CHAPTER 1

INTRODUCTION

1.1 Prelude

This project is about making a Smart Floor Cleaning System that can clean the floor automatically and be controlled using a mobile phone. Today, many people are busy and don’t have time to clean their homes or offices regularly. Also, manual cleaning can be tiring and may not reach all corners properly. Our aim is to create a simple and smart machine that can help make cleaning easier, faster, and more effective. It is especially useful for elderly people, office workers, or anyone who wants to save time and effort in daily cleaning tasks.

We used different components like an Arduino UNO, L298N motor driver, Bluetooth module (HC-05), servo motors for moving the arms, and a water pump for spraying water. All these parts work together to clean the floor when the system is controlled through a mobile app. The user can move the cleaner forward, backward, left, or right and turn the water spray on or off as needed. This gives the user full control from their phone. The arms of the cleaner can also lift or lower the cleaning pads using servo motors, making it easier to clean different surfaces.

While making this project, we tried different ideas and tested each part carefully. Sometimes the system didn’t work as expected, so we made changes and improved it. We faced problems with Bluetooth connection, water pump flow, and motor movement, but by testing and learning, we found solutions. We learned how to connect the components properly, write the Arduino code, and make the cleaner move and spray water in the right way. We also focused on power supply, balance, and design to make the system simple and stable.

This project helped us improve our knowledge in electronics, programming, and problem-solving. It also taught us how to work as a team, plan our work, and be patient when solving technical issues. We learned how to test, fix errors, and think creatively. In the future, this system can be improved with more features like automatic movement using sensors, obstacle detection, dust collection, and scheduled cleaning. For now, it is a good and affordable solution to help people keep their floors clean with less effort. We are proud of this project and believe it is a small step toward smart home automation.

1.2 Importance of Smart Floor Cleaning System in Modern Era

In today's fast-paced, technology-driven world, smart floor cleaning systems have revolutionized hygiene management in homes and businesses. These intelligent devices—from robotic vacuums to industrial-grade autonomous scrubbers—offer superior efficiency, technological sophistication, and health and environmental benefits. As urban living intensifies and time becomes increasingly valuable, these systems provide an effective, automated alternative to traditional, labour-intensive cleaning methods. Using AI, machine learning, IoT, and sensor mapping, they navigate complex environments, adjust to various surfaces, and optimize cleaning routes over time, delivering consistent results with minimal human input.

Integrated into smart home or building ecosystems, users can control them remotely via smartphones or voice assistants, enabling seamless cleaning even in their absence. In a post-pandemic era, their relevance has grown due to their ability to reduce allergens, dust, and harmful particles. Many feature HEPA filters and even disinfection technologies such as UV-C light, vital for healthcare and elderly care environments where hygiene is critical. These systems reduce human exposure to contaminants and cleaning chemicals while ensuring consistent sanitation—especially in high-traffic or fatigue-prone areas.

Environmentally, smart cleaners are designed for sustainability. They optimize water and chemical use, some recycle cleaning water, and many operate efficiently with improved battery life. These features align with global efforts toward eco-friendly practices, helping businesses meet environmental regulations and CSR goals. In commercial and industrial settings—malls, airports, warehouses—autonomous machines reduce labor costs and increase productivity, operating during off-hours and gathering performance data for smarter facility management.

Despite their benefits, smart cleaning systems face challenges such as high upfront costs, maintenance needs, and data privacy concerns. However, falling prices, new financing models, and advances in cybersecurity are addressing these issues. Though they may struggle in cluttered or uneven areas, ongoing innovations in AI and navigation are enhancing adaptability.

Ultimately, smart floor cleaning systems embody the shift toward automation, sustainability, and intelligent living. They offer convenience and independence in homes, boost hygiene and efficiency in public and commercial spaces, and represent a crucial tool for meeting modern cleanliness standards. As innovation continues, these systems are set to become indispensable in creating cleaner, safer, and smarter environments.

1.3 Block Diagram

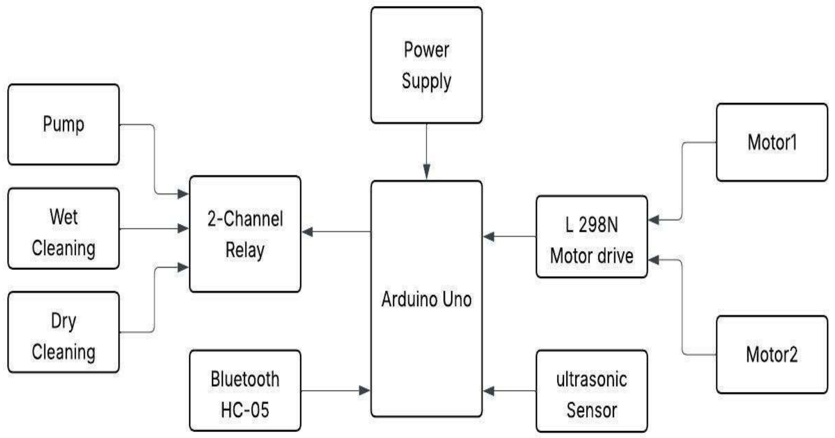


Fig 1.1: Block Diagram

The block diagram shown in Figure 1.1 illustrates the architecture of a smart floor cleaning system that is designed to perform both wet and dry cleaning tasks with automation and remote control capabilities. At the heart of the system lies the Arduino Uno, which functions as the central processing unit responsible for coordinating the operations of various components. The system begins with a power supply that provides the required voltage and current to operate all modules, including sensors, actuators, and the Arduino itself. One of the critical input devices connected to the Arduino is the Bluetooth module

(HC-05), which allows wireless communication between the user and the cleaning robot. Through a smartphone or Bluetooth-enabled device, the user can send commands to control

the movement and cleaning modes of the system. Once the Arduino receives the user's input, it determines whether to activate the wet or dry cleaning mechanism. This decision is executed via a 2-channel relay module, which switches on either the pump (used for wet cleaning by spraying water or cleaning liquid) or the dry cleaning system (typically involving brushes or vacuum mechanisms). The relay acts as an electronic switch, ensuring that only one cleaning mode is active at a time depending on the user's instruction.

1.4 Motivation

The motivation behind developing a Smart Floor Cleaning System stems from the increasing need for efficient, hygienic, and time-saving cleaning solutions in today’s fast-paced world. With modern lifestyles becoming more demanding, individuals often struggle to maintain clean environments, and traditional cleaning methods prove time-consuming, inconsistent, and labor-intensive. This system offers a smart, automated alternative that reduces human effort while ensuring consistent cleanliness. Equipped with technologies like sensors, microcontrollers, and motor control, the Smart Floor Cleaning System can navigate spaces autonomously, detect dirt or obstacles, and clean floors systematically. It is especially beneficial for elderly individuals, people with disabilities, and those with busy schedules who need reliable, hands-free cleaning.

In commercial and institutional environments—such as hospitals, schools, and offices—where hygiene is crucial, smart cleaning systems provide continuous, high-quality cleaning without the limitations of manual labor. The COVID-19 pandemic has further emphasized the importance of surface sanitation, boosting the relevance of automated cleaning technologies. Additionally, these systems promote sustainability by optimizing the use of water, electricity, and cleaning agents, reducing environmental impact. From a technical and educational perspective, building such a system fosters innovation and interdisciplinary learning. Overall, the Smart Floor Cleaning System addresses key challenges of modern living through automation, convenience, and environmental responsibility.

1.5 Objective of the Project

The Floor Cleaning Robot project faces significant challenges that impact its design, functionality, and real-world usability. One major issue is the accuracy and reliability of sensors, such as infrared and ultrasonic, which struggle with obstacles like shiny surfaces or low-profile objects, resulting in missed spots or collisions. Navigation is further complicated by varying floor types and cluttered environments. Power management is another hurdle, as the robot's battery often depletes mid-operation, leading to interrupted cleaning. The robot's mechanical design must balance compactness with functionality, as poor designs may compromise suction power or debris capacity. Wet cleaning adds complexity, requiring durable, integrated systems that resist leakage or electrical damage. Software limitations also exist, as basic robots lack efficient movement algorithms, while advanced features like SLAM and AI require expensive hardware. Cost is a barrier, as quality components drive up manufacturing expenses, limiting affordability. Environmental constraints such as uneven floors or cables hinder effectiveness, and regular maintenance remains necessary. Additionally, user interface challenges, scalability, and sustainability issues pose further concerns. Overcoming these challenges will require ongoing research, innovation, and careful design trade-offs.

- ❖ Sensors often struggle with detecting shiny, transparent, or low-profile obstacles, affecting navigation and cleaning efficiency.
- ❖ Cluttered spaces and varying floor types complicate navigation, while battery life limitations require frequent recharging mid-operation.
- ❖ Balancing compact design with adequate cleaning components is challenging, especially for wet cleaning systems that risk leakage or electrical damage.
- ❖ Basic software algorithms result in inefficient cleaning paths, and high-quality components drive up manufacturing costs, making robots less affordable.
- ❖ Environmental challenges like uneven floors, cables, and debris reduce effectiveness, while regular maintenance is still necessary for emptying bins and refilling water tanks.
- ❖ Complex controls, app integration issues, and limited customization hinder user experience, and scalability remains a concern with one-size-fits-all designs.

❖ Sustainability issues arise from battery replacements, non-recyclable parts, and synthetic cleaning agents, contributing to environmental impact.

1.6 Issues of the Floor Cleaning Robot Project

The development of autonomous floor cleaning robots offers clear benefits but faces numerous challenges affecting performance, usability, and cost. Sensor limitations, such as difficulty detecting shiny or transparent surfaces, can lead to poor navigation and incomplete cleaning. Battery life is another issue, as high power demands from motors and cleaning systems often result in limited operation time and require frequent recharging. Mechanical design must balance compactness with effective cleaning, yet components like brushes and bins can wear down or be inefficient. Advanced software like SLAM enhances coverage but requires significant processing power, raising costs and complexity. Budget models often compromise on performance, while high-end units remain costly and inaccessible for many. Environmental factors like clutter or spills can impair functionality, and regular maintenance is essential but often neglected. User interfaces may lack customization or ease of use, especially for non-tech users. Addressing these issues requires integrated engineering, intelligent design, and ongoing innovation.

1.7 Tools Used

1.7.1 Hardware Tools

- Arduino Uno
- Servo Motor
- Motor Driver
- Water Pump
- DC Motor
- Bluetooth Module HC-05
- Ultrasonic Sensor
- IR Sensor
- Jumper Wires
- Roker switch
- 3.7 V Lithium Rechargeable Battery
- 9v HW Battery
- Robot Wheels

1.7.2 Software Tools

- **Microcontroller Programming Tool**
- **Arduino Bluetooth Controller**

1.8 Applications

- ❖ Residential Cleaning: The robot can automatically clean floors in homes, reducing manual effort and maintaining hygiene, especially in urban households with busy lifestyles.
- ❖ Office and Commercial Spaces: It can be used in offices, shops, and commercial buildings for routine cleaning of floors during non-working hours without human intervention.
- ❖ Hospitals and Clinics: In medical environments, the robot helps maintain cleanliness and reduces contamination risk, especially by automating wet mopping with disinfectants.
- ❖ Educational Institutions: Schools, colleges, and universities can utilize the robot to keep classrooms, labs, and corridors clean efficiently and consistently.
- ❖ Hotels and Hospitality: It ensures clean and presentable floors in hotel lobbies, hallways, and guest rooms, improving guest satisfaction and reducing labor costs.
- ❖ Airports and Railway Stations: The robot can contribute to maintaining hygiene in high-footfall public areas like terminals, waiting halls, and lounges.
- ❖ Factories and Warehouses: In industrial environments, it can help in cleaning dust and debris from smooth concrete floors, supporting workplace safety.
- ❖ Smart Homes and IoT Integration: When integrated with smart home systems, the robot enhances automation and remote control through mobile apps or voice assistants.
- ❖ Elderly and Disabled Assistance: It is especially useful for people with limited mobility who may find manual cleaning difficult or impossible.

CHAPTER 2

LITERATURE SURVEY

2.1 Literature Review

Kerkar et al. [1] developed an autonomous floor cleaner designed for efficient indoor navigation and cleaning. The system uses an Arduino microcontroller to control sensors and motors. An ultrasonic sensor detects obstacles and measures distances, while an infrared sensor improves detection of low or small objects. The robot includes a Bluetooth module for wireless control via a smartphone app. Powered by a rechargeable battery, the cleaner offers portability and user convenience.

Ramalingam et al, [2] developed a smart floor-cleaning robot that focuses on cleaning only dirty areas using deep learning algorithms and RGB-D vision. They employed SORT for human traffic analysis and SSD MobileNet for detecting stains and waste. The system optimizes cleaning efficiency by targeting high-use zones. Path planning is done using waypoint-based logic to save energy and reduce wear. This approach increases cleaning effectiveness while lowering accessory consumption.

Yatmono et al, [3] presented a smart floor-cleaning robot capable of vacuuming and polishing using omni wheels and controlled via an Android app. The development followed Pressman's R&D model including design, implementation, and testing. Powered by an Arduino microcontroller and Bluetooth communication, the robot offers automated and user-directed cleaning. Its multifunctional design aims to reduce manual cleaning efforts.

Das et al, [4] proposed a Robotic Automated Floor Cleaner to minimize human labor and promote hygiene in public and private spaces. It aims to address the environmental downsides of fuel-powered cleaners. The robot focuses on efficient and safe cleaning for places like homes and hospitals. Their system is designed to be cost-effective and sustainable, improving daily cleaning routines.

Yadav et al, [5] introduced a wireless automatic floor cleaning robot built with a lightweight foam board frame and equipped with advanced electronics. The robot detects fires and temperature anomalies using onboard sensors and alerts users via LED indicators. Its design emphasizes adaptability, frequent cleaning, and user safety. The system operates entirely wirelessly, allowing remote monitoring and control.

Dhole et al, [6] tackled power-related issues in public places by proposing a solar-powered mobile cleaning robot. Built with commonly available materials, the robot is analyzed using commercial modeling tools. It provides a cost-effective alternative to conventional cleaners, especially in power-deficient regions. A finite element analysis confirmed structural durability, ensuring long-term usability.

Rumane et al, [7] developed a Bluetooth-controlled floor-cleaning system using DC motors, a scrubber, a cleaning solution reservoir, and a CPU fan for drying. Housed in a wheeled plastic body, the robot is user-friendly and suitable for various environments like hospitals and homes. The Bluetooth module allows remote operation through a smartphone or remote control.

Wayker et al, [8] built an affordable Android-controlled smart cleaning robot using Arduino UNO. Designed for users with limited access to high-end technology, it includes a cleaning mechanism and robotic arm. Bluetooth communication allows users to send commands from their phones. The system ensures efficient, low-cost cleaning automation.

Sudam et al, [9] created a wireless self-driving floor cleaner with ultrasonic and infrared sensors for obstacle avoidance. It works on multiple surface types and supports remote control, making it user-friendly and safe. The robot is positioned as a convenient and efficient tool for modern households, addressing the hassle of traditional corded devices.

Rathee et al, [10] designed a Bluetooth-enabled Arduino-based smart cleaner to simplify floor cleaning via wireless smartphone commands. The robot performs basic cleaning functions like movement and washing, reducing human effort. Its cost-effective and low-maintenance nature makes it ideal for daily home use.

Kumar et al, [11] presented a smart vacuum cleaner robot with an ultrasonic sensor for obstacle avoidance and 12V battery power. Built for cleaning hazardous or hard-to-reach places, it reduces human risk and effort. Its autonomous design targets modern homes and workplaces with frequent cleaning needs.

Gholap et al, [12] introduced a multipurpose cleaning robot using IoT and image processing. It supports functions like window and solar panel cleaning with mechanisms including line-following and gesture control. Water spraying is automated using a pump and DC motor setup. The system integrates automation into everyday maintenance tasks.

Kukde et al, [13] developed a dual-mode vacuum cleaner robot for automatic and manual operation. It includes components like DC motors, brushes, mops, and sensors. RF modules enable remote control within a 50m range, and IR sensors help avoid obstacles. The robot emphasizes adaptability and user convenience.

Raviraj et al, [14] proposed a manually operated floor cleaner designed for areas with frequent power outages. This low-cost system doesn’t rely on electricity, making it practical for Indian transportation hubs. Materials and components are selected for affordability and durability. Finite element analysis confirms the structure’s reliability.

Jain et al, [15] created an autonomous cleaner suitable for homes and industries, capable of navigating and cleaning on its own. It uses sensors to detect obstacles and a controller to power the cleaning mechanism. The design enhances living standards by automating a routine but necessary chore.

Irawan et al, [16] developed a cleaning robot using Arduino UNO and ultrasonic sensors for autonomous navigation. The system avoids obstacles by adjusting direction when they’re detected within 15 cm. The motor shield and servo motors coordinate movement and cleaning, offering a flexible and effective floor-cleaning method.

. Simsek et al, [17] integrated fuzzy logic with path-planning algorithms to improve time and energy efficiency in smart cleaning robots. They addressed battery limitations and optimized path selection using CSP and search algorithms. The system adjusts cleaning behavior based on floor type and dirt levels, enhancing battery life and cleaning efficiency.

Das et al, [18] (duplicate of [4]) focused on an automated floor cleaner combining mechanical and electrical engineering. It targets hygiene improvement in schools, hospitals, and homes by replacing manual and fuel-based devices. The robot aims for efficient and eco-friendly cleaning.

Watkar et al, [19] introduced a dual-mode automatic cleaner controlled via phone app. Built with Arduino UNO, L293D driver, and ultrasonic sensors, the system supports both manual and autonomous functions. Its goal is to improve household cleaning with simple controls and efficient components.

Md Som et al, [20] presented a cleaning robot with Android-based control for both dry and wet cleaning. Using Bluetooth (HC05) and Arduino, the robot responds to smartphone commands. It simplifies cleaning with intelligent switching between cleaning modes, ideal for homes and offices.

Gupta et al, [21] developed a floor-cleaning robot that performs dry, wet, and dust collection functions. Controlled via an Android app and Arduino system, the robot simplifies traditional vacuuming. Its intelligent interface and autonomous features enhance indoor hygiene with minimal user effort.

Kulkarni et al, [22] created an autonomous robot capable of both wet and dry cleaning for homes and offices. The system aims to minimize manual labor while boosting efficiency. Designed for multi-surface use, it simplifies daily cleaning with integrated sensors and automated motion.

Tiwari et al, [23] developed a Bluetooth-controlled cleaner with both dry and wet modes. It focuses on simplifying the cleaning process for users in schools, homes, and offices. The robot combines efficiency and ease of use through mobile-based control and automation.

Singh and Verma [24] developed a multifunctional floor cleaning robot designed for efficient indoor operation. The system uses an Arduino Mega microcontroller to manage sensors and actuators. An ultrasonic sensor detects obstacles for safe navigation, while an infrared sensor enhances the detection of low-level impediments. The robot features Wi-Fi connectivity for remote operation via a smartphone app. Powered by a rechargeable battery, the cleaner offers both portability and reliable performance in varied indoor environments.

Ruser et al, [25] highlighted micro-sensor enhancements in central vacuum cleaners to improve user comfort and air quality. These systems reduce indoor dust and allergens while remaining quieter and more powerful. The integration of microcontrollers and multi-chip technology enhances adaptability to floor conditions.

Sharada L. Kore et al. [26] developed a smart floor cleaning robot that reduces human involvement by integrating both mopping and vacuuming functions. The system includes a water container with a mop and a vacuum pump to suck dust particles. Controlled by an Arduino Mega microcontroller, it uses a GSM module to wirelessly communicate operation status to users through acknowledgment messages. The design targets domestic spaces by improving cleaning efficiency and user interaction with real-time notifications.

Dattatray M. Kumbhar et al. [27] designed a smart robot for autonomous cleaning in homes and commercial areas, combining mopping, sweeping, and sanitizing functionalities. The robot uses dual-mode cleaning with brushes and vacuum technology and is controlled by an Arduino Mega 2560. It navigates using sensors like magnetometers, ultrasonic, and encoders to avoid obstacles, enabling it to cover large spaces effectively while maintaining cleaning precision.

Pooja D. Rathod et al. [28] presented a multipurpose smart floor cleaner controllable through an Android device via Bluetooth. The robot uses PMDC motors and obstacle-detection sensors for dry and wet cleaning tasks. It incorporates indoor GPS-like localization to optimize area coverage and systematic cleaning strategies, offering remote user control to enhance cleaning performance in residential settings.

Shritika Wayker et al. [29] proposed a cost-effective floor cleaning robot managed via an Android app using Bluetooth communication. The robot employs an Arduino Uno, IR sensors for obstacle detection, and a cleaning arm with water spraying and drying features. Ultrasonic sensors help autonomous navigation around obstacles, making it suitable for both home and office environments with a focus on affordability and simplicity.

Monika S. et al. [30] introduced a mobile-controlled cleaning robot featuring an Arduino Uno and Bluetooth communication. Equipped with a robotic arm that sprays water and stores it for wet floor cleaning, the system uses IR sensors to detect obstacles and motor drivers to control movement. Designed for household and office use, it aims to improve cleanliness with minimal manual effort.

Sonali A. Gaikwad et al. [31] developed a smart cleaning robot capable of wet and dry cleaning using sweeping and mopping mechanisms. It supports both automatic and semi-automatic modes controlled by Arduino ATmega328. Ultrasonic and IR sensors detect obstacles, while Bluetooth enables wireless communication. Motor and pump operations are managed via L293D and ULN2003 driver ICs, ensuring efficient cleaning in indoor spaces.

Shree Kande et al. [32] designed a wireless floor-cleaning robot that navigates and avoids obstacles using ultrasonic and infrared sensors. Controlled by an Arduino, the robot operates autonomously and adapts to various floor types with a spinning brush mechanism. Its lightweight design improves maneuverability, making it suitable for domestic use.

Suyog Patil et al. [33] created an autonomous floor cleaning robot that uses a Zig-Zag path strategy for optimal area coverage. Powered by an Arduino Mega, the robot integrates vacuum and dry cleaning capabilities and operates wirelessly. Sensor-based obstacle avoidance enhances safety, and the system’s cost-effectiveness makes it appropriate for home and office cleaning.

Zarinabegam Mundargi et al. [34] presented a cleaning robot controlled by Arduino Uno for dry and wet cleaning tasks. It employs DC motors, ultrasonic sensors for navigation, and an L298 motor shield for power distribution. The design emphasizes modularity and user-friendliness to facilitate autonomous operation in homes and offices.

Khaleel et al. [35] developed a low-cost smart cleaning robot that functions on both flat and complex surfaces using a differential drive model. Controlled via Bluetooth from an Android app, the robot uses ultrasonic sensors for obstacle detection and an Arduino Nano microcontroller for precise navigation. The system is affordable (USD 28), highly mobile, and designed for flexibility, making it accessible for everyday users.

Kumar et al. [36] proposed an Arduino-based autonomous vacuum cleaner robot intended for hazardous or inaccessible areas. It employs ultrasonic sensors to detect obstacles and motor shields for drive control. Powered by a 12V battery and using CPU fans for suction, the system focuses on efficiency and portability to enable cleaning without human intervention in sensitive environments.

Akash Kumbar et al. [37] designed a dust-cleaning robot featuring sweeping and mopping capabilities with detachable mops. It uses ultrasonic sensors and RF modules to allow both automatic and manual control within a 50-meter wireless range. The robot includes an automatic water sprayer and drying fan, targeting home users with its flexible and simple design.

Pranal Mahajan et al. [38] developed a smart home cleaning robot that integrates vacuuming, mopping, water spraying, and UV sterilization. The system is controlled using an Arduino UNO and a dual-axis joystick, powered by a 12V battery with a detachable charging unit. It features motor drivers and UV sterilization to enhance hygiene, designed for multifunctional use in domestic environments.

Adithya et al. [39] presented an automatic cleaning and mopping robot suitable for Indian households and industrial applications. Controlled via Bluetooth, it uses sensors and motors selected based on design evaluations. The robot aims to minimize human effort with a sturdy chassis and can operate up to one hour per charge, focusing on affordability and simplicity.

Uman Khalid et al. [40] introduced the CLEAR robot supporting both autonomous and manual cleaning modes. It complies with IEEE 1621 standards, using ultrasonic, infrared, and tactile sensors for navigation. A GUI facilitates manual control, and an auto-dirt disposal system is included. The design emphasizes power efficiency, user-friendliness, and local manufacturability for consumer and industrial use.

Bergman and Lind [41] explored a cost-effective robot vacuum cleaner design using Arduino Mega, ultrasonic sensors, and stepper and DC motors. The project focused on driving patterns, obstacle avoidance, and automated return-to-base charging. The prototype used 3D-printed parts and low-cost components, with iterative design testing comparing performance against commercial models.

Samuel Twum et al. [42] developed a smart floor-cleaning robot incorporating wet cleaning, controlled via an Android app. Using a NodeMCU (ESP32) microcontroller with Bluetooth and Wi-Fi, it manages movement, water-level monitoring, and obstacle detection through ultrasonic sensors. Alerts are provided by a buzzer and an LCD displays operational status. The system targets homes, schools, and offices for efficient, automated cleaning.

Swathi R. et al. [43] designed a Bluetooth-controlled floor cleaning robot using a Raspberry Pi 3, L298N motor drivers, and DC motors. It features brushing and mopping with a submersible pump for wet cleaning. Navigation is controlled via the Blue Dot app, and the robot is powered by a 12V battery. The system offers thorough cleaning with local components while remaining user-friendly and affordable.

Singh and Verma [44] developed a hybrid autonomous floor-cleaning robot combining vacuum suction, mopping, and UV sterilization features. The system uses an Arduino Mega microcontroller interfaced with ultrasonic and IR sensors for precise navigation and obstacle avoidance. Controlled through a custom mobile app via Bluetooth, it supports both automatic and manual modes. The design prioritizes energy efficiency and adaptability for diverse floor types, aiming for enhanced hygiene in residential and commercial spaces

Ruri Ashari Dalimunthe et al. [45] presented a floor-cleaning robot controlled by voice commands through an Android app using Bluetooth. An Arduino Uno microcontroller with HC-05 Bluetooth module interprets voice commands to move the robot forward, backward, left, right, or stop. Intended for urban homes, it reduces manual cleaning effort with simple motor control for small residential spaces.

Sharada L. Kore et al. [46] developed a smart cleaning robot integrating vacuuming and mopping, controlled by an Arduino Mega. The robot includes a water-dipping mop and a vacuum pump for dust suction. A GSM module sends status messages to users, and obstacle detection allows smooth navigation. Compact design and wireless communication enhance usability in home environments.

Pranal Mahajan et al. [47] illustrated a smart home cleaning robot with vacuuming, mopping, drying, water spraying, and UV sterilization. Powered by a 12V battery and controlled with a dual-axis joystick and Arduino UNO, it has a detachable charging unit and UV sterilizer box. The design is cost-effective, portable, and multifunctional, ideal for residential and small commercial cleaning.

Hind Zuhair Khaleel et al. [48] proposed a low-cost, high-precision floor cleaning robot employing a differential drive and ultrasonic sensors. Controlled via Android and Bluetooth using Arduino Nano, it performs well on flat and complex surfaces with minimal navigation errors. Priced at $28, the system emphasizes affordability, robustness, and user accessibility for everyday cleaning tasks.

Bergman and Lind [49] explored the design of a cost-effective robotic vacuum cleaner using Arduino Mega, ultrasonic sensors for obstacle detection, and a combination of stepper and DC motors for movement and brush control. The project focused on developing effective driving patterns, obstacle avoidance, and automated return-to-base charging. Components were 3D-printed and sourced affordably, with iterative testing to optimize performance compared to commercial vacuum robots.

Jadhav and Thombare [50] proposed an autonomous floor cleaning robot powered by Arduino Uno and equipped with ultrasonic sensors for navigation and obstacle avoidance. The robot features dual motor drive with L298N motor drivers and utilizes a simple cleaning mechanism combining a brush and vacuum suction. It is designed to operate autonomously in indoor environments, emphasizing low cost and ease of maintenance

2.2 Summary

The literature survey reviews various smart and autonomous floor-cleaning robots developed to reduce manual effort and enhance cleaning efficiency in domestic and commercial environments. Most systems are built using cost-effective microcontrollers like Arduino Uno, Mega, or Nano, integrating features such as ultrasonic and infrared sensors for obstacle detection and navigation. Common functionalities include dry sweeping, wet mopping, vacuuming, water spraying, and in some cases, UV sterilization for disinfection. Wireless communication through Bluetooth, GSM, or Wi-Fi allows control via Android applications or smartphone interfaces, with some systems even supporting voice commands or graphical user interfaces. Path planning techniques such as Zig-Zag motion and indoor GPS-based localization improve area coverage and cleaning effectiveness. Power is typically supplied by 12V rechargeable batteries, and motor drivers like L298N or ICs such as L293D manage the movement of wheels, brushes, and pumps. Most designs focus on affordability, portability, and ease of use, often featuring manual, semi-automatic, and fully autonomous modes to suit user needs. These systems demonstrate a blend of mechanical design and embedded technology, paving the way for future advancements in smart cleaning solutions for homes, offices, and small industries.

CHAPTER 3

PROPOSED SYSTEM

3.1 Introduction

The increasing demand for automation in household and industrial cleaning applications has led to the development of smart floor cleaning systems. Traditional floor cleaning methods require manual effort and are often time-consuming, leading to inefficiencies in maintaining hygiene. Additionally, labour-intensive methods may result in inconsistent cleaning quality and increased physical strain on users. To overcome these challenges, we propose a Smart Floor Cleaning System that integrates automation, mobility, and user control for efficient cleaning operations.

This system is designed using an Arduino UNO microcontroller, a motor driver L298N, a Bluetooth module HC-05, servo motors for arm lifting, a water pump for spraying water, and is controlled via a mobile application. The Smart Floor Cleaning System enhances cleaning efficiency, reduces human effort, and ensures a systematic cleaning process through real-time control and automation. The mobile application interface allows users to select cleaning modes, start or stop cleaning, and monitor real-time progress. This makes the system more convenient and accessible for users across different environments, including homes, offices, and public spaces.

The system operates efficiently on both hard and smooth surfaces such as tile, marble, and wood, ensuring a versatile cleaning experience. It integrates a combination of wet and dry cleaning functionalities, allowing users to select the appropriate mode based on the level of dirt and surface type. The cleaning mechanism consists of rotating brushes for sweeping dust, a water pump for spraying, and a suction unit for drying the floor, ensuring a spotless finish. The system's modular design enables easy maintenance and upgrades, ensuring long-term usability and flexibility.

Furthermore, the system is adaptable and scalable, making it suitable for a wide range of applications. It can be modified to include additional features such as obstacle detection, automated path planning, and sensor-based navigation, allowing it to function autonomously in future iterations. Integration with artificial intelligence and IoT technology can further improve its efficiency, enabling smart scheduling and voice control compatibility through virtual assistants like Google Assistant and Alexa. The Smart Floor Cleaning System represents a step toward a more advanced, automated, and intelligent approach to maintaining cleanliness in residential and commercial spaces.

3.2 Advantages

- ❖ Automated Cleaning Process: The system automates floor cleaning, reducing manual effort and ensuring consistent and efficient cleaning operations.
- ❖ Remote Control and Operation: Users can control the system wirelessly via a mobile application using Bluetooth, providing flexibility and ease of use.
- ❖ Dual-Mode Cleaning: Supports both wet and dry cleaning modes, making it suitable for different floor types and levels of dirt accumulation.
- ❖ Time and Energy Efficiency: Reduces cleaning time while optimizing energy consumption, making it an efficient alternative to manual cleaning.
- ❖ Customizable Cleaning Modes: Users can select different cleaning modes based on surface type and cleaning requirements.
- ❖ Enhanced Hygiene and Health Benefits: Automated cleaning ensures thorough and regular maintenance, reducing dust, allergens, and bacterial growth.
- ❖ User-Friendly Interface: The mobile application offers an intuitive and easy-to-use interface for controlling the system.
- ❖ Low Maintenance and Cost-Effective: Designed with affordable and replaceable components, making it easy to maintain and repair.
- ❖ Smart Data Logging and Performance Tracking: The system records cleaning activities, helping users track performance and optimize future cleaning sessions.
- ❖ Adaptability and Scalability: Can be upgraded with additional features like obstacle detection, automated navigation, and AI-based cleaning optimization.

3.3 Work flow of the system

The Smart Floor Cleaning System follows a structured workflow to ensure efficient and automated floor cleaning. The process begins with the power supply and system initialization, where a rechargeable battery powers the entire system, including the Arduino UNO, motor driver, ECU (Electronic Control Unit), and various sensors. Once the system is powered on, the Arduino initializes all components and establishes communication with the Bluetooth module, sensors, and motors.Next, the user controls the system via the Bluetooth module, which allows for wireless communication between a mobile application and the Arduino UNO. Through the mobile app, users can send commands to start or stop the cleaning operation, control movement, and select different cleaning modes. This enhances user convenience and flexibility in managing the cleaning process.

Once activated, the system collects sensor data and detects obstacles using multiple ultrasonic sensors and an IR sensor. The ultrasonic sensors help in avoiding obstacles and navigating the environment, while the IR sensor aids in detecting surfaces and ensuring smooth operation without collisions. This real-time feedback allows the system to make necessary adjustments for effective cleaning.

For movement, the motor driver controls the motion of the Smart Floor Cleaner by operating four motors (M1, M2, M3, and M4). The front motors (M1 and M2) and rear motors (M3 and M4) work together to ensure smooth navigation across the floor. The movement is coordinated based on user commands and sensor feedback to prevent accidents and optimize coverage.As the system moves, the cleaning mechanism is activated. The ECU regulates the Brushless DC (BLDC) motor, which drives the cleaning brush or suction mechanism. If wet cleaning is selected, the water pump sprays water before the brushes start scrubbing the floor, ensuring effective dirt and stain removal. The servo motors also adjust the cleaning arm height to adapt to different surfaces, improving efficiency.

During operation, the system continuously processes data for real-time navigation and adjustments. If an obstacle is detected, the system alters its path or stops to prevent damage. The motor speed and cleaning functions are dynamically adjusted based on environmental conditions. Finally the system completes the cleaning process and notifies the user. Once the cleaning cycle is finished, the Smart Floor Cleaning System stops

operation and sends a notification to the user through the Bluetooth module. The system logs cleaning data for future reference, allowing users to track performance and maintenance needs.This structured workflow ensures that the Smart Floor Cleaning System operates efficiently, providing an automated, user-friendly, and effective cleaning solution for homes, offices, and public spaces.

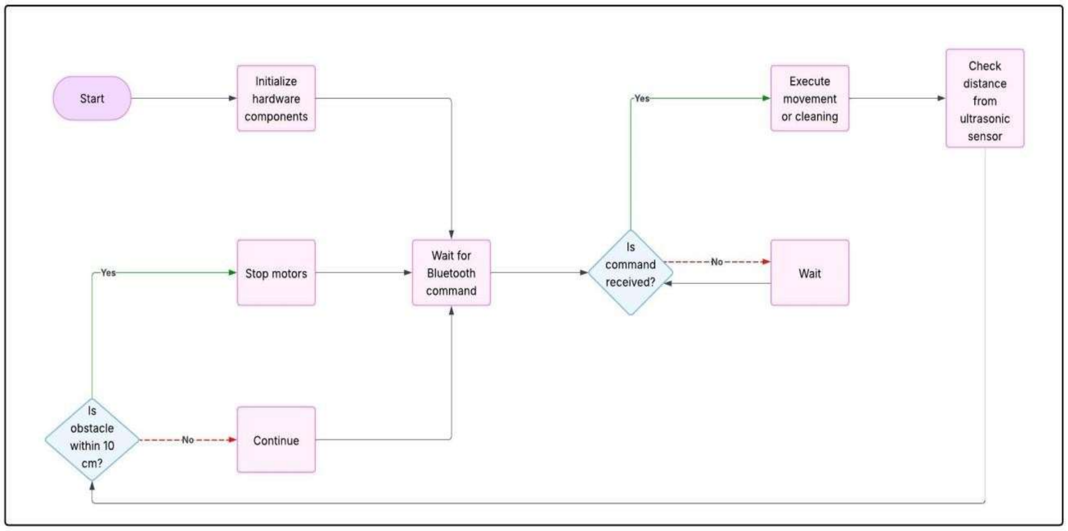

Start
Initialize hardware components
Wait for Bluetooth command
Is command received?
Yes
No
Execute movement or cleaning
Check distance from ultrasonic sensor
Wait
Stop motors
Yes
Is obstacle within 10 cm?
No
Continue

Fig 3.1: Flow Chart Of The Proposed System

3.4 Methodology

3.4.1 System Design and Architecture

This phase lays the groundwork for the entire project by outlining the components and how they interact.

- ❖ Microcontroller/Processor: Acts as the robot’s “brain.” Popular choices like Arduino or ESP32 handle input from sensors, control the motors, and execute cleaning algorithms. Arduino is often preferred for simpler setups, while ESP32 offers more processing power and Wi-Fi/Bluetooth capabilities.

- Motors and Actuators: These provide mechanical motion. DC motors drive the wheels, allowing the robot to navigate the environment. Servo motors or stepper motors handle more precise tasks like lifting arms, rotating brushes, or adjusting the mopping head.

- Sensors: Ultrasonic sensors help measure distances to obstacles, IR sensors detect nearby objects, and optionally, cameras or visual sensors help the robot identify dirty spots or map the surroundings.

- Power Supply: A rechargeable lithium-ion (Li-ion) battery pack powers the entire system. The capacity is chosen based on the energy requirements of the motors, sensors, and processor.

- User Interface (UI): This can be a physical control panel with buttons or more advanced interfaces like a mobile app or web dashboard, allowing users to start, stop, schedule, or monitor cleaning tasks.

3.4.2. Path Planning and Navigation

Efficient movement is key for cleaning coverage and battery conservation.

- Mapping: Sensors like ultrasonic or LIDAR scan the room and create a digital map, enabling the robot to know where it has cleaned and where it needs to go.

- Localization: The robot constantly updates its position relative to the environment using techniques like SLAM (Simultaneous Localization and Mapping). SLAM lets it adapt to changes, like a chair being moved or a new obstacle appearing.

- Obstacle Detection and Avoidance: Algorithms ensure the robot doesn’t collide with objects. A* or Dijkstra’s algorithm helps find the shortest path around detected obstacles. This step is crucial for smooth navigation in dynamic, cluttered spaces.

3.4.3 Cleaning Mechanism

The robot’s hardware must deliver effective floor cleaning.

- Brushes or Mopping Mechanism: Rotating brushes sweep dirt into a collection bin, while a mop or microfiber pad wipes surfaces. Some designs also incorporate suction for vacuuming.

- Water or Cleaning Solution Dispensing: A small pump system sprays water or cleaning fluid onto the surface before the mop engages. Controlled dispensing prevents excess water usage and ensures even coverage.

- Dirt Detection and Adaptive Cleaning: Capacitive or optical sensors can detect particularly dirty areas (e.g., mud spots), prompting the robot to perform focused cleaning or multiple passes in those regions.

3.3.4 Energy Management

Without good energy planning, the robot may stop mid-task or fail to return to its charger.

- Energy-Efficient Design: Prioritizing low-power sensors, lightweight materials, and efficient motor control reduces battery consumption. Smart path planning also minimizes unnecessary movement.

- Solar Assistance (optional): Integrating solar panels (e.g., on top of the robot) can help recharge batteries between tasks, especially useful in bright environments like office lobbies or sunlit rooms.

- Energy Monitoring: Battery voltage sensors and power management ICs track battery levels and predict when the robot should return to its charging dock.

3.4.5 Control System and Software Implementation

The “intelligence” of the robot lives here.

- ❖ Microcontroller Programming: Using tools like Arduino IDE, developers write code that connects all components: reading sensor inputs, calculating paths, controlling motors, and triggering the cleaning mechanisms.

- ❖ Algorithm Implementation: Path-planning strategies (like random, spiral, or systematic grid cleaning) and obstacle-avoidance routines are implemented to guide the robot through its tasks efficiently.

- ❖ Communication Protocols: For user interaction, Bluetooth allows local wireless control (e.g., from a smartphone), while Wi-Fi enables broader control, including integration with smart home systems or cloud services.

3.4.6 Testing and Validation

Before releasing the system for real-world use, thorough testing ensures reliability.

- ❖ Simulation: Software tools simulate the environment and the robot’s actions, helping spot issues in navigation logic or sensor interpretation before building the prototype.

- ❖ Prototype Testing: A physical robot is tested in various real-world settings (homes, offices, uneven floors) to observe cleaning performance, navigation accuracy, battery life, and sensor reliability.

- ❖ Iterative Improvement: Feedback from testing is used to refine the design — optimizing code, tweaking hardware setups, or improving mechanical components.

3.4.7 Integration with IoT and Remote Monitoring

Adding smart features elevates the system's functionality.

- ❖ Sensor Data Upload: The robot periodically uploads status reports — such as battery level, cleaning completion, or error logs — to a cloud server or local database.
- ❖ Remote Control: Users can schedule cleanings, change modes, or check progress from anywhere via a mobile app or web portal.
- ❖ Real-Time Feedback: Push notifications or app updates inform users about tasks (e.g., "cleaning complete," "low battery," "obstacle detected") to improve user interaction.

3.4.8 Final Evaluation and Optimization

In this last stage, the system is polished for long-term use.

- ❖ Performance Evaluation: Data from multiple cleaning cycles is analyzed to assess how efficiently and thoroughly the robot cleans, how much energy it uses, and how satisfied users are with its operation.
- ❖ Optimization: Hardware (like motor speed), software (like better pathfinding algorithms), or user experience (like app interface improvements) are fine-tuned. Over-the-air software updates can be delivered if Wi-Fi capability is included.

3.5 Summary

The proposed Smart Floor Cleaning System is an automated, Bluetooth-controlled robot built using an Arduino UNO, motor driver L298N, HC-05 Bluetooth module, servo motors, and a water pump, designed to efficiently clean various surfaces with both wet and dry functionalities. Its system architecture integrates key components such as microcontrollers, sensors, motors, and a mobile app interface for remote operation, allowing users to select cleaning modes, control movement, and monitor progress in real time. The robot employs path planning and navigation techniques, including obstacle detection and algorithms like SLAM and A*, to ensure thorough and energy-efficient coverage. The cleaning mechanism combines brushes, mopping, and controlled water dispensing, while energy management focuses on low-power design and smart battery monitoring. The software side involves microcontroller programming for sensor integration, movement control, and communication protocols, with extensive testing through simulations and real-world trials to refine performance. With potential IoT integration, users can remotely schedule and monitor cleaning, receiving real-time feedback, and the system is optimized through iterative improvements for enhanced efficiency, scalability, and long-term usability across home and commercial environments.

CHAPTER 4

HARDWARE & SOFTWARE REQUIRMENTS

4.1 Introduction

The Smart Floor Cleaning System is an innovative automation project designed to enhance the efficiency, convenience, and effectiveness of floor cleaning in residential, commercial, and public spaces. At its core lies the Arduino UNO microcontroller, which coordinates several hardware components such as servo motors for lifting and adjusting the cleaning arms, the L298N motor driver to control the DC motors for smooth mobility, a water pump to spray cleaning solution or water onto the surface, and the HC-05 Bluetooth module for wireless connectivity. This configuration allows users to remotely control the system using a mobile application, offering features such as switching between wet and dry cleaning modes, adjusting cleaning parameters, and receiving real-time updates. The system is designed to reduce manual effort while improving the overall consistency and quality of floor cleaning.

In terms of design, the system is modular and scalable, making it easy to maintain and upgrade. It supports future enhancements such as obstacle detection sensors and AI-based path planning for autonomous navigation. The adaptive cleaning arms help the device reach corners and clean over uneven surfaces more efficiently. Power consumption is carefully optimized to extend battery life, enabling the cleaner to operate for extended periods without frequent recharging. With customizable scheduling features, users can automate cleaning tasks according to their preferred timings. The user-friendly mobile interface ensures ease of use and is compatible with both Android and iOS platforms. This smart cleaning system represents a practical and cost-effective step toward everyday home automation.

Further enhancements can significantly extend the system’s capabilities. For example, LED indicators can be added to display battery levels, mode selection, and connectivity status. A buzzer module may be included to provide audio alerts during startup, shutdown, or fault detection. Rubber-coated wheels ensure better traction on various surfaces, while a detachable water tank makes refilling easy. Additional components like microfiber rollers or rotating brushes can improve cleaning efficiency. With the potential to incorporate a self-charging dock, the system could operate even more autonomously. Obstacle detection using infrared or ultrasonic sensors would prevent collisions and improve path planning. Advanced versions could include data logging features to analyze cleaning performance or integrate IoT capabilities for cloud-based monitoring. Overall, the system offers a valuable learning platform for embedded systems and robotics, while delivering a practical solution for smart, automated floor cleaning.

4.2 Block Diagram

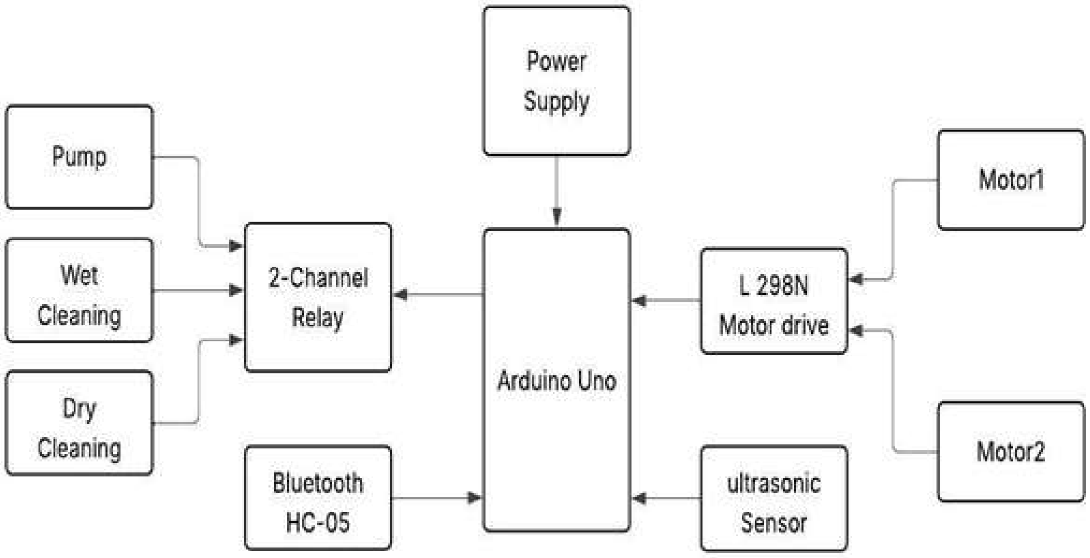


Fig 4.1: Block Diagram

The block diagram of the Smart Floor Cleaning System illustrates the integration and interaction of various components that enable efficient and automated cleaning. At the center of the system is the Arduino UNO, which acts as the main controller, processing inputs and controlling outputs based on programmed logic.The power supply provides the necessary voltage to all components, ensuring stable and uninterrupted operation. Two DC motors (Motor1 and Motor2) are connected through the L298N motor driver, which allows the Arduino to control the direction and speed of the motors, enabling the system to move in multiple directions.

For user interaction, the system uses the HC-05 Bluetooth module, which connects wirelessly to a mobile application. Through this connection, users can control the cleaner remotely, selecting cleaning modes or directing its movement.To handle different cleaning modes, a 2-channel relay module is used. The relay switches between wet cleaning, which activates a water pump to spray a cleaning solution, and dry cleaning, which typically involves brushes or suction mechanisms. This relay is also controlled by the Arduino based on the user's input.

An ultrasonic sensor is integrated for obstacle detection. It helps the system avoid collisions by sending distance data to the Arduino, which can then make movement decisions to navigate safely around objects.Each module works in synchronization under the control of the Arduino to perform efficient floor cleaning. The modular setup ensures the system is easily upgradable, such as adding AI for autonomous navigation or more sensors for environmental awareness. This architecture allows for flexibility, remote operation, and reliable performance in various settings like homes, offices, and public spaces.

4.3 Circuit Diagram

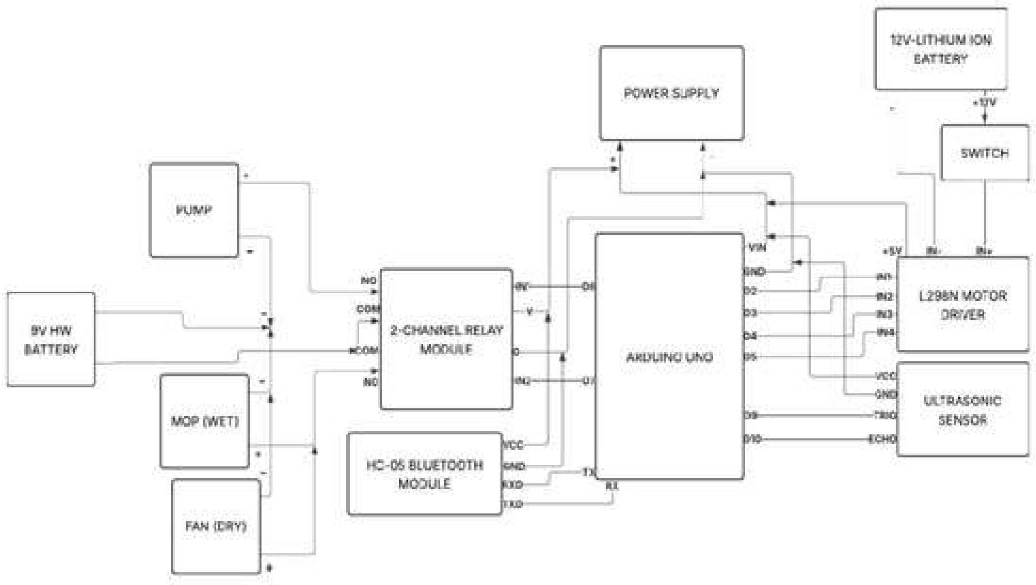


Fig 4.2: Circuit Diagram

The Smart Floor Cleaning System integrates multiple components to deliver efficient and autonomous cleaning. At the heart of the system is the Arduino UNO, which acts as the central controller, processing inputs and managing outputs. It receives commands from the HC-05 Bluetooth module, allowing users to wirelessly control movement and switch between wet and dry cleaning modes via a mobile app. The Arduino drives the L298N motor driver using digital pins D2 to D5, which in turn powers two DC motors connected to a 12V lithium-ion battery via a switch, enabling forward, backward, and turning motion. For obstacle avoidance, an ultrasonic sensor connected to pins D9 and D10 of the Arduino detects nearby objects using echo pulses. The system features a two-channel relay module controlled via pins D6 and D7, which switches between the water pump, mop (wet cleaning), and fan (dry cleaning), all powered by a separate 9V HW battery. This separation of power sources ensures system stability by isolating the high-current motor operations from the cleaning mechanisms. The water pump sprays liquid for wet cleaning, the mop wipes the floor, and the fan is used to dry surfaces or blow away dust during dry mode. A power supply module ensures proper voltage regulation from the 12V battery to the Arduino and motor driver, maintaining consistent performance. Altogether, this setup creates a modular, flexible, and remotely operable cleaning solution capable of navigating obstacles and performing dual cleaning functions, making it suitable for modern households and commercial spaces.

4.4 Hardware Components

4.4.1 Arduino UNO

Fig 4.3: Arduino UNO

The ATmega328P microcontroller operates at a stable 5V voltage, with a recommended input voltage range of 7–12V and a maximum input limit between 6–20V, providing flexibility in power supply options. It is equipped with 14 digital I/O pins, 6 of which support Pulse Width Modulation (PWM) output on pins 3, 5, 6, 9, 10, and 11, allowing for control of motors, LED dimming, and other analog-like operations. Additionally, the microcontroller features 6 analog input pins (A0 to A5), which enable it to read sensor data and perform analog-to-digital conversions. Each I/O pin can handle a maximum DC current of 20 mA, while the 3.3V pin can supply up to 50 mA, providing power for low-voltage components.

The ATmega328P offers 32 KB of flash memory (with 0.5 KB reserved for the bootloader), 2 KB of SRAM, and 1 KB of EEPROM, providing ample space for program storage and persistent data storage. The microcontroller runs at a clock speed of 16 MHz, driven by a quartz crystal oscillator, ensuring reliable performance for a variety of applications. It is equipped with a Type-B USB connector for both programming and power, and it includes an In-Circuit Serial Programming (ICSP) header, allowing for direct programming of the microcontroller. For communication, it supports UART serial communication on pins 0 (RX) and 1 (TX), as well as SPI through pins 10 (SS), 11 (MOSI), 12 (MISO), and 13 (SCK). It also offers I2C communication on pins A4 (SDA) and A5 (SCL).

Additional features of the ATmega328P include a manual hardware reset button and an onboard LED indicator connected to digital pin 13, which can be used for debugging and visual feedback. The microcontroller includes both 5V and 3.3V onboard voltage regulators, providing necessary power for external components. With compact dimensions of 68.6 mm by 53.4 mm and a weight of approximately 25 grams, it is a lightweight and space-efficient option for embedded systems and electronics projects, offering robust performance and versatility in a wide range of applications.

4.4.2 Servo Motor

Fig 4.4: Servo motor

A servomotor is a precise electromechanical device used in closed-loop control systems, where it receives a control signal—either analog (voltage level) or digital (typically PWM, i.e., Pulse Width Modulation)—that represents the desired output position. The motor is fitted with a position sensor such as a potentiometer or an encoder that provides real-time feedback of the shaft's position to the internal controller. This feedback mechanism enables the servomotor to continuously compare the current position with the target position. Based on the difference, known as the error signal, the motor is driven in the appropriate direction to reduce the error and align the output shaft to the commanded position.

When the shaft nears the desired angle, the error signal decreases, reducing the motor's speed, and eventually becomes zero when the position is matched, causing the motor to stop or hold. This feedback control loop ensures high accuracy and repeatability. In advanced servomotor systems—typically used in industrial automation, robotics, and CNC machinery—additional feedback such as speed or torque is also utilized, often involving high-resolution optical encoders or Hall-effect sensors. These systems usually implement PID (Proportional-Integral-Derivative) control algorithms to achieve smooth, accurate, and dynamic performance.

The servomotor unit often includes a DC or AC motor (brushed or brushless), a control circuit, a gear reduction system to increase torque while lowering speed, and the feedback sensor. In low-cost applications such as hobby robotics or radio-controlled models, servos typically use potentiometers for simple position feedback and implement bang-bang control, where the motor runs at full speed in either direction or is off—without any intermediate control. These types are less suited for industrial-grade motion control due to their limited resolution and lack of smooth operation but are ideal for compact, cost-sensitive applications.

4.4.3 Motor Driver

Fig 4.5: Motor Driver

The L298N is a dual H-bridge motor driver integrated circuit commonly used for controlling DC and stepper motors in a wide range of embedded and robotics applications. Based on the H-bridge configuration, it can independently drive two motors with both bidirectional control and variable speed. The H-bridge arrangement allows current to flow in either direction through the motor coils, enabling forward and reverse rotation. The IC supports an input voltage range from 5V to 46V, and each channel can deliver a continuous output current of up to 2A, with peak currents up to 3A for short durations. This makes the L298N suitable for medium-power motor control applications, such as mobile robots, conveyor belts, and small automated systems.

The L298N includes built-in flyback diodes (clamp diodes) that protect the circuit from voltage spikes caused by the inductive load of motors during switching—commonly known as back-EMF. It also features an on-board 5V voltage regulator, which can be optionally enabled to provide regulated power to external logic circuits or the IC itself, depending on the design requirements. The logic input pins accept standard TTL/CMOS logic levels, making it compatible with microcontrollers like Arduino, Raspberry Pi, and

STM32. The driver has four input pins (IN1–IN4) and two enable pins (ENA, ENB) for controlling motor direction and speed, typically using PWM signals for speed modulation.

From a thermal and electrical protection standpoint, the L298N IC is mounted on a multiwatt-15 package with a metal tab that can be attached to a heat sink to manage heat dissipation. However, the internal voltage drops across the transistors result in significant power loss, especially under high current loads, so external heat sinking is often necessary for sustained high-current operation. The IC does not have built-in current sensing or overcurrent protection, so these features must be implemented externally if required. Despite being slightly outdated compared to modern MOSFET-based drivers, the L298N remains a reliable and cost-effective solution for educational and prototyping purposes where moderate performance is acceptable.

4.4.4 Water Pump

Fig 4.6: Water Pump

The design of a small pumping system for efficient water transfer requires careful selection and integration of mechanical, electrical, and hydraulic components. The choice of pump—whether submersible, centrifugal, diaphragm, or peristaltic—depends on parameters such as required flow rate (Q), head (H), pressure, and the nature of the fluid being transferred. For low to moderate head applications with clear water, centrifugal pumps are typically preferred due to their high flow capability and energy efficiency. In contrast, submersible pumps are more suitable for deep well or underground water extraction, where the pump must operate while fully submerged. The system is powered either by an AC/DC power source or, in off-grid scenarios, a solar PV system with a charge controller and battery storage for autonomous operation.

Hydraulic design considerations include pipe diameter sizing based on flow rate and velocity to reduce head losses, typically calculated using the Darcy-Weisbach or Hazen-Williams equations. Properly sized non-return valves, pressure relief valves, and isolation valves are critical for maintaining unidirectional flow, avoiding water hammer, and allowing for safe maintenance. Sensors such as float switches, ultrasonic level detectors, and flow meters provide real-time feedback for monitoring and automation. The use of microcontrollers or programmable logic controllers (PLCs) enables integration of control logic, such as automatic start/stop based on tank levels, leak detection, and dry-run protection using current sensing or pressure switches.

To improve system efficiency and reliability, materials such as HDPE, uPVC, or stainless steel are chosen for piping and fittings due to their corrosion resistance and durability in varied water quality conditions. The design should incorporate minimal bends and fittings to reduce frictional losses and should avoid cavitation by maintaining positive suction head. Energy consumption can be minimized through the use of variable frequency drives (VFDs) or PWM-based speed controllers for pumps, allowing the system to adapt to changing flow demands. Redundancy, such as dual pumps or backup power supplies, should be included in critical systems to ensure continuous operation during maintenance or failure. With proper design, sizing, and automation, a small pumping system can deliver high operational efficiency, low maintenance, and reliable performance across domestic, agricultural, and small-scale industrial water management applications.

4.4.5 DC Motors

Fig 4.7: DC Motor

Electric motors are available in a wide range of types and sizes to suit diverse motion control applications. Among the most common are brushless DC motors (BLDCs), servo motors, and gear motors. Brushless motors use electronic commutation instead of mechanical brushes, providing higher efficiency, longer lifespan, and better speed-torque characteristics. Servo motors, typically used in precision positioning systems, integrate a motor with a feedback sensor such as an encoder or resolver and a control system to maintain accurate control over position, speed, and torque. Gear motors combine a motor with a gear reduction unit to increase torque output while reducing speed, making them ideal for applications that require high force at low rotational speeds.

Each of these motor types consists fundamentally of a rotor (the rotating part) and a stator (the stationary part). The stator produces a magnetic field, either through permanent magnets or electromagnetic windings, which interacts with the rotor to produce torque. The rotor may be a wound armature or contain permanent magnets depending on the motor type. In brushless designs, the stator windings are energized in a sequence controlled by an electronic controller based on feedback from position sensors, allowing precise modulation of motor speed and direction. Advanced control methods, such as field-oriented control (FOC), are used in high-performance motors to optimize torque production and minimize power loss.

These motors and their associated motion control components form one of the most comprehensive categories in automation and electromechanical systems. The ecosystem includes bearings for smooth rotation, bushings, clutches, brakes for motion interruption, and drives that act as electronic interfaces to regulate motor behavior. Devices like encoders and resolvers provide feedback for position and speed control, while limit switches and position sensors help prevent overtravel and ensure accurate operation. Linear actuators and rotary actuators convert motor output into linear or rotational movement, and components like slides, guides, and positioning stages enable precise mechanical alignment. Supporting elements like pneumatics, springs, solenoids, and mechanical seals complete the motion control system, offering tailored solutions for robotics, industrial automation, automotive, aerospace, and medical devices.

4.4.6 Bluetooth Module HC05

Fig 4.8: Bluetooth Module XC05

The HC-05 Bluetooth module is a popular and cost-effective solution for wireless serial communication in embedded systems. It operates using the Bluetooth v2.0+EDR (Enhanced Data Rate) standard and supports a Class 2 range, typically up to 10 meters, with a maximum data rate of around 2.1 Mbps (theoretically). However, in practice, its USART (Universal Synchronous Asynchronous Receiver Transmitter) interface typically runs at a default baud rate of 9600 bps, though it can be configured between 1200 and 1382400 bps in command mode. It supports full-duplex communication, allowing simultaneous bidirectional data transfer, which makes it ideal for controlling microcontrollers such as Arduino, STM32, or PIC and communicating with Bluetooth-enabled devices like smartphones and laptops.

The HC-05 module operates in two distinct modes: Command mode (AT mode) and Data mode. In command mode, users can send AT commands to configure settings such as baud rate, device name, password, and operating roles (Master or Slave). By default, the module powers up in slave mode, but it can be reprogrammed to operate as a master, enabling it to initiate connections to other Bluetooth devices. The mode is determined by the logic level of the KEY pin, which, when pulled high during power-up, enables AT command mode. The module operates at 3.3V logic level, though the power supply input can tolerate 3.6V to 6V, thanks to an onboard voltage regulator. Direct connection of 5V logic without a voltage divider may damage the module.While the HC-05 is excellent for basic wireless serial communication—such as sensor data transmission, wireless robot control, or UART debugging—it has limitations in bandwidth and protocol support. It is not suitable for high-bandwidth applications like audio streaming or multimedia file transfer. For such applications, advanced Bluetooth modules like the CSR8645, which supports Bluetooth A2DP (Advanced Audio Distribution Profile) and other audio-centric protocols, are preferred. The HC-05 does not support Bluetooth Low Energy (BLE), so for BLE-specific use cases, modules like HM-10 or nRF52-based modules are more appropriate. Despite these constraints, the HC-05 remains a reliable choice for UART-based wireless communication in embedded and hobbyist projects.

4.4.7 Ultrasonic Sensor

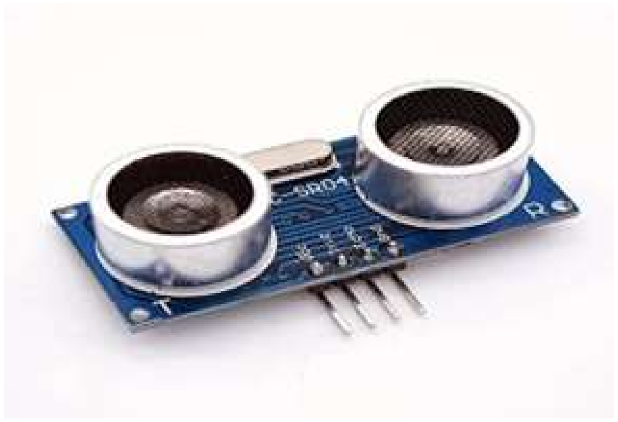


Fig 4.9: Ultrasonic sensor

An ultrasonic sensor is an electronic device that uses high-frequency sound waves, typically around 40 kHz, to detect the presence or measure the distance of objects without making physical contact. It consists of a transmitter that emits ultrasonic pulses and a receiver that detects the echo reflected from a surface. These sensors are widely used in applications such as obstacle detection in robotics, material level monitoring in industrial tanks, and proximity sensing in automation systems because they work reliably in various lighting and environmental conditions.A common example is the HC-SR04 ultrasonic sensor module, which provides accurate distance measurements typically ranging from 2 cm to 400 cm. It uses two digital pins—one to trigger the sound pulse and the other to receive the echo—making it compatible with most microcontrollers like Arduino or Raspberry Pi. Some advanced ultrasonic sensors include features like temperature compensation and built-in signal filtering to enhance measurement accuracy in changing or noisy environments.

Ultrasonic sensors offer several advantages, such as non-contact operation, resistance to dust and light interference, and low cost, which make them suitable for a wide range of consumer and industrial applications. However, they may not perform well with soft, absorbent, or angled surfaces, as such materials can scatter or absorb the sound waves, reducing accuracy. Despite these limitations, their simplicity, durability, and effectiveness have made ultrasonic sensors a key component in modern automation and embedded systems.

4.4.8 IR Sensor

Fig 4.10: IR Sensor

An infrared (IR) sensor is a device that uses infrared radiation to detect objects, measure distance, or determine the presence of a material. These sensors operate in the infrared spectrum (typically wavelengths between 700 nm to 1 mm), which is just beyond the visible spectrum of light. An IR sensor usually consists of two components: an IR LED that emits infrared light and a photodiode or phototransistor that detects the reflected infrared light. When the emitted IR light hits an object, it reflects back toward the sensor, allowing it to detect the presence and sometimes the distance to the object based on the intensity or time taken for the light to return.

In applications like proximity sensing, IR sensors work by measuring the reflected light intensity. The sensor's output is typically analog or digital, where higher reflected light indicates a closer object. These sensors can detect objects at a range of distances, depending on the sensor's power, design, and the reflective properties of the object's surface. They are widely used in applications such as object detection, line following robots, and remote controls, as well as in safety systems like motion detection and alarm systems.

The main advantages of IR sensors include their low cost, low power consumption, and non-contact nature, which make them ideal for various consumer electronics and industrial automation applications. However, their performance can be affected by ambient light, particularly in the presence of sunlight or other infrared sources, which may reduce their accuracy. Additionally, IR sensors may struggle to detect objects with dark or non-reflective surfaces, as these surfaces absorb infrared light rather than reflecting it back to the sensor. Despite these limitations, IR sensors are still widely used in many applications due to their simplicity and versatility.

4.4.9 Jumper Wire

Fig 4.11: Jumper Wire

Jumper wires are essential components in electronics and embedded systems used for making temporary connections between different parts of a circuit, such as between a microcontroller and external devices like sensors, LEDs, or motors. These wires are typically made of insulated copper or tinned copper and come in different lengths, colors, and connector types, including male-to-female, male-to-male, and female-to-female configurations. The color coding of jumper wires helps to visually organize connections, but it doesn't indicate electrical properties. They are designed for use with breadboards, where they can be inserted into the breadboard's rows and columns of holes, enabling easy prototyping and experimentation without soldering.

Jumper wires are commonly used for prototyping and debugging, as they allow for quick adjustments and changes in a circuit design. The wires are typically constructed with a 22 AWG (American Wire Gauge) size, which strikes a balance between flexibility and current-carrying capability. They can handle currents up to 2-3 Amps depending on the wire length and the insulation used. Stranded wire is often preferred for jumper wires due to its increased flexibility, while solid-core wires are stiffer and can be easier to use for stable connections in temporary setups.

In addition to their use in prototyping, jumper wires are also employed in testing and debugging applications where a temporary connection is required to check the functionality of a specific part of the circuit. They offer convenience for tasks such as connecting a microcontroller to sensors, testing voltage levels at different points, or creating connections between power and ground rails. While jumper wires are versatile and inexpensive, they are primarily intended for low-power, low-frequency applications, and are not suitable for permanent, high-load, or high-frequency connections.

4.4.10 2-Channel Relay Module

Fig 4.12: 2-Channel Relay Module

2-channel relay module, which is commonly used in electronics to control high-voltage or high-current devices using low-voltage signals. This particular module is designed with two relays, typically capable of controlling devices like motors, lights, or other electronic appliances. The relays are usually mechanical switches that act as intermediaries between the low-power control circuit (e.g., a microcontroller like Arduino) and the high-power load (e.g., a household appliance). The relays can be controlled via digital signals from the microcontroller, with the module typically operating at 5V or 12V for the relays' activation.

The module uses transistor-driven relay drivers, ensuring that the microcontroller's low-power output can safely control the relays. It also often includes diodes to protect against voltage spikes caused by the inductive load when the relay opens or closes (back-EMF protection). The relay contacts are rated for various voltages and currents, and this particular model likely supports up to 250V AC or 30V DC with a current rating of 10A per channel. These relays allow for both switching the high-power load on and off, making them ideal for applications in home automation, robotics, and automotive systems where precise control of high-power devices is required.

The relay module typically has several input pins: VCC (5V/12V) for powering the module, GND for the ground connection, and IN1 and IN2 for controlling the individual relays. Additionally, there are NO (Normally Open), NC (Normally Closed), and COM (Common) terminals for the relays, where the load is connected. These relay modules are widely used due to their simplicity, ease of integration with microcontrollers, and ability to interface low-voltage electronics with high-power systems. The status LEDs on the module indicate whether each relay is activated, providing visual feedback during operation.

4.4.11 Rocker Switch

Fig 4.13: Rocker Switch

Standard rocker switch, commonly used in various electronic devices and systems for controlling power. The rocker switch operates on a simple mechanism where a button-like component pivots over a central axis, toggling between two positions. These positions are typically marked with an "I" (for ON) and an "O" (for OFF), corresponding to the electrical connections inside the switch. When the switch is toggled to the "I" position, the circuit is completed, allowing current to flow; when toggled to "O," the circuit is broken, and no current flows.

Rocker switches are often used in applications requiring a simple on/off function. They can handle a variety of voltages and currents, making them suitable for many uses in consumer electronics, household appliances, and industrial equipment. The size and construction of the switch make it easy to integrate into panels and control boards, as it can be mounted through a hole and secured by a locking mechanism or clips. These switches typically feature a durable design and are built to withstand repeated use.

In terms of electrical specifications, a typical rocker switch like the one shown might be rated for a voltage of 125V or 250V AC, with current ratings commonly in the range of 10 to 15 amps. Some rocker switches are also available with additional features such as LED illumination or dustproof and waterproof designs, depending on the application. The polarity of the switch's electrical connection and the material quality of the contacts also influence the performance and durability of the switch over time.

4.4.12 3.7V Lithium ion Rechargeable Battery

Fig 4.14: 3.7V Lithium ion Rechargeable Battery

A 3.7V rechargeable battery is a commonly used power source, especially for portable electronics such as smartphones, tablets, drones, and electric vehicles. These batteries typically use lithium-ion (Li-ion) or lithium-polymer (LiPo) chemistry, which allows them to store a substantial amount of energy in a compact form. Lithium-ion batteries are known for their high energy density, longer lifespan, and light weight compared to other battery types. The 3.7V rating refers to the nominal voltage of a fully charged cell, which typically ranges from 4.2V when fully charged to 3.0V or 3.2V when discharged.

The battery’s capacity is an important factor, usually measured in milliampere-hours (mAh) or ampere-hours (Ah). A higher mAh value indicates a larger capacity, meaning the battery can provide more energy for a longer period. For example, a 2000mAh battery can theoretically supply 1 amp of current for 2 hours. Lithium-based rechargeable batteries also have a higher cycle life than traditional alkaline batteries, typically ranging from 300 to 500 full charge-discharge cycles, depending on the quality and usage.

One key feature of these batteries is their protection circuits, which prevent overcharging, deep discharging, and short-circuiting, ensuring safe usage. The batteries are often integrated with a Battery Management System (BMS) that monitors voltage, temperature, and charging current to enhance performance and safety. However, due to their chemical composition, they require proper handling, including avoiding exposure to extreme temperatures or overcharging, which could lead to degradation or, in rare cases, thermal runaway.

4.4.13 9V HW Battery

Fig 4.15: 9V HW Battery

A 9V hardware (HW) battery is a small, portable power source commonly used in electronic devices such as smoke detectors, radios, and portable gadgets. It typically consists of six 1.5V cells arranged in series within a rectangular casing. The 9V battery is known for its compact size and ability to deliver a steady voltage output, making it suitable for low-power devices. Its standard size allows it to fit easily into a variety of consumer electronics.

These batteries are commonly of two types: alkaline and lithium. Alkaline 9V batteries are the most widely used due to their cost-effectiveness and availability. They provide a reliable power source but have a limited lifespan, usually lasting around 5 to 10 hours in continuous use, depending on the device. Lithium 9V batteries, on the other hand, offer longer battery life, more stability, and better performance at extreme temperatures, but they tend to be more expensive than alkaline ones.

In terms of technical specifications, a typical 9V battery provides a nominal voltage of 9 volts, with a capacity ranging from 400mAh to 600mAh for standard alkaline versions. The discharge rate of the battery varies depending on the load but can provide a steady voltage for extended periods at lower currents. However, these batteries are not ideal for high-drain devices and may lose voltage rapidly under heavy loads. As a result, proper selection based on device requirements is crucial for optimal performance.

4.4.14 Robot Wheels

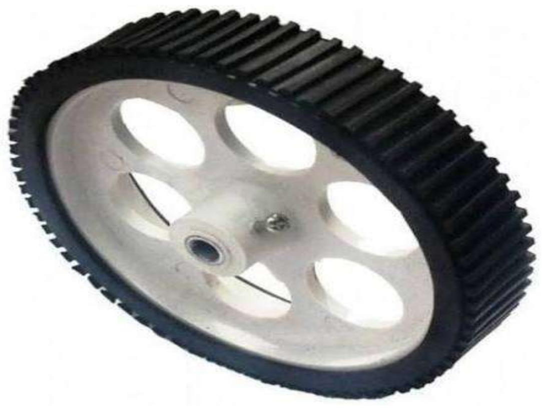


Fig 4.16: Robot wheels

Plastic robotic wheel with a rubber grip, often used in educational and hobby robotics. It typically has a diameter of around 70mm to 100mm and a width of about 8mm to 20mm. The center hub is designed to fit standard motor shafts (commonly 6mm diameter) and may include a screw or locking system to secure it to the motor. The wheel's body is made of lightweight yet strong plastic, reducing the overall weight of the robot and minimizing the load on the motor.

The outer tread is made of synthetic rubber with deep grooves or ridges that improve traction on different surfaces like wood, tile, or carpet. This design ensures better grip, prevents slippage, and allows for smoother motion even at moderate speeds. The tread also absorbs minor shocks, which helps maintain stability and reduces vibrations during movement. The rubber ring is generally press-fitted or glued to the plastic base, offering a balance between grip and ease of manufacturing.

From a performance perspective, this type of wheel is ideal for low- to medium-speed robotic applications, such as line-following robots, maze solvers, and prototype delivery bots. It performs best with DC geared motors ranging from 100 to 300 RPM. The wheel's lightweight and rigid structure allow for quick acceleration and precise turns. It’s compatible with various mounting options, such as set-screw hubs or D-shaped motor shafts, making it a versatile and essential component in mobile robotic platforms.

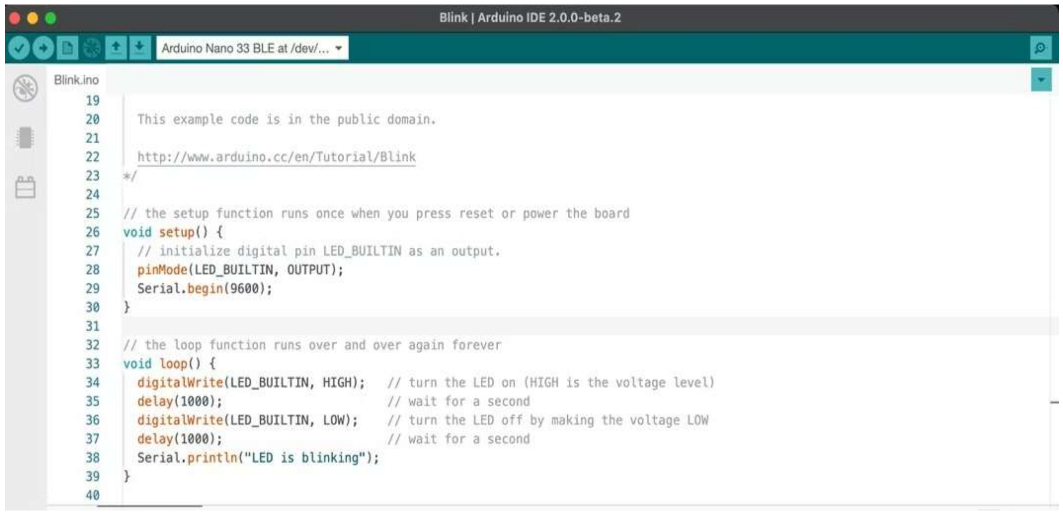
4.5 Software Requirements

4.5.1 Microcontroller Programming Tool (Arduino IDE)

The Arduino IDE is a beginner-friendly integrated development environment that supports programming microcontrollers like the ESP8266 NodeMCU, making it well-suited for projects such as the Smart Polyhouse Irrigation Management System. It provides a simple and intuitive platform for writing, uploading, and debugging code written in C/C++, as shown in Fig 4.5.1One of the main advantages of the Arduino IDE is its widespread support for various microcontrollers and its compatibility with a broad range of community-contributed libraries. For the ESP8266, it allows seamless integration after installing the appropriate board support package through the Board Manager. This enables developers to write efficient firmware that can interface with sensors and modules used in Smart Polyhouse applications.

The Arduino IDE makes it easy to work with critical sensors used in the Smart Polyhouse project. It supports integration with devices such as the DHT11 sensor for temperature and humidity monitoring, soil moisture sensors to measure water content, and NPK sensors for analysing essential soil nutrients like Nitrogen (N), Phosphorus (P), and Potassium (K). Libraries for these sensors are readily available and can be installed directly through the Library Manager, streamlining the coding process for sensor data acquisition and analysis.A notable feature of the Arduino IDE is its built-in support for serial communication through the Serial Monitor, which allows real-time debugging and monitoring of data from the microcontroller. This is especially useful for verifying sensor readings, checking system responses, and ensuring the correct behaviour of control logic in the irrigation system.

The IDE also supports direct uploading of code to the ESP8266 via USB, eliminating the need for external flashing tools. Additionally, developers can organize their code using modular sketches, and manage different versions or configurations of the irrigation control logic easily.Another powerful feature is the use of Serial Plotter, which provides a graphical view of real-time data, useful for visualizing trends in temperature, humidity, or soil moisture. Combined with a vast community and rich documentation, the Arduino IDE offers a practical, reliable, and efficient development environment for IoT applications such as the Smart Polyhouse Irrigation Management System.


Blink | Arduino IDE 2.0.0-beta.2
Arduino Nano 33 BLE at /dev/...
Blink.ino

```

  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
  Serial.begin(9600);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
  Serial.println("LED is blinking");
}

```

Fig 4.17: Arduino IDE

4.5.2 Arduino Bluetooth controller (By Broxcode)

The Arduino Bluetooth Controller app by Broxcode is a mobile application designed to provide a simple and efficient way to control Arduino-based devices wirelessly using Bluetooth. It is specifically developed to work with Bluetooth modules like the HC-05 and HC-06, which are commonly used in projects involving microcontrollers such as the Arduino UNO. This app allows users to send predefined ASCII characters or custom commands from their Android smartphones to their Arduino devices, enabling real-time control of various robotic or automation systems.

In the floor cleaning robot project, this app plays a critical role in enabling wireless manual control. Once the HC-05 module on the robot is paired with the smartphone, users can connect to it via the app and choose from multiple control interfaces such as Switch Mode, Terminal Mode, D-Pad Mode, or Accelerometer Mode. Among these, the D-Pad Mode is most commonly used for directional control, sending commands like ‘F’ for forward, ‘B’ for backward, ‘L’ for left, ‘R’ for right, and ‘S’ for stop. These commands are received by the Arduino via serial communication and interpreted in the program to control the motors, servos, or other components of the cleaning mechanism.

A major advantage of this app is its customization feature, where each button can be configured to send a specific character or command, making it highly suitable for projects like a smart floor cleaner. Additional buttons can also be set up to activate or deactivate the wet or dry cleaning features by sending commands such as ‘W’ for water spray or ‘D’ for dust cleaning. This approach eliminates the need for physical buttons or remote controls, making the system more user-friendly and flexible for operation.Overall, the Arduino Bluetooth Controller app is an effective and reliable solution for controlling Arduino-based robots like the floor cleaning robot. Its ease of use, multiple control options, and seamless Bluetooth communication make it ideal for DIY projects that require remote control and automation through mobile devices.

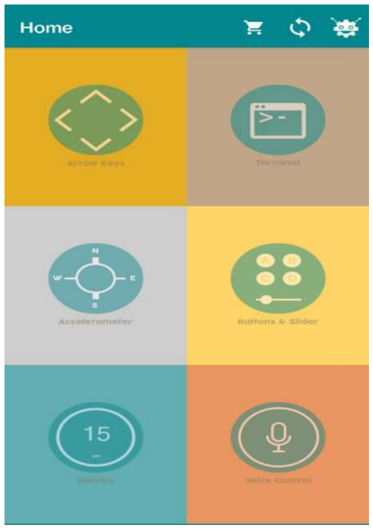


Fig 4.18: App Dashboard

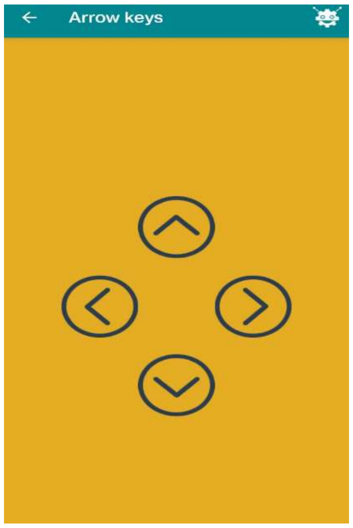


Fig 4.19: Controller

4.5.3 Library Required

4.5.3.1 Servo.h

The Servo.h library is a standard Arduino library used to control servo motors with precision and ease. Servo motors are commonly used in robotics for tasks that require controlled movement, such as arm lifting, steering, or adjusting positions. In the context of the Floor Cleaning Robot project, the servo motor might be used for lifting or lowering a cleaning arm, rotating a nozzle, or adjusting the direction of a cleaning brush. The Servo.h library provides simple functions that allow you to move the servo to a specific angle between 0 and 180 degrees.

One of the key advantages of using the Servo.h library is its simplicity. With just a few lines of code, a servo can be attached to a digital pin, and its position can be controlled by specifying the desired angle. This eliminates the need for complex calculations or manual pulse-width modulation (PWM) code. For example, the function myServo.write(90); will move the servo to the 90-degree position. Additionally, the library allows multiple servo motors to be controlled simultaneously, which is useful in more advanced robots that perform multiple mechanical actions.

Another benefit of the Servo.h library is its compatibility with a wide range of Arduino boards and its efficient use of resources. It uses hardware timers to generate precise control pulses, ensuring smooth and accurate motor positioning. This is especially important in applications like floor cleaning robots, where mechanical arms or sprayers must operate accurately and reliably.Overall, the Servo.h library is a crucial component when servo motors are involved in a project. It simplifies the process of controlling angular motion, making it ideal for both beginners and experienced developers working on robotics or automation projects. In your floor cleaning robot, it allows for dynamic mechanical functions like lifting or rotating parts to enhance the cleaning process.

4.5.3.2 New Ping.h

The NewPing.h library is a powerful and efficient Arduino library used to interface with ultrasonic sensors, such as the HC-SR04, which is commonly used in robotic projects for obstacle detection and distance measurement. In the Floor Cleaning Robot project, the ultrasonic sensor plays a crucial role in helping the robot detect obstacles and avoid collisions while navigating around the cleaning area. While it is possible to use the ultrasonic sensor with basic Arduino functions like digitalWrite() and pulseIn(), the NewPing.h library significantly simplifies the code and improves accuracy and

performance.One of the main advantages of using the NewPing.h library is that it reduces the complexity of the code required to measure distance. It allows the user to initialize the sensor with just one line of code and provides functions to get distance readings in centimeters or inches with minimal delay. For example, sonar.ping_cm(); can be used to get the distance to the nearest object in centimeters. The library also supports features like setting a maximum distance range, which helps to ignore unwanted distant readings and improve detection in close-range environments.

Another benefit of NewPing.h is its ability to handle multiple ultrasonic sensors simultaneously without interference, which is particularly useful in complex robotic systems. The library is designed to minimize CPU usage and avoid blocking functions, which is important in real-time systems where the microcontroller needs to perform multiple tasks, such as motor control, communication, and sensor reading, all at once. This makes it an ideal choice for projects like the floor cleaning robot, where responsive obstacle detection is essential.

In summary, the NewPing.h library enhances the functionality and efficiency of ultrasonic sensors in Arduino projects. It simplifies implementation, increases accuracy, reduces CPU load, and supports multiple sensor setups. By using this library in the floor cleaning robot project, you can ensure that the robot navigates more intelligently and safely, avoiding obstacles smoothly while maintaining effective cleaning operations.

4.5.3.3 SoftwareSerial.h

The SoftwareSerial.h library is an essential Arduino library that allows users to create additional serial communication ports using digital pins on the Arduino board. By default, Arduino boards like the UNO have only one hardware serial port, which is shared between the USB connection (used for programming and debugging) and other serial devices. In projects like the Floor Cleaning Robot, where a Bluetooth module (HC-05) is used for wireless communication, this limitation can be problematic. The SoftwareSerial library solves this issue by enabling serial communication on any digital pins, allowing you to keep the USB port free for debugging while still communicating with external devices like the Bluetooth module.

The primary purpose of SoftwareSerial is to allow the Arduino to talk to multiple serial devices by emulating serial ports in software. This is particularly useful when you're using components that require serial communication, such as GSM modules, GPS receivers, or Bluetooth modules, in addition to the built-in USB serial connection. For example, in your robot, you might use pins 10 and 11 for TX and RX communication with the HC-05 module. The library lets you define these pins and manage data transmission and reception easily.

One of the main benefits of SoftwareSerial is its simplicity and flexibility. With a few lines of code, you can define a new serial port, begin communication at a specified baud rate (like 9600 for HC-05), and use familiar functions like read(), write(), and print(). This enables seamless integration of Bluetooth communication into the robot’s control system, allowing real-time commands from the mobile app to be received and executed by the Arduino.In conclusion, the SoftwareSerial.h library is a vital tool in Arduino projects that require more than one serial connection. It provides a straightforward way to expand communication capabilities beyond the single hardware port, making it ideal for Bluetooth-controlled robots like your floor cleaning robot. By using SoftwareSerial, you ensure smooth and uninterrupted communication with the HC-05 module while still having access to the main USB port for monitoring and debugging.

CHAPTER 5

HARDWARE IMPLEMENTATION

5.1 Introduction

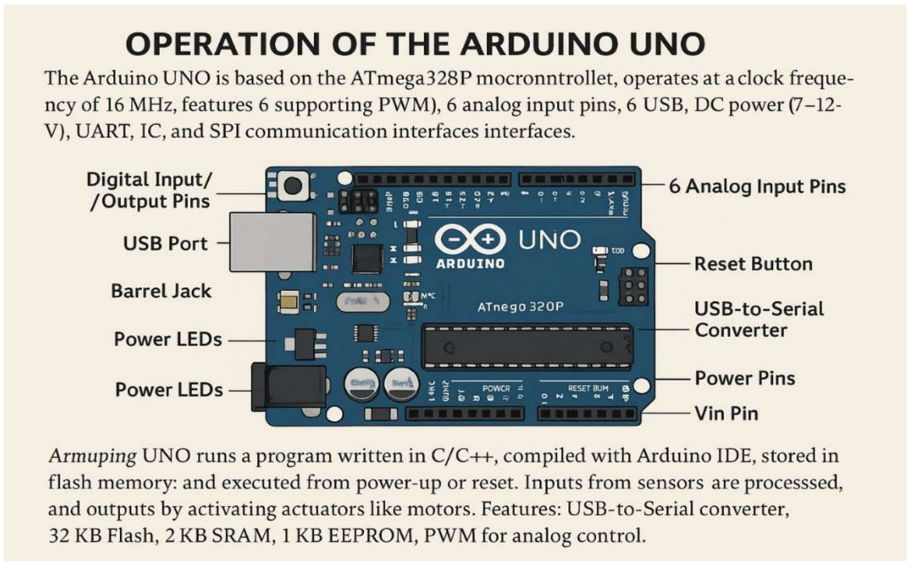
The hardware implementation of the Floor Cleaning Robot forms the backbone of its functionality, combining essential electronic components to enable movement, cleaning, and obstacle avoidance. At the center of the system is the **Arduino UNO** microcontroller, which coordinates inputs from sensors and outputs commands to various actuators. Movement is handled by **two DC motors**, driven through an **L298N motor driver**, allowing the robot to move in all directions.

For cleaning, the robot features **two pumps**—one for wet cleaning (dispensing water or cleaning fluid) and one for dry cleaning (such as vacuuming)—controlled via a **2-channel relay module**. A **Bluetooth HC-05 module** enables wireless control from a smartphone, giving users the ability to direct the robot remotely. To enhance safety, an **ultrasonic sensor** mounted at the front detects obstacles and halts the robot if an object is too close.All components are powered by a **rechargeable battery pack**, ensuring autonomous operation. This simple yet effective hardware setup demonstrates how affordable components can be integrated into a functional cleaning robot suitable for home and small-office environments.

5.2 Operation of Arduino UNO

The Arduino UNO is a microcontroller development board based on the ATmega328P microcontroller, designed to facilitate the development and testing of embedded systems and automation projects. It operates at a clock frequency of 16 MHz and is equipped with 14 digital input/output pins (6 of which support PWM), 6 analog input pins, and multiple communication interfaces including UART, I2C, and SPI. The board is powered either via a USB connection (typically 5V) or an external DC power supply ranging from 7–12V through a barrel jack or the Vin pin.At its core, the Arduino UNO operates by continuously running a program written in C/C++ and compiled using the Arduino IDE. This program is stored in the flash memory of the ATmega328P chip and is executed from the moment the board is powered on or reset. The microcontroller reads inputs from various sensors (such as ultrasonic sensors, IR sensors, or switches), processes the data based on the logic defined in the program, and produces outputs by activating actuators like motors, relays, LEDs, or communication modules.

One of the key features of the Arduino UNO is its built-in USB-to-Serial converter, which allows easy programming and serial communication through the USB port. The microcontroller itself includes separate memory regions: 32 KB of flash memory for code storage, 2 KB of SRAM for variable handling, and 1 KB of EEPROM for permanent data storage. It uses interrupt-driven or polling-based input handling for real-time responsiveness, and supports PWM for analog control of motors and servos.In practical applications, such as the Floor Cleaning Robot, the Arduino UNO coordinates various components by interpreting commands from a Bluetooth module via serial communication, reading distance measurements from an ultrasonic sensor, and controlling motors through a motor driver IC (like the L298N). Its precise timing, GPIO handling, and real-time responsiveness make it suitable for managing both logic and mechanical operations in such automation tasks. Overall, the Arduino UNO serves as a compact yet powerful embedded control unit capable of executing complex tasks with deterministic behavior.


OPERATION OF THE ARDUINO UNO

The Arduino UNO is based on the ATmega328P mocronntrollet, operates at a clock freque-ncy of 16 MHz, features 6 supporting PWM), 6 analog input pins, 6 USB, DC power (7–12-V), UART, IC, and SPI communication interfaces interfaces.

Armuping UNO runs a program written in C/C++, compiled with Arduino IDE, stored in flash memory: and executed from power-up or reset. Inputs from sensors are processsed, and outputs by activating actuators like motors. Features: USB-to-Serial converter, 32 KB Flash, 2 KB SRAM, 1 KB EEPROM, PWM for analog control.

Fig 5.1: Operation of Arduino UNO

5.3 Operation of IR Sensor

In the Smart Floor Cleaning Robot, the IR sensor plays a critical role in ensuring operational safety and intelligent navigation, particularly for edge detection and surface monitoring. The main objective of using an IR sensor in this system is to prevent the robot from falling off elevated surfaces such as stair edges, table tops, or uneven flooring. This enhances the robot's autonomy by enabling it to detect boundaries and avoid accidents during the cleaning process.The IR sensor module in the robot typically consists of an IR transmitter (IR LED) and an IR receiver (photodiode or phototransistor) mounted on the underside of the chassis, close to the floor. During operation, the IR LED emits a continuous beam of infrared light directed downward. When the robot moves over a normal flat surface, the emitted IR light reflects off the floor and returns to the photodiode. The reflected signal is detected, and the sensor outputs a logical HIGH (usually 1), indicating that the surface is present beneath the sensor.

However, when the robot approaches the edge of a surface — such as the top of stairs or the boundary of a table — the infrared beam is not reflected back to the receiver due to the absence of a reflecting surface. In this case, the IR receiver detects little or no reflected IR light, and the sensor output drops to a logical LOW (0). This change is immediately processed by the Arduino controller, which executes an emergency response action, such as stopping the motors, reversing briefly, or altering direction to avoid falling.This real-time feedback mechanism allows the floor cleaning robot to operate safely in both domestic and semi-structured environments without requiring constant human supervision. The IR sensor thus acts as a non-contact, rapid-response safety system that complements other sensors like ultrasonic modules used for obstacle detection. By integrating the IR sensor into the robot’s decision-making algorithm, the system becomes more robust, adaptive, and capable of operating intelligently on varied floor terrains.

5.4 Operation of Ultrasonic Sensor

In a Smart Floor Cleaning Robot, the ultrasonic sensor is a vital component for obstacle detection and avoidance, enabling the robot to navigate intelligently within its environment. This sensor operates on the principle of echolocation, similar to how bats and submarines detect objects. It uses high-frequency sound waves to determine the distance between the robot and nearby obstacles, ensuring that the robot avoids collisions with furniture, walls, or other objects during its cleaning operation.The ultrasonic sensor typically consists of two main elements: a transmitter (trigger) and a receiver (echo). During operation, the transmitter emits an ultrasonic pulse (typically at 40 kHz) when triggered by the microcontroller, such as the Arduino UNO. This pulse travels through the air and reflects off any object in its path. The reflected wave is then detected by the receiver. The time taken for the pulse to travel to the object and back is measured by the microcontroller, which then calculates the distance using the formula:

Distance = (Time × Speed of Sound) / 2,

where the speed of sound in air is approximately 343 meters per second.

In the context of the smart floor cleaning system, this distance measurement allows the robot to constantly monitor its surroundings. If the ultrasonic sensor detects an obstacle within a predefined range (for example, less than 10 cm), the Arduino initiates a programmed response such as stopping the motors, reversing, or changing direction. This ensures that the robot doesn't bump into obstacles or get stuck in tight spaces, maintaining uninterrupted cleaning performance.Additionally, the ultrasonic sensor is typically mounted on the front of the robot to provide forward-facing environmental awareness. In more advanced systems, multiple ultrasonic sensors may be used to cover different directions, enabling 360-degree object detection.The non-contact nature of ultrasonic sensing makes it ideal for operating in dusty or wet cleaning environments where physical sensors like touch bumpers may fail.Overall, the ultrasonic sensor significantly enhances the autonomy, reliability, and safety of the smart floor cleaning robot by allowing it to detect and avoid obstacles effectively, contributing to a more intelligent and efficient cleaning process.

5.5 Operation of Bluetooth Module

In a Smart Floor Cleaning System, the HC-05 Bluetooth module plays a vital role in enabling wireless communication between the user and the robot. This module facilitates remote control and monitoring by allowing a smartphone or computer to send and receive commands over a serial Bluetooth connection. The HC-05 operates using the Serial Port Profile (SPP), which emulates RS-232 serial communication over Bluetooth, making it an ideal choice for integration with microcontrollers like the Arduino UNO.Technically, the HC-05 module functions over UART (Universal Asynchronous Receiver/Transmitter) and is typically connected to the Arduino’s RX and TX pins. It receives movement or control commands (such as ‘F’ for forward, ‘B’ for backward, or ‘S’ for stop) from a mobile application via Bluetooth. The Arduino reads these characters using its serial interface (Serial.read()) and responds accordingly by controlling actuators such as DC motors, relay modules, or servo mechanisms. This forms the basis of manual or semi-autonomous operation in the floor cleaning robot, where the user can intervene in real time to navigate the robot or trigger cleaning actions.

The HC-05 module operates in either Command Mode or Data Mode. In the Smart Floor Cleaning System, it typically runs in Slave Mode and Data Mode, meaning it passively waits to pair with a smartphone and then transmits/receives data. When powered on, the module begins advertising its presence, allowing the mobile device to discover and connect to it. Once paired (usually using a default PIN like 1234 or 0000), the Bluetooth link becomes a transparent bridge, where data sent from the phone is directly forwarded to the Arduino for execution.Another important technical feature is the baud rate configuration, which by default is set to 9600 bps, matching Arduino’s typical serial speed. However, if needed, the module can be placed into AT Command Mode to modify parameters such as the device name, role (Master/Slave), or baud rate. In this project, such configuration is rarely necessary unless multiple Bluetooth modules are involved or custom behavior is desired.Overall, the HC-05 Bluetooth module provides a reliable and low-power solution for wireless control in the Smart Floor Cleaning Robot. It reduces the need for complex user interfaces by enabling smartphone-based command delivery, thus enhancing user convenience while maintaining simplicity in hardware and software integration.

5.6 Operation of Motor Driver

In a Smart Floor Cleaning System, the motor driver—typically an L298N dual H-bridge driver module—acts as the crucial interface between the microcontroller (Arduino UNO) and the DC motors that drive the robot's wheels. Since the microcontroller alone cannot supply the required current and voltage to power the motors directly, the motor driver functions as a power amplifier that receives low-current control signals from the Arduino and delivers high-current output to the motors accordingly. Its operation is based on the H-bridge circuit principle, which allows current to flow in both directions through a motor, enabling bidirectional motion (forward and reverse).The L298N motor driver consists of two H-bridges and can control two DC motors independently, making it suitable for differential drive systems like in the smart floor cleaner. The driver has four input pins—IN1, IN2, IN3, and IN4—which are connected to the Arduino. These pins determine the direction of current flow through each motor. For example, setting IN1 HIGH and IN2 LOW causes Motor A to spin in one direction, while reversing the logic (IN1 LOW, IN2 HIGH) causes it to spin in the opposite direction. The enable pins (ENA and ENB) are used to turn the motors ON or OFF and are often connected to PWM (Pulse Width Modulation) pins on the Arduino to regulate motor speed by varying the duty cycle.

In operation, when the user sends a directional command via Bluetooth (e.g., 'F' for forward), the Arduino processes the command and sets the appropriate HIGH/LOW signals on the L298N inputs to drive both motors forward. For turning, one motor is stopped or reversed while the other continues forward, enabling smooth navigation. The use of PWM allows for speed control, which is important for maneuvering around obstacles or adjusting cleaning intensity.The motor driver is also responsible for handling the power supply to the motors. It typically accepts input voltages from 7V to 12V and features onboard terminal blocks for external power connections. This allows the motors to be powered separately from the Arduino's logic circuitry, ensuring system stability and preventing voltage drops.In summary, the motor driver in a smart floor cleaning robot enables safe, efficient, and controlled operation of the drive motors, providing directional control, speed regulation, and power management. It acts as a bridge between the low-power logic of the Arduino and the high-power demands of the cleaning robot's mechanical system, ensuring reliable and responsive movement throughout the cleaning process.

5.7 Operation of 2-Channel Relay Module

In a Smart Floor Cleaning System, a 2-channel relay module is used to control high-power devices such as water pumps for wet cleaning or vacuum motors for dry cleaning. Since the Arduino UNO operates at low logic levels (typically 5V and a few milliamps), it cannot directly drive these high-current cleaning components. The relay module acts as an electromechanical switch, allowing the Arduino to control the ON/OFF state of external 12V or 24V devices safely and efficiently, without direct electrical connection.Each channel of the relay module consists of a relay coil, a switching mechanism, and a driver circuit (usually based on a transistor and an optocoupler). When the Arduino sends a HIGH or LOW signal to one of the input pins (IN1 or IN2), the relay coil is energized, causing the internal switch (contacts) to change state. This connects or disconnects the external device from its power source, effectively turning it ON or OFF. The module supports two independent channels, enabling the robot to activate wet and dry cleaning functions separately or simultaneously.

In practice, the relay's control inputs are connected to digital output pins of the Arduino. For example, when the robot receives a command like 'W' (wet cleaning), the Arduino sets the corresponding pin HIGH, which activates the first relay, powering the water pump. Similarly, sending a command 'D' may activate the second relay to start the vacuum or dry cleaning mechanism. The normally open (NO) and normally closed (NC) terminals on the relay provide flexibility in how the connected devices behave when the relay is inactive or energized.One key feature of the relay module is its electrical isolation, typically provided by optocouplers. This ensures that voltage spikes or faults on the high-voltage side do not damage the Arduino or affect other low-voltage components. Additionally, onboard indicator LEDs show the status of each relay, providing a visual confirmation of operation during testing and use.Overall, the 2-channel relay module serves as a reliable and safe interface for controlling the high-power components of the smart floor cleaning robot. It enables automated activation of cleaning tools in response to user commands or programmed routines, enhancing the robot's functionality while maintaining the safety and integrity of the control circuitry.

5.8 Mechanism of Servomotor

A servo motor is a type of actuator that allows for precise control of angular or linear position, velocity, and acceleration. It operates based on the principles of closed-loop control systems, where feedback is continuously monitored to maintain the desired output. Internally, a typical servo motor comprises a small DC motor, a gear reduction unit, a position-sensing device (usually a potentiometer), and an integrated control circuit. These components work together to ensure that the motor shaft accurately moves to and holds a position as determined by the control input.The core mechanism of a servo motor begins with the DC motor, which provides the mechanical rotation. The gearbox connected to the motor reduces its rotational speed while simultaneously increasing torque. This gearing system is critical for achieving high positional accuracy and mechanical advantage. Attached to the output shaft of the gearbox is a potentiometer, which acts as a position sensor. As the shaft rotates, the potentiometer's resistance changes, providing an analog voltage that represents the current angular position of the motor.

The motor receives a control signal in the form of a Pulse Width Modulation (PWM) signal, which encodes the desired position. The control circuit inside the servo interprets this PWM signal and compares the desired position (target) with the actual position from the potentiometer (feedback). If a difference is detected, an error signal is generated and the motor is driven in the direction required to minimize this error. As the shaft moves, the potentiometer updates the feedback voltage, and once the target position is reached, the control circuit stops the motor.This feedback loop enables the servo motor to achieve high precision in angular positioning, usually within a range of 0 to 180 degrees, though some servo motors offer extended ranges. The response time, holding torque, and resolution depend on the quality of the motor, gearing, and control electronics. In robotics and embedded systems—like your floor cleaning robot—servo motors are particularly useful for tasks such as actuating arms, adjusting nozzles, or steering mechanisms, where deterministic and accurate control of movement is essential.

5.9 Mechanism of DC Motor

A DC motor (Direct Current motor) is an electromechanical device that converts direct electrical energy into mechanical rotational motion. The working principle of a DC motor is based on Lorentz force, which states that a current-carrying conductor placed in a magnetic field experiences a force. This force acts in a direction perpendicular to both the magnetic field and the current, resulting in rotational motion when the conductor is part of a closed loop mounted on a shaft. The core components of a typical DC motor include the stator, rotor (or armature), commutator, brushes, and windings.The stator of a DC motor is the stationary part that provides a constant magnetic field, either through permanent magnets or electromagnets. The rotor, on the other hand, is the rotating part and includes a coil of wire (the armature winding) mounted on a shaft. When DC voltage is applied to the motor, current flows through the brushes and commutator into the armature winding. According to the right-hand rule, the interaction between the magnetic field from the stator and the current in the rotor produces a torque that causes the rotor to spin.The commutator and brush assembly play a crucial role in maintaining unidirectional torque. As the rotor turns, the commutator reverses the direction of current through the armature windings at the appropriate moment to ensure that the magnetic poles of the rotor remain repelled by and attracted to the corresponding poles of the stator. This continuous reversal allows the motor to keep spinning in the same direction without interruption.

The speed and direction of a DC motor can be controlled by varying the applied voltage and the polarity of the supply, respectively. Lower voltages result in slower speeds, while reversing the polarity of the supply changes the direction of rotation. In robotic applications, such as in the floor cleaning robot project, DC motors are used for driving wheels or actuating moving parts due to their ease of control, high starting torque, and simple interfacing with motor drivers like the L298N.Overall, the DC motor offers a straightforward yet highly effective method for achieving rotational motion in embedded and mechatronic systems. Its well-understood mechanism, rapid response, and controllability make it indispensable in mobile robotic platforms, small automation devices, and consumer electronics.

5.10 Mechanism of Water Pump

In a Smart Floor Cleaning System, the water pump is a critical component used to dispense water or a cleaning solution onto the floor surface for wet cleaning operations. Its mechanism involves electromechanical fluid transfer, where electrical energy is used to drive a small DC pump that moves liquid from a reservoir to the floor through a nozzle or pipe. The operation is coordinated with the robot's movement, typically activating the pump only when the robot is moving forward or when a specific cleaning command is issued.The type of water pump used in such systems is usually a diaphragm or submersible DC pump, operating at 6V to 12V, which makes it compatible with standard battery power supplies and relay-based switching. When power is applied to the pump's terminals, the internal motor drives a rotating mechanism or diaphragm that creates pressure, drawing liquid from the tank and pushing it through the outlet. This mechanism ensures a steady flow rate and sufficient pressure for even distribution of water across the cleaning path.

The pump is controlled via a 2-channel relay module, which acts as a switch between the Arduino microcontroller and the pump. When the Arduino receives a cleaning command (such as via Bluetooth or a predefined routine), it sends a signal to the relay, closing the circuit and supplying power to the pump. This allows precise control of when and for how long water is dispensed, conserving resources and improving cleaning efficiency.For optimal performance, the water pump is strategically positioned near the water reservoir and connected to a nozzle system or drip pipe located just in front of or behind the cleaning brush or mop. This ensures that water is applied directly to the area being scrubbed or wiped, maximizing the cleaning effect while preventing wastage. In more advanced systems, flow can also be controlled using valves or PWM (Pulse Width Modulation), but in most simple smart cleaning robots, on/off relay control is sufficient.In summary, the water pump mechanism in a smart floor cleaning robot provides automated liquid dispensing, enabling effective wet cleaning. Through controlled electrical activation, the pump delivers water in sync with the robot's cleaning operation, enhancing the overall hygiene and functionality of the system. Its integration with relay control and microcontroller logic ensures that the wet cleaning process is efficient, programmable, and responsive to user commands.

CHAPTER 6

SOFTWARE IMPLEMENTATION

6.1 Introduction

The software implementation of the Autonomous Floor Cleaning Robot plays a pivotal role in orchestrating the robot's actions, ensuring efficient cleaning and seamless operation. At the core of the system is a microcontroller, such as the Arduino UNO, which acts as the central control unit. The code uploaded to the microcontroller is written using Arduino IDE in C/C++ and handles real-time decision-making based on sensor inputs and user commands. It governs all aspects of the robot's behavior, including motion control, obstacle detection, water spraying, and cleaning arm movement.

The movement of the robot is managed through motor drivers (such as the L298N), which are controlled by PWM (Pulse Width Modulation) signals generated from the microcontroller. The software determines the direction and speed of the motors by interpreting sensor data and remote commands. Obstacle avoidance is achieved using ultrasonic sensors, which continuously send distance readings to the microcontroller. When an obstacle is detected within a predefined threshold, the robot executes an avoidance maneuver such as stopping, reversing, or changing direction. These responses are handled through conditional programming logic within the Arduino sketch.

Cleaning functions are implemented by integrating actuators like servo motors and water pumps. The software activates the servo motors to lift and lower the cleaning arms based on the cleaning cycle logic, while the water pump is triggered to spray water intermittently to aid in dirt removal. Timing functions (such as millis() and delay()) are utilized to ensure that cleaning operations are synchronized with the robot's movement and direction changes. The modularity of the code allows for easy modifications and improvements, making the system scalable for more complex cleaning tasks or additional features.

Bluetooth-based wireless control is also embedded into the software using the HC-05 module, enabling remote operation from a smartphone. The mobile app sends predefined commands to the microcontroller, which are parsed and interpreted through serial communication. This functionality is particularly useful for manual control, diagnostics, and mode switching (e.g., auto mode or spot cleaning mode). Overall, the software implementation integrates multiple components cohesively, ensuring the robot functions autonomously while maintaining flexibility and control for the user.

6.2 Software Development Environment

The software development for the Smart Floor Cleaning System begins with selecting a suitable Integrated Development Environment (IDE), which acts as the platform for writing, compiling, and uploading code to the robot's microcontroller. Since this project uses the Arduino UNO as the central control unit, the Arduino IDE is used for software implementation. The Arduino IDE is widely preferred due to its simple interface, compatibility with multiple microcontrollers, and access to a vast collection of built-in libraries for controlling sensors, motors, servos, and communication modules. In case the system is later upgraded to a more powerful board like the ESP32, alternative IDEs such as PlatformIO or ESP-IDF can be used for advanced features and real-time operating system support.

The core of the software is written in **C/C++**, the primary programming languages supported by the Arduino ecosystem. These languages are efficient and offer low-level control over hardware, making them ideal for embedded systems like autonomous cleaning robots. The code is structured using the setup() and loop() functions. The setup() function is responsible for initializing various components such as the ultrasonic sensors, motor driver (L298N), servo motors, water pump, and the HC-05 Bluetooth module. The loop() function continuously executes the main logic, including reading sensor data, navigating the floor, avoiding obstacles, and performing cleaning operations in a cyclical manner.

Using the Arduino IDE, the software integrates multiple libraries such as Servo.h for controlling servo motors used in the lifting mechanism of the cleaning arms, AFMotor.h or custom PWM control for driving motors via L298N, and SoftwareSerial.h for managing Bluetooth communication. The ultrasonic sensor readings are processed to detect nearby obstacles, allowing the robot to dynamically alter its path by adjusting motor directions. Similarly, commands received via Bluetooth from a mobile app are interpreted using serial communication and used to switch between manual and automatic cleaning modes.

Moreover, the program includes timing and logic-based controls for activating the water pump, which is used to spray water periodically or on command. This enhances the cleaning efficiency by moistening the surface before the brushes or cloth arms pass over it. The use of **C/C++** provides flexibility in handling these operations through precise timing (using millis() and delay() functions) and real-time responsiveness. Overall, the combination of the Arduino IDE and C/C++ programming creates a cohesive and efficient software structure that enables the Smart Floor Cleaning System to operate autonomously while allowing user interaction and future scalability.

6.3 Key Functions Of The Software

The software serves as the central intelligence of the Smart Floor Cleaning Robot, managing all essential tasks required for autonomous operation. It controls the robot's movement by sending precise signals to the motor driver, which adjusts the direction and speed of the wheels based on environmental conditions. Simultaneously, it processes real-time data from onboard sensors—particularly ultrasonic sensors—to detect obstacles and avoid collisions by triggering actions such as stopping, reversing, or changing direction. The software also oversees the cleaning functions by managing servo motors to lift or lower the cleaning arms and activating the water pump to spray water either periodically or based on specific triggers. All of these operations work in coordination to ensure that the robot performs efficient and intelligent floor cleaning with minimal user intervention.

Below are the primary functions handled by the software:

6.3.1 Setup Function

The first and one of the most critical steps in the software implementation of the Smart Floor Cleaning Robot is the initialization of all hardware components. This is done within the setup() function of the Arduino program, which runs once when the microcontroller is powered on or reset. During this phase, the microcontroller initializes all necessary components, including motors, ultrasonic sensors, and communication modules like Bluetooth. Each of these components must be configured properly to ensure smooth interaction between hardware and software.

A key part of this process is the configuration of input and output (I/O) pins. Each sensor or actuator connected to the Arduino is assigned a specific pin number, and its direction (input or output) is set using the pinMode() function. For example, the motor control pins are set as output so they can receive signals from the Arduino, while the ultrasonic echo pin is set as input to receive distance readings. Similarly, serial communication is initialized for Bluetooth control using the Serial.begin() function. The following sample code demonstrates this setup:

```
void setup() {
  // Initialize motor control pins
  pinMode(motor1, OUTPUT);
  pinMode(motor2, OUTPUT);

  // Initialize ultrasonic sensor pins
  pinMode(ultrasonicTrig, OUTPUT);
  pinMode(ultrasonicEcho, INPUT);

  // Initialize serial communication for Bluetooth module
  Serial.begin(9600);
}
```

6.3.2 Main Loop (Loop Function)

The loop() function is the heart of the Smart Floor Cleaning Robot’s software and is responsible for continuously monitoring sensor inputs and updating the robot's behavior in real time. This function runs in an infinite cycle after the initial setup and acts as the core of the robot's decision-making process. It allows the robot to adapt to changing environmental conditions by processing data from sensors, controlling movement, and managing cleaning mechanisms dynamically.One of the primary tasks performed in the loop is sensor data acquisition. The ultrasonic sensor is frequently triggered to measure the distance between the robot and nearby obstacles. This real-time data helps the robot determine whether the path ahead is clear or blocked. Infrared sensors (if included) may also contribute to more precise object or edge detection. Once the data is acquired, the software evaluates it to make movement decisions. For instance, if no obstacle is detected within the predefined threshold, the robot continues moving forward.

However, if an obstacle is detected—typically within 10 cm—the robot activates its obstacle avoidance routine. This usually involves stopping the motors momentarily, reversing to create space, and then turning to avoid the object. These actions are controlled through conditional logic using if statements. Additionally, the robot checks for a cleaning trigger, such as being in cleaning mode or detecting a dirty surface. If the conditions are met, the cleaning mechanism (such as a brush or water spray) is activated accordingly.

Below is a sample code snippet representing this logic inside the loop() function:

```
void loop() {
  // Read distance from ultrasonic sensor
  int distance = readUltrasonicSensor();

  // Obstacle detection and avoidance logic
  if (distance < 10) { // Obstacle detected within 10 cm
    stop();             // Stop movement
    delay(500);
    reverse();          // Move backward
    delay(1000);
    turnRight();         // Turn to avoid obstacle
    delay(500);
  } else {
    moveForward();       // Continue moving if path is clear
  }
  // Check for cleaning condition
  if (cleaningMode) {
    activateCleaning();  // Start cleaning mechanism
  }
}
```

This loop ensures that the robot operates intelligently by reacting to its surroundings and maintaining a balance between navigation and cleaning. It allows the robot to work autonomously without requiring constant human input, making it both efficient and reliable in performing floor cleaning tasks.

6.3.3 Sensor Data Processing

The Ultrasonic Sensor plays a vital role in the Smart Floor Cleaning Robot's ability to perceive its surroundings and avoid obstacles. It operates using the principle of echolocation, much like a bat. The sensor emits a short ultrasonic pulse from the trigger pin, which travels through the air until it hits an obstacle. The sound wave then reflects back and is detected by the echo pin. By measuring the time delay between sending the pulse and receiving the echo, the software calculates the distance to the object using the known speed of sound in air, which is approximately 0.034 cm/µs.

This real-time distance measurement enables the robot to make intelligent navigation decisions. If the calculated distance is below a certain threshold (e.g., 10 cm), the robot identifies the presence of an obstacle and initiates an avoidance maneuver. Otherwise, it continues on its path. The sensor is frequently polled in the loop() function to maintain constant environmental awareness.
Here is a sample function used to read the distance using the ultrasonic sensor:

```
long readUltrasonicSensor() {
  // Send a short LOW pulse to ensure clean signal
  digitalWrite(ultrasonicTrig, LOW);
  delayMicroseconds(2);

  // Send a 10-microsecond HIGH pulse to trigger the ultrasonic burst
  digitalWrite(ultrasonicTrig, HIGH);
  delayMicroseconds(10);
  digitalWrite(ultrasonicTrig, LOW);

  // Measure the duration of the echo pulse
  long duration = pulseIn(ultrasonicEcho, HIGH);

  // Calculate distance in cm (speed of sound = 0.034 cm/µs)
  long distance = duration * 0.034 / 2;

  return distance;
}
```

In this function, the pulseIn() function is used to measure the time (in microseconds) that the echo pin stays HIGH, which corresponds to the time taken for the ultrasonic wave to travel to the object and back. The distance is then calculated by multiplying the duration by 0.034 and dividing by 2, as the wave travels to the object and back (round trip). This distance data is then used in the main control loop to decide whether the robot should continue moving forward or take evasive action. The ultrasonic sensor's simplicity, accuracy, and reliability make it an ideal choice for basic obstacle detection in autonomous robotic systems.

6.3.4 Movement Control

The movement of the Smart Floor Cleaning Robot is governed by the motor driver module, typically the L298N, which acts as an interface between the microcontroller (like Arduino UNO) and the DC motors. The motor driver allows the robot to execute various directional movements such as moving forward, reversing, and turning left or right. These movements are achieved by sending digital HIGH or LOW signals from the Arduino to specific input pins on the motor driver, which in turn controls the power supply to the motors.

To simplify motor control and improve code readability, dedicated functions are created in the program—such as moveForward(), reverse(), turnRight(), turnLeft(), and stop(). Each of these functions sets the appropriate logic levels on the motor driver inputs to trigger the desired motor behavior. For example, when one motor is set to rotate forward and the other in reverse, the robot can rotate in place (turning), whereas setting both motors to move forward propels the robot straight ahead.

Below is an example of how these functions are typically implemented:

```
// Function to move the robot forward
void moveForward() {
  digitalWrite(motor1, HIGH);  // Left motor ON
  digitalWrite(motor2, LOW);   // Right motor OFF (depends on wiring)
}

// Function to reverse the robot
void reverse() {
  digitalWrite(motor1, LOW);   // Left motor OFF
  digitalWrite(motor2, HIGH);  // Right motor ON
}

// Function to stop all motor movement
void stop() {
 digitalWrite(motor1, LOW);
  digitalWrite(motor2, LOW);
}

// Function to turn the robot right
void turnRight() {
  digitalWrite(motor1, HIGH);   // Both motors ON may result in a spin
  digitalWrite(motor2, HIGH);   // Adjust wiring or add PWM for smoother turns
}
```

These movement functions are called within the main loop depending on the robot's surroundings. For example, when the ultrasonic sensor detects an obstacle, the stop() function is called to halt movement, followed by reverse() and turnRight() to navigate around the obstacle. In the absence of obstacles, moveForward() is called to continue cleaning operations. This modular approach to motor control makes the robot's behavior both intuitive and easy to debug, while also making it simple to expand with additional motion types or sensors in the future.

6.3.5 Obstacle Avoidance Logic

One of the most critical features of the Smart Floor Cleaning Robot is its ability to autonomously detect and avoid obstacles while performing its cleaning operation. This function ensures that the robot doesn't collide with walls, furniture, or other objects in its path, maintaining both the robot's safety and operational efficiency. The robot relies on real-time data from the ultrasonic sensor, which continuously measures the distance between the robot and nearby obstacles by emitting sound waves and measuring their echo time.

When the sensor detects that an obstacle is within a predefined safe distance—for example, 15 cm—the robot executes a sequence of actions to prevent a collision. This typically involves stopping the robot, reversing for a brief period to create space, and then turning to a new direction before resuming forward motion. This behavior mimics basic path-planning logic, allowing the robot to navigate around obstacles without human intervention.

Below is a sample snippet that represents this obstacle avoidance logic:

```
if (distance < 15) { // If obstacle is within 15 cm
  stop();           // Stop the robot
  delay(500);       // Pause for half a second

  reverse();        // Move backward to create space
  delay(1000);      // Continue reversing for 1 second

  turnLeft();       // Turn the robot to the left
  delay(500);       // Pause to complete the turn
}
```

This simple yet effective logic enables the robot to adjust its path in real time based on sensor feedback. The use of delays ensures that each action is completed before the next begins, which is especially important for physical hardware with moving components. The threshold distance (15 cm in this case) can be fine-tuned based on the robot's speed and reaction time to optimize performance in different environments. This modular and responsive approach to obstacle handling greatly enhances the robot’s autonomous navigation capability, making it reliable in dynamic and cluttered spaces.

6.3.6 Cleaning Mechanism

In the Smart Floor Cleaning Robot, the cleaning mechanism is a key functional component that distinguishes it from simple mobile robots. When the robot enters cleaning mode, the software takes charge of activating the appropriate cleaning hardware, which could include a vacuum motor, rotating brushes, or a water sprayer depending on the robot’s design. This cleaning mechanism is usually powered by a small DC motor, and its operation is controlled directly through the microcontroller by sending HIGH or LOW signals to a designated output pin connected to the cleaning motor.

To manage this functionality efficiently, the software includes dedicated functions such as activateCleaning() and deactivateCleaning(). These modular functions ensure that the cleaning motor is only powered on when necessary—typically when the robot is moving over a surface that needs cleaning or when a manual or automatic cleaning mode is triggered. This not only improves energy efficiency but also extends the lifespan of the motor and other mechanical parts.

Below is the sample code for controlling the cleaning mechanism:

```
// Function to activate the cleaning mechanism
void activateCleaning() {
  // Activate cleaning motor (e.g., vacuum or brush)
  digitalWrite(cleaningMotor, HIGH);
}

// Function to deactivate the cleaning mechanism
void deactivateCleaning() {
  // Turn off cleaning motor
  digitalWrite(cleaningMotor, LOW);
}
```

These functions are generally called within the main loop() function based on conditions such as whether cleaning mode is enabled or a dirty surface is detected. For instance, when cleaningMode is true, the activateCleaning() function is triggered to start the cleaning motor. If the robot is paused or no longer in cleaning mode, deactivateCleaning() is called to stop the cleaning action. This approach ensures the cleaning process is fully automated, responsive to the environment, and easy to expand or modify in future upgrades (e.g., adjusting motor speed via PWM or integrating with a dirt detection sensor).

6.3.7 Communication

The communication module in the Smart Floor Cleaning Robot enables remote control and monitoring, significantly enhancing its usability. This functionality is particularly useful during manual overrides, testing, or when the user wants to guide the robot into hard-to-reach areas. The robot can be controlled through either Bluetooth (using modules like HC-05) or Wi-Fi (using ESP8266/ESP32), depending on the hardware used. These modules connect to the microcontroller via serial communication, allowing the robot to receive commands from a smartphone or computer and respond accordingly.

When using Bluetooth, for instance, a smartphone app or a serial terminal can send characters that represent specific commands. The microcontroller continuously checks if data is available through the serial port. Once it receives a character, it interprets it and executes the corresponding movement or action. This simple interface is very effective for testing and controlling the robot in real-time without requiring physical buttons or pre-set logic.

Here's an example of a basic Bluetooth command handling routine in code:

```
void loop() {
 if (Serial.available()) {
  char command = Serial.read();  // Read the incoming command
  if (command == 'F') {
   moveForward();              // Move forward on 'F'
  } else if (command == 'B') {
   reverse();                  // Move backward on 'B'
  } else if (command == 'L') {
   turnLeft();                 // Turn left on 'L'
  } else if (command == 'R') {
   turnRight();                // Turn right on 'R'
  } else if (command == 'S') {
   stop();                     // Stop the robot on 'S'
  } else if (command == 'C') {
   activateCleaning();         // Start cleaning on 'C'
  } else if (command == 'X') {
   deactivateCleaning();       // Stop cleaning on 'X'
  }
 }
}
```

This flexible communication structure can be easily extended to include status updates (like battery level, current operation mode, or obstacle alerts) by having the robot send data back to the controller using Serial.print() or over a Wi-Fi network through a web interface or mobile app. By integrating Bluetooth or Wi-Fi, the robot transitions from being purely autonomous to being interactively controllable, offering users a seamless hybrid experience of automation and manual precision.

6.3.8 Energy Management

For battery-powered autonomous robots like the Smart Floor Cleaning Robot, energy management is a critical aspect of reliable and uninterrupted operation. Without effective power monitoring, the robot could stop mid-task due to battery depletion, potentially causing inconvenience or failure to complete cleaning routines. To prevent such issues, the software includes a mechanism to continuously monitor the battery voltage using an analog pin connected to a voltage divider circuit from the battery.

This energy management system allows the robot to make informed decisions based on its current power status. For instance, when the battery level drops below a predefined threshold, the robot can reduce activity, enter low-power mode, or even navigate back to a charging station if such a setup is integrated. This ensures that the robot avoids complete power loss and can recharge in time for future tasks, promoting autonomy and self-maintenance.

Below is a simple implementation of a battery check function:

```
void checkBatteryLevel() {
  int batteryLevel = analogRead(batteryPin);  // Read analog voltage level

  if (batteryLevel < 100) { // If battery level is below the threshold
    goToChargingStation();  // Initiate return to charging base
  }
}
```

This function can be called periodically in the main loop() to keep track of battery health. The value 100 represents the analog threshold, which can be calibrated based on your specific battery and voltage divider setup. When this threshold is crossed, a function like goToChargingStation() can be triggered to begin autonomous docking. If an auto-docking feature is not yet implemented, the robot could simply halt operations and send a Bluetooth/Wi-Fi alert to the user, prompting manual charging.By integrating power monitoring into the software, the robot becomes more self-aware and resilient, capable of sustaining long-term autonomous behavior without risking sudden shutdowns or inefficient battery usage. This is an important step toward developing a truly intelligent and reliable floor cleaning system.

6.4 Interfacing of ultrasonic sensors with Arduino UNO

In the Smart Floor Cleaning System, the ultrasonic sensor helps the robot move safely by finding how far objects are. This sensor sends out sound waves that bounce off nearby things and come back. The robot measures how long it takes for the sound to return and uses that time to figure out the distance. This helps the robot know when something is in front of it so it can avoid hitting it.

To connect the HC-SR04 ultrasonic sensor to the Arduino UNO, we use four pins: VCC (5V), GND (ground), Trig (to send the signal), and Echo (to receive the signal). Usually, the Trig pin is connected to pin 9 and the Echo pin to pin 10 on the Arduino. When the Trig pin sends a short sound pulse, the Echo pin waits for it to bounce back. The Arduino then calculates the distance using this formula:
distance = (duration × 0.0343) / 2,
which uses the speed of sound in air.

The robot uses this sensor to avoid bumping into objects. If it sees something closer than a set distance (like 10 or 15 cm), the robot will stop, back up, and turn to find a new direction. This helps the robot move around safely and continue cleaning without getting stuck. The robot keeps checking for obstacles as it moves, so it can quickly respond and change direction when needed.

Here is a simple code example to read distance using the sensor:

```
long readUltrasonicSensor() {
  digitalWrite(ultrasonicTrig, LOW);
  delayMicroseconds(2);
  digitalWrite(ultrasonicTrig, HIGH);
  delayMicroseconds(10);
  digitalWrite(ultrasonicTrig, LOW);
  duration = pulseIn(ultrasonicEcho, HIGH);
  distance = duration * 0.0343 / 2;
  return distance;
}
```

Even though this sensor works well, it has some limits. It may not detect soft things or objects with slanted surfaces. Also, changes in temperature or noise around the robot can affect the results. But still, it is a good choice for small indoor robots because it is easy to use, low cost, and does the job well. In future versions, we can add more sensors or mix this one with other types like IR sensors to improve how well the robot sees its surroundings.

6.5 Interfacing of Bluetooth Module with Arduino UNO

In the Smart Floor Cleaning Robot project, wireless communication is important because it lets users control the robot without wires. We use a Bluetooth module called HC-05, which allows the robot to receive commands from a smartphone or other Bluetooth device. This means users don’t have to push buttons on the robot directly. Instead, they can move the robot using their phone while staying at a distance. This makes the robot easier and more fun to use indoors.The HC-05 Bluetooth module is simple to connect. It has four main pins: VCC (connected to 5V), GND (connected to ground), TXD (transmit), and RXD (receive). TXD goes to the Arduino's RX pin, and RXD goes to Arduino's TX pin.

Since the module works at 3.3V logic and the Arduino works at 5V, we usually use a voltage divider on the RXD line to protect the module. Once the connections are made, the HC-05 is ready to talk to the Arduino and receive control commands.In the Arduino code, we start Bluetooth communication using Serial.begin(9600); this sets the speed of communication. A user can now open a Bluetooth terminal app on their phone, pair with the robot, and send single letters to control movement. For example, sending 'F' makes the robot move forward, 'B' for backward, 'L' for left, and 'R' for right. The robot reads this command and sends the right signal to the motors using the L298N motor driver.

Here's a simple code example for Bluetooth control:

```
void loop() {
  if (Serial.available()) {
    char command = Serial.read();
    if (command == 'F') {
      moveForward();
    } else if (command == 'B') {
      reverse();
    } else if (command == 'L') {
      turnLeft();
    } else if (command == 'R') {
      turnRight();
    } else if (command == 'S') {
      stop();
    }
  }
}
```

Using Bluetooth like this makes the robot easy to control in small or tight spaces. The user just needs to open a mobile app and type the commands. The Bluetooth module usually has a default password like "1234" or "0000" for pairing. Once paired, the robot listens and acts on each command sent by the phone. This is very helpful when the robot needs to move around furniture or objects where it's hard to reach by hand.

Though the HC-05 has a short range, around 10 meters, it is still enough for most home or office use. It cannot connect to the internet or send large amounts of data, but for simple control, it works very well. In the future, we could improve the robot by letting it send back information to the phone, such as battery level or obstacle alerts. This way, the robot would not only follow orders but also keep the user informed. Overall, Bluetooth makes the Smart Floor Cleaning Robot more flexible, easier to use, and more enjoyable to operate.

6.6 Interfacing of IR sensor with the Arduino UNO

In the Smart Floor Cleaning Robot, IR (Infrared) sensors are used to detect objects or edges on the floor. They help the robot avoid falling off edges like stairs or detect walls and obstacles. An IR sensor works by sending infrared light and checking if it reflects back. If the light reflects back, it means something is in front of the sensor. If there is no reflection, it means there is no object or the robot is near an edge.

The IR sensor usually has three pins: VCC, GND, and OUT. The VCC pin is connected to the 5V pin on the Arduino, the GND pin goes to ground, and the OUT pin is connected to a digital pin on the Arduino, like pin 7. When an object is detected, the sensor sends a LOW signal (0) to the Arduino. If nothing is detected, the sensor sends a HIGH signal (1). The robot uses this signal to decide if it should stop or change direction.For example, if the IR sensor detects an obstacle or edge, the robot should stop or turn to avoid it. This makes the cleaning process safe and smart. The IR sensor is small, cheap, and easy to use, so it's a good choice for home robots. You can use one IR sensor in the front or place two sensors on the left and right for better detection. This way, the robot can better understand its surroundings.

Here is a simple code to show how the IR sensor can be used with Arduino:

```
int irSensorPin = 7; // IR sensor connected to pin 7
int sensorValue;
void setup() {
  pinMode(irSensorPin, INPUT);
  Serial.begin(9600);
}
void loop() {
  sensorValue = digitalRead(irSensorPin);
  if (sensorValue == LOW) {
    // Object detected or edge found
    stop();         // Stop the robot
    reverse();      // Go back
    delay(1000);
    turnRight();    // Turn to avoid
    delay(500);
  } else {
    moveForward();  // Keep moving if no obstacle
  }
}
```

This code checks the sensor all the time. If the sensor sends LOW, it means an object or edge is near. The robot will stop, move back, and then turn. If there is nothing, the robot keeps moving forward. You can adjust the delay() time and direction as needed for your robot's design.Using IR sensors with Arduino helps the Smart Floor Cleaning Robot move safely and smartly. It can clean the floor without bumping into furniture or falling from edges. These sensors are low-cost and easy to use, which makes them perfect for student and DIY robotics projects. In the future, more IR sensors can be added to make the robot even smarter by covering more directions.

CHAPTER 7

EXPERIMENTAL RESULTS

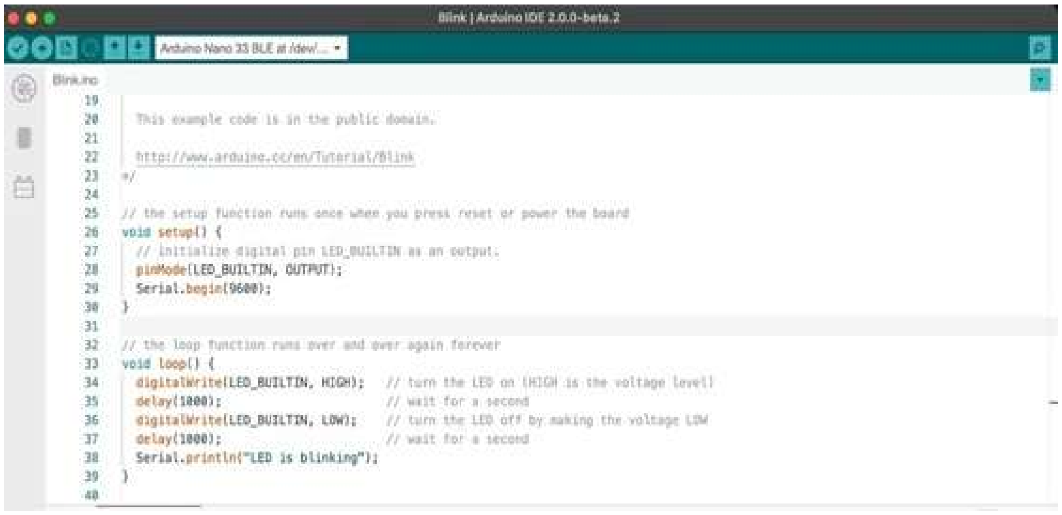
7.1 Introduction

After the construction and programming of the Smart Floor Cleaning Robot, a comprehensive series of tests were conducted to evaluate its performance. These tests focused on key functionalities such as floor cleaning, navigation, obstacle detection and avoidance, and water spraying. The robot was placed in various environments including smooth floors, slightly uneven surfaces, and areas with common household obstacles like furniture and walls. It was observed that the robot could effectively navigate around the obstacles using the ultrasonic sensors, maintain a steady path using motor control, and operate the water spraying mechanism when needed. The cleaning mechanism worked efficiently, and the servo-controlled lifting arms performed well, especially when transitioning between different sections of the floor.

The testing phase also examined the robot's response time, coverage area, battery consumption, and Bluetooth connectivity with the mobile device. The system demonstrated consistent and reliable behaviour across multiple test runs, indicating a good level of robustness. While a few minor improvements could enhance precision in tight spaces, the overall results confirmed that the robot meets the design objectives. It was able to perform autonomous floor cleaning under real-world conditions with satisfactory efficiency and adaptability, making it a promising prototype for smart home cleaning applications.

Step 1: Code Upload and Configuration

The first step was uploading the Arduino code to the Arduino UNO board. We used the Arduino IDE for this process. After uploading the code, we connected all the components like the motor driver, Bluetooth module, ultrasonic sensor, and water pump. Once the setup was complete, we turned on the system to check if the code was working. All parts responded correctly, which showed that the configuration was successful.


Blink | Arduino IDE 2.0.0-beta.2
Arduino Nano 33 BLE at /dev/...
Blink.ino

```
  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
  Serial.begin(9600);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
  Serial.println("LED is blinking");
}
```

Fig 7.1: Code Upload and Configuration

Step 2: Movement

We tested the movement of the Smart Floor Cleaner to check if it could go forward, backward, left, and right using the mobile app. The cleaner moved properly in all directions without any problems. It worked well on smooth floors like tiles and marble.

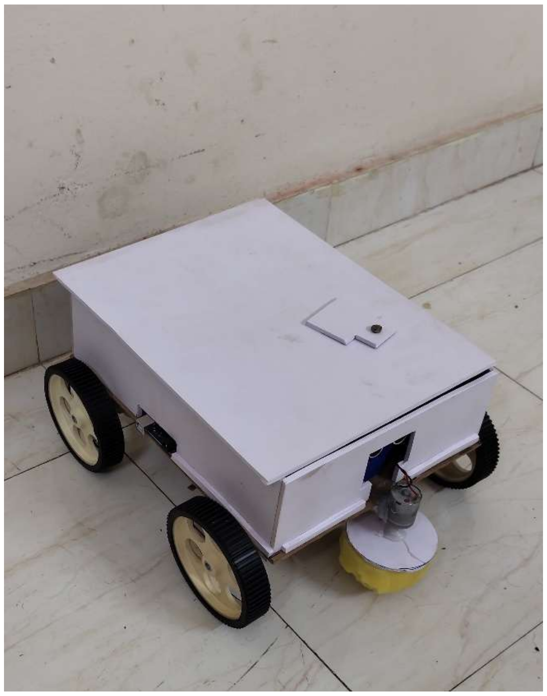


Fig 7.2: Movement

Step 3: Bluetooth Communication

We tested the Bluetooth connection using the HC-05 module to control the robot wirelessly. The mobile phone connected easily to the cleaner. Commands were sent without delay, and the cleaner followed them correctly within a range of about 10 meters.

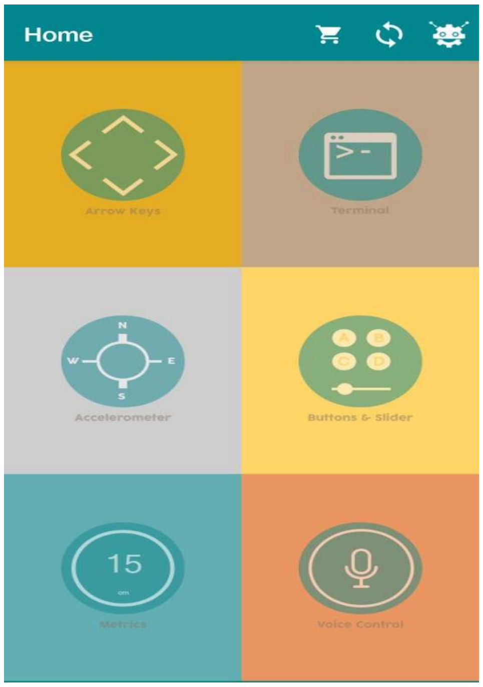


Fig 7.3: App Dashboard

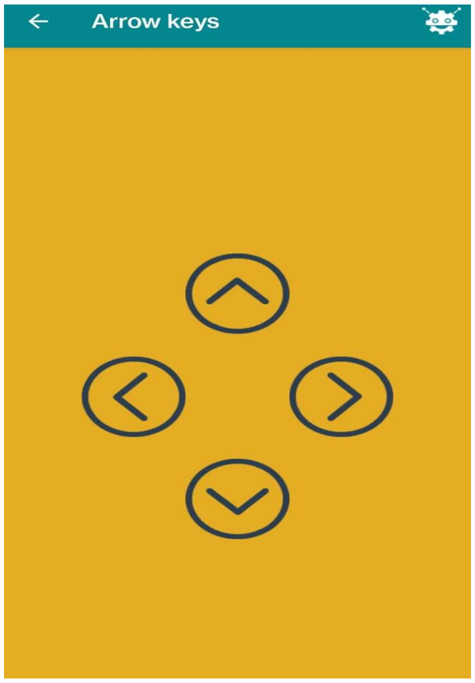


Fig 7.4: Controller

Step 4: Water Spraying Test

The water spray feature was tested by turning on the pump from the mobile app. The pump started when the command was given and sprayed water evenly. It was enough to wet the floor before cleaning

Step 5: Obstacle Detection Test

We checked if the ultrasonic sensor could detect obstacles in front of the robot. When something came close, the sensor noticed it and the cleaner stopped or changed direction. This helped the cleaner avoid hitting things while moving..

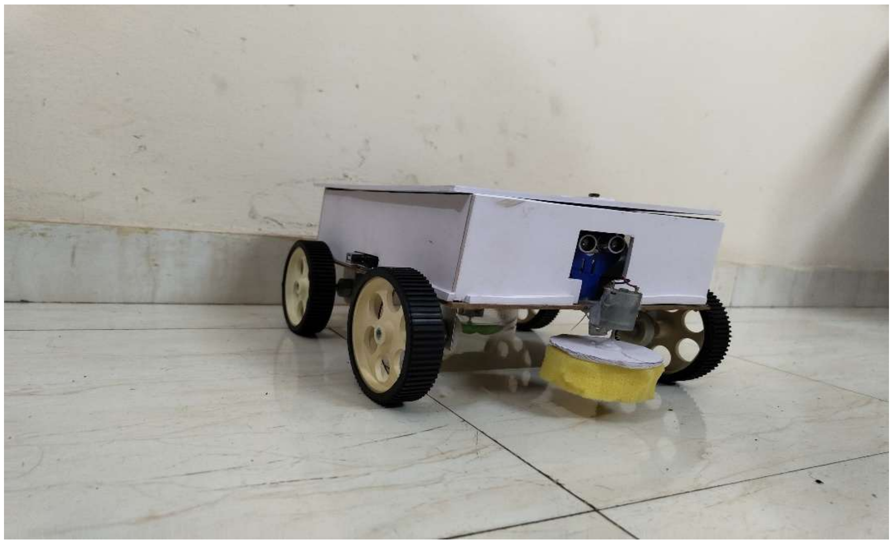


Fig 7.5: Obstacle Detection Test

Step 6: Cleaning Effectiveness

To test how well the robot cleaned the floor, we ran it on surfaces with light dust and small dirt. The cleaner picked up the dust properly and cleaned the surface well in both dry and wet cleaning modes.

Step 7: Power Performance Evaluation

We tested how long the battery lasted during cleaning. On a full charge, the cleaner worked for around 30 to 40 minutes. This was enough to clean a normal-sized room without stopping.

CHAPTER 8

CONCLUSION

8.1 Conclusion

The Smart Floor Cleaning System was successfully designed and tested. It can move in different directions, spray water, avoid obstacles, and clean the floor using both wet and dry modes. The system is controlled using a mobile phone through Bluetooth, which makes it easy to use. All the components like Arduino UNO, motor driver, ultrasonic sensor, Bluetooth module, and water pump worked together as expected. This project helped us understand how to build and control a robotic cleaning system using basic electronic parts. It also improved our skills in wiring, coding, and testing. We learned how to solve technical problems through trial and error and how to make sure each part works correctly with the others. This experience taught us teamwork, time management, and how to handle real-world challenges in electronics and automation. Overall, the project achieved its main goal of reducing human effort in floor cleaning and gave us a strong foundation for building more advanced robotic systems in the future.

8.2 Future Scope

In the future, this system can be made more advanced by adding new features. One useful upgrade is to add voice control using virtual assistants like Google Assistant or Alexa, so the user can give voice commands to start or stop cleaning. Another feature can be auto-charging, where the robot goes back to a charging dock when the battery is low. We can also add water level and dust bin sensors to alert the user when it needs to be refilled or emptied. Using solar panels for partial charging can help save energy. A mobile app with cleaning history and room mapping features can also be created to monitor performance. The robot can be improved to work better on carpets or rough surfaces by using stronger motors or better wheels. It can also be made smaller and more compact so it can clean under furniture or in narrow spaces. In the long run, this system can be upgraded for commercial use in malls, hospitals, and offices where regular and smart cleaning is important. These improvements will make the system more efficient, eco-friendly, and fully automatic.

REFERENCES

[1] Prof. Mahadik S. C., Tupe Raviraj, Khengare Sharad, Patil Akshay, Doshi Nikhil, “Remote Operated Floor Cleaner: An Ease In Floor Cleaning,” *International Journal of Research and Analytical Reviews (IJRAR)*, Volume 6, Issue 2, May 2019, E-ISSN 2348-1269, P-ISSN 2349-5138.

[2] Manya Jain, Pankaj Singh Rawat, Assist. Prof. Jyoti Morbale, “Automatic Floor Cleaner,” *International Research Journal of Engineering and Technology (IRJET)*, Volume 4, Issue 4, April 2017, e-ISSN: 2395-0056, p-ISSN: 2395-0072.

[3] Yuda Irawan, Muhardi, Rian Ordila, Roni Diandra, “Automatic Floor Cleaning Robot Using Arduino and Ultrasonic Sensor,” *Journal of Robotics and Control (JRC)*, Volume 2, Issue 4, July 2021, ISSN: 2715-5072, DOI: 10.18196/jrc.2485.

[4] Hakan Simsek, Faize Nur Erturk, Recep Seker, “A Fuzzy Logic Approach and Path Algorithm for Time and Energy Management of Smart Cleaning Robots,” *Gazi University Journal of Science*, Volume 36, Issue 3, 2023, Pages 1034-1048, DOI: 10.35378/gujs.1037741.

[5] Nabamita Ramkrishna Das, Rashmi Daga, Sneha Avte, Prof. Kavita Mhatre, “Robotic Automated Floor Cleaner,” *International Research Journal of Engineering and Technology (IRJET)*, Volume 6, Issue 3, March 2019, e-ISSN: 2395-0056, p-ISSN: 2395-0072.

[6] Nitesh Bhoyarkar, Ashwinkumar Watkar, Sachin Thak, Pratik Bobade, Vishal Tanale, Guide: Pooja Raut, Shweta Totade, “Automatic Floor Cleaning Machine,” *International Research Journal of Engineering and Technology (IRJET)*, Volume unspecified.

[7] Devabalan Sinnapatchai, Ellysa Mu'izz Zaini, Syahira Salleh, Mohamad Md Som, "Cleaning Robot with Android Application Controller," *Multidisciplinary Applied Research and Innovation (MARI)*, Volume 3, Issue 2, 2022, Pages 354-361, DOI: https://doi.org/10.30880/mari.2022.03.02.041.

[8] Rahul Gupta, Deepak Sajnekar, Mohan Dhadaga, Suraj Gupta, "Review Paper on Electrical Drive Based Floor Cleaning Robot," *International Research Journal of Engineering and Technology (IRJET)*, Volume 9, Issue 12, December 2022, e-ISSN: 2395-0056, p-ISSN: 2395-0072.

[9] Vijayalaxmi S. Kumbhar, Dnyaneshwari Jagtap, Mansi Kulkarni, Salim Lakade, "Autonomous Floor Cleaning Robot," *International Research Journal of Engineering and Technology (IRJET)*, Volume 8, Issue 5, May 2021, e-ISSN: 2395-0056, p-ISSN: 2395-0072.

[10] Veerajagadheswar Prabakaran, Mohan Rajesh Elara, Thejus Pathmakumar, Shunsuke Nansai, "Floor Cleaning Robot with Reconfigurable Mechanism," *Automation in Construction*, Elsevier, 2021, journal homepage: www.elsevier.com/locate/autcon.

[11] Benjamin Andreas Ulsmåg, "Private Information Exposed by the Use of Robot Vacuum Cleaner in Smart Environments," *Master's Thesis*, Norwegian University of Science and Technology (NTNU), June 2023.

[12] Roshan Kerkar, Sadguru Rane, Sanchit Rane, Jaysing Sawant, Eliyan Fernandes, Shweta Jadhav, "Automatic Floor Cleaning Robot", International Research Journal of Engineering and Technology (IRJET), Vol. 11, Issue 3, March 2024, e-ISSN: 2395-0056, p-ISSN: 2395-0072.

[13] S. Yatmono, et al., "Development of Intelligent Floor Cleaning Robot", Journal of Physics: Conference Series, Vol. 1413, 01201, 2019.

[14] Dr. J. Ashok, M. Pavithra, T. Periyanayaki, C. Midhuna, "Revolutionizing Floor Cleaning: Design and Development of a Smart Solar-Powered Floor Cleaning Machine", JETIR, Vol. 10, Issue 4, April 2023, ISSN: 2349-5162.

[15] Rhutuja Patil, Mohini Kulkarni, Sejal Mhadgut, Prashant Titare, D.G. Khairnar, "Smart Floor Cleaner using Mobile Application", International Journal of Scientific Research in Engineering and Management (IJSREM), Vol. 7, Issue 3, March 2023, ISSN: 2582-3930.

[16] Nabamita Ramkrishna Das, Rashmi Daga, Sneha Avte, Prof. Kavita Mhatre, "Robotic Automated Floor Cleaner", IRJET, Vol. 6, Issue 3, March 2019, e-ISSN: 2395-0056, p-ISSN: 2395-0072.

[17] Ankit Tejbahadur Yadav, Sushant Anand Sarvade, Rahul Ramesh, Suryamani Yadav, "Wireless Automatic Floor Cleaning and Safety Indicator Robot", IRJET, Vol. 5, Issue 6, June 2018, e-ISSN: 2395-0056, p-ISSN: 2395-0072.

[18] Alexander Smirnov, Alexey Kashevnik, Andrew Ponomarev, "Multi-Level Self-Organization in Cyber-Physical-Social Systems: Smart Home Cleaning Scenario", Procedia CIRP, Vol. 30, 2015, pp. 329–334.

[19] Abdul Hafiz Kassim, Mohamad Yusof Mat Zain, Mohd Abdul Talib Mat Yusoh, et al., "Internet of Things-Based Floor Cleaning Robot", Indonesian Journal of Electrical Engineering and Computer Science, Vol. 32, No. 3, December 2023, pp. 1353–1360, ISSN: 2502-4752.

[20] Prof. Dattatray M. Kumbhar, Miss Vidya Shetti, "Smart Floor Cleaning Robot", JETIR, Vol. 7, Issue 10, October 2020, ISSN: 2349-5162.

[21] Prof. Vaishnavi Dhole, Palash Lakhe, Vinod Lanjewar, Mayur Bowade, Mayuri Jaypurkar, "Smart Multifunction Floor Cleaning Robot", IJRASET, Vol. 10, Issue 3, March 2022, ISSN: 2321-9653.

[22] Prof. Heena B. Kachhela, Ms. Mamta S. Khadse, Prof. Sneha R. Bhange, et al., "Design and Implementation of Floor Cleaning Robot Using IoT", Vidyabharati

[23] International Interdisciplinary Research Journal, Vol. 18(1), March–May 2024, ISSN: 2319-4979.

[24] Manasa M., Vidyashree T. S., Bindushree V., Sanjana Rao, Gowra P. S., "Smart Vacuum Cleaner", Global Transitions Proceedings, BMS College of Engineering, VTU, Bangalore.

[25] Larissa Nicholls, Yolande Strengers, "Robotic Vacuum Cleaners Save Energy? Raising Cleanliness Conventions and Energy Demand in Australian Households with Smart Home Technologies", Centre for Urban Research, RMIT University, Melbourne, Australia.

[26] Manisha Kukde, Sanchita Nagpurkar, Akshay Dhakulkar, Akshay Amdare, "Automatic & Manual Vacuum Cleaning Robot", IRJET, Vol. 5, Issue 2, February 2018, e-ISSN: 2395-0056, p-ISSN: 2395-0072.

[27] Akshata Gholap, Abhinay Rasal, Aishwarya Pawar, Chaitali Dighe, Prof. S.R. Gagare, "Multipurpose Cleaning Robot using IoT and Image Processing", IRJET, Vol. 7, Issue 6, June 2020.

[28] K. Sarath Kumar, M. Pavan, N. Kishore Karthikeyan, P. Sai Venkata Lokesh, K. Sasidhar, T. Haritha, "Arduino Based Smart Vacuum Cleaner Robot", IJRASET, Vol. 11, Issue 3, March 2023, ISSN: 2321-9653.

[29] Indronil Dey Niloy, Sayed Mohaiminul Hoque, Mashfiqul Hoque, Afif Bin Arfan, Israt Jahan, Islam Bin Mursalin, Mohammad Shidujaman, Mohammad Rejwan Uddin, Mahady Hasan, “Smart Floor Cleaning Robot”, Fab Lab IUB, Independent University, Bangladesh.

[30] Sharada L. Kore, Sonal M. Patil, Roshana J. Sapkal, Savita A. Itkarkar, Roma R. Jain, “Floor Cleaning Smart Robot”, International Journal of Engineering Research and Applications, ISSN: 2248-9622, Vol. 12, Issue 9, September 2022, pp. 61-65.

[31] Prof. Dattatray M. Kumbhar, Miss. Vidya Shetti, “Smart Floor Cleaning Robot”, Journal of Emerging Technologies and Innovative Research (JETIR), ISSN: 2349-5162, Vol. 7, Issue 10, October 2020.

[32] Pooja D. Rathod, Puja V. Wandile, Kiran S. Mohitkar, Pallavi G. Jiwtode, "Multipurpose Smart Floor Cleaning System by Using Android Device", IJSRSET, Vol. 4, Issue 7, 2018.

[33] Pooja D. Rathod, Puja V. Wandile, Kiran S. Mohitkar, Pallavi G. Jiwtode, "Multipurpose Smart Floor Cleaning System by Using Android Device", IJSRSET, Vol. 4, Issue 7, 2018.

[34] Mrs. Shritika Wayker, Prashant Tiwari, Vishal Kumar, Kunal Limbu, Amay Tawade, "Smart Floor Cleaning Robot Using Android", International Research Journal of Engineering and Technology (IRJET), e-ISSN: 2395-0056, Vol. 9, Issue 1, Jan 2022.

[35] S. Monika, K. Aruna Manjusha, S. V. S. Prasad, B. Naresh, "Design and Implementation of Smart Floor Cleaning Robot using Android App", International Journal of Innovative Technology and Exploring Engineering (IJITEE), ISSN: 2278-3075, Vol. 8, Issue 4S2, March 2019.

[36] Sonali A. Gaikwad, Shital P. Gambhir, Prajakta J. Pawar, Prof. Sadashiv R. Badiger, "Smart Floor Cleaning Robot", International Journal of Creative Research Thoughts (IJCRT), ISSN: 2320-2882, Vol. 6, Issue 1, March 2018.

[37] Prof. A. S. Shirkande, Chavan Aniket Nitin, Bhosale Gorakh Sudam, Parekar Datta Ashroba, "Wireless Floor Cleaning Robot", International Research Journal of Modernization in Engineering, Technology and Science, Vol. 5, Issue 11, November 2023.

[38] Prof. Zarinabegam Mundargi, Deepak Jadhavar, Vishal Bolke, Anandita Singh, Atharva Athanikar, "Floor Cleaning Robot Using Arduino-UNO", Vishwakarma Institute of Technology, Pune, India.

[39] J. Lee, A. S. Ab Ghafar, N. Mohd Nordin, F. A. Saparudin, N. Katiran, "Autonomous Multi-function Floor Cleaning Robot with Zigzag Algorithm", Indonesian Journal of Electrical Engineering and Computer Science, Vol. 15, No. 2, August 2019, pp. 1653–1663.

[40] Hind Zuhair Khaleel, Bashra Kadhim Oleiwi, "Design and Implementation Low Cost Smart Cleaner Mobile Robot in Complex Environment", Mathematical Modelling of Engineering Problems (MMEP), Vol. 11, No. 3, 2024.

[41] K. Sarath Kumar, M. Pavan, N. Kishore Karthikeyan, P. Sai Venkata Lokesh, K. Sasidhar, T. Haritha, "Arduino Based Smart Vacuum Cleaner Robot", International Journal for Research in Applied Science & Engineering Technology (IJRASET), Vol. 11, Issue 3, March 2023.

[42] Mr. Akash Kumbar, Mr. Arunkumar N., Mr. Chethan Naik L., Ms. N. Hema Latha, "Implementation of Dust Cleaning Robot", Project Ref. No.: 47S_BE_2503, Nagarjuna College of Engineering and Technology, Bengaluru.

[43] Pranal Mahajan, Nilesh Ponde, Abhishek Malvi, Akash Gupta, Dr. Dheeraj S. Deshmukh, “Design and Development of Smart Home Cleaning Robot”, G. H. Raisoni College of Engineering, Nagpur, Maharashtra, India.

[44] P. S. Adithya, R. Tejas, V. Sai Varun, B. N. Prashanth, “Design and Development of Automatic Cleaning and Mopping Robot”, IOP Conf. Series: Materials Science and Engineering, Vol. 577 (2019), ICONAMMA2018.

[45] Uman Khalid, Muhammad Faizan Baloch, Haseeb Haider, Muhammad Usman Sardar, Muhammad Faisal Khan, Abdul Basit Zia, Tahseen Amin Khan Qasuria, “Smart Floor Cleaning Robot (CLEAR)”, Ghulam Ishaq Khan Institute of Engineering Sciences and Technology, Pakistan.

[46] Joel Bergman, Jonas Lind, “Robot Vacuum Cleaner”, Degree Project in Electronics and Computer Engineering, KTH Royal Institute of Technology, Sweden, 2019.

[47] Ruri Ashari Dalimunthe, Maulana Dwi Sena, William Ramdhan, “Floor Cleaning Robot Control System with Android Based Voice Command”, Journal of Physics: Conference Series, Vol. 1783, ACOSTER 2020.

[48] Swathi R, R. Ananya, Kshama K. M, Dr. M. V. Sreenivas Rao, “Design and Implementation of a Bluetooth Controlled Floor Cleaning Robot”, International Journal of Advanced Research in Computer and Communication Engineering, Vol. 12, Issue 10, October 2023.

[49] Samuel Twum, Richard Sarpong, Alpha Agusah, “Design and Development of a Smart Floor Cleaning Robot Controlled by a Mobile App”, Lovely Professional University, Punjab, India.

[50] Akshata Gholap, Abhinay Rasal, Aishwarya Pawar, Chaitali Dighe, Prof. S.R. Gagare, "Multipurpose Cleaning Robot using IoT and Image Processing", IRJET, Vol. 7, Issue 6, June 2020.

[51] Dr. J. Ashok, M. Pavithra, T. Periyanayaki, C. Midhuna, "Revolutionizing Floor Cleaning: Design and Development of a Smart Solar-Powered Floor Cleaning Machine", JETIR, Vol. 10, Issue 4, April 2023, ISSN: 2349-5162.