الجمهورية الجزائرية الديمقراطية الشعبية
الجمهورية الجزائرية الديمقراطية الشعبية
PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
وزارة التعليم العالي والبحث العلمي
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

University Mohamed Boudiaf - M’sila
Faculty of Technology
Department of Electronics
Sector : Electronics
Specialization: Instrumentation

Final Dissertation

In order to obtain the diploma

MASTER

Presented by :

- Heraiz Osema

- Chadi Mohamed

Theme

Object-Tracking and Obstacle-Avoiding Mobile Robot

Publicly supported on: 18/06/2025 before the jury composed of :

- Dr Zerdoumi Zohra .	**Université Med Boudiaf –M’sila**	**Président**
- Djalab Abd Elhak	**Université Med Boudiaf –M’sila**	**Rapporteur**
- Pr. LALAOUI Lahouaoui	**Université Med Boudiaf –M’sila**	**Co- Rapporteur**
- Dr Khalfa Ali	**Université Med Boudiaf –M’sila**	**Examinateur**

The memoir contains, in order of appearance:

- **Acknowledgements**
- **List of Tables**
- **List of Figures**
- **Abstracts**

Table of Contents:

- General Introduction
- Chapter 1, 2, 3
- (Each chapter begins with an Introduction and ends with a Conclusion)
- General Conclusion
- Bibliography
- Appendix

Formatting Style:

- Font: Times New Roman, size 12
- Titles in **bold**
- Line spacing: 1.5
- Equations and Figures are numbered by chapter:
    - Example: Fig. Chapter Number.Figure Number → (Fig. I.1) or (Eq. I.1)

Acknowledgements

First and foremost, we would like to express our deepest gratitude to Almighty God for granting us the strength, patience, and determination to complete this work.

We extend our sincere thanks to our supervisor, Dr. Lahouaoui Laalaoui, Professor at University Mohamed Boudiaf – M'sila, for his valuable guidance, insightful remarks, and constant support throughout this project. His trust and encouragement were instrumental in helping us move forward during every stage of our work. We are truly honored to have worked under his supervision.

We would also like to express our heartfelt appreciation to our families for their unwavering moral support, their encouragement, and their faith in us during this journey.

Our thanks also go to our friends and colleagues, especially those in the laboratory, whose advice, assistance, and motivation contributed greatly to the completion of this project.

Finally, we are grateful to the members of the jury for accepting to evaluate this modest work. Their feedback and evaluation are of great importance to us.

List of Tables

List of Figures

Chapter I – Robots Mobile

Chapter II – Components of Robot Mobile

Chapter III – Practical Implementation

ABSTRACT

This project presents the design and implementation of an autonomous mobile robot capable of tracking a predefined path and avoiding obstacles in real time. The system is built using an Arduino Uno microcontroller, DC motors driven by an L298N H-bridge module, and a combination of ultrasonic and infrared (IR) sensors for environmental perception. The ultrasonic sensor is responsible for detecting obstacles in the robot’s path, while the IR sensors enable line-following and side detection functionalities.

The robot's behavior is controlled by a decision-making algorithm embedded in the Arduino, which processes sensor data and commands the motors accordingly to ensure smooth navigation. Servo-controlled scanning is also integrated to improve environmental awareness. Simulation of the hardware circuit was conducted using Cirkit Designer to validate the wiring and logic before physical assembly.

Testing was performed in a controlled environment with various obstacle configurations and line paths. Results indicate high reliability in obstacle detection and path correction, with quick reaction times and stable motor control. This low-cost and adaptable solution can serve as a foundation for more advanced autonomous systems used in smart vehicles, warehouse automation, and educational robotics.

ملخص

يقدم هذا المشروع تصميم وتنفيذ روبوت متحرك ذاتي الحركة قادر على تتبع مسار محدد مسبقًا وتجنب العوائق آنيًا. بُني النظام باستخدام متحكم أردوينو أونو، ومحركات تيار مستمر تعمل بواسطة وحدة جسرH L298N ، ومجموعة من مستشعرات الموجات فوق الصوتية والأشعة تحت الحمراء (IR) للإدراك البيئي. يتولى مستشعر الموجات فوق الصوتية مسؤولية اكتشاف العوائق في مسار الروبوت، بينما تُمكّن مستشعرات الأشعة تحت الحمراء من تتبع المسار والكشف الجانبي.

يُتحكم في سلوك الروبوت من خلال خوارزمية اتخاذ قرار مدمجة في أردوينو، والتي تُعالج بيانات المستشعر وتُصدر الأوامر للمحركات وفقًا لذلك لضمان سلاسة الحركة. كما تم دمج المسح الضوئي المُتحكم به بواسطة محرك سيرفو لتحسين الوعي البيئي. أُجريت محاكاة لدائرة الأجهزة باستخدام برنامج Cirkit Designer للتحقق من صحة الأسلاك والمنطق قبل التجميع الفعلي.

أُجري الاختبار في بيئة مُتحكم بها مع تكوينات مختلفة للعوائق ومسارات خطوط. تُشير النتائج إلى موثوقية عالية في اكتشاف العوائق وتصحيح المسار، مع أوقات استجابة سريعة وتحكم مستقر في المحرك. يمكن أن يكون هذا الحل منخفض التكلفة والقابل للتكيف بمثابة أساس لأنظمة مستقلة أكثر تقدمًا تستخدم في المركبات الذكية وأتمتة المستودعات والروبوتات التعليمية.

Keys words: Mobile Robot, Arduino Uno, controlled environment, Cirkit Designer.

SOMMAIRE

Introduction General

General Conclusion

Introduction General

Robotics technology has significantly advanced in recent years, with mobile robots becoming essential in various industries such as healthcare, logistics, and security. These robots are designed to perform tasks autonomously, aided by advancements in sensors, artificial intelligence (AI), and machine learning. Mobile robots are especially valuable in environments that require precision and speed, allowing them to interact with and navigate dynamic surroundings without human intervention[1].

One promising application of mobile robots is object tracking, where robots autonomously follow moving targets. This capability is crucial in fields like surveillance, monitoring, and logistics. To achieve this, robots need reliable sensors to detect objects and navigate efficiently in real time. In this project, a mobile object-tracking robot with ultrasonic sensors is developed. The ultrasonic sensor allows the robot to detect objects by emitting sound waves and measuring the time it takes the waves to return after hitting an object. This data helps the robot track the object and navigate autonomously while avoiding obstacles.

The goal of this project is to design a robot capable of tracking moving objects using ultrasonic sensors, enabling it to follow a target at a constant distance while reacting to obstacles and changes in the environment. This technology can be applied in practical scenarios like warehouse management and robotic surveillance, where real-time object tracking is essential[2].

The project involves both hardware and software development. The hardware consists of a mobile platform with wheels, motors, and the ultrasonic sensor. The software will include path planning and object detection algorithms to enable the robot to track objects and avoid collisions in real time.

This approach, using ultrasonic sensors for tracking, offers advantages such as simplicity and reliability, making it a cost-effective choice for mobile robot designs. By developing a mobile object-tracking robot with these sensors, this project contributes to the growing field of autonomous systems, with potential applications in various industries.

PART I: THEORETICAL STUDY

Chapter I – **R**obots Mobile

I.1. Introduction

The rapid advancements in robotics over the past few decades have fundamentally transformed industries, economies, and societies. What was once confined to science fiction has now become a real, integral part of modern life. Robots are no longer just machines for industrial tasks, but autonomous entities capable of performing complex functions across a wide range of fields such as healthcare, transportation, manufacturing, and even customer service. The growth of artificial intelligence (AI) and machine learning technologies has contributed greatly to the evolution of robots, making them more autonomous and adaptable[3].

Mobile robots are those robots, which work as a machine, which is highly controlled by software programming that, use sensors and other technologies like Artificial Intelligence to identify their surroundings and move around its environment. Mobile robots function using a combination of artificial intelligence (AI) and physical robotic elements, such as wheels, tracks, and legs. Mobile robots are becoming increasingly popular across different business sectors. They are used to assist with work processes and even accomplish tasks that are impossible or dangerous for human workers.

The purpose of this chapter is to provide a broad understanding of Mobile robots—what they are, how they function, and how they have evolved over time. It will also examine their main components, classifications, defining characteristics, and the methods used to program them. By the end of this chapter, the reader will gain an understanding of the different types of Mobile robots, their capabilities, and the important role they play in shaping the future of work and technology[4].

I.2. Definitions

In general, the term **mobile robots** refers to all robots equipped with a mobile base.

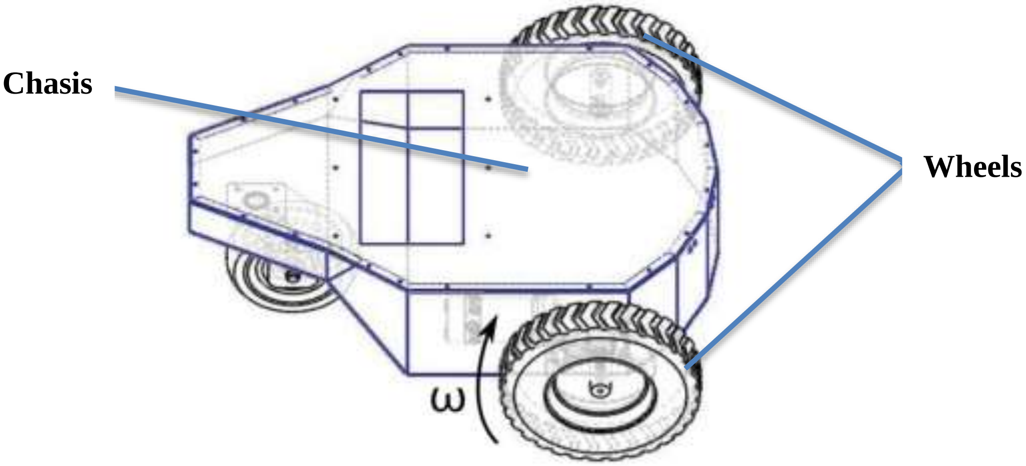


Figure I.1: the mobile robots base

These machines consist of a **chassis** and a set of **wheels**, whose function is to ensure the **stability and mobility** of the system **(Figure I.1)**. (In this context, we will focus only on **wheeled mobile robots**, excluding other types of platforms such as walking or crawling robots.)

The particularity of these robots lies in their ability to move through relatively large environments without altering their structure, thanks to their locomotion system.

I.3. HISTORICAL OVERVIEW

Robotics has evolved through several generations as follows:

- **1947**: First electrically tele-operated manipulator.
- **1954**: First programmable robot.
- **1961**: Use of the first industrial robot, commercialized by UNIMATION (USA), on a General Motors assembly line.
- **1961**: First robot with force control.
- **1963**: Use of vision to control a robot.

- **1978**: The ARGOS robot, developed at Paul Sabatier University in Toulouse (France). ARGOS simulated the navigation of a mobile robot equipped with a vision system during its movement.
- **1979**: The HILARE robot. Researchers at LAAS in Toulouse (France) studied path planning for a point-like mobile robot in a fully known environment.
- **1981**: The VESA robot, built at INSA Rennes (France), was equipped with a safety arch to detect obstacles in a completely unknown environment.
- **1984**: The **FLAKEY** robot, designed and built at the **Stanford Research Institute**, reflects the advancements made over 14 years of development. It is equipped with two motorized wheels with encoders, and its maximum speed reaches **66 cm/s**, compared to only a few centimeters per second in earlier robots. FLAKEY is capable of navigating in real-world environments.
- **1993**: The **ERRATIC** and **PIONEER** robots. ERRATIC was designed by **Kurt Konolige** at the Stanford Research Institute as a low-cost mobile robot for his robotics courses.
- **Modern Mobile Robots**: Today, most research focuses on **perception problems**, **trajectory planning**, **environment analysis and modeling**, all applied to commercial mobile robots. There is also active research in the **mechanical design** of mobile robots for highly specialized applications, such as **underwater exploration**, **flying robots**, and **micro-robots**.

I.4. Components of a Mobile Robot

A mobile robot consists of several core components that work together to enable its functionality. These components can be broadly classified into five categories: sensors, actuators, controllers, power supplies, and communication systems. This image explain this **(Figure I.2)**.

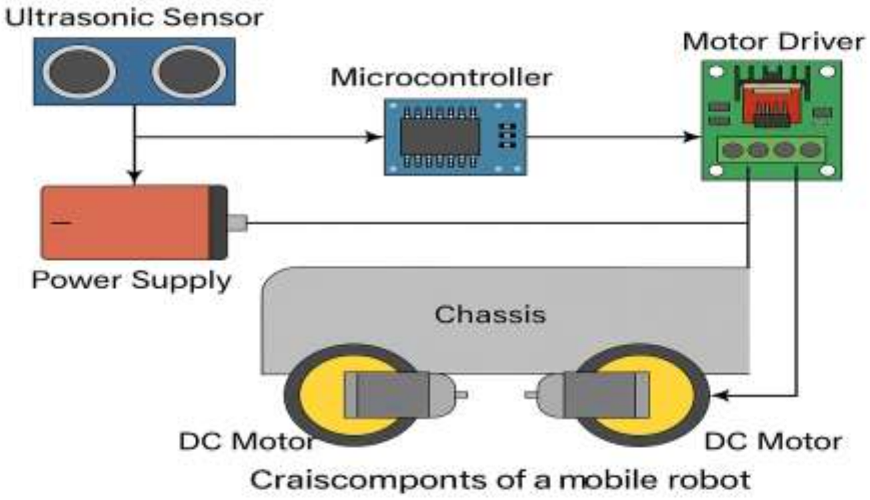


Figure I.2: Diagram showing the basic components of a mobile robot

Sensors: Sensors are crucial for enabling robots to perceive their environment. They gather data about the robot's surroundings, which the robot uses to make decisions about its actions. Some common sensors include:

- Vision sensors such as cameras and LIDAR (Light Detection and Ranging) systems help robots "see" their environment. These sensors are particularly useful in applications like autonomous vehicles and robots performing medical surgeries, where precise object detection is critical[5].
- Proximity sensors help robots detect objects and obstacles, which is essential for navigation and safety, especially in mobile robots.
- Force sensors are used in robots that handle delicate objects or need to apply specific pressure, such as robotic arms in assembly lines or in surgeries.
- Environmental sensors such as temperature and humidity sensors help robots monitor conditions in settings like laboratories, agricultural fields, or deep-sea environments.

Actuators: Actuators are the mechanical components that allow robots to perform actions. They convert electrical energy into mechanical motion, enabling robots to manipulate objects, move, and interact with their environment. There are different types of actuators, including:

- Electric motors, which are common in smaller robots, provide precise and reliable movement for tasks such as turning, lifting, or rotating.
- Hydraulic actuators, used in robots requiring high force output, such as industrial robots or large-scale machines in construction.
- Pneumatic actuators, using compressed air, are ideal for robots that need rapid, lightweight movements[6].

Controller: The controller is the brain of the robot. It processes the data from sensors and executes the commands based on its programming. The controller makes decisions and sends signals to actuators, enabling the robot to carry out its tasks. These controllers can be simple microcontrollers or more advanced embedded systems capable of complex calculations. The power of the controller determines how intelligent the robot can be, particularly in tasks requiring learning, adaptation, and problem solving[1].

Power Supply: Robots need a power source to operate. This can come in several forms:

Batteries: Most mobile robots and autonomous systems, such as drones, rely on rechargeable batteries. These batteries may include lithium-ion, lead-acid, or other energy-dense technologies.

Direct electrical connections: Industrial robots and other stationary systems are often connected directly to the power grid.

Communication Systems: Communication systems enable robots to transmit and receive data from external sources. These systems are essential for multi-robot coordination, remote operation, and real-time data analysis. Common communication technologies used in robots include:

- Wireless communication like Wi-Fi, Bluetooth, and ZigBee allow robots to communicate with other robots, control centers, or the cloud.
- Wired communication systems, such as Ethernet cables, are often used in industrial robots to ensure secure, high-speed data transfer.

I.5. Classification of Mobile Robots

Mobile robots exist in many forms, each developed with a specific function and environment in mind. From domestic assistants that clean our homes to robotic explorers traversing distant planets, the diversity in their design reflects the wide range of challenges they are meant to address. To better understand these systems, mobile robots are typically classified based on their locomotion mechanisms, level of autonomy, operational environment, intended purpose, and navigation methods[7].

This image explain this **(Figure I.2)** .

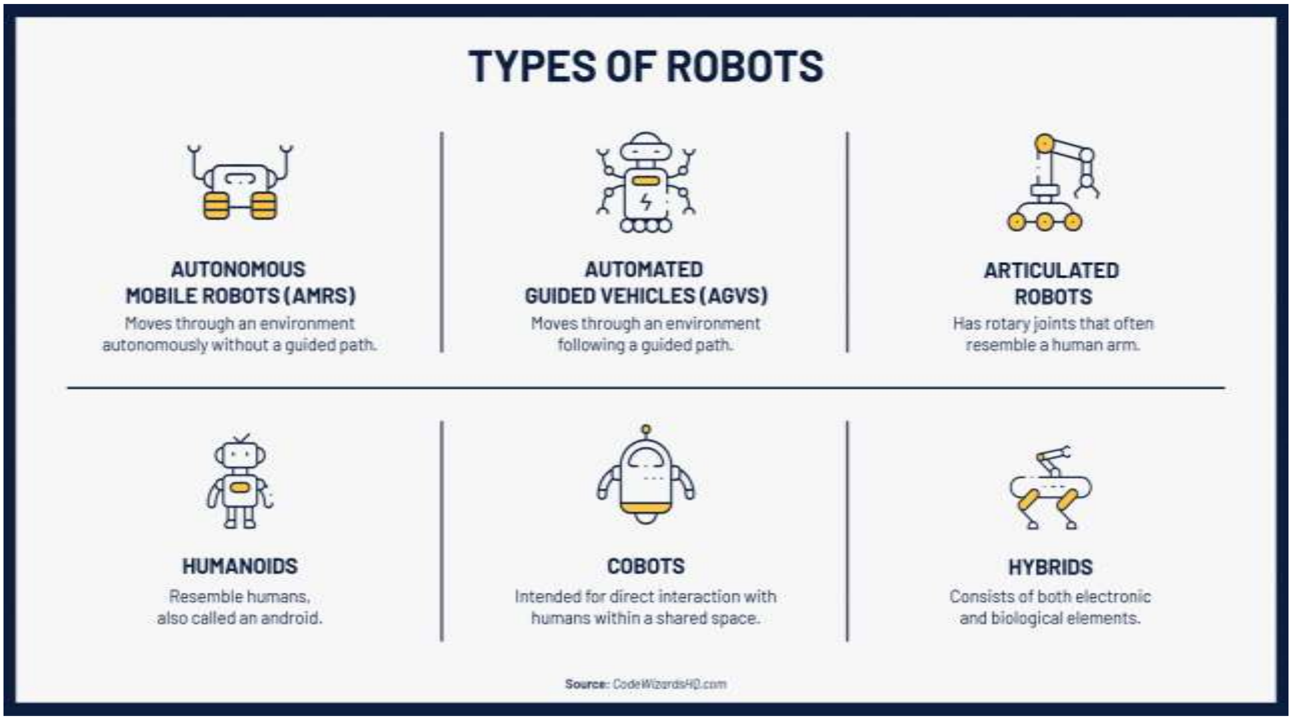


Figure I.3: Diagram showing the Classification of** Mobile **Robots

I.5.1. Locomotion Mechanisms

One of the most fundamental ways to categorize mobile robots is by how they move within their environment.

- **Wheeled Robots**

Wheeled robots are the most commonly used due to their simplicity, speed, and efficiency on smooth surfaces. They come in various configurations such as two-wheeled differential drive, three-wheeled tricycle models, and four-wheeled or omni-directional systems using specialized wheels like Mecanum wheels.

- **Legged Robots**

Inspired by biological organisms, legged robots are designed to traverse uneven or complex terrain. Bipedal robots mimic human walking, while quadrupedal or hexapod designs resemble animals or insects. These robots offer versatility in natural environments where wheels may not be effective.

- **Tracked Robots**

Utilizing continuous tracks similar to those on tanks, tracked robots provide excellent stability and traction on rough or slippery terrain. They are often deployed in military, rescue, or agricultural applications where robustness is critical.

- **Hybrid Locomotion Robots**

Some robots combine multiple forms of movement, such as wheels and legs, to increase their adaptability in dynamic or unpredictable environments. These hybrid systems are commonly found in research or exploratory missions.

I.5.2. Levels of Autonomy

Mobile robots vary widely in terms of how much decision-making is handled by the machine versus a human operator.

- **Manual Robots**

These are fully controlled by humans, often through remote systems. They are useful in hazardous environments where human presence would be risky, such as bomb disposal or underwater inspection.

- **Semi-Autonomous Robots**

These systems can perform specific tasks independently but still rely on human supervision or occasional input. They strike a balance between automation and control.

- **Fully Autonomous Robots**

Fully autonomous robots can make decisions, navigate, and perform complex tasks without direct human intervention. They rely on sensors, algorithms, and often artificial intelligence to interact with their environment intelligently.

I.5.3. Operational Environment

The environment in which it is intended to operate heavily influences the design and function of a mobile robot.

- **Indoor Robots**

Typically used in structured settings such as homes, hospitals, and factories, indoor robots must navigate around furniture, people, and narrow corridors. Examples include robotic vacuum cleaners and automated guided vehicles (AGVs) in manufacturing.

- **Outdoor Robots**

Built to endure environmental variability, outdoor robots operate in agricultural fields, urban streets, or natural terrains. These robots are rugged and often equipped with advanced sensing systems to cope with unpredictable conditions.

I.5.4. Purpose and Application

Another important aspect of classification relates to the robot’s end goal—what it is designed to accomplish.

- **Service Robots**

Designed to assist humans in domestic or commercial settings, service robots perform tasks such as cleaning, delivery, or customer interaction. They are increasingly common in homes, hotels, and hospitals.

- **Industrial Robots**

These robots are central to automation in manufacturing and logistics. Mobile industrial robots move materials, tools, or products across factory floors with high efficiency and reliability.

- **Military and Security Robots**

Used for surveillance, reconnaissance, and tactical support, these robots reduce the risk to human soldiers and enhance operational capability in dangerous environments.

- **Exploration Robots**

These include terrestrial, underwater, and space robots built to explore environments that are inaccessible or hazardous to humans. Examples include Mars rovers and deep-sea gliders.

I.5.5. Navigation and Perception Systems

A mobile robot’s ability to move purposefully in an environment depends heavily on its navigation method.

- **Line-Following Robots**

Often used in educational and industrial settings, these robots follow pre-defined paths marked by lines or magnetic strips.

- **SLAM-Based Robots**

SLAM (Simultaneous Localization and Mapping) allows robots to build a map of an unknown environment while simultaneously keeping track of their position within it. This method is commonly used in autonomous vehicles and advanced indoor robots.

- **GPS-Based Robots**

These robots use satellite positioning to navigate over large outdoor areas, making them ideal for agriculture, delivery, or surveying.

- **Sensor-Based Robots**

Employing a variety of sensors such as ultrasonic, infrared, LiDAR, or vision systems, these robots perceive their surroundings and make real-time decisions to avoid obstacles and complete tasks.

I.6. Characteristics of a Mobile Robot

Mobile robots are autonomous or semi-autonomous machines capable of navigating and performing tasks in various environments without continuous human intervention. They are designed to move within their environment and are not fixed to one physical location[7].

I.6.1. Mobility and Locomotion

The primary feature of mobile robots is their ability to move within an environment. This mobility can be achieved through various locomotion mechanisms, including:

- **Wheeled Robots**: Utilize wheels for movement, offering simplicity and efficiency on flat surfaces.
- **Legged Robots:** Employ articulated limbs, enabling traversal over uneven terrains.
- **Tracked Robots:** Use continuous tracks, providing stability and traction on rough or slippery surfaces.
- **Hybrid Designs:** Combine multiple locomotion methods to enhance adaptability in diverse environments.

I.6.2. Autonomy and Control Systems

Mobile robots vary in their level of autonomy:

- **Manual Robots:** Fully controlled by human operators, often via remote systems.

- **Semi-Autonomous Robots:** Perform specific tasks independently but require human supervision or input.
- **Fully Autonomous Robots:** Make decisions, navigate, and execute tasks without human intervention, relying on onboard sensors and algorithms.

These robots often incorporate hierarchical control structures and predefined algorithms to manage their operations.

I.6.3. Perception and Sensing Capabilities

To interact effectively with their environment, mobile robots are equipped with various sensors, such as:

- **Ultrasonic and Infrared Sensors:** Measure distances to detect obstacles.
- **Cameras and Vision Systems:** Capture visual information for object recognition and navigation.
- **GPS:** Determines the robot's global position, essential for outdoor navigation.

These sensors enable robots to perceive their surroundings, facilitating tasks like obstacle avoidance and environment mapping.

I.6.4. Navigation and Localization

Effective navigation requires robots to determine their position and plan paths to destinations. Techniques employed include:

- **Simultaneous Localization and Mapping (SLAM):** Builds a map of an unknown environment while keeping track of the robot's location within it.
- **Predefined Path Following:** Utilizes markers or wires in the environment to guide movement.
- **GPS-Based Navigation:** Employs satellite signals for positioning, suitable for outdoor applications.

Advanced robots integrate these methods to navigate complex and dynamic environments.

I.6.5. Energy Supply and Power Management

Mobile robots require onboard power sources to operate independently. Common energy solutions include:

- **Rechargeable Batteries:** Provide mobility without tethering, though they require efficient energy management to maximize operational time.

- **Alternative Power Sources:** Such as fuel cells or solar panels, depending on the application and environment.

Efficient power management is crucial to ensure that robots can perform their tasks without frequent recharging or power depletion.

I.6.6. Task-Specific Functionality

Depending on their intended applications, mobile robots may be equipped with specialized tools or features, such as:

- **Manipulators or Grippers:** For handling objects.
- **Payload Capacities:** Tailored to transport goods or materials.

These functionalities enable robots to perform tasks ranging from industrial automation to healthcare assistance.

I.7 Effectors

Here we present the different types of mobile bases used in robotics, focusing on indoor environments.

I.7.1. Differential Platforms

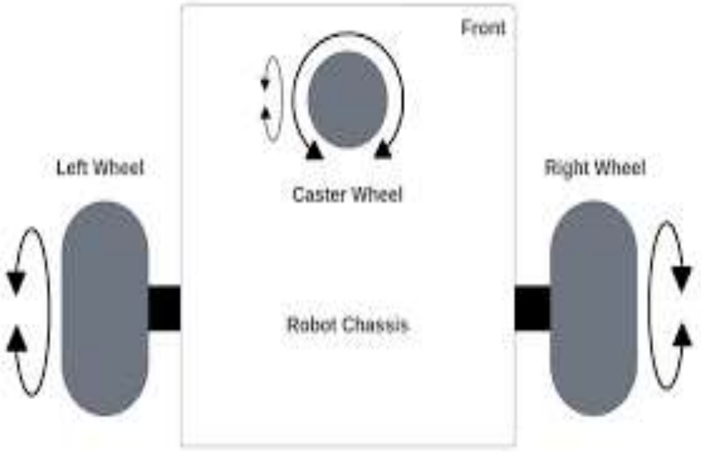


Figure I.4: Diagram showing Differential Platforms

One of the most commonly used configurations for indoor mobile robots is the differential configuration, which includes two independently driven wheels. One or more caster wheels are added at the front or rear of the robot to provide stability **(Figure I.4)** .

This platform is very simple to control, as it only requires specifying the speed of the two wheels. It also allows the robot to rotate in place, enabling it to behave like a holonomic robot—that is, one that can move in any direction from its current position (unlike a car, which is non-holonomic). This property greatly simplifies path planning and control[8].

I.7.2. Omnidirectional Platforms

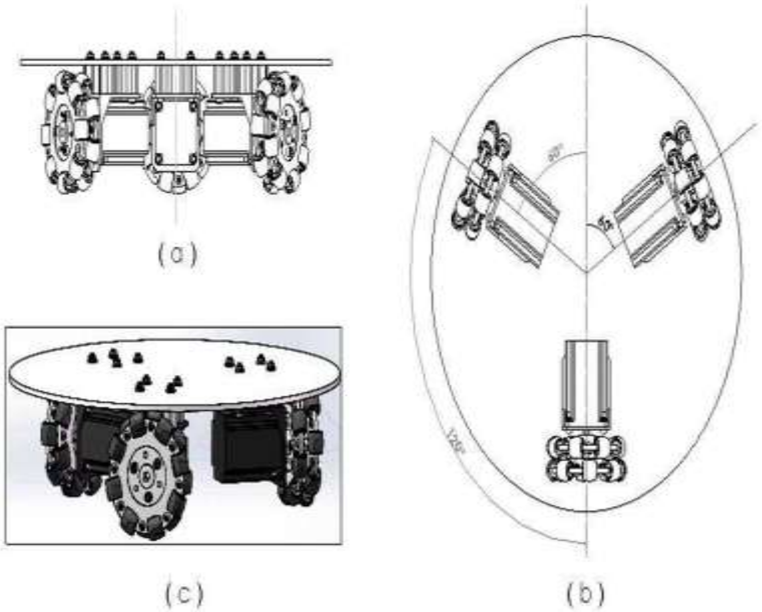


Figure I.5: Diagram showing Omnidirectional Platforms

Omnidirectional platforms allow for an even clearer separation between rotation and translation control, making them truly holonomic platforms **(Figure I.5)** . They typically use three or four wheels that rotate at the same speed to provide translation, along with a mechanism that simultaneously orients the wheels toward the desired direction of movement.

The robot's body itself does not rotate—only translations occur. This system enables simple and relatively fast control, since directional changes involve only the wheels and can happen quickly. However, these platforms have limited terrain-handling capacity and require very flat surfaces[8].

I.7.3. Non-Holonomic Platforms

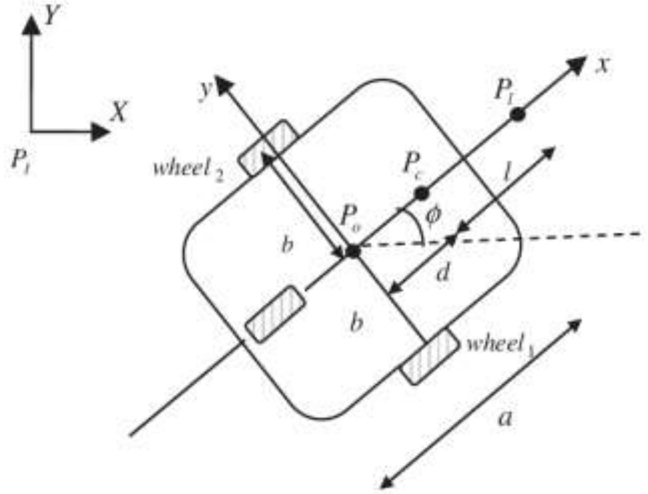
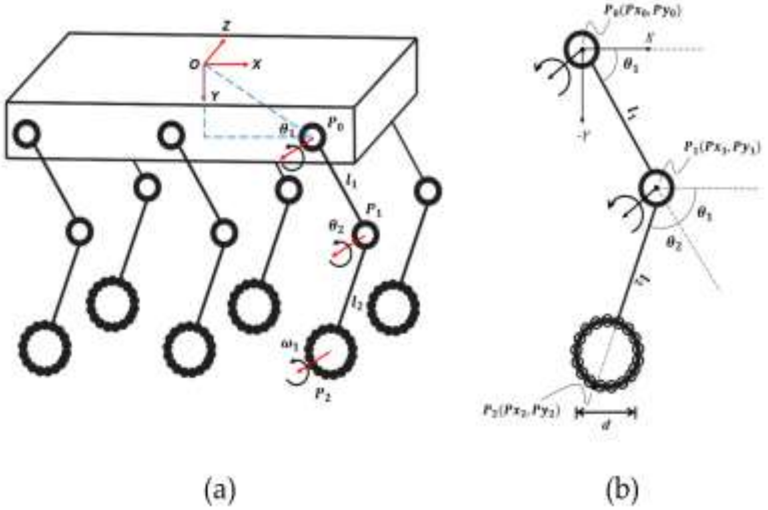


Figure I.6: Diagram showing Non-Holonomic Platforms

Non-holonomic platforms, such as car-like robots, are also used in mobile robotics. However, these are more difficult to control because they cannot turn in place and must maneuver, which can be challenging in cluttered environments. Controlling such platforms to perform specific movements is a complex problem in itself **(Figure I.6)** .

I.7.4. Legged Platforms



(a) (b)

Figure I.7: Diagram showing Legged Platforms

Platforms with two, four, or six legs can also be used. Six-legged platforms are relatively practical because the robot remains balanced at all times, making control easier. Two- or four-legged platforms are more complex to control, and simply maintaining stability and a proper gait remains difficult, making them generally slow **(Figure I.7)**.

Moreover, odometry for this type of platform is usually of low quality. These factors make such platforms rarely used—except in applications that specifically require advanced positioning and navigation capabilities[8].

I.8. Path Planning in Mobile Robots

Path planning is one of the most critical tasks in mobile robotics. It refers to the process of determining a route that a robot can follow to move safely and efficiently from its starting position to a desired destination. Depending on the complexity of the environment, this task can range from straightforward to highly challenging.

In contrast to simple obstacle avoidance which is purely reactive—path planning involves a degree of foresight. The robot must consider the layout of its surroundings, anticipate potential obstacles, and decide in advance which path to take. A well-designed planning strategy not only prevents collisions but also minimizes the distance traveled and ensures smooth, reliable motion[9].

I.8.1. Global and Local Planning Approaches

In general, path-planning strategies fall into two categories: global and local.

- **Global path planning** is used when the robot has access to a full map of the environment. Based on this map, algorithms such as A* or Dijkstra’s can compute an optimal path to the goal before the robot even begins to move. This type of planning works well in static environments where obstacles don’t change.
- **Local path planning**, on the other hand, is more dynamic and reactive. It relies on real-time sensor data—such as ultrasonic or LiDAR—to guide the robot moment by moment, helping it avoid unexpected obstacles. Local methods like the Dynamic Window Approach (DWA) are especially useful in environments where things move or where the layout is only partially known.

In most real-world applications, robots use a combination of both: the global planner provides an overall route, while the local planner handles the details and ensures that the robot responds properly to changes in its surroundings **(Figure I.8)**.

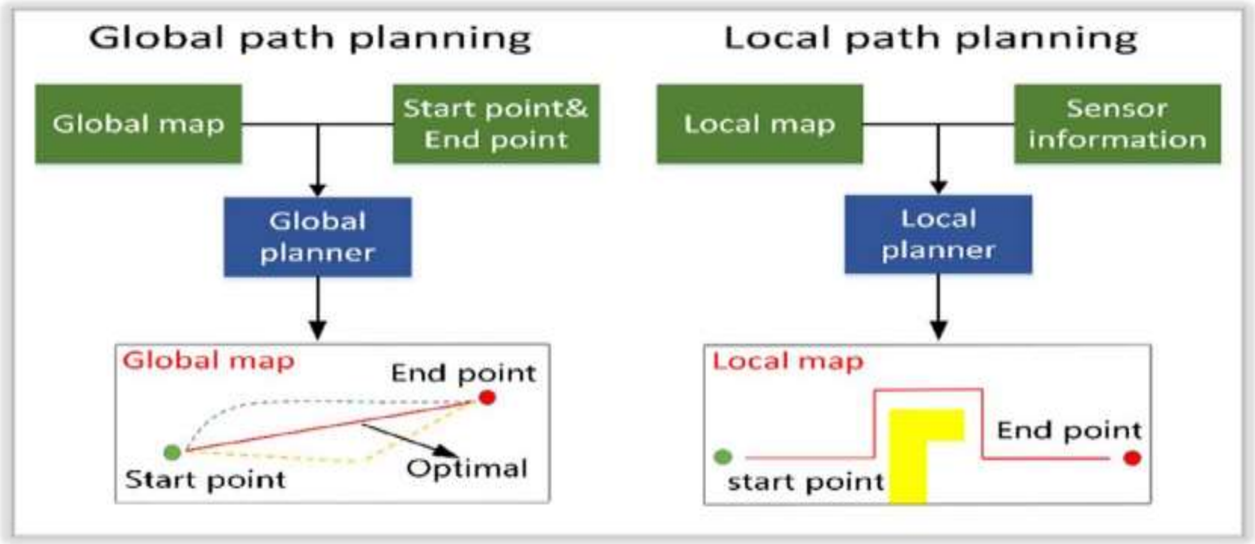


Figure I.8: Comparison of Global and Local Path Planning

I.8.2. Real-World Constraints

Path planning must take into account several constraints that affect how a robot can move:

Mechanical limitations such as minimum turning radius or restricted motion directions (especially in non-holonomic robots) **(Figure I.9)**.

Speed and acceleration limits, which affect how quickly the robot can change direction or stop.

Environmental factors, including narrow passages, uneven surfaces, and moving objects.

Ignoring these constraints can result in paths that are theoretically possible but impossible for the robot to follow in practice. That’s why modern planning systems carefully balance feasibility with efficiency[20].

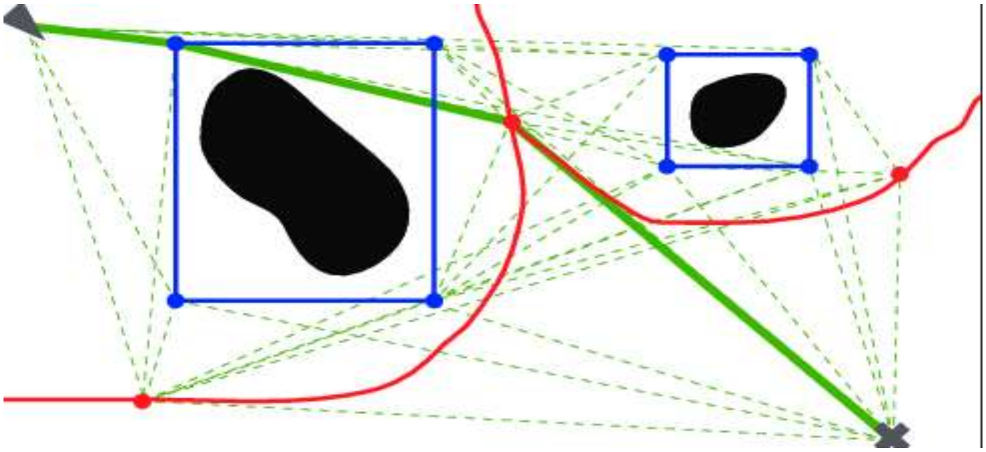


Figure I.9: Real-World Constraint-Aware Path Planning

I.8.3. Practical Implementation

In practical use such as warehouse automation, delivery robots, or autonomous service robots path planning often involves the creation of cost maps. These maps highlight areas that are risky or difficult to navigate, assigning them higher "costs" so that the planner can avoid them.

As the robot moves, it continuously updates its path using fresh sensor data **(Figure I.10)**. This ability to re-plan in real time is what allows mobile robots to navigate busy or unpredictable environments safely and efficiently[20].

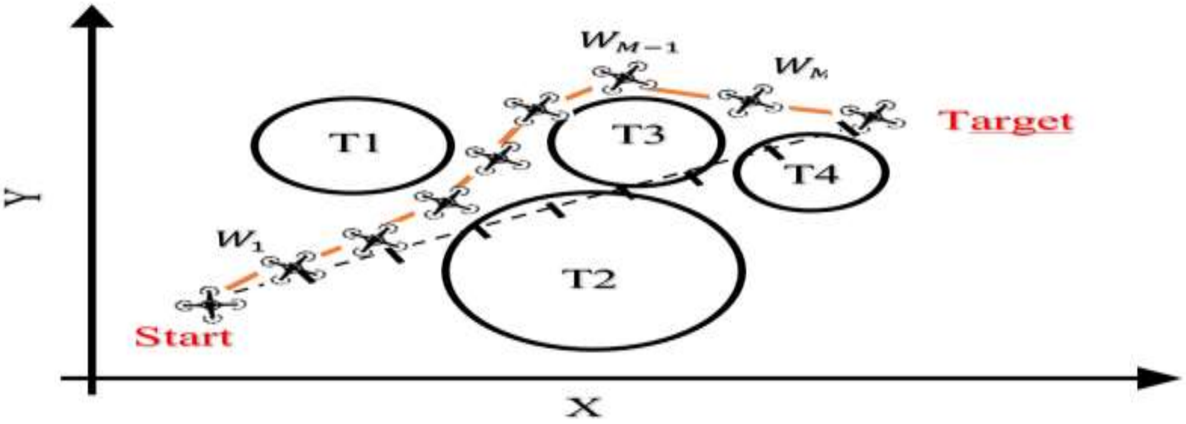


Figure I.10: Cost Map Utilization in Real-Time Path Re-Planning

I.9. Reactive Obstacle Avoiding

In this section, we present a concise overview of the main reactive obstacle avoidance methods, with the goal of highlighting their benefits and limitations in the context of our mobile robot system.

I.9.1. Analytical Methods

Navigating a mobile robot from a starting point to a desired destination is already a complex task, especially when the system is non-holonomic meaning it cannot move freely in all directions due to its mechanical constraints (like a car that cannot move sideways). Even in environments without obstacles, finding a valid trajectory remains a challenge[10].

Currently, there is no universal analytical algorithm that can solve trajectory planning for all types of non-holonomic systems. Some analytical solutions exist, but only for specific classes of robots with well-defined behaviors. For most practical cases, engineers rely on numerical or iterative approaches.

When obstacles are introduced into the environment, the situation becomes even more difficult. Analytical methods generally lose their applicability, since the presence of unpredictable or dynamic obstacles makes it nearly impossible to derive closed-form, collision-free paths for non-holonomic systems.

I.9.2. Dynamic Window Approach (DWA)

The Dynamic Window Approach is a local navigation method designed for real-time obstacle avoidance. It operates within the control space of the robot—that is, it selects safe combinations of linear and angular velocities that the robot can adopt without colliding with nearby obstacles.

What makes this method effective is that it directly takes into account the robot's kinematic constraints (like maximum speed, turning radius, and acceleration). Instead of considering all possible movements, it focuses only on those that are feasible and safe over a short time horizon.

The robot uses a cost function to evaluate different possible movements, aiming to find a trajectory that is smooth, efficient, and brings it closer to its goal. This makes the DWA particularly useful for robots in motion, especially those with limited acceleration and braking abilities, such as wheeled mobile robots.

However, one of the main limitations of the Dynamic Window Approach is its lack of flexibility in multi-robot systems. When multiple robots are involved, the local nature of this method can lead to coordination problems, as each robot acts independently without global awareness or communication[5].

I.10. Programming Mobile Robots

Programming is one of the most critical aspects of robotics, as it defines how robots behave and interact with the world. There are several approaches to programming robots, ranging from low-level programming to advanced AI-driven methods[16].

- Low-Level Programming: Low-level programming involves directly controlling the hardware of the robot. This is typically done using languages like C or C++, which provide precise control over the robot's movements and sensors. These languages allow developers to write detailed instructions that define exactly how the robot should behave.
- Robot Operating System (ROS): ROS is a popular framework for developing robot applications. It provides a set of tools and libraries that simplify the development of robotic systems. ROS supports a range of programming languages, including Python and C++, and allows developers to build complex systems by integrating various software components.

- AI and Machine Learning: AI enables robots to learn from experience and make decisions based on data, without requiring explicit programming for every task. Machine learning algorithms allow robots to improve their performance over time, making them adaptable to new situations.
- Demonstration-Based Programming: In some cases, robots can be programmed by demonstrating the desired behavior. This method, known as "teach-by-showing," allows a robot to learn by observing a human perform a task. The robot then records the movements and repeats them autonomously.

I.11. Conclusion

The field of robotics has come a long way, from simple machines performing repetitive tasks to advanced, intelligent systems that can learn, adapt, and collaborate with humans. With the rapid advancements in AI and machine learning, the future of robotics looks promising, with robots becoming more autonomous and capable of performing tasks in increasingly complex environments. As we move into the next generation of robots, they will continue to transform industries, revolutionize workplaces, and open up new possibilities for human-machine interaction. The possibilities are vast, and robots will continue to play an essential role in the future of technology and societ

Chapter II – Components of Robot **M**obile

II.1. Introduction

The development of a mobile object-tracking robot involves meticulous planning and the careful selection of both components and hardware architecture to ensure optimal performance. In this chapter, we delve into the fundamental physical building blocks that constitute the robot, each contributing to its ability to perceive its surroundings, process sensor data, and respond effectively to obstacles encountered in its environment.

The design philosophy centers on utilizing widely available, reliable, and cost-effective modules. This approach not only simplifies the construction process but also enhances the robot's accessibility for replication and modification by other developers or researchers. The chosen components encompass sensors, microcontrollers, actuators, and power supply units, each selected based on their compatibility, functionality, and ease of integration.

By systematically exploring each module's role and detailing how they interconnect within the overall system, this chapter aims to provide a comprehensive understanding of the robot's hardware setup. Such integration is critical, as it ensures seamless communication between the sensing and control elements, allowing the robot to navigate its environment with accuracy and efficiency[1].

II.2. System Overview

The hardware system consists of the following major components:

- Arduino Uno: Acts as the central control unit.
- Ultrasonic Sensor (HC-SR04): Detects objects and provides distance measurements.
- L298N Motor Driver: Controls the speed and direction of the motors.
- DC Motors (x2): Enable the robot to move.
- Power Supply: A battery pack to provide the required energy.
- Chassis and Wheels: Support structure and mobility.

The system architecture is built around simplicity and modularity. The ultrasonic sensor sends distance data to the Arduino, which processes the information and decides motor commands. The L298N receives control signals and powers the motors accordingly, driving the robot toward or away from detected objects (***Figure II.1***).

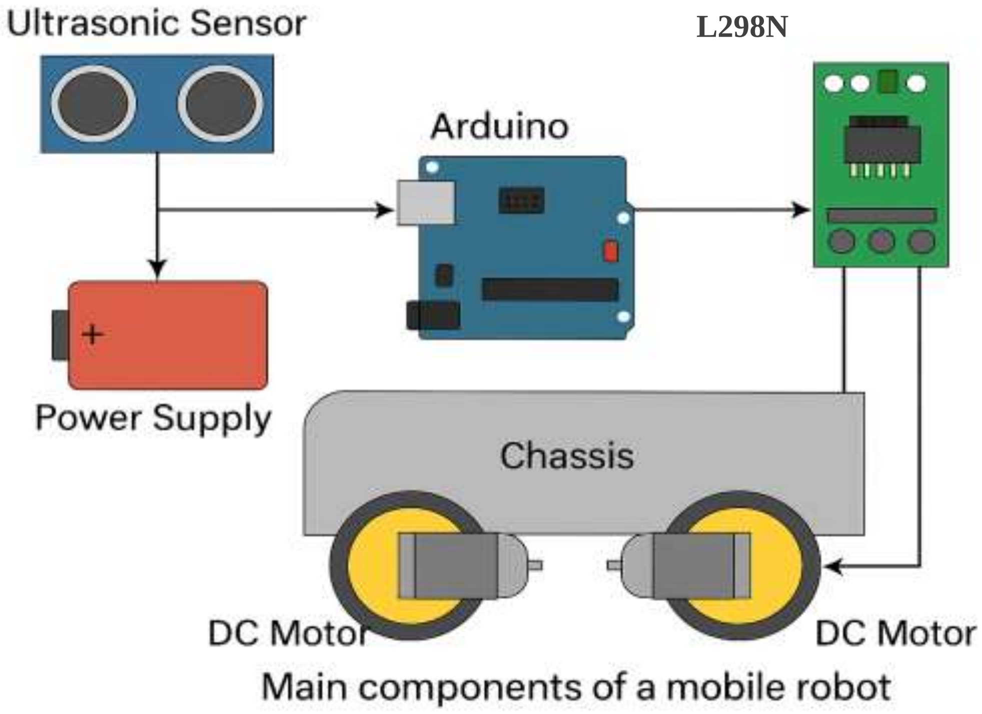


Figure II.1: Diagram showing the basic components of a robot

II.3. Arduino Uno

The **Arduino Uno** board is the most popular product among Arduino boards. Perfect for beginners in Arduino programming, it includes all the basic components needed to build relatively low-complexity projects.

As its name suggests, the Arduino Uno was the first to use the Arduino 1.0 programming version, and it has become the symbol of the Arduino ecosystem.

The Arduino Uno board consists of 14 digital input/output pins, six of which can be used as PWM outputs, 6 analog input pins, a USB connector, a power jack, an ICSP header, and a RESET button[11].

At the heart of the system lies the Arduino Uno, a microcontroller board based on the ATmega328P. It serves as the brain of the robot, managing sensor input and motor output. Its ease of use, open-source environment, and wide community support make it ideal for robotics projects and educational applications.

A description of all the connectors on the Arduino Uno board is shown in the image below (***Figure II.2***):

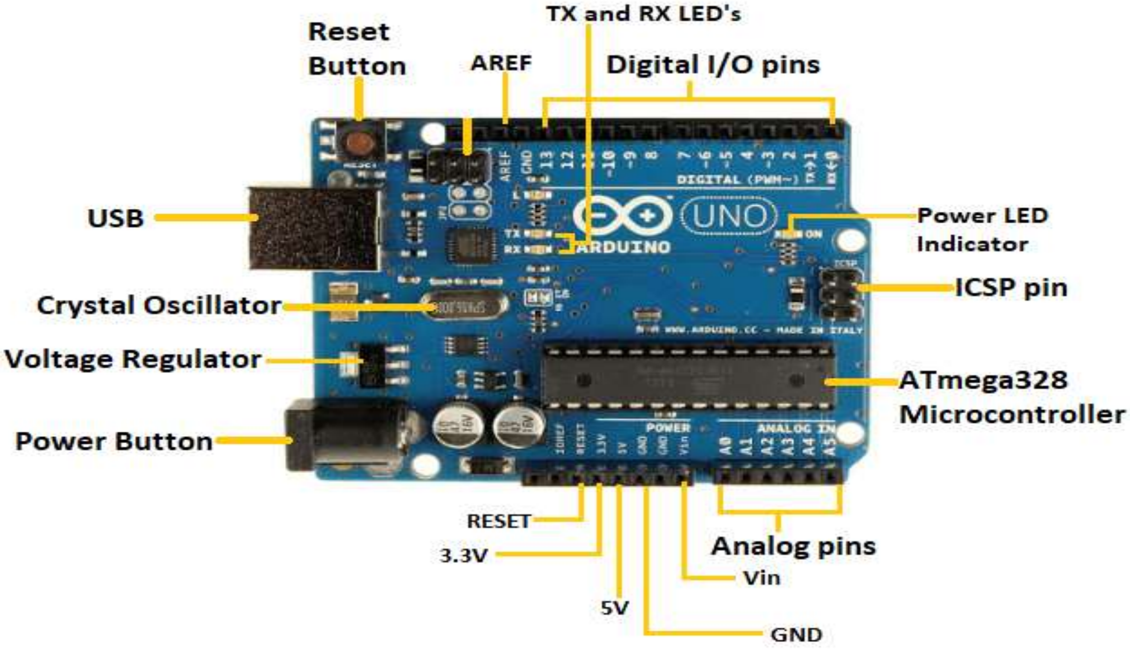


Figure II.2: Arduino Uno

- **Technical specification :**

Component	**specification**
Microcontroller	ATmega328
Operating voltage	5 V
Input voltage (recommended)	7-12 V
Input voltage (limits)	6-20 V
Digital I/O pins	14 ()
Digital I/O pins PWM	6
Analog input pins	6
DC current per I/O pin	20 mA
DC current for 3.3V pin	50 mA
Flash memory	32 Ko
SRAM	2 KB
EEPROM	1 KB
Clock speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Table II.1: Technical specification of Arduino UNO

- **Key Features**
  - 14 digital I/O pins (6 PWM outputs)
  - 6 analog inputs
  - Clock speed: 16 MHz
  - Operating voltage: 5V
  - USB interface for programming and communication

The Arduino collects real-time data from the ultrasonic sensor and processes it according to the programmed logic. It then sends appropriate control signals to the motor driver to change the direction or speed of the motors.

- **Advanced**
  - Cost-effective and widely available
  - Extensive support libraries for sensors and actuators
  - Can be powered by USB or external power supply

II.4. Motor Driver: L298N

To control the movement of the robot's wheels, the system employs the L298N motor driver. This integrated circuit is capable of driving two DC motors simultaneously in both forward and reverse directions (***Figure II.3***) . The L298N is a dual H-bridge motor driver, which allows the microcontroller to manage high-current motors using low-power control signals.

Since the Arduino cannot directly power motors due to current limitations, the L298N acts as a bridge between the controller and the motors. It receives logic-level signals from the Arduino and provides the necessary voltage and current to the motors.

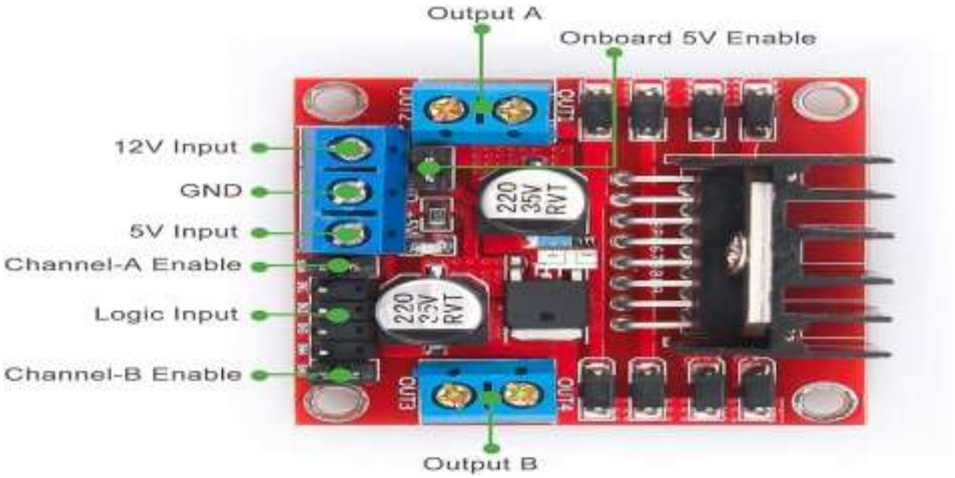


Figure II.3: Motor Driver: L298N

- **Technical specification**

Component	specification
Driver Model	L298N
Dual full-bridge driver	can control 2 motors
Driver Chip	Double H Bridge L298N
Motor Supply Voltage (Maximum)	46V
Motor Supply Current (Maximum	2A
Logic Voltage	5V
Driver Voltage	5-35V
Driver Current	2A
Logical Current	0-36mA
Maximum Power (W)	25W

Table II.2: Technical specification of L298N

- **Key Features**
  - Current Sense for each motor
  - Heatsink for better performance
  - Power-On LED indicator

- **Typical Wiring with Arduino**
  - IN1/IN2 for Motor A direction control
  - IN3/IN4 for Motor B direction control
  - ENA/ENB for enabling PWM speed control
  - 12V and GND for power supply

II.5. DC Motors

The robot uses two DC motors for locomotion, which form the basis of a differential drive system. These motors are inexpensive, easy to control, and provide sufficient torque and speed for small robotic platforms (***Figure II.4***) . Each motor is connected to one of the rear wheels of the robot and is powered via the L298N driver.

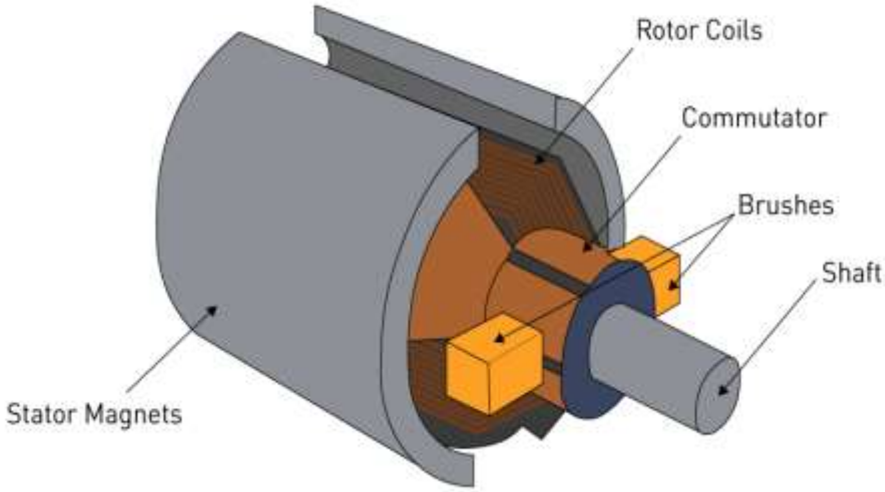


Figure II.4: DC Motors

By controlling the rotation direction and speed of each motor independently, the robot can move forward, backward, and rotate left or right.

- **Technical specification**
  - Operating voltage: 6–12V
  - Typical speed: 100–300 RPM
  - Average current: 200–500 mA
  - Shaft diameter: 5–6 mm

II.6. Ultrasonic Sensor (HC-SR04)

The HC-SR04 ultrasonic sensor is the primary tool for detecting objects and measuring distances , By this formula :

$$\{\, Distance = \frac{v\times t}{2} \qquad \textbf{(Equ II.1)}$$

It operates by sending out an ultrasonic pulse and measuring the time, it takes the echo to return (***Figure II.5***) . This time delay is converted into distance using a simple speed-of-sound formula[15].

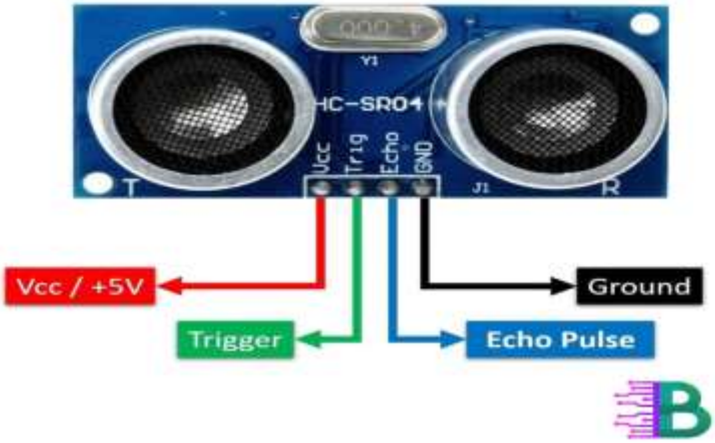


Figure II.5: Ultrasonic Sensor (HC-SR04)

- **Key Features**
  - Detection range: 2 cm to 400 cm
  - Accuracy: ±3 mm
  - Operating voltage: 5V
  - Easy to interface with Arduino

The sensor is placed at the front of the chassis to provide a clear path for object detection.

II.7. Infrared Sensor (IR Sensor)

Infrared (IR) sensors are widely used in robotic applications for **obstacle detection**, **line following**, and **short-range distance estimation**. They operate by emitting infrared light (usually from an IR LED) and detecting the reflection using a photodiode or phototransistor. When an object is present in front of the sensor, the emitted IR light reflects back and is detected, signalling the presence of an obstacle (***Figure II.6***).

In our project, we use **two IR sensors** mounted at the front left and right of the chassis. These sensors enhance the robot's awareness of its immediate surroundings, especially near the ground, making it effective at **detecting nearby objects or lines**, such as black strips or borders in a controlled environment[1].

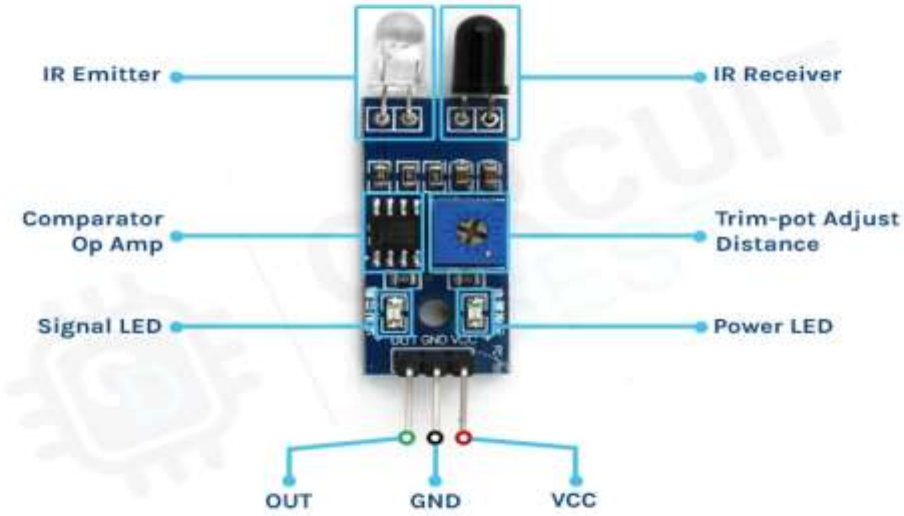


Figure II.6: IR Sensor

- **Key Features**
  - **Short-Range Detection:** Typically, effective between **2 cm to 30 cm**, depending on ambient light and surface reflectivity.
  - **Fast Response Time:** The sensors respond almost instantly, making them suitable for real-time applications like object tracking or line-following.
  - **Analog or Digital Output:** Most IR sensors offer either an analog voltage proportional to the reflected IR light or a digital HIGH/LOW signal indicating object presence.

- **Low Power Consumption:** Efficient for battery-powered projects, drawing minimal current.
- **Compact and Lightweight:** Ideal for mobile robots where space and weight are limited.
- **Environmental Sensitivity:** Performance may vary with the colour and texture of the surface (dark matte surfaces reflect less IR).

II.8. Power Supply

A reliable power supply is crucial to ensure the stable operation of all components in a mobile robot. In this project, the power system is designed to independently supply sufficient voltage and current to the Arduino Uno, L298N motor driver, DC motors, and the ultrasonic sensor.

The chosen setup uses a rechargeable battery pack, typically a 7.4V or 12V Li-ion battery, as the primary power source. This voltage is suitable for powering the L298N motor driver, which in turn powers the DC motors. The Arduino Uno can be powered either from the L298N's 5V regulated output or directly through its barrel jack or USB input[12].

II.9. Chassis and Mechanical Frame

The chassis serves as the physical foundation of the mobile robot, providing structural support for all electronic and mechanical components. It houses the DC motors, Arduino Uno, L298N driver, battery pack, and ultrasonic sensor, ensuring that all parts remain securely in place during movement.

For this project, a two-wheeled differential drive chassis is used. It consists of two driven wheels mounted on either side, each powered by a DC motor, and a free-rotating caster wheel at the front or back to maintain balance. This configuration enables the robot to rotate in place and perform smooth forward/backward movement[12].

II.10. Conclusion

In this chapter, we have detailed the hardware architecture of the mobile object-tracking robot. The project leverages a simple yet effective combination of components, including the Arduino Uno microcontroller, ultrasonic sensor, infrared (IR) sensors, the L298N motor driver, and two DC motors. Each module was selected for its reliability, ease of integration, and cost-effectiveness, making this design ideal for educational and prototype-level applications.

The ultrasonic sensor provides a real-time, non-contact method for object detection, while the pair of IR sensors enhances the robot's ability to detect nearby obstacles or follow lines, offering a more responsive and adaptable navigation system. The Arduino Uno serves as the processing unit, interpreting sensor data and directing motor actions through the L298N driver. The robot is powered by a compact battery pack and mounted on a stable, lightweight chassis that supports smooth movement in indoor environments.

Altogether, this configuration lays a strong foundation for the robot's object tracking and obstacle avoidance capabilities, which will be further explored in the next chapter dedicated to system design, programming logic, and functional behaviour.

PART II: IMPLEMENTATION AND TESTING

Chapter III –Practical Implementation

III.1. INTRODUCTION

This chapter presents the **real-world construction and implementation** of our object-tracking and obstacle-avoiding mobile robot. It reflects the transition from theoretical planning to a fully functioning physical system, demonstrating how various disciplines—mechanics, electronics, and programming—come together to create intelligent and autonomous behavior.

Throughout the development process, we focused on building a reliable and responsive system capable of detecting and reacting to its environment in real time. The robot was carefully assembled using accessible components selected for their affordability, ease of use, and compatibility. These include distance sensors, motor drivers, a microcontroller, and a robust chassis—each playing a critical role in ensuring coordinated motion and accurate detection[13].

Programming the robot was just as essential as assembling it. The software governs how the system interprets sensor data and how it decides to move or stop when obstacles are detected. Special attention was given to writing clear and efficient code that ensures fast response times and stable performance during operation.

To better illustrate the inner workings of the robot, this chapter includes a variety of visual and technical elements: detailed photographs of the hardware setup, functional diagrams that explain system interactions, simulations validating the design, and code excerpts that reveal the robot's logic.

The overall aim is to provide a complete and accessible view of the robot's development—from concept to realization—while offering insight into the decisions made, the problems solved, and the lessons learned during its construction[13].

III.2. MECHANICS

III.2.1. The Chassis

The chassis is the physical structure that houses all the components of our robot. We selected a plastic chassis with two large driving wheels and a front caster wheel for stability and maneuverability. It offers a compact and durable frame that holds the motors, Arduino Uno board, sensors (ultrasonic and IR), battery, and motor driver. This layout ensures easy accessibility for maintenance and wiring.

III.2.2. Operating Principle

Our mobile robot operates based on continuous detection of objects using two types of sensors: an ultrasonic sensor (HC-SR04) for forward obstacle detection, and two IR sensors for basic side proximity and line detection. When the ultrasonic sensor detects an object closer than 15 cm, the robot stops or changes direction. The IR sensors, mounted on the sides, help refine obstacle detection and avoid side collisions.

The Arduino Uno processes all the data in real-time, executing a control algorithm that decides when to move forward or stop. The L298N driver then executes these commands, controlling the motors accordingly.

III.2.3. System Structure

The main components of our robot system are:

- Chassis: 3-wheel platform for movement.
- Two DC motors: Rear wheels propulsion.
- Arduino Uno: Central microcontroller unit.
- L298N Motor Driver: Dual H-bridge for motor direction and speed control.
- Ultrasonic Sensor (HC-SR04): Forward obstacle detection.
- Two IR Sensors: Side obstacle detection.
- Power Supply: 9V battery pack.

III.3. ELECTRONICS

III.3.1. Power Supply

A 9V battery supplies power to both the Arduino and the motor driver via the L298N module. The power is distributed efficiently using onboard voltage regulators and capacitors to prevent noise and power drops during motor load changes.

III.3.2. DC Motor Control

III.3.2.1. H-Bridge

The L298N module contains two H-bridges, which control the direction of the current through the motors. This enables both forward and reverse motion depending on the combination of logic signals sent from the Arduino to the motor driver inputs.

III.3.2.2. Motor Driver Implementation

Each motor is connected to the L298N through two input pins (IN1/IN2 and IN3/IN4), and two enable pins (ENA and ENB) for PWM-based speed control (***Figure III.1)***. The driver interfaces with the Arduino Uno which sends signals based on the sensor input.

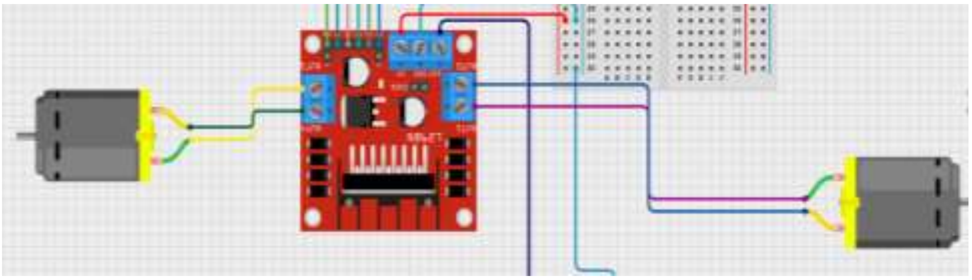


Figure III.1: Wiring Diagram of L298N and Motors

III.3.3. Object Detection

The ultrasonic sensor is connected to digital pins (TRIG and ECHO). It continuously emits sound pulses and listens for their echoes. The time difference is used to calculate distance using the equation:

$$Distance\ (cm) = Time\ (\mu s)\ /\ 58 \quad \textbf{(Equ } \textbf{\textit{III.1)}}$$

If the distance is less than 15 cm, the robot stops. Otherwise, it moves forward. The IR sensors also contribute by detecting nearby objects on the sides.

III 4. Typical Wiring

III 4.1. Inputs / Outputs

In our object-tracking and obstacle-avoiding mobile robot, the Arduino Uno acts as the central controller, interfacing with all the sensors and actuators. It receives input data from the ultrasonic sensor and two IR sensors and sends output control signals to the L298N motor driver, which in turn drives the two DC motors (***Figure III.2)***.

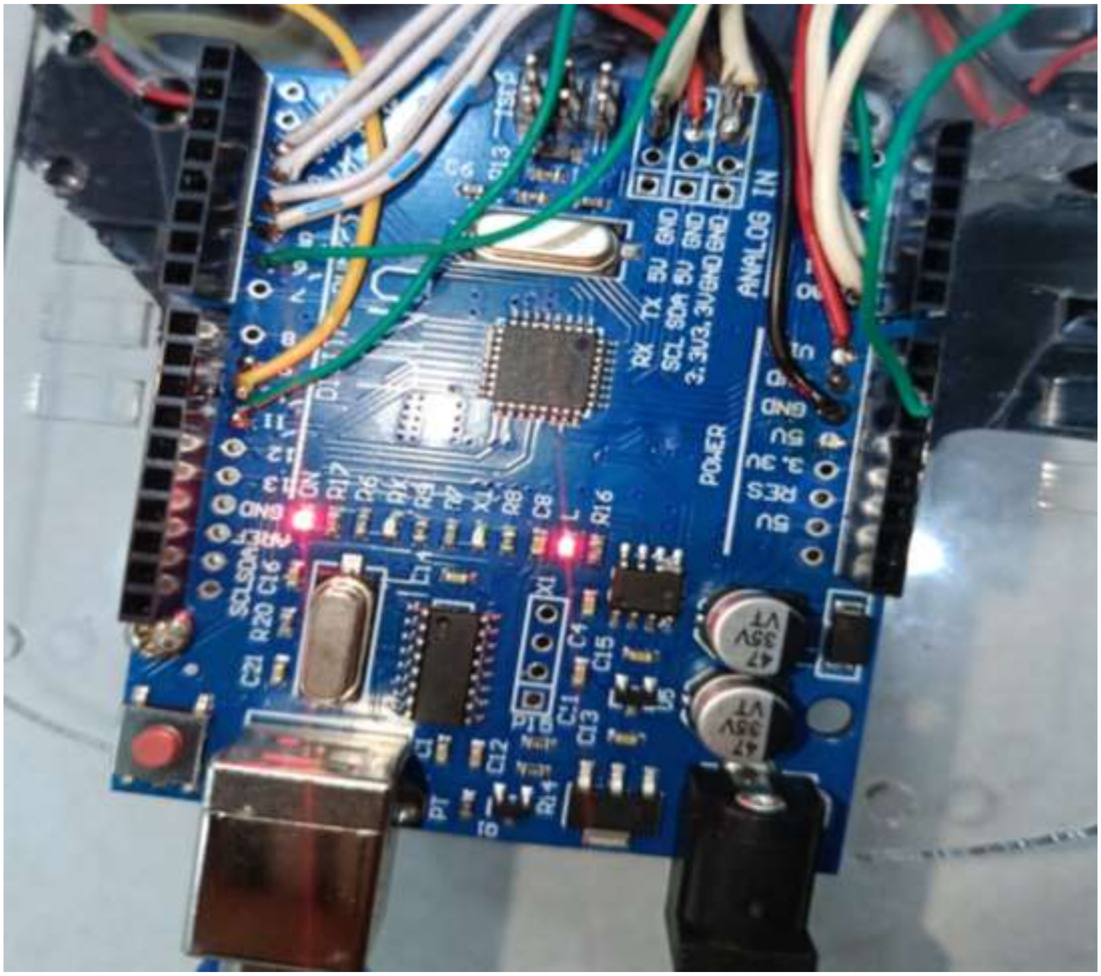


Figure III.2: Inputs / Outputs in Arduino Uno

Inputs – Sensors to Arduino:

- The **ultrasonic sensor** (HC-SR04) has two main pins connected to the Arduino:
    - **TRIG pin** is connected to **analog pin A0**: It sends out an ultrasonic pulse.
    - **ECHO pin** is connected to **analog pin A1**: It listens for the reflected pulse to measure the distance to obstacles.
- The **IR sensors** (two units used for line or object detection) are connected as follows:
    - **IR Sensor 1 (Left)**: OUT pin connected to analog pin **A3**.
    - **IR Sensor 2 (Right)**: OUT pin connected to analog pin **A2**.

These inputs provide environmental data, allowing the robot to detect objects and follow or avoid obstacles (***Figure III.3)***.

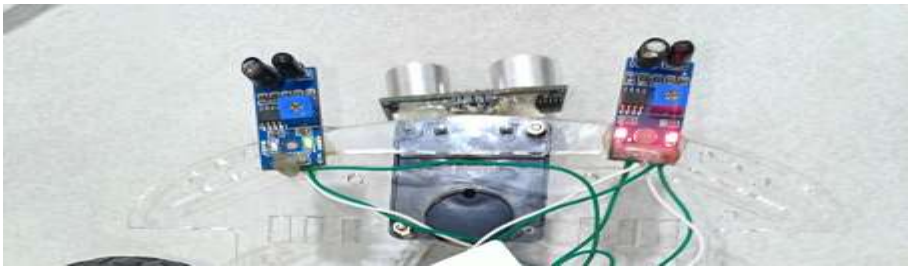


Figure III.3: Inputs – Sensors to Arduino

Outputs – Arduino to L298N Motor Driver:

- The **L298N motor driver** has control pins for each motor:
    - **Motor A (left wheel):**
        - **IN1** connected to **D2**.
        - **IN2** connected to **D3**.
        - **ENA (speed control)** connected to **D6** (PWM capable).
    - **Motor B (right wheel):**
        - **IN3** connected to **D5**.
        - **IN4** connected to **D4**.
        - **ENB (speed control)** connected to **D11** (PWM capable).

These outputs allow the Arduino to control motor direction and speed based on sensor readings (***Figure III.4)***.

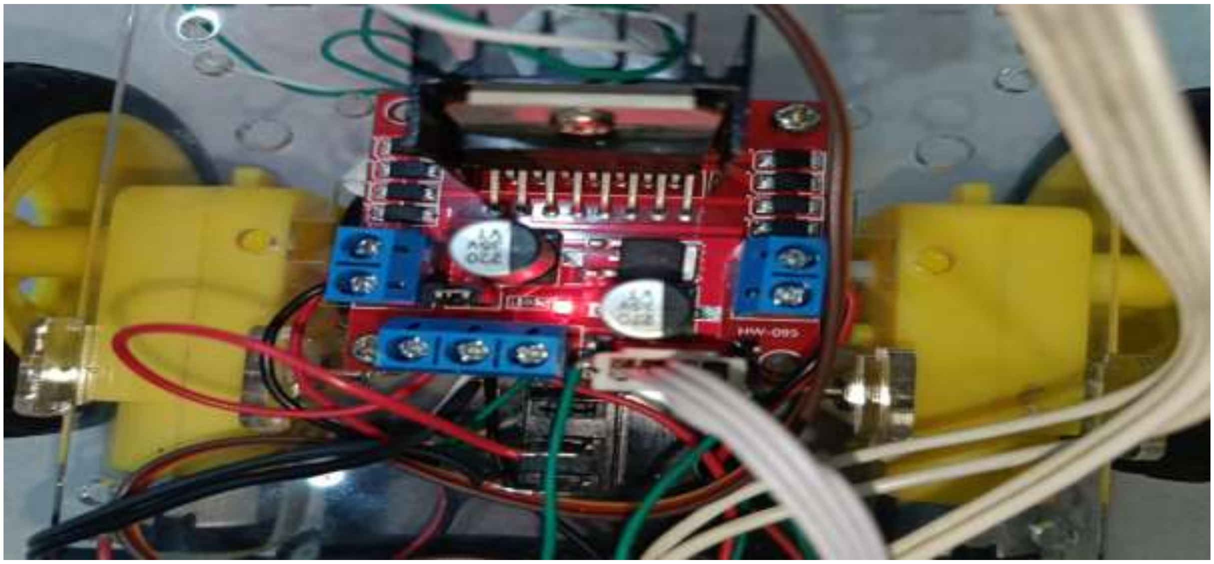


Figure III.4: Outputs – Arduino to L298N Motor Driver

Power Connections:

- The entire system is powered using a 9V battery pack connected to the L298N motor driver.
- The L298N module powers the DC motors directly and also provides regulated voltage (5V) to the Arduino Uno through its onboard 5V pin.
- All sensors (ultrasonic and IR) receive 5V and GND connections directly from the Arduino.

Summary:

- Inputs: Ultrasonic sensor (D9 & D10), IR sensors (A3 & A2).
- Outputs: Motor driver control pins (D2, D3, D4, D5, D6, D11).
- Power: Centralized through the L298N and 12V battery.

This wiring setup enables the robot to detect objects using both proximity and infrared reflection, make movement decisions using the Arduino, and drive motors accurately using PWM signals (***Figure III.5)***.

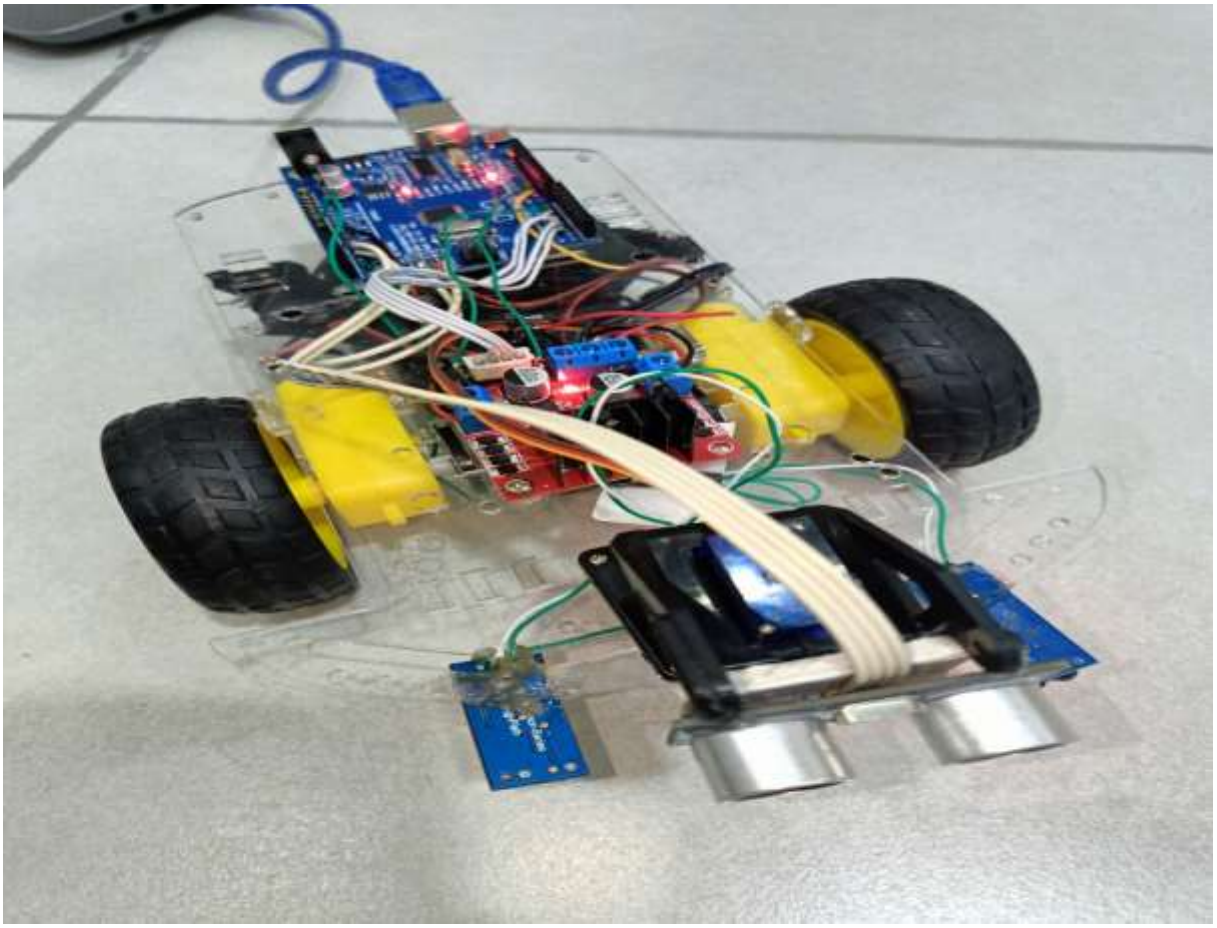


Figure III.5: Final Result

III. 4.2. Communication

In our mobile robot system, communication refers to how the Arduino exchanges data both internally (between components) and externally (with a computer or other devices, if needed). Although the robot functions autonomously, communication is essential during development, debugging, and possible upgrades.

III. 4.2.1. Serial Communication (UART)

The Arduino Uno uses **UART (Universal Asynchronous Receiver/Transmitter)** to send and receive data over its **TX (D1)** and **RX (D0)** pins. During programming and testing, the Arduino communicates with the computer via USB through a built-in USB-to-serial converter.

- This communication allows us to:
  - Send debugging messages (e.g., distance measured by the ultrasonic sensor).

  - Monitor sensor readings in real-time using the Arduino Serial Monitor.
  - Test logic without the need to interact with the physical robot.

III. 4.2.2. Sensor and Actuator Communication

- **Ultrasonic Sensor (HC-SR04)** communicates with the Arduino via **digital pins (D9 and D10)** using timing signals. The Arduino sends a trigger pulse and measures the time taken for the echo to return, which is then converted into distance using a basic formula.
- **IR Sensors** output a simple **digital HIGH or LOW** signal depending on object presence or surface reflection. They are connected to **digital pins D11 and D12**.
- **Motor Driver (L298N)** communicates with the Arduino via **digital and PWM signals**, allowing control of motor direction and speed.

III. 4.2.3. Optional Communication Extensions

Although not used in this version of the project, the Arduino Uno supports:

- **I2C** communication (using A4 for SDA and A5 for SCL)
- **SPI** communication (using pins 10–13)

III.5. Development And Simulation Tools

To ensure our mobile robot functioned as expected before physically assembling the components, we relied heavily on a set of software tools that supported development, simulation, and testing.

III.5.1. Development Tools

The primary development tool used in this project was the **Arduino Integrated Development Environment** (IDE). This platform allowed us to write, compile, and upload our control algorithm to the Arduino Uno microcontroller. The IDE's simplicity and strong community support made it ideal for rapid prototyping and troubleshooting. Additionally, the built-in Serial Monitor feature provided real-time insights into sensor readings and robot behavior during development[11]. This was particularly useful for debugging and fine-tuning the obstacle avoidance logic (***Figure III.6)*.**

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

}

voi
  //

}
```

AN OPEN PROJECT WRITTEN, DEBUGGED, AND SUPPORTED BY ARDUINO.CC AND THE ARDUINO COMMUNITY WORLDWIDE

LEARN MORE ABOUT THE CONTRIBUTORS OF ARDUINO.CC on arduino.cc/credits

Figure III.6: Arduino Integrated Development Environment (IDE)

We also used version control practices, even informally, to track changes in the Arduino code and compare different algorithm versions during testing.

III.5.2. Simulation Tools

- **Cirkit Designer**: A modern and intuitive electronic design tool that was used for schematic design and wiring visualization (***Figure III.7)***. Unlike traditional simulation platforms, Cirkit Designer helped us clearly lay out the connections between components like the Arduino, L298N motor driver, ultrasonic sensor, IR sensors, and power supply. It also allowed us to present professional-looking schematics for documentation and troubleshooting purposes[14].

Figure III.7: Cirkit Designer

- **Serial Monitor**: An integrated tool within the Arduino IDE, used for live testing and data tracking. It allowed us to monitor sensor readings such as distance and IR detection, ensuring the software logic responded accurately during test scenarios.

III.6. PROGRAMMING

III.6.1. Arduino Code Structure

The Arduino code is written using the Arduino C/C++ language. It handles data collection, processing, and command execution. The loop function continuously reads sensor values and makes decisions about movement.

```
#include <Servo.h>

// Motor pins
const int IN1 = 2;
const int IN2 = 3;
const int IN3 = 5;
const int IN4 = 4;
const int ENA = 6;   // PWM for left motor
const int ENB = 11;  // PWM for right motor

// Servo and ultrasonic sensor pins
const int servoPin = 10;
const int trigPin = A0;
const int echoPin = A1;

// IR sensor pins
const int irLeftPin = A3;
const int irRightPin = A2;

// Speed
int leftSpeed = 60;
int rightSpeed = 60;

// Servo
Servo myServo;

// Sensor values
bool leftSensorState, rightSensorState;

// Obstacle distance threshold (in cm)
const int obstacleThreshold = 15;

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

```
  pinMode(irLeftPin, INPUT); pinMode(irRightPin, INPUT);
  pinMode(trigPin, OUTPUT); pinMode(echoPin, INPUT);

  myServo.attach(servoPin);
  myServo.write(90);  // Face forward

  Serial.begin(9600);
}

void loop() {
  int distance = measureDistance();

  if (distance < obstacleThreshold) {
    stopMotors();
    avoidObstacle();
    return;  // Skip line following during avoidance
  }

  leftSensorState = digitalRead(irLeftPin);
  rightSensorState = digitalRead(irRightPin);

  if (leftSensorState == LOW && rightSensorState == LOW) {
    moveForward();
  }
  else if (leftSensorState == LOW && rightSensorState == HIGH) {
    turnLeft();
  }
  else if (leftSensorState == HIGH && rightSensorState == LOW) {
    turnRight();
  }
  else {
    stopMotors();
  }

  delay(100);
}

int measureDistance() {
  digitalWrite(trigPin, LOW); delayMicroseconds(2);
  digitalWrite(trigPin, HIGH); delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  long duration = pulseIn(echoPin, HIGH);
  int distance = duration * 0.034 / 2; // cm
  return distance;
}

void avoidObstacle() {
  myServo.write(150); // Scan right
  delay(500);
  int rightDist = measureDistance();

  myServo.write(30); // Scan left
  delay(500);
  int leftDist = measureDistance();

  myServo.write(90); // Return to center
```

```
  if (leftDist > rightDist && leftDist > obstacleThreshold) {
    turnLeft(); delay(600);
  } else if (rightDist > leftDist && rightDist > obstacleThreshold) {
    turnRight(); delay(600);
  } else {
    stopMotors();
  }
}

// Movement functions
void moveForward() {
  digitalWrite(IN1, HIGH); digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH); digitalWrite(IN4, LOW);
  analogWrite(ENA, leftSpeed);
  analogWrite(ENB, rightSpeed);
  Serial.println("Moving Forward");
}

void stopMotors() {
  digitalWrite(IN1, LOW); digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW); digitalWrite(IN4, LOW);
  Serial.println("Stopped");
}

void turnLeft() {
  digitalWrite(IN1, LOW); digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH); digitalWrite(IN4, LOW);
  analogWrite(ENA, 0);
  analogWrite(ENB, rightSpeed);
  Serial.println("Turning Left");
}

void turnRight() {
  digitalWrite(IN1, HIGH); digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW); digitalWrite(IN4, LOW);
  analogWrite(ENA, leftSpeed);
  analogWrite(ENB, 0);
  Serial.println("Turning Right");

}
```

Obstacle Avoidance Algorithm

The robot decides how to move using the combination of ultrasonic and IR sensor input. The control logic can be described as:

- If an obstacle is detected in front (ultrasonic < 15 cm), the robot stops.
- If the left IR detects an object, it turns right.
- If the right IR detects an object, it turns left.
- If no sensors are triggered, the robot moves forward.

Pseudocode

```
IF (ultrasonic distance < 15 cm) OR (left IR detects object) OR (right IR
detects object)
    STOP
    IF left IR detects object
        TURN RIGHT
    ELSE IF right IR detects object
        TURN LEFT
    ELSE
        STOP (obstacle ahead)
ELSE
    MOVE FORWARD
```

III.6.2. Cirkit Designer Simulation

A complete simulation of the robot's electronic circuit was created in Proteus. It includes the Arduino board, the L298N motor driver, both DC motors, the ultrasonic sensor, and two IR sensors. The simulation allowed us to validate the logic and wiring prior to the physical assembly (***Figure III.8)***.

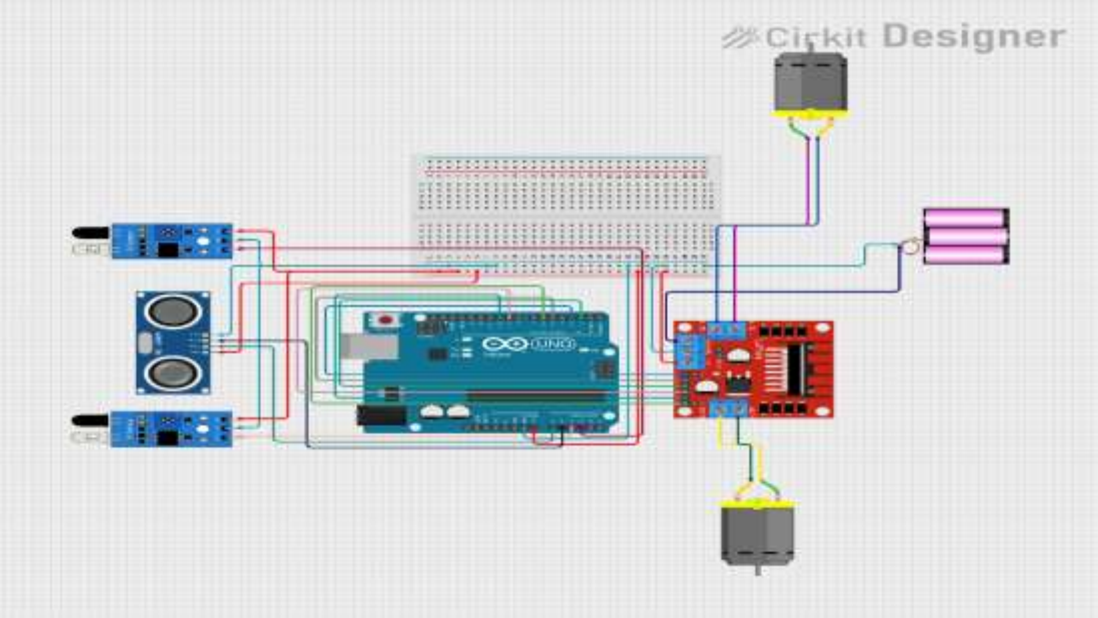


Figure III.8: Cirkit Designer Simulation

III.6.3. Characteristics of the Programming Language (Arduino – C/C++)

The programming of our mobile robot was carried out using the Arduino IDE, which is based on C and C++ languages. This environment is specially tailored for ease of use with embedded systems and microcontrollers, particularly for beginners and educational projects. Below are some of the key characteristics that make this language and platform well-suited for our robot system[13].

1. Simplicity and Readability

Arduino's version of C/C++ is simplified, with a structured and readable syntax. It allows developers to focus on hardware logic without worrying about low-level programming intricacies.

2. Hardware-Level Control

The language allows direct access to hardware components (such as setting pin modes, sending PWM signals, or reading sensor values), which is crucial for real-time robotics applications.

3. Event-Driven Structure

Arduino code runs inside two main functions:

- `setup()`: Runs once when the system starts, ideal for initialization.
- `loop()`: Continuously executes and handles real-time events like object detection and motor control.

4. Modular Programming Support

Arduino supports function creation and library inclusion, which helps in writing clean and modular code. This was useful in our project for separating motor control, distance checking, and decision-making functions.

5. Wide Library Ecosystem

Arduino provides a large number of open-source libraries (e.g., NewPing, Servo, IRremote) that simplify the interaction with sensors and actuators, making it easier and faster to prototype.

6. Community and Documentation

Due to its popularity, Arduino has a vast community and extensive documentation, allowing quick troubleshooting and learning through forums and tutorials.

III.7. Test and Results

Testing Methodology:

- The robot was tested across 5 different obstacle courses arranged with varying object types (books, bottles, cups).
- Data was monitored via the Arduino Serial Monitor to verify distance measurements and logic execution.
- Each test assessed how the robot handled sharp turns, unexpected obstacles.
- Some limitations included false detections on shiny surfaces and slight misalignment on very narrow objects.

These results confirm that the robot functions effectively under standard indoor conditions and validates the control logic and sensor integration (***Figure III.9)***.

Figure III.9: Test Of Mobile Robot

Key Results:

Following the implementation and simulation stages, our robot was tested in a controlled indoor environment. The results showed:

- Obstacle detection success rate: 85% of the time the robot correctly identified and avoided obstacles placed within 10–15 cm.
- Reaction time: The robot took approximately 150–200 milliseconds to stop or change direction after detecting an obstacle.
- IR sensors: Effectively detected side obstructions and prevented minor collisions.
- Motor control: The L298N driver responded well to PWM commands, ensuring smooth speed regulation and direction changes.
- Limitations: The robot occasionally misjudged very narrow objects or glossy surfaces that caused echo distortions.

These results confirm that the system functions reliably under normal conditions and responds effectively to environmental changes.

III.8 CONCLUSION

This chapter demonstrated how the robotic system was assembled and programmed to detect and avoid obstacles. The synergy between hardware and software components results in a functional mobile robot capable of tracking and responding to its environment. From mechanical construction to software logic and simulation, each phase was critical to the overall success of the system.

General Conclusion

At the end of this project, we successfully designed, built, and tested an autonomous mobile robot capable of object tracking and obstacle avoidance in real-time. This experience allowed us to deepen our theoretical knowledge in electronics, embedded programming, and mechatronic system design, while also developing practical skills through the realization of a functional and educational prototype.

The robot's architecture is based on a well-integrated combination of components: an Arduino Uno board, two DC motors controlled via an L298N motor driver, an ultrasonic sensor for frontal obstacle detection, and two infrared sensors for side detection and line following. The control algorithm, developed in Arduino C, enables the robot to react intelligently to its surroundings and make decisions dynamically. Using Cirkit Designer, we simulated and validated the electronic design before implementing it physically.

The results obtained during testing demonstrate that the system operates reliably in indoor environments, with effective object detection and fast response times. Although simple in scope, this project provides a strong foundation for further development in mobile robotics applications, such as autonomous navigation, educational platforms, or small-scale logistics systems.

Finally, this project highlights the importance of teamwork, systematic planning, and technical precision across all stages—from concept to final validation.

BIBLIOGRAPHY

1. **Siegwart, R., Nourbakhsh, I. R., & Scaramuzza**, D. (2011). *Introduction to Autonomous Mobile Robots*. MIT Press.
2. **K. Kim, D. Kim, and Y. Cho**, "Ultrasonic Sensor-Based Target Tracking System for Autonomous Mobile Robots," *Sensors*, vol. 18, no. 9, 2018. https://doi.org/10.3390/s18092955
3. **M. Bughin et al.**, *"Notes from the AI frontier: Modeling the impact of AI on the world economy,"* McKinsey Global Institute, 2018. https://www.mckinsey.com/featured-insights/artificial-intelligence
4. **IEEE Robotics and Automation Society**, ongoing. https://www.ieee-ras.org
5. **ROS Wiki**, "Navigation Stack Overview," ongoing. https://wiki.ros.org/navigation
6. **H. M. Paynter**, *Analysis and Design of Engineering Systems*, MIT Press, 1961. A foundational reference in describing actuator systems, including pneumatic, hydraulic, and electromechanical systems in automation.
7. **J. Borenstein, H. R. Everett, L. Feng, and D. Wehe**, "Mobile Robot Positioning: Sensors and Techniques," Journal of Robotic Systems, vol. 14, no. 4, 1997.
8. **B. Asfora**, "Embedded Computer Vision System Applied to a Four-Legged Line Follower Robot," *arXiv preprint arXiv:2101.04804*, Jan 2021. arxiv.org
9. **GeeksforGeeks**, "Path Planning Algorithms in Artificial Intelligence," ongoing. https://www.geeksforgeeks.org/path-planning-algorithms-in-artificial-intelligence/
10. **A. Rossi et al.**, "Real-time Lane Detection and Motion Planning in Raspberry Pi and Arduino for an Autonomous Vehicle Prototype," *arXiv preprint arXiv:2009.09391*, Sep 2020. arxiv.org
11. **Arduino**, "Arduino Homepage & Tutorials," ongoing. https://www.arduino.cc
12. **T. Braunl**, *Embedded Robotics: Mobile Robot Design and Applications with Embedded Systems*, 3rd ed., Springer, 2012.
13. **M. Banzi & M. Shiloh,** *Getting Started with Arduino***:** *The Open Source Electronics Prototyping Platform*, Maker Media, 2014.
14. **Cirkit Designer** – Modern Electronics Design Tool. Available: https://www.cirkitdesigner.com/
15. **Y. D. Zaman and S. A. Alim**, "Obstacle Detection and Anti-Collision Robot Using Ultrasonic Sensor," *ELEKTRIKA: J. Electrical Engineering*, vol. 22, no. 1, pp. 11–14, Apr. 2023. DOI:10.11113/elektrika.v22n1.404

16. **R. Mothe, S. Tharun Reddy, G. Sunil, and C. Sidhardha**, “An IoT Based Obstacle Avoidance Robot Using Ultrasonic Sensor and Arduino,” *IOP Conf. Ser.: Mater. Sci. Eng.*, vol. 981, p. 042002, 2020. DOI:10.1088/1757-899X/981/4/042002

17. **J. Azeta, C. Bolu, D. Hinvi, and A. Abioye**, “Obstacle Detection Using Ultrasonic Sensor for a Mobile Robot,” *IOP Conf. Ser.: Mater. Sci. Eng.*, vol. 707, p. 012012, 2019. DOI:10.1088/1757-899X/707/1/012012 iopscience.iop.org

18. **B. K. Krishna, R. K. Rachananjali, and C. H. N. Narasimha Rao**, “Obstacle Avoiding Robotic Vehicle with Arduino and Ultrasonic Sensor,” *Int. J. Health Sci.*, vol. 6, S1, pp. 6888–6898, May 2022. DOI:10.53730/ijhs.v6nS1.6463

19. **Q. Hidayati**, “Obstacle Avoidance Mobile Robot With Ultrasonic Sensors,” *Jurnal Teknologi Terpadu (JTT)*, vol. 5, no.1 (SINTA indexed), 2022. DOI:10.32487/jtt.v5i1.215 jurnal.poltekba.ac.id

20. **Y. Srivastava, S. Singh, and S. P. S. Ibrahim**, “Autonomous Bot with ML-based Reactive Navigation for Indoor Environment,” *arXiv preprint arXiv:2111.12542*, Nov 2021. arxiv.org