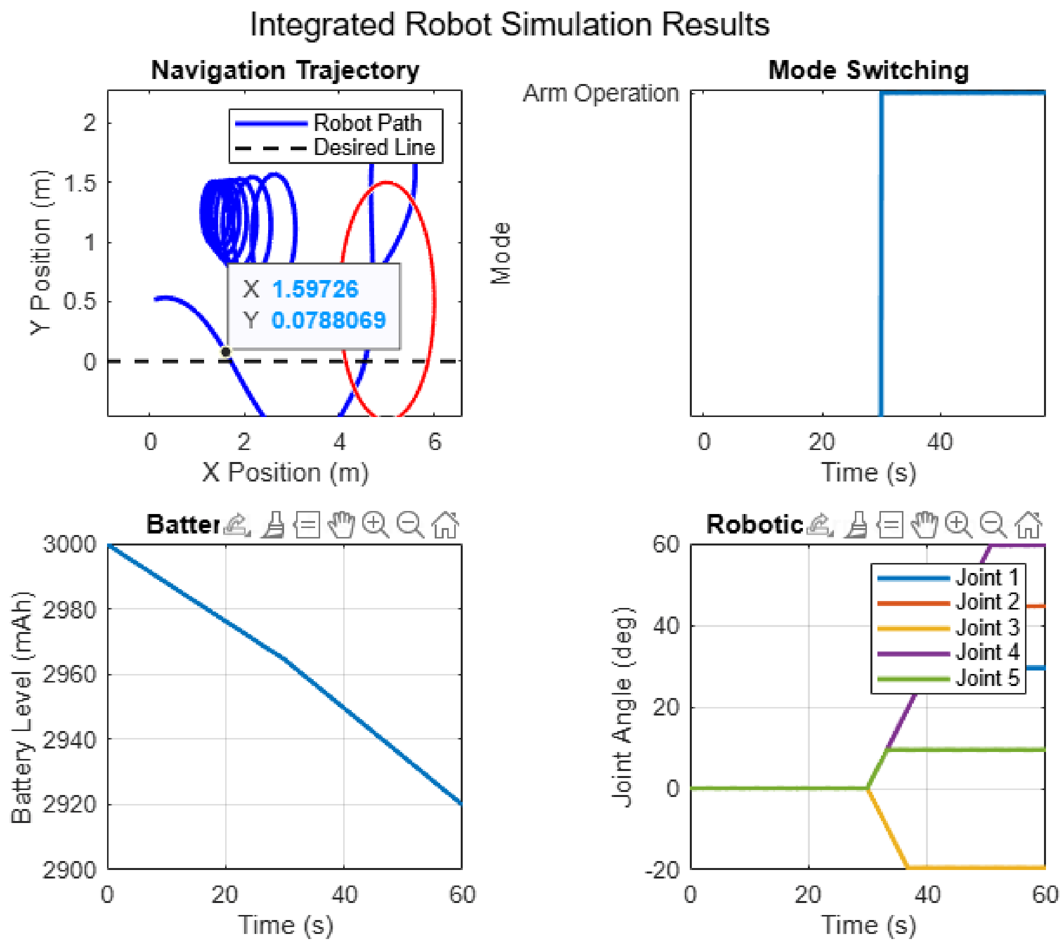
6.1.1. Virtual Modeling and Simulation Environment

Software Tools:
Model Components:

- **Mechanical Simulation:** Detailed CAD models of acrylic chassis, motor mounts, and the 5 DOF robotic arm were imported. Kinematic and dynamic analyses provided insights into chassis behavior under various load conditions.
- **Electrical Simulation:** Comprehensive circuit schematics—including the H-bridge motor driver, power distribution network, sensor interfaces (IR sensor array and ultrasonic sensor), and dual Arduino setups—were modeled. Critical components like DC-DC buck

converters, voltage divider circuits, and diode isolation networks were incorporated with real datasheet parameters to simulate realistic behavior.

- **Control Algorithms:** Virtual implementations of the control logic for line following, obstacle avoidance, and robotic arm operations were embedded into the simulation.



This figure illustrates the results of the integrated robot simulation, including navigation trajectory, battery level over time, mode switching, and robotic arm joint angles. By using MATLAB

6.1.2. Simulation Scenarios and Performance Metrics

- **Motor Dynamics and H-Bridge Evaluation:**

  - **Evaluate Conditions:** Simulations modeled acceleration, deceleration, and turning maneuvers under varying load conditions. The motor dynamics were analyzed in conjunction with the H-bridge performance to observe current draw, transient voltage drops, and torque distribution.
  - **Thermal Analysis:** Simulated heat dissipation in the H-bridge and diode isolation network ensured that junction temperatures remained within safe limits under peak load conditions (up to 6 A per motor circuit).
- **Sensor Fusion and Navigation:**
  - **Line Following:** The 5-channel IR sensor array was evaluated under different track widths, contrasts, and ambient lighting conditions. Parameters such as sensor response times and threshold levels were fine-tuned through simulation.
  - **Obstacle Avoidance:** The ultrasonic sensor’s range and accuracy were evaluated with virtual obstacles of varying sizes and materials. Simulations measured the response time (ideally below 100 MS) from obstacle detection to motor command adjustments, and the servo’s angular scanning profile was optimized accordingly.
- **Robotic Arm Performance:**
  - **Kinematics and Dynamics:** The 5 DOF arm, driven by six servos (3 SG-90 and 3 MG-996R), was simulated to execute complex pick-and-place motions. The virtual model optimized PWM signal ramp profiles, angular acceleration, and deceleration to ensure smooth, precise movements.
  - **Load Handling:** Virtual loads were applied to verify that the servo torque outputs were sufficient for the intended pick-and-place tasks without overloading the MG-996R or SG-90 servos.
- **Power Management & BMS Integration:**
  - **Battery Discharge Profiles:** The battery was simulated under various conditions, including simultaneous high-current draws from motors

and servos. The charging circuit and BMS were evaluated for effective overcharge, over-discharge, and thermal protection.
    - **Voltage Regulation:** Efficiency and ripple voltages of the DC-DC buck converters were modeled to ensure a stable 5 V output for sensitive electronics, even during rapid load changes.

6.1.3. Simulation Outcomes

- **Identification of Weaknesses:**
    - Transient voltage dips were detected during simultaneous high-current events, leading to increased decoupling capacitor values and revised wiring layouts.
    - Thermal simulations indicated the need for additional heat sinks on the H-bridge to prevent overheating.
- **Algorithm Refinement:**
    - Adjustments to sensor thresholds improved the robot's line-following accuracy.
    - The obstacle avoidance algorithm was enhanced to reduce reaction delays by optimizing the servo scanning speed.
- **Component Verification:**
    - Diode ratings and wiring gauges were confirmed to manage the expected current loads without a significant voltage drop, ensuring the overall integrity of the design.

6.2. Field Testing

6.2.1. Testing Environment and Setup

- **Evaluate Tracks and Real-World Scenarios:**
    - **Indoor Environment:** A controlled indoor track with marked lines and strategically placed obstacles was set up to simulate navigation challenges.

    - **Outdoor Environment:** Field tests in outdoor settings assessed the robot's performance under variable lighting, surface textures, and weather conditions.
    - **Obstacle Courses:** Custom courses featuring random obstacle placements were used to evaluate the real-time responsiveness of the ultrasonic sensor and the efficacy of the navigation algorithms.
- **Instrumentation and Data Logging:**
    - Onboard sensors, data loggers, and high-speed cameras captured key metrics such as motor currents, battery voltage, sensor outputs, and system response times.
    - Temperature sensors monitored critical components (e.g., the H-bridge and BMS) during extended operations.

6.2.2. Field Test Procedures and Measured Parameters

- **Navigation & Obstacle Avoidance:**
    - **Path Accuracy:** The robot’s ability to follow the designated line was assessed by measuring deviations and recovery times when encountering obstacles.
    - **Response Times:** The interval between obstacle detection and the initiation of avoidance maneuvers was recorded, with an acceptable target below 100 Ms.
    - **Environmental Robustness:** Testing under various ambient lighting conditions validated the IR sensor array’s reliability and the system’s adaptive performance.
- **Remote Control and Autonomous Switching:**
    - **Bluetooth Performance:** The HC-05 module was evaluated for communication range, latency, and connection stability, with successful operations observed up to a 10-meter range.
    - **Mode Transition:** The responsiveness of the switching mechanism between autonomous navigation and manual control was rigorously evaluated and fine-tuned based on operator feedback.
- **Robotic Arm Operations:**

    - **Task Repetition:** The pick-and-place operations were repeated multiple times to assess consistency, precision, and reliability.
    - **Load Testing:** Different object weights and sizes were used to ensure the arm could manage varying torque requirements without performance degradation.
    - **Motion Smoothness:** Real-time feedback and high-speed recordings verified that PWM signal adjustments resulted in fluid and controlled arm movements.
- **Power System Evaluation:**
    - **Extended Run-Time Tests:** Continuous operation over several charge/discharge cycles confirmed the efficiency of the charging circuit and the accuracy of the battery level indicator.
    - **Voltage Stability:** Under peak loads, the DC-DC buck converters maintained a stable 5 V supply, ensuring proper operation of all electronic components.
    - **Thermal Management:** Temperature measurements confirmed that the implemented cooling strategies effectively managed component heat during prolonged operation.

6.2.3. Field Testing Outcomes

- **Performance Verification:**
    - The robot demonstrated robust navigation capabilities, reliably following the line, and avoiding obstacles in both indoor and outdoor environments.
    - Bluetooth connectivity and mode-switching functions were stable, ensuring seamless transitions between autonomous and remote-controlled operations.
    - The pick-and-place arm performed accurately and consistently, with the servo operations remaining smooth even under variable loads.
- **Areas for Improvement:**
    - Minor sensor calibration tweaks were necessary to optimize performance under extreme lighting conditions.

        - Occasional transient power fluctuations were observed during simultaneous high-current events, prompting further enhancements in decoupling and wiring arrangements.

6.3. Improvements & Iterations

6.3.1. Iterative Development Process

- **Feedback Collection:**
    - Data from simulation tests and field trials were meticulously logged, highlighting discrepancies between expected and observed performance.
    - A systematic feedback loop ensured that all issues were documented, analyzed, and addressed in subsequent design iterations.
- **Targeted Modifications:**
    - **Algorithm Enhancements:**
        - Sensor thresholds for the IR array were adjusted based on ambient light data.
        - The obstacle avoidance algorithm was refined to reduce the servo scanning delay and improve reaction times.
    - **Hardware Adjustments:**
        - Additional decoupling capacitors were installed near critical components to further stabilize voltage under peak conditions.
        - Revised wiring schemes and enhanced heat sinks were implemented to address transient power fluctuations and thermal issues.
    - **Power Management Tuning:**
        - The voltage divider for the battery level indicator was recalibrated to ensure full utilization of the ADC range.
        - BMS firmware updates provided more precise protection thresholds against overcharge and over-discharge conditions.

6.3.2. Iterative Testing Cycles

- **Cycle Execution:**
  - After each round of modifications, targeted simulation tests were run to validate improvements before conducting additional field tests.
  - Each cycle was documented with detailed test results, component adjustments, and performance benchmarks, creating a comprehensive record of development progress.
- **Documentation & Version Control:**
  - Detailed records of all hardware and software changes were maintained. Version control systems ensured that any modifications could be tracked, reviewed, and reversed if necessary.
  - Final validation reports were generated after each iteration, summarizing the improvements, and outlining any remaining issues to be addressed.

6.3.3. Outcome and Scalability

- **Enhanced Performance:**
  - The iterative process resulted in a highly refined system with robust autonomous navigation, precise robotic arm operation, and stable remote control.
  - Power management improvements led to a reliable system capable of handling high-current demands without significant voltage drops.
- **System Robustness and Future Enhancements:**
  - The extensive testing and iterative refinement process has produced a resilient and scalable robotic platform.
  - Lessons learned and documented improvements provide a solid foundation for future enhancements and potential feature expansions.

7. Project Management & Budgeting

Effective project management played a crucial role in ensuring the successful execution of our robotic system. This section outlines the structured approach we followed, covering the project timeline, budgeting considerations, and team roles.

7.1 Project Timeline

The project was divided into four structured weeks, ensuring a smooth workflow and efficient time management. Each phase was designed to systematically develop the robot while optimizing cost and performance.

Week	Milestones	Tasks Completed
Week 1: Research & Ideation	Team Formation & Initial Research	- Assembled the team and assigned roles - Conducted extensive research on robotics, control systems, and existing solutions - Identified key challenges and potential solutions
	Idea Generation & Selection	- Each team member proposed an initial concept - Conducted feasibility testing of different ideas - Selected the best approach based on performance, efficiency, and feasibility

Week 2: Design & Procurement	Component Selection & Budgeting	- Identified required electrical, mechanical, and control components - Conducted a cost vs. performance analysis to select optimal materials - Purchased necessary components while ensuring budget efficiency
	Design & Simulation	- Created detailed CAD models for the chassis and robotic arm - Simulated navigation, obstacle avoidance, and pick & place functions in MATLAB - Verified system response through virtual testing
Week 3: Assembly & Development	Hardware Integration	- Assembled the acrylic-based chassis and robotic arm - Installed sensors (IR, Ultrasonic), actuators, and control modules - Ensured proper wiring, circuit connections, and cable management

	Software Development	- Developed a mobile application using MIT App Inventor for remote control - Implemented Arduino codes for both microcontrollers - Created switching logic between different modes to avoid system conflicts
Week 4: Testing & Validation	System Integration & Final Testing	- Constructed a testing arena simulating real-world conditions - Conducted test runs for line following, obstacle avoidance, and robotic arm operations - Debugged and refined the system for optimized performance
	Final Adjustments & Documentation	- Enhanced aesthetic appeal and cable organization - Prepared detailed documentation, reports, and presentation materials - Successfully demonstrated the working prototype

7.2 Budget & Cost Analysis

Efficient budget management was essential to ensure cost-effectiveness without compromising performance. The following table provides a breakdown of the actual costs:

Component	Quantity	Cost (EGP)
Microcontrollers (Arduino Uno)	2	370 ×2 = 740
Motor Drivers (H-Bridge L298N)	1	80
DC Motors With wheel	4	45 × 4 = 180
Servo Motors (SG91R)	3	70 × 3 = 210
Servo Motors (MG996R)	3	160 × 3 = 480
IR Sensors (5-Channel Array)	1	175
Ultrasonic Sensor	1	40
Servo Motor (SG90)	1	95
Bluetooth Module (HC-05)	1	215
Battery	6	25 × 6 = 150
BMS Circuit	1	130
Acrylic Chassis (Two layers)	1	170
ARM	1	750
Miscellaneous (Wiring, Connectors, Switches, Indicators, etc.)	Multiple	150
Ultrasonic sensor holder	1	11
Total	---	3576

Through careful material selection and component sourcing, we reduced costs while maintaining high efficiency.

7.3 Team Roles & Contributions

A well-structured team successfully conducted the project, where each member played a crucial role in both technical and managerial aspects.

Team Member	Role	Key Responsibilities
Moataz Rafik	Project Manager & Documentation Specialist	- Managing workflow, deadlines, and overall project execution - Preparing detailed project reports and presentations - Creating technical documentation, diagrams, and budgeting analysis - Ensuring clarity and professionalism in reporting
Adham Abd El-Fattah	Hardware & Electronics Engineer	- Designing and assembling electrical circuits - Handling power distribution and battery management - Ensuring integration between hardware components

		- Managing sensors, motors, and connectivity modules
Omar Ayman	Software & Control Systems Engineer	- Writing Arduino and MATLAB simulation codes - Implementing control algorithms for movement and navigation - Debugging and optimizing system response - Managing Bluetooth communication and sensor interfacing
Hazem Abd El-Fattah	Mechanical Engineer & Testing Specialist	- CAD modeling and chassis assembly - Mounting servos, sensors, and mechanical components - Conducting field tests and debugging - Optimizing speed and accuracy of navigation

8. Final Results

The robot successfully achieves its intended functionalities, including independent navigation(line following and obstacle avoidance), Bluetooth- grounded homemade control, and pick- and- place operations using a robotic arm. The binary- Arduino system with a switching medium ensures flawless operation without conflicts. The H- Bridge motor driver effectively controls the four- wheel drive system, while the 5- channel IR detector and ultrasonic sensor with a servo motor enable accurate path- following and obstacle detection.

The robotic arm, equipped with six servo motors(three SG- 91R) and(three MG- 996R), performs object manipulation with perfection. The battery position indicator and integrated charging circuit enhance usability, furnishing real- time monitoring and accessible recharging. The overall design, featuring a modular acrylic chassis, ensures easy assembly and conservation.

9. Conclusion

The robot demonstrates a functional, effective, and modular design, successfully integrating multiple factors to achieve independent mobility, remote control, and object- handling capabilities. The system's binary- microcontroller armature, combined with smart power distribution and switching mechanisms, prevents conflicts and ensures smooth operation.

Despite its success, farther advancements can improve effectiveness, continuity, and intelligence. Incorporating machine literacy for adaptive navigation, advanced materials for structural advancements, and expanded wireless communication options would elevate its capabilities. Unborn duplications could also optimize power consumption and sensor integration for better performance in different surroundings.

10. Future Improvements

1. **Enhanced AI and Machine Learning:** Implementing AI-based vision systems using a camera module for more advanced object detection and line tracking.
2. **Improved Power Efficiency:** Upgrading to a more efficient power management system to extend battery life and optimize energy distribution.
3. **Wireless Communication Expansion:** Adding Wi-Fi connectivity for remote monitoring and control via the internet.
4. **Stronger Structural Design:** Using lightweight yet durable materials such as aluminum to improve robustness without adding excessive weight.
5. **Advanced Sensor Integration:** Incorporating additional sensors like IMUs (Inertial Measurement Units) for improved stability and navigation.

These enhancements would significantly improve the robot's performance, making it more versatile and adaptable for real-world applications.

11. References

1) *(PDF) design and fabrication of robots for surveillance using Arduino. (n.d.-e). https://www.researchgate.net/publication/364661129_Design_and_Fabrication_of_Robot_for_Surveillance_using_Arduino*
2) *(PDF) development of a low-cost experimental quadcopter testbed using an Arduino controller and software. (n.d.-f). https://www.researchgate.net/publication/281144471_Development_of_a_Low-Cost_Experimental_Quadcopter_Testbed_Using_an_Arduino_Controller_and_Software*
3) *(PDF) remote and autonomous controlled robotic car based on Arduino with real time obstacle detection and avoidance. (n.d.-g). https://www.researchgate.net/publication/330702209_Remote_and_Autonomous_Controlled_Robotic_Car_based_on_Arduino_with_Real_Time_Obstacle_Detection_and_Avoidance*

4) (PDF) robot arm control with Arduino. (n.d.-h). https://www.researchgate.net/publication/317277584_ROBOT_ARM_CONTROL_WITH_ARDUINO
5) (PDF) smartphone controlled multipurpose robot car. (n.d.-i). https://www.researchgate.net/publication/341874361_Smartphone_Controlled_Multipurpose_Robot_Car
6) Alvarado, D., & Asif, S. (2024a, March 12). A framework for controlling multiple industrial robots using mobile applications. arXiv.org. https://arxiv.org/abs/2403.07639
7) Alvarado, D., & Asif, S. (2024b, March 12). A framework for controlling multiple industrial robots using mobile applications. arXiv.org. https://arxiv.org/abs/2403.07639
8) Arduino programming with MATLAB and simulink. with MATLAB and Simulink - MATLAB & Simulink. (n.d.). https://www.mathworks.com/discovery/arduino-programming-matlab-simulink.html
9) Arduino Project Hub. (n.d.). https://create.arduino.cc/projecthub/embeddedlab786/pick-and-place-robotic-arm-vehicle-0589da
10) Arduino Robotic hand: Survey paper | IEEE conference publication | IEEE xplore. (n.d.-b). https://ieeexplore.ieee.org/document/8537312
11) Bhatia, A., & Panchal, P. (2024, April 18). Development of an Arduino-based smart robot car for object detection and navigation. SSRN. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4763020#:~:text=This%20paper%20presents%20the%20design,servo%20motor%20for%20steering%20control .
12) Bluetooth controlled robot using Arduino. (n.d.-c). https://www.researchgate.net/profile/Sudha-Yelmareddy/publication/349303261_ISSN_2581-4451_C_IJRAD/links/6029484b92851c4ed5707706/ISSN-2581-4451-C-IJRAD.pdf
13) Cakir, N. K., & Guven, G. (2019). Arduino-assisted robotic and coding applications in science teaching: Pulsimeter activity in compliance with the 5E learning model. Science Activities, 56(2), 42–51. https://doi.org/10.1080/00368121.2019.1675574
14) Delivery Line Tracking Robot. (n.d.-d). https://ijsra.net/sites/default/files/IJSRA-2024-0099.pdf

15) *A framework for controlling multiple industrial robots using ... (n.d.-a).* https://dspace.lib.cranfield.ac.uk/server/api/core/bitstreams/346a57df-7ded-4919-8582-25184fda94f8/conten t

16) *IEEE CONECCT 2025. (n.d.).* https://ieee-conecct.org/

17) *KGD, N., & Instructables. (2020, July 21). Obstacle avoiding robot (Arduino). Instructables.* https://www.instructables.com/Obstacle-Avoiding-Robot-Arduino-1/

18) *Line follower robot using Arduino: Code, Circuit, components, working. Line Follower Robot using Arduino | Code,Circuit,Components. (n.d.).* https://www.rfwireless-world.com/ApplicationNotes/Line-Follower-Robot-Using-Arduino.html

19) *Mariselvan, Alain, Raza, H., Steve, Hos, Nedelkovski, D., Cardwell, D., Hjt, Plinio, Talha, Logdonio, K., Konrad, Richard, Sharma, S., Medina, B., Subhasis, Pradhan, T. R., Rishi, Shah, A., … Alexandre, J. (2022, February 17). How to build custom android app for your arduino project using MIT app inventor. How To Mechatronics.* https://howtomechatronics.com/tutorials/arduino/how-to-build-custom-android-app-for-your-arduino-project-using-mit-app-inventor/

20) *My droid robot controlled by APP INVENTOR. (n.d.).* https://appinventor.mit.edu/explore/stories/my-droid-robot-controlled-app-inventor.html

21) *rocketboss13, & Instructables. (2017, September 21). Using mit app inventor to control Arduino - the basics. Instructables.* https://www.instructables.com/Using-MIT-App-Inventor-to-Control-Arduino-the-Basi/

22) *Witty. (2016, July 9). Robotic arm: Arduino + MATLAB. Arduino Forum.* https://forum.arduino.cc/t/robotic-arm-arduino-matlab/396392