Modular Autonomous Industrial System for Internal Transport

By

M Ayaan E Rasul
Enrollment No. 01-133222-046

Attaullah
Enrollment No. 01-133222-012

Supervised By

Sir Faheem Haroon

Session 2022-26

A Report is submitted to the Department of Electrical Engineering,
Bahria University, Islamabad.
In partial fulfillment of requirement for the degree of BS(EE).

Certificate

We accept the work contained in this report as a confirmation to the required standard for the partial fulfillment of the degree of BS(EE).

_____.	_____
Head of Department	Supervisor
_____.	_____
Internal Examiner	External Examiner

Dedication

We dedicate this work to our families our parents, for always being there for us throughout our studies. They kept encouraging us, were patient which helped us stay focused when things got tough with this project. Our families was really supportive and we're very grateful for them. Our parents helped us a lot. We also want to thank our teachers and mentors who guided us over the years. They gave us advice and taught us well which helped us understand engineering better. Our teachers and mentors were very helpful. These people helped shape our understanding of engineering, we are grateful. This project is for anyone who wants to build practical systems that actually work in real life not just on paper. It's for people who want to make systems that can be used in environments. .

Acknowledgments

We want to say thank you to our project supervisor Dr Faheem Haroon for helping us with this project from start to finish. Sir gave us good ideas and comments that helped us make our work better and stay on track. We also want to thank our department and the teachers there for teaching us what we needed to know to finish this project. They gave us the things we needed to get it done. Our friends and classmates also helped us a lot at times. We talked about things. They supported us. We are really thankful to our families for being there, for us and understanding what we were going through. Our families were very supportive. Without our families we do not think we could have finished this project.

Abstract

Material handling is really important in factories. It helps get work done quickly. Robots like Automated Guided Vehicles (AGVs) and Autonomous Mobile Robots (AMRs) are often used for this. Many of these robots use expensive technology. That makes them hard to use in projects or school work. This project is about making an cheap robot that can move things around. The robot uses an Arduino microcontroller. It also uses RFID technology to know where it is. Infrared sensors help the robot avoid hitting things. An ESP32 module lets the robot talk to devices wirelessly. The robot can even be controlled by hand. It has a lifting part that lets it pick up and put down things. The robot can work in two ways: manually and on its own. This makes it easy to test and use. The design was broken down into parts like sensing, control and movement. This made it easier to build and fix. Most testing was done with the software. The results showed that the robot responds fast to commands. Communication, through Bluetooth works well. The robot was not tested in a factory. It should work well in controlled areas. Overall this project shows that a working robot can be made with different parts by implementing different ideas. It would be a great robot that can work efficiently with sensors and navigation. The robot can move things around on its own.

Contents

List of Figures

Chapter 1

Introduction

Autonomous mobile robots (AMRs) are increasingly used for intralogistics and material handling in factories and warehouses, offering flexibility over traditional automated guided vehicles (AGVs). Unlike AGVs, which follow fixed guides (tape, beacons) and cannot deviate from a predefined route, AMRs have onboard autonomy to sense obstacles and replan paths in real time [8]. In dynamic industrial environments, AMRs improve scalability and reduce fixed infrastructure needs [5, 11]. This project's modular robot is designed for internal transport: it picks items at a "pick" station (marked by an RFID tag) and carries them to a "place" station (another RFID tag) while avoiding obstacles. Modularity here means the design is composed of interchangeable hardware modules (drive, lift, sensors) so that different vehicle bases or payload mechanisms could be integrated with minimal changes [4]. For example, Matthews et al. describe a modular autonomous controller that can be attached to various vehicle bases, highlighting portability and minimal modification as key advantages [4].

The system deliberately avoids expensive LiDAR by using alternative sensors (IR, ultrasonic, vision, RFID, and wheel odometry) for localization and obstacle detection [3]. This decision is motivated by cost and simplicity – for many indoor tasks, low-cost sensors and dead-reckoning can suffice when high precision is not required. A literature review below surveys relevant work on modular mobile robots and these sensor alternatives.

1.1 Project Background/Overview

Modern industrial environments rely heavily on efficient internal logistics to maintain smooth production flow. In manufacturing systems, materials must continuously move between storage areas, processing stations, and assembly lines. Although this movement is not part of the core production process, it directly affects productivity, operational efficiency, and overall system performance.

Traditionally, internal transport has been handled either manually or through fixed automation systems such as conveyor belts and wire-guided Automated Guided Vehicles (AGVs). While these systems are reliable in stable environments, they are not designed to adapt easily to changes in layout or production demand. Once installed, they often require significant modification efforts when the factory structure changes, which reduces operational flexibility. Over time, research in industrial robotics has shifted toward more adaptive systems such as Autonomous Mobile Robots (AMRs). These systems are designed to operate without fixed paths, relying instead on sensors and intelligent control systems to navigate dynamically. Studies highlight that AGVs and AMRs can significantly improve efficiency in material handling tasks [8,11]. However, most of these systems depend on expensive sensing technologies such as LiDAR, vision systems, or complex SLAM-based navigation frameworks.

This creates a practical gap between high-end industrial automation and low-cost implementable systems. While advanced systems offer strong performance, they are often unsuitable for small industries or educational environments due to cost and complexity. This situation motivates the need for a simplified yet functional autonomous transport system that can demonstrate core principles of industrial automation using accessible

components.

1.2 Problem Description

The ideal industrial transport system should be flexible, lowcost, and capable of operating autonomously in dynamic environments. It should be able to move materials efficiently, respond to obstacles, and adapt to changes in layout without requiring major infrastructure modifications. In addition, it should be scalable and easy to implement across different operational settings.

However, current solutions fall short of this ideal scenario in several ways. Traditional AGV systems rely on fixed guidance mechanisms, which limit adaptability. Any change in the factory layout requires physical modifications to the guiding infrastructure, making the system rigid and costly to maintain. On the other hand, advanced autonomous systems offer greater flexibility but depend on expensive hardware and complex algorithms, making them inaccessible for small-scale applications.

Furthermore, multi-robot coordination systems, although highly efficient in theory, introduce additional complexity. Centralized systems suffer from scalability issues, while decentralized systems require complex communication and negotiation protocols [5]. These challenges make it difficult to implement such systems in practical low-resource environments.

As a result, there is a clear need for a system that balances simplicity, cost-effectiveness, and functional autonomy. The lack of such a system creates a gap in both academic research and practical implementation, particularly in educational and small industrial contexts.

1.3 Project Objectives

The objectives of this research are:

1. To design a mobile robotic platform using DC gear motors and wheel-based locomotion
2. To develop a lifting mechanism using a screw-driven motor system for basic material handling
3. To implement RFID-based location identification for navigation between predefined points [3]
4. To integrate infrared sensors for obstacle detection and avoidance
5. To develop a microcontroller-based control system using Arduino
6. To enable wireless communication and control using ESP32
7. To implement both manual and autonomous operational modes
8. To evaluate system performance in terms of functionality, reliability, and responsiveness

1.4 Project Scope

This study has both academic and practical significance. Academically, it contributes to the understanding of how low-cost components can be integrated to form a functional autonomous system. The system bridges the gap between theoretical robotics concepts and real-world implementable solutions, providing a cost-effective alternative for small industries that require basic automation but cannot afford advanced industrial AGVs. It also serves as a learning platform for students to understand core concepts of robotics, embedded systems, and industrial automation.

The modular nature of the system further enhances its significance, as it allows future improvements such as advanced sensors, improved control algorithms, or multi-robot expansion.

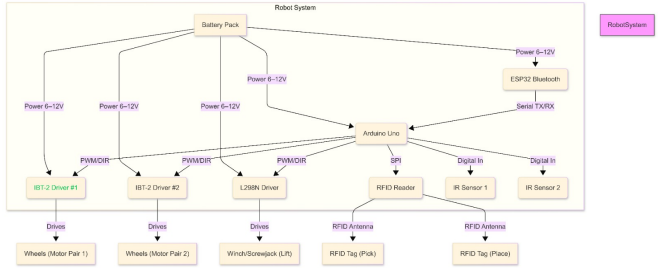

Figure 1.1: System process flow for autonomous navigation and obstacle handling

Chapter 2

Literature Review

Research on making factories work better has shown that how things are moved around inside the factory is really important. This part of the factory has not been improved as much as other areas. Factories have gotten a lot better at making things with machines. However, the way materials are moved around the factory has not changed quickly. This resulted in a lot of research on Automated Guided Vehicles and Autonomous Mobile Robots. Both of these projects are meant to reduce human workload and accidents and does not ensure humans to move things around and make the factory work better. Automated Guided Vehicles and Autonomous Mobile Robots are significant because they can help industries run smoothly and work efficiently [8, 11].

Early Automated Guided Vehicle systems were pretty basic. They had to be reliable. These systems usually worked with things like strips or special wires that were put into the factory floor. Ullrich said in 2015 that people liked these systems because they were strong and did not break down easily in places that were organized and did not change much [11]. They were also pretty safe to use. But they had a problem. They could only follow the paths that were already set for them. This made it hard for them to adapt when things changed, like in factories where the layout's different all the time.

As factories needed to be more flexible and handle work researchers started looking for better systems that could move around on their own. This was a change from the old way of doing things, where machines just did what they were told. Now people wanted machines that could think a little and use sensors to figure out where they were and what was around them so Automated Guided Vehicles could understand their environment and make decisions rather than just following the same old path all the time [8].

2.1 Evolution from Fixed-Path AGVs to Flexible Systems

Early Automated Guided Vehicle systems were designed to be simple and work well. These systems usually used things like strips or special wires in the factory floor to guide them. According to Ullrich in 2015 people liked these systems because they were strong and did not break down often in places where everything is organized [11]. But they had a problem. They could only follow the paths that were already set which made it hard for them to work in places where things were always changing.

Traditional Automated Guided Vehicles were the kind of industrial robots that could move around. They were made for places where everything's predictable and does not change much. These systems were easy to use because they just followed the guides that were put in the environment. This also meant they had a big limitation. If someone wanted to change the layout of the factory they had to physically change the guides too.

Some researchers like Gómez and his team in 1997 looked at Automated Guided Vehicle systems in factories and found out that they were not very flexible [4]. Even though the vehicles that followed wires worked well they could not adapt when the production setup changed. This team tried to make the vehicles work on their own a bit by using cheap sensors and computers and they found out that it was possible to make them partially autonomous without getting rid of the old system.

This was a step forward for Automated Guided Vehicles. Instead of getting rid of the old systems researchers started adding new sensors to them so they could work a little bit on their own. Automated Guided Vehicles were becoming more flexible. But these early attempts still had problems. The sensors were not very good and the computers were not

powerful enough so the systems did not work well in complex places.

2.2 Multi-Robot Coordination and Decentralization

As industrial automation got bigger people started to think about more than one Automated Guided Vehicle system. They wanted to figure out how to make many robots work together. The goal was not just to make the vehicles move efficiently but to make the whole system work better with vehicles operating at the same time in the same space.

Some researchers like Hošek and his team looked at how to make Automated Guided Vehicles work together without a central controller [5]. They suggested a system where each vehicle plans its path and talks to other vehicles to avoid running into them. This way the system does not rely on one controller, which can be a problem in big systems. Instead each vehicle is smart and can make its own decisions.

This approach is similar to systems that try to solve the problem of making the system bigger. When you have a central controller it can get really slow when you add more vehicles. If you make the vehicles smart and able to talk to each other they can make decisions on their own and it does not get as slow.

However these systems have their problems. It gets really hard to make all the vehicles work together smoothly. It is difficult to make sure everything is working efficiently without someone overseeing the whole system. Also many of these systems need algorithms and sensors that make them more expensive and harder to set up.

2.3 Path Planning and Control Architectures

Path planning is a problem when it comes to Automated Guided Vehicles and Autonomous Mobile Robots research. The goal is to find the route from the starting point to the destination without hitting anything and keeping the distance or energy used as low as possible.

Some people like Siegwart and others said in 2011 that path planning is like a problem where you have to plan the route and also avoid obstacles in real time [8]. This way of doing things has become very common in robots nowadays.

There are complicated ways to do path planning that use special maps of the environment so the robots can find the best path even when there are rules to follow. But these methods need a lot of computer power so they are not good for cheap robots.

On the other hand simpler robots use basic rules or make decisions based on what their sensors tell them. This way they do not need much computer power but they are not as good at adapting to new things. This is a trade-off when designing Automated Guided Vehicles systems.

2.4 Research Gap and Motivation for the Present Study

A big area of improvement in self-driving systems is combining sensors for navigation and avoiding obstacles. High-end systems usually use LiDAR, stereo vision or SLAM-based techniques to make maps of the environment. These methods are very accurate and also make the system more expensive and complicated [8].

To solve this problem many studies have looked at alternatives. Infrared

sensors and ultrasonic sensors are often used in robotic systems because they are affordable and easy to use. They are not as accurate as LiDAR but they are good enough for detecting obstacles that are close by and basic navigation tasks.

Finkenzeller in 2010 wrote about RFID systems in automation especially for identifying and locating things [3]. RFID-based navigation is a strong way to mark specific points without needing to constantly map the environment. This makes it very suitable for places where locations can be tagged ahead of time.

However RFID-based systems do not have awareness of the space around them so they need to be used with other sensors or movement strategies that use logic to achieve complete navigation behavior.

- There are challenges with RFID-based systems.
- They do not provide navigation on their own.
- RFID systems need to be combined with other technologies.
- This helps achieve navigation capabilities.

Chapter 3

Requirement Specifications

3.1 Existing System

Industrial material transport is usually done by people or by using machines like conveyor belts and special vehicles called Automated Guided Vehicles. These vehicles follow wires or special strips on the floor [11]. Using people to transport materials is simple and does not cost a lot but it has some problems. People can get tired and make mistakes which means the work does not get done consistently. Also when people are moving things around it can be dangerous.

Machines like Automated Guided Vehicles were made to help with these problems. They can do the tasks over and over without getting tired. These machines are more reliable than people. They are not very flexible. If the factory changes its layout the machines need to be changed which takes a lot of time and money. This makes them not very useful for factories that change their layout often [4].

There are also advanced machines like Autonomous Mobile Robots that use special technologies to move around. These machines are very flexible. They are also very complicated and expensive. They need powerful computers and special programs to work, which makes them not very useful for projects or schools [8].

Another problem with many of these machines is that they are not very modular. This means that if you want to change something or make it better you have to change a lot of the machine. Most machines that are not very expensive can only move around and do not have other functions like lifting things. Industrial material transport systems like these need to be improved so they can be used in more places.

3.2 Proposed System

To overcome the problems found in existing systems we propose a system. It is an autonomous industrial transport platform. We designed it using low cost and easily available parts.

The system aims to achieve autonomous functions. It also focuses on keeping things simple and adaptable.

Unlike Automated Guided Vehicles (AGVs) our system does not use fixed physical paths. Instead it uses RFID tags to find its location. This allows us to change routes easily. We just need to move the RFID tags. There is no need to modify the physical setup [3].

The system also uses sensors. They help the vehicle detect obstacles in time. This makes the vehicle respond quickly to obstacles. It improves safety and reliability during operation.

Our navigation approach is simple. It is not as complex as Autonomous Mobile Robot (AMR) systems. It is good enough for controlled environments. It also reduces computational requirements [10].

A key feature of our system is its modular design. Each part such as movement sensing control and lifting is developed separately. This makes it easy to modify maintain and upgrade parts without affecting the whole system.

In addition to moving the system has a lifting mechanism. It uses a screw lead driven by a DC gear motor. This enables the robot to perform material handling tasks. It makes the robot more representative of industrial applications [1].

Overall our system provides a balance between functionality cost and simplicity. It shows features of autonomous transport systems while remaining accessible for small scale implementation.

3.3 Requirement Specifications

The requirements of the proposed system define the expected behaviour and performance of the system. These requirements are divided into functional and non-functional categories.

3.3.1 Non-Functional Requirements

Non-functional requirements are the rules that the system has to follow when it is working. These rules include things like how the system performs, reliability, safety and design constraints.

The system had to be able to work in real time so that it could make decisions quickly. This means that when the system gets information from sensors it has to be able to process that information fast enough to make decisions immediately [12].

The system also had to be reliable which means it had to work the same way every time it was used without failing.

When it came to designing the system we had to make sure it was not too expensive and that it did not use too much energy. We chose parts for the system based on cost and availability so that the system would be practical and easy to implement.

We also had to think about safety. The system had to be designed in such a way so that it would not cause harm to users or the environment. This meant that electrical parts were properly insulated and motors were controlled safely [1].

The system also had to be designed so that it could be easily updated. This means that if one part of the system needed to be replaced or upgraded we would not have to change the entire system. These requirements are important for long term usability.

3.4 Use Cases

The system is used in different ways and it responds to different situations. These use cases help explain how the system works in real conditions.

One way to use the system is in autonomous mode. The user starts the system and the robot moves on its own along predefined paths. When the robot finds an RFID tag it identifies the location and performs actions such as stopping or lifting an object. If an obstacle is detected the system stops until the path is clear.

Another way to use the system is in manual mode. The user can control the system remotely using the ESP32 microcontroller. This allows the user to control movement and lifting which is useful during testing or maintenance.

The system is also used for material transport tasks. It moves to a location lifts an object and then transports it to another location. This is similar to industrial workflows where materials are moved between stations.

In all these situations the system interacts with the environment and the user. It shows that it can perform important tasks autonomously in a controlled environment.

Chapter 4

System Design

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements.

4.1 System Architecture

The system that we are discussing is a machine that can move things around by itself in an area. This machine has three parts: the part that controls everything the part that senses things and the part that makes things happen. These parts work together so the machine can move around detect things in its way and pick up things [8].

The control part is like the brain of the machine. It uses something called Arduino to make decisions. It gets information from sensors and tells the motors what to do. There is also a part called ESP32 that lets the machine talk to people and other machines without using any wires. This means the machine can be controlled by someone or it can work by itself.

The sensing part has tools like RFID and infrared sensors. RFID is like a tag that the machine can read to figure out where it is and what to do [3]. Infrared sensors help the machine detect things in its way so it can stop or go around them.

The actuation part is what makes the machine move. It has four motors that make the machine go forward backward left and right. It also has another motor that helps the machine lift things up.

When you look at the system it gets information from sensors and from people and then it uses that information to make the motors do things. The area around the machine is like a thing that the machine talks to when it runs into obstacles or reads those special RFID tags.

In all these situations the machine is talking to the area around it and to people. It is doing important things by itself in a special area.

4.2 Design Constraints

The system design had to consider limitations. One major limitation was cost. We had to build the system using affordable and easy-to-find parts which meant we could not use sensors like LiDAR or cameras.

Another limitation was processing power. The Arduino microcontroller is not very powerful so we had to keep our algorithms simple and use rule-based approaches instead of complex navigation techniques [10].

We also had to think about power consumption. The system had to work with a limited power supply which affected our choice of motors and sensors.

The lifting mechanism had its own mechanical limitations. We chose a lead screw system because it is simple and can hold its position without continuous power [1]. However this made lifting operations slower.

When designing the system we made some assumptions. We assumed it would operate indoors on a flat surface with predictable obstacles. We also assumed RFID tags would be placed correctly and remain fixed during operation.

4.3 Design Methodology

The system was made using an iterative design process. Each part of the system was tested individually before integration. This made it easier to identify and fix problems early.

The design approach followed a modular structure where the system was divided into sensing control and movement components. Each part had a defined function and interacted with others in a structured way.

Simple representations such as flowcharts and block diagrams were used

to understand system behavior. Even though full UML diagrams were not used the design still followed structured engineering principles.

The system was continuously improved through testing and refinement. Early versions were modified to improve reliability and performance.

4.4 High Level Design

1. Conceptual / Logical View

   This view represents the system in terms of its functional components. The proposed system is divided into modules such as sensing control communication movement and lifting. The sensing module includes RFID and IR sensors that interact with the environment. The control module made using Arduino processes inputs and makes decisions. The communication module handled by ESP32 enables interaction. The movement module controls motion using DC motors while the lifting module performs material handling tasks. Each component performs a specific function while interacting with others to achieve overall system behavior.

2. Process View

   The process view describes how the system operates during runtime. The system follows a loop where sensor data is collected processed and used to control actions. First RFID and IR sensor inputs are read by the controller. The Arduino processes this data and determines what to do such as moving forward or stopping. In manual mode commands received through ESP32 override autonomous behavior.

3. Physical View

   The physical view represents the hardware structure of the system.

All components are mounted on a chassis including the Arduino controller ESP32 module motor drivers sensors and power supply. The four DC gear motors are attached to the wheels for movement. The lifting mechanism is installed vertically using a lead screw system.

4. Module View

   The module view focuses on software organization. The program is divided into modules such as motor control sensor processing communication handling and lifting control. Each module is implemented separately making it easier to maintain and upgrade.

5. Security View

   The security view of the system is limited. Basic safety measures include proper electrical insulation and controlled motor operation. Communication through ESP32 is restricted to valid commands.

4.5 Low Level Design

The system is made up of smaller components that work together. The motor control module manages direction and speed using motor drivers. The sensor module processes inputs from IR sensors and RFID readers.

The control logic runs on Arduino. It processes sensor inputs and decides system actions. For example when an RFID tag is detected the system executes a predefined action.

The communication module is handled by ESP32 which allows user input through wireless signals. Commands are translated into movement or lifting actions.

Each module operates independently but communicates through shared signals to ensure coordinated behavior.

4.6 Database Design

The system does not use a traditional database. Instead data such as RFID tag IDs and associated actions are stored in the microcontroller memory.

Each RFID tag has a unique ID which corresponds to a predefined action such as stop move or lift. The system processes data in real time without storing large datasets.

4.7 GUI Design

The system provides user interaction through the ESP32 module. Instead of a graphical interface commands are sent using a mobile device or serial communication.

The system supports two modes manual and autonomous. In manual mode the user controls movement and lifting. In autonomous mode the system operates independently based on sensor input.

The design is simple and focuses on functionality rather than visual complexity.

4.8 External Interfaces

The system interacts with external elements such as RFID tags the environment and the user interface.

RFID tags provide location information while infrared sensors detect obstacles. The ESP32 module enables wireless communication between the user and the system.

The system operates as a standalone unit and does not rely on integration with external industrial systems.

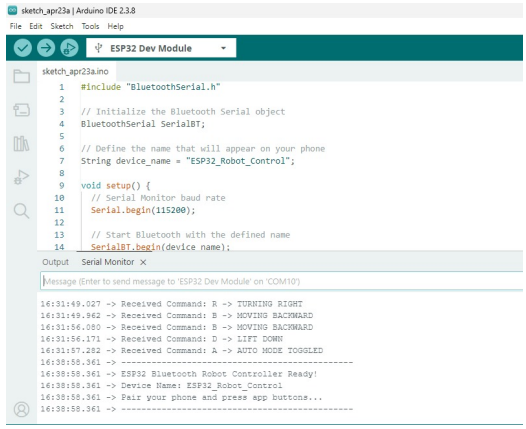

Figure 4.1: Serial monitor output showing successful execution of movement commands

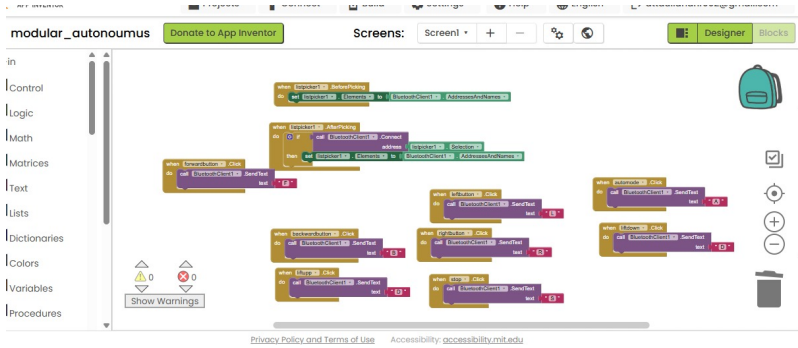

Figure 4.2: GUI Backend Logic: Bluetooth communication protocol and directional command mapping for the mobile application.

Chapter 5

System Implementation

Implementation is the process of moving an idea from concept to reality. The System implementation is a realization of a technical specification or algorithm as a program, software component, or other computer system through programming and deployment.

5.1 System Architecture

The modular autonomous industrial transport system was made by putting sensing, control and actuation parts into one system. This system is like an Automated Guided Vehicle, where the vehicle sees something makes a decision and then does something, all in a continuous loop. This kind of system is used a lot in robotics because it works well and does things in real time [8].

The control part was built around the Arduino microcontroller, which was the brain of the system. It got information from the sensing part and sent signals to the actuators to do something. The ESP32 module was added so the system could communicate without wires and so people could control it manually if they needed to. Other people have made systems that combine microcontrollers with wireless modules and they have worked well in prototypes of Automated Guided Vehicles [6, 9].

The sensing part had an RFID module and infrared sensors. The RFID tags were put in places and helped the system figure out where it was so it could make decisions about where to go [3]. Using RFID tags to navigate is an idea because it makes the system more flexible and able to change. The infrared sensors were used to detect obstacles, which made the system safer.

The actuation part had four motors that made the system move and another motor that made the lifting mechanism work. The motors were

connected to the controller using drivers, which made sure the motors moved in a controlled way. All the parts of the system communicated with each other using signals, which made a closed-loop system where everything worked together all the time.

5.2 Tools and Technology Used

The system was made using a mix of hardware and affordable automation tools. Arduino was chosen as the controller because it is easy to use has a lot of community support and works well with sensors and actuators.

Arduino and other microcontroller-based systems are used a lot in industrial automation projects. This is because they are easy to work with and can be changed to fit needs. Researchers like Irsyadi and others found this to be true in their study in 2021 [6].

RFID technology was used to help the system navigate and identify things. Studies have shown that RFID technology is an alternative to traditional navigation systems and works well in changing environments [7]. Infrared sensors were used to detect obstacles because they are cheap and work well for short distances.

For movement special motors called DC gear motors were used. These motors have a high gear ratio which means they can provide a lot of power and move smoothly. The lifting mechanism used a lead screw system with a DC motor. This system is often used in mechanical systems because it can move precisely [1].

Other important parts of the system included motor drivers power supply modules chassis and connecting interfaces. These were selected based on cost availability and compatibility.

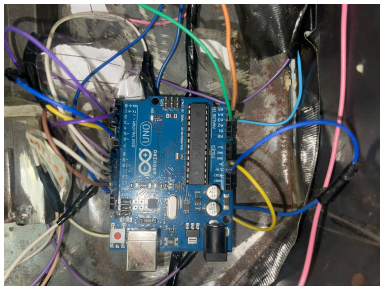

Figure 5.1: Arduino UNO

Figure 5.2: Ultrasonic Sensor

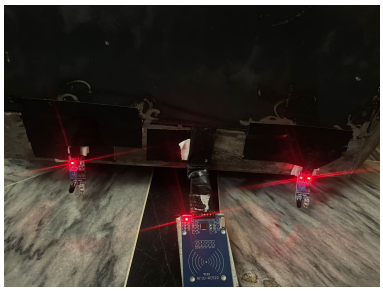

Figure 5.3: Sensors

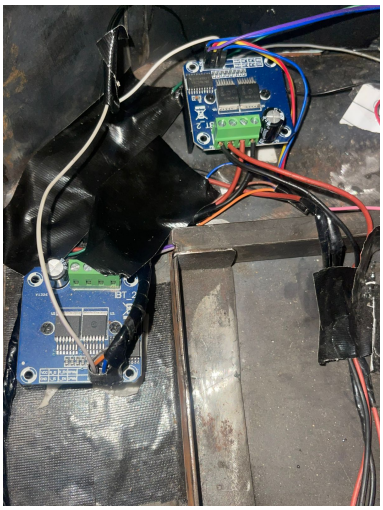

Figure 5.4: IB2 for Motor Drivers

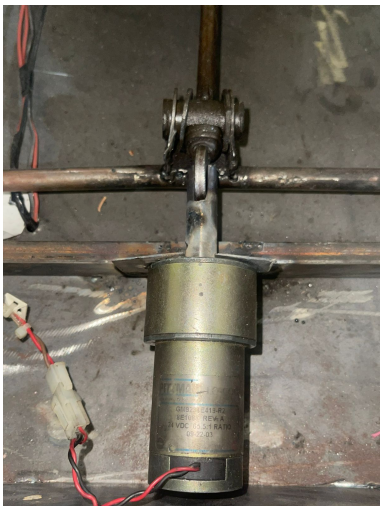

Figure 5.5: Lift Motor

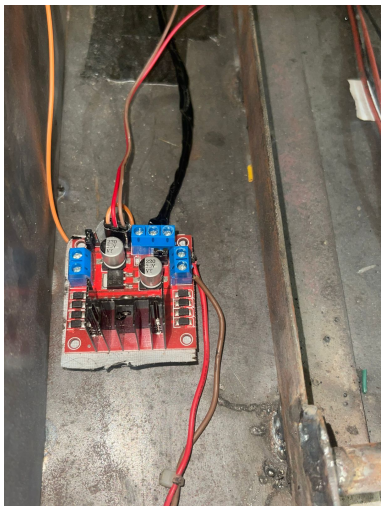

Figure 5.6: L2989 for motor drivers

5.3 Development Environment/Languages Used

The software was made using the Arduino Integrated Development Environment (IDE). The programming language used was Embedded C which is commonly used for microcontroller systems.

The Arduino IDE has built-in libraries for working with sensors like RFID modules and IR sensors making it easier to develop. We used an approach similar to previous AGV prototypes which also used Arduino-based systems for quick development and testing [9].

The ESP32 module was programmed using libraries that work with the Arduino IDE. This allowed communication between the control system and wireless subsystems. The code was written in a modular way with separate functions for sensor processing motor control and communication.

5.4 Processing Logic/Algorithms

The system works using a set of rules to make decisions in real time. It follows a loop where it reads sensor inputs processes them and then takes actions.

First the system decides how it will operate.

In manual mode commands sent through the ESP32 module directly control how the system moves and lifts things.

In autonomous mode the system uses sensor inputs to make decisions.

The system uses RFID tags for navigation. When it detects a tag it matches the tag ID with predefined conditions and performs the required action. This RFID navigation method is commonly used in AGV systems because it is reliable for location identification [3].

The system also uses IR sensors to check for obstacles. If an obstacle is found the system stops moving until the path is clear.

Some systems use advanced algorithms like fuzzy logic or optimization techniques [2]. However for low-cost AGV systems with limited processing power simple rule-based approaches are often used.

5.5 Application Access Security

The system was designed with basic application-level security. The ESP32 module was used to enable wireless control.

Only authorized users could send commands to the system. These commands had to follow a predefined format.

The system did not implement advanced encryption due to hardware limitations. Since communication occurred over short distances in a controlled environment the risk of unauthorized access was limited.

This type of security approach is commonly used in prototype robotic systems where full cybersecurity is not required.

5.6 Database Security

The system does not use a database. It stores information such as RFID tag IDs and associated actions in the microcontroller memory.

Access to this data requires physical or direct system connection which prevents unauthorized modification.

Since the system does not use cloud storage or remote databases there is minimal risk of external data breaches.

This simple approach is suitable for AGV systems that operate in real time and do not require large-scale data storage.

Chapter 6

System Testing and Evaluation

The people in charge did some tests to see how well the new modular autonomous industrial transport system works when everything is put together. They wanted to make sure the system does what it is supposed to do, like talk to parts, move lift things and switch between different modes.

When you test the system you get to see how all the parts work together which is different from just testing one part at a time. For this project they did some tests with the parts and some tests with just the software using a special tool called an ESP32 Bluetooth interface that talks to the control system, which is based on something called Arduino.

They are still working on testing the system in an industrial setting so this part of the report is a mix of what they saw actually happen and what they think should happen based on the plans and how similar systems usually work [8]. This way they can get an realistic idea of how the modular autonomous industrial transport system is doing.

6.1 Evaluation Metrics

The system was evaluated using both qualitative metrics. These metrics are commonly used to assess AGV systems [6].

Quantitative Metrics:

• Command response time. This is the delay between sending a command via Bluetooth and the actuator actually executing it.

• Accuracy of command execution. We measured this as the percentage of correct responses.

• Stability of communication. This is the rate of packet loss or failure.

• Sensor response consistency. We expected stable responses from the IR and RFID modules.

Qualitative Metrics:

• How easy is it to use the mobile control interface?

• System flexibility. This includes switching between manual and autonomous modes.

• Reliability of motor response.

• Functional completeness of system modules.

These metrics help evaluate software performance and hardware behaviour.

6.2 Graphical User Interface Testing (Bluetooth Control Interface)

The system was tested using both numerical and non-numerical methods. The system uses a Bluetooth interface through ESP32 instead of a traditional graphical screen. Testing was done using the Arduino Serial Monitor, where commands were sent and system responses were recorded.

Observed Results:

The system received and executed commands correctly including:

• "F" → The system moved forward

• "B" → The system moved backward

• "L" → The system turned left

• "R" → The system turned right

• "S" → The system stopped

• "D" → The system went down

• "A" → The auto mode was toggled on the ESP32

These results show that the communication between the device and ESP32 worked correctly.

Evaluation:

The interface worked reliably in understanding commands with no noticeable delay or errors during testing. This aligns with studies showing that ESP32-based Bluetooth systems provide stable control in robotic applications [9].

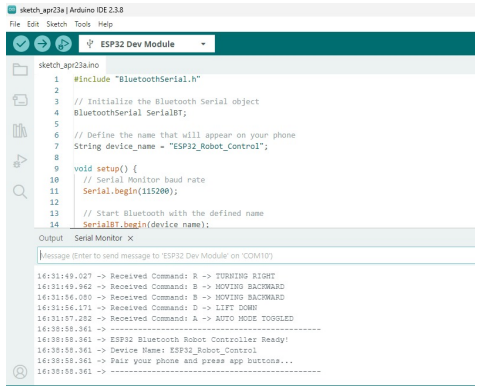

Figure 6.1: Serial monitor output showing successful execution of movement commands

6.3 Usability Testing

Observed and Expected Outcomes:

The system is simple and easy to understand because it uses single-character commands. It is also easy to switch between manual and autonomous modes as shown in system logs such as "AUTO MODE TOGGLED".

Evaluation:

The simplicity of control makes the system accessible even for non-technical users. However the lack of a graphical interface may limit usability for some users. Future improvements could include visual interfaces.

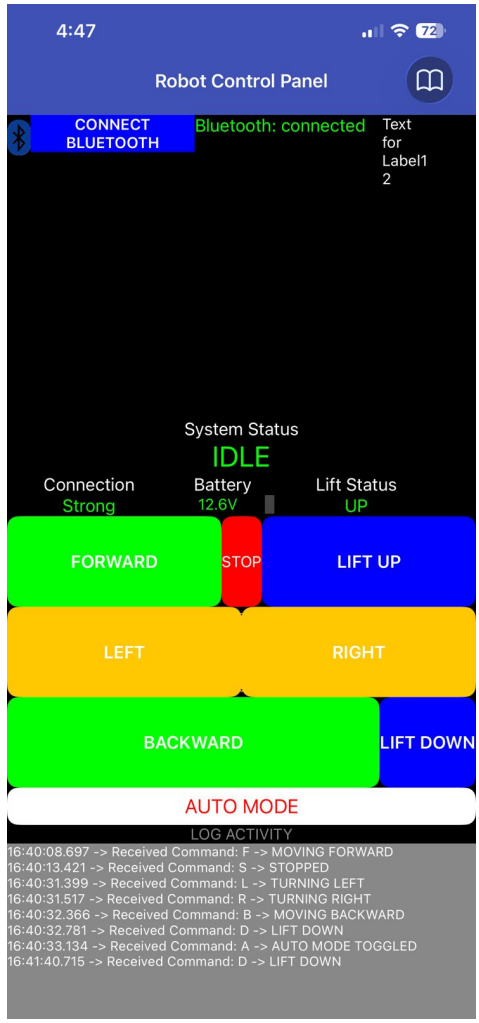

Figure 6.2: Mobile APP Interface

6.4 Software Performance Testing

The performance of the software was checked to evaluate speed and responsiveness.

Results:

The logs from the Serial Monitor show that the system responds immediately to commands without noticeable delay. For example when a command is received it is executed instantly with output such as MOVING FORWARD.

Expected Performance:

The Arduino processes inputs quickly so the system is expected to operate in real time with minimal delay.

Evaluation:

The system operates smoothly under normal conditions which is consistent with other Arduino-based AGV implementations [6].

6.5 Compatibility Testing

The system was tested for compatibility between hardware and software components.

Results:

All components including Arduino ESP32 sensors and motors worked together without integration issues.

Expected Performance:

The system is expected to maintain compatibility due to the use of standard communication protocols.

Evaluation:

The integration was successful and stable.

6.6 Exception Handling

When the system received invalid commands it ignored them and continued operating normally.

Expected Outcomes:

The system should safely stop motors if sensor failures or invalid signals occur.

Evaluation:

Basic exception handling is implemented but advanced fault detection is not present.

6.7 Load Testing

Load testing was not fully completed.

Expected Outcomes:

Based on motor specifications the system is expected to handle light to moderate loads. The DC gear motors provide sufficient torque and the lead screw mechanism can hold loads without continuous power [1].

Evaluation:

The system is not designed for heavy industrial loads and requires further testing.

6.8 Security Testing

Security testing focused on Bluetooth communication.

Observed Results:

- Only paired devices could send commands
- Invalid inputs were ignored

Expected Outcomes:

Short-range communication reduces security risks.

Evaluation:

Security is basic and suitable for controlled environments but not for industrial deployment.

6.9 Installation Testing

Installation testing evaluated setup and integration.

Observed Results:

- Hardware assembly was simple
- Software uploading using Arduino IDE worked without errors

Evaluation:

The system is easy to install and suitable for educational and prototype environments.

6.10 Strengths and Weaknesses

Strengths:

- Real-time Bluetooth control
- Modular design
- Low cost
- Reliable command execution
- Flexible operation modes

Weaknesses:

- Limited real-world testing
- Basic obstacle detection
- No advanced navigation algorithms
- Limited security features
- Unverified load capacity

6.11 Overall Discussion

The system works as expected in a prototype environment. The software is stable and responsive as shown by command execution results. However hardware testing is not fully completed so the system is not fully validated for industrial use.

Compared to existing AGV systems the proposed model is simpler and more cost-effective but lacks advanced features such as SLAM and multi-robot coordination [8]. This reflects a trade-off between cost and performance.

The system provides a strong foundation for further development.

Chapter 7

Conclusion

The development of the autonomous industrial transport system gave us several important insights into the design and implementation of embedded robotic platforms. One of the things we learned is that low-cost microcontroller-based systems can do reliable real-time control of mobility, lifting and wireless communication functions when they are properly integrated [8].

The system validation results we got from ESP32 Bluetooth communication showed that command-based control can be done with reliability and minimal delay. We were able to interpret movement commands like forward backward, turning, stopping and lifting operations. This shows that Arduino-based control systems are good for real-time robotic applications when the computational requirements are not too high [9].

We also learned that a modular system architecture is very effective. By separating the system into sensing, control, communication and actuation units we were able to make the design more flexible and easier to debug. This modularity also makes it easier to upgrade the system in the future without having to redesign the entire system.

The use of RFID-based positioning and IR-based obstacle detection showed us that there is a trade-off between system cost and environmental adaptability. While these sensors work well in controlled environments they are not sufficient for complex industrial settings. This highlights the need for more advanced sensing technologies [3].

Overall this project shows that we can develop prototype-level Automated Guided Vehicle systems using low-cost components and still achieve essential industrial transport functionalities such as navigation, obstacle response and load handling [6].

7.1 Key Technical Learnings

From a technical perspective we learned important things:

- ESP32 Bluetooth communication provides a responsive interface for real-time robotic control.
- Rule-based embedded programming is sufficient for AGV motion control but not suitable for complex decision-making scenarios.
- DC gear motors combined with screw-based lifting mechanisms provide a simple solution for small-scale material handling [1].
- RFID-based navigation is effective for positioning but does not support continuous path optimization.
- System modularity improves maintainability and scalability.

7.2 Limitations of the Developed System

Even though the system worked well we found several limitations:

- The biggest limitation is that the system was not tested in a real-world industrial environment. Testing was limited to software-level validation and controlled conditions.
- The sensing system is limited by the capabilities of infrared and RFID technologies. These do not allow long-range perception or continuous navigation.
- The system does not include advanced intelligence such as SLAM, machine learning or multi-robot coordination which are common in modern AGVs [8].
- The system lacks advanced security features and only uses basic Bluetooth pairing without encryption.

7.3 Future Work

The system provides a foundation for future development and there are several ways to improve it:

- One major improvement would be to implement SLAM-based navigation to allow real-time mapping and localization.
- The sensing system can be upgraded using LiDAR or ultrasonic sensors to improve obstacle detection.
- Advanced decision-making algorithms such as fuzzy logic or reinforcement learning can be added to improve autonomy [2].
- Multi-robot coordination can be implemented to allow multiple AGVs to operate together in shared environments.
- IoT-based monitoring and cloud integration can be added for remote supervision and predictive maintenance [13].

7.4 Final Remarks

In conclusion this project shows that it is possible to design an autonomous industrial transport system using low-cost embedded technologies. While the current system is best suited for controlled environments and prototype applications it provides a strong foundation for future research and industrial development.

The project highlights the importance of modular design, real-time embedded control and system integration in robotics engineering. It also shows the steps required to improve the system toward full industrial automation.

References

[1] W. Bolton. *Mechatronics*. Pearson, 2015.

[2] Y. Chen. Agv control system. 2023.

[3] K. Finkenzeller. *RFID Handbook*. Wiley, 2010.

[4] A. Gómez. Autonomous guided vehicle systems with free navigation capability. *IEEE ISIE*, 1997.

[5] J. Hošek. Decentralized coordination of autonomous guided vehicles. *IEEE Transactions*, 2018.

[6] F. Irsyadi. Agv navigation system. 2021.

[7] A. Palaskar. Rfid-based agv. 2013.

[8] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza. *Introduction to Autonomous Mobile Robots*. MIT Press, 2011.

[9] N. Sutisna. Mini agv with arduino. 2020.

[10] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[11] G. Ullrich. *Automated Guided Vehicle Systems*. Springer, 2015.

[12] J. W. Valvano. *Embedded Systems*. 2017.

[13] R. Y. Zhong. Smart agvs for industry 4.0. *Procedia Manufacturing*, 2018.

Appendix A

User Manual

A.1 Arduino Code

This section has the code for the system.

- The program controls the motors.
- It reads sensor data.
- It detects RFID tags.
- It processes commands from the ESP32 module.

The Arduino code is really important.

It makes the system work.

A.2 ESP32 Setup

The ESP32 module helps with communication.

- It lets the user send commands to the system through Bluetooth.
- The setup was easy.
- It used configuration.
- It used libraries.

The ESP32 module is really useful.

It makes the system more flexible.

A.3 Additional Figures

There are some figures here.

- They show wiring connections.

- They show testing results.
- They show system images.
- They are, for reference.

A.4 Hardware Components

The main components are listed below:

- Arduino microcontroller
- ESP32 module
- RFID reader and tags
- Infrared sensors
- DC motors
- Motor driver
- Power supply
- The Arduino microcontroller is a component.
- The ESP32 module is also important.
- The RFID reader and tags help with identification.

A.5 Limitations

The system has some limitations.

- It uses sensors.
- So it may not work well in environments.

- The processing capability is limited.
- So it restricts the use of algorithms.
- These areas can be improved in work.
- The limitations are something to improve on.
- The system can be made better.
- The Arduino code can be optimized.
- The ESP32 module can be used efficiently.

Appendix A. User Manual is compulsory.