A

PROJECT REPORT

On

Autonomous Trolly

Submitted in partial fulfilment of the requirement for the award of the

Degree of

BACHELOR of TECHNOLOGY

In

ELECTRONICS &TELECOMMUNICATION ENGINEERING

SUBMITTED BY

OMKAR BAJRANG KASHID (2162701372004)

SHUBHAM RAJENDRA SALUNKHE (2162701372008)

VEDANT ANAND MOHITE (2162701372059)

SAI VIJAY JADHAV (2162701372051)

Under the Guidance of

DR. Y. K. KANASE

DEPARTMENT OF ELECTRONICS & TELECOMMUNICATION ENGINEERING

Karmaveer Bhaurao Patil College of Engineering, Satara

of

Dr. Babasaheb Ambedkar Technological University, Lonere

2024-2025

Rayat Shikshan Sanstha's

Karmaveer Bhaurao Patil College of Engineering, Satara

CERTIFICATE

This is to certify that, the work that is being presented in this dissertation entitled "Autonomous Trolly" in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology in Electronics & Telecommunication Engineering of Dr. Babasaheb Ambedkar Technological University, Lonere submitted by,

Omkar Bajrang Kashid	2162701372004
Shubham Rajendra Salunkhe	2162701372008
Vedant Anand Mohite	2162701372009
Sai Vijay Jadhav	2162704372051

To the Department of Electronics & Telecommunication Engineering, Karmaveer Bhaurao Patil College of Engineering, Satara, is the record of the student's own work that is carried out under our supervision and guidance.

Date: / /20

Place: Satara

Dr. Y. K. Kanase	*Prof. Nanaware J. D.*	*DR. Attar A. C.*
Guide	*Head, E&TC Engg. Dept.*	*Principal*

DECLARATION

I hereby declare that the dissertation entitled "Autonomous Trolly" completed and written by me has not previously formed the basis for award of any degree or diploma or other similar title of this or any other university or examining body.

Date: / /20

Place: Satara

Omkar Bajrang Kashid

Shubham Rajendra Salunkhe

Vedant Anand Mohite

Sai Vijay Jadhav

Acknowledgement

It is my proud privilege and duty to acknowledge the kind of help and guidance received from several people in preparation of this report. It would not have been possible to prepare this report in this form without their valuable help cooperation and guidance.

First and foremost, I wish to record my sincere gratitude to our beloved Principal **Dr. A. C. Attar** for his constant support and encouragement in preparation of this report and for making available library and laboratory facilities needed to prepare this report.

I express my sincere gratitude to my guide **Dr. Y. K. Kanase**, for guiding us in investigations for this project and in carrying out experimental work. My sincere thanks to **Prof. J. D. Nanaware** Head of Department of Electronics and Telecommunication Engineering, for his valuable suggestions and guidance throughout the period of this report. Finally, I would like to wish special thanks to Management of Karmaveer Bhaurao Patil College of Engineering, Satara, all staff members, my family members and friends, without whose patience, encouragement, and support this Project work might never have been completed.

Omkar Bajrang Kashid
Shubham Rajendra Salunkhe
Vedant Anand Mohite
Sai Vijay Jadhav

LIST OF FIGURES

TABLE OF CONTENTS

ABSTRACT

The growing need for automation in industries, warehouses, and institutional campuses has led to the development of autonomous systems that enhance efficiency, reduce human effort, and improve safety. This project presents the design and implementation of an Autonomous Trolley system capable of navigating a predefined path using time-based control, while also intelligently detecting and avoiding obstacles in real time. The trolley is powered by an ESP8266 microcontroller, chosen for its processing power, wireless capabilities, and low energy consumption. It uses a VL53L0X LiDAR distance sensor to detect obstacles and make real-time decisions based on environmental feedback.

The trolley follows a predetermined sequence of movements defined by timing logic rather than GPS or vision-based navigation. During operation, if an object obstructs its path, the trolley stops and waits for 5 seconds, giving the obstacle a chance to clear. If the object remains stationary beyond this period, the trolley initiates a rerouting maneuver consisting of a series of directional movements (left, forward, right, etc.) designed to avoid the object and rejoin the original path without human intervention.

A buzzer is used to provide auditory feedback whenever the trolley encounters obstacles or reroutes, enhancing the system's interactive capabilities. Motion control is achieved using an L298 motor driver module, which allows the microcontroller to direct two DC motors effectively. Power is supplied via a rechargeable battery, ensuring portability and independence from external power sources.

This autonomous trolley provides a cost-effective, simple, and scalable solution for short-range transportation tasks in controlled environments. It demonstrates the integration of sensors, actuators, and embedded software to achieve intelligent behavior with minimal components. The project is especially relevant for educational, prototyping, and light-duty industrial applications where automation is beneficial but full-scale robotic systems are unnecessary or cost-prohibitive.

Chapter 1

Introduction

Automation has increasingly become a cornerstone of modern technological advancements across multiple domains including manufacturing, logistics, healthcare, and education. The ability of machines and embedded systems to perform tasks without direct human input enhances efficiency, reduces operational costs, and improves workplace safety. One such innovation is the autonomous trolley—a self-driving cart capable of transporting materials or items from one location to another with minimal supervision.

Traditional autonomous navigation systems often rely on GPS, cameras, or complex sensor arrays, making them expensive and difficult to maintain. Moreover, in indoor or constrained environments such as factory floors, GPS signals are weak or unavailable, and camera-based systems require heavy processing and are susceptible to environmental lighting conditions. To overcome these challenges, this project introduces a simple, time-based autonomous trolley system that navigates using predefined motion sequences and handles obstacles with basic yet effective logic.

The proposed trolley is built using an ESP8266 microcontroller, which handles logic control, sensor data processing, and wireless communication. The VL53L0X time-of-flight (ToF) sensor is used for precise distance measurement, enabling the trolley to detect obstacles within its immediate path. Upon detecting an obstacle, the trolley waits for a short period (5 seconds), allowing time for the object to potentially move. If the object remains in place, the system initiates a rerouting mechanism—a series of timed motor movements that circumvent the obstacle and bring the trolley back to its original course.

The trolley's movement is driven by two DC motors, controlled via an L298 motor driver, and powered by a rechargeable battery, ensuring mobility and autonomy. A buzzer is used to notify users when the trolley stops or reroutes, improving system transparency and safety. A web-based interface hosted by the ESP8266 allows for manual control during testing or emergencies.

The aim of this project is to deliver a low-cost, easily replicable, and efficient autonomous transport solution using readily available components. This makes it ideal not only for industrial use but also for educational purposes, where students can gain hands-on experience with embedded systems, sensor integration, and automation logic. Furthermore, the system can be scaled or adapted to various environments, proving its flexibility and practical utility.

1.1) Problem statement

Manual transportation of materials or goods in environments such as factories, hospitals, or campuses often results in inefficiencies, human errors, and safety concerns. Traditional autonomous systems typically rely on complex and costly technologies such as GPS, computer vision, or track-based navigation. There is a need for a simple, affordable, and reliable trolley system that can autonomously follow a predefined path, detect obstacles in real time, and make intelligent decisions to pause or reroute when required. This project aims to design and develop an autonomous trolley using basic embedded components that can navigate based on time control, detect obstacles using a LiDAR sensor, and avoid collisions through a predefined rerouting strategy.

1.2) Objectives

1. To design and develop an autonomous trolley system capable of navigating along a predefined time-based path without human intervention.
2. To integrate a VL53L0X LiDAR sensor for real-time obstacle detection and distance measurement.
3. To implement an intelligent decision-making mechanism that allows the trolley to pause and wait if an obstacle is detected.
4. To develop a rerouting algorithm that enables the trolley to bypass obstacles and return to its main path.
5. To provide audible feedback using a buzzer when the trolley encounters an obstacle or enters rerouting mode.
6. To utilize the ESP8266 microcontroller for controlling all components and hosting a web interface for manual control and testing.
7. To build a cost-effective and efficient solution suitable for applications in warehouses, hospitals, campuses, and indoor logistics

Chapter 2

BACKGROUND AND LITERATURE REVIEW

2.1) Background

Automation has become increasingly important in industries, hospitals, and smart campuses to improve efficiency and reduce manual effort in material transport. Traditional autonomous systems often rely on GPS, cameras, or line-following mechanisms, which can be costly and complex—especially in indoor or semi-structured environments.

To address these challenges, this project proposes a low-cost, sensor-based Autonomous Trolley that navigates a predefined path using time-based logic. It uses an ESP8266 microcontroller, VL53L0X LiDAR sensor for obstacle detection, L298N motor driver for motion control, and a buzzer for alerting during interruptions. The trolley can stop, wait, or reroute around obstacles and resume its path, making it ideal for structured indoor use cases.

This system demonstrates how affordable components can be used to build intelligent, autonomous transport solutions without the need for complex navigation systems.

2.2) Literature Review

In[1]"Robotic Autonomous Trolley Collection with Progressive Perception and Nonlinear Model Predictive Control": This work presents a fully autonomous mobile manipulation system designed for unattended collection of luggage trolleys, medical supply carts, and similar objects an application especially relevant in logistics and healthcare contexts. The proposed system integrates a progressive perception module that incrementally refines object location and orientation estimates through multi-view sensing as the robot approaches the target. This enables reliable grasp planning despite initial uncertainty. A lightweight online deep-learning model detects trolleys, then a point-cloud based pose refinement method improves accuracy in real time. For motion control, the authors employ a nonlinear model predictive controller (NMPC) that accounts for the robot's nonholonomic constraints and actuator limits, generating smooth trajectories that align the arm and base for successful pickup maneuvers. Extensive real-world experiments show the system achieving over 90% success in varied environments, demonstrating resilience to perception errors and dynamic disturbances. Compared to baseline methods using static perception and simple PID control, the proposed approach improves reliability and efficiency, significantly reducing failed grasp attempts or collision risks. This research highlights how coupling adaptive perception with advanced control yields robust autonomous solutions for mobile manipulation in unstructured real-world settings.

In[2]"Robotic Autonomous Trolley Collection with Progressive Perception and Nonlinear Model Predictive Control": This paper introduces a mobile robotic system engineered for fully autonomous collection of luggage trolleys aimed at high-traffic environments like airports. The solution features a compact hardware platform equipped with a chassis, multi-modal sensors (RGB-D camera, 2D/3D LiDARs), and a bespoke robotic manipulator with feedback encoding for precise grasping. A progressive perception strategy enables robust long-range detection via monocular 3D pose estimation combining object detection (e.g., YOLOv5) and six-point keypoint regression and then switches to high-precision short-range LiDAR-based plane fitting as the robot nears the trolley. For autonomy, the authors develop a three-stage hierarchical framework: approaching, docking, and returning. During approach, the robot aligns behind the trolley; in docking, it refines pose and deploys the fork; finally, it transports the trolley to a designated location.

Crucially, motion planning leverages a Nonlinear Model Predictive Control (NMPC) architecture augmented with Control Barrier Functions (CBFs), ensuring obstacle avoidance and persistent alignment between sensors and trolley during close-range maneuvers. Experiments conducted on a real robotic prototype within dynamic, crowded scenarios (including moving humans) show successful autonomous trolley collection, validating system robustness and safety.

In[3]"Robotic Autonomous Trolley Collection with Progressive Perception and Nonlinear Model Predictive Control": This paper presents an advanced mobile robotic system capable of autonomously collecting luggage trolleys, particularly suited for environments like airports and hospitals. It integrates a compact hardware platform featuring a differential-drive base, RGB-D cameras, multi-mode LiDAR, and a custom manipulator. The core innovation lies in a progressive perception framework: long-range 3D detection combines object detection and keypoint estimation to identify trolleys from afar, while close-range docking uses LiDAR-based point-cloud plane fitting to fine-tune pose estimates in real time. This layered approach boosts both detection reliability and grasp accuracy.

On the autonomy front, the paper introduces a Nonlinear Model Predictive Control (NMPC) planner augmented with Control Barrier Functions (CBF). This planner dynamically generates smooth, nonholonomic motion trajectories that respect system constraints, actively avoid obstacles including humans and maintain visual tracking of the trolley during close maneuvers. The system undergoes real-world evaluations in cluttered, dynamic settings, consistently demonstrating high success rates in docking, picking, and transporting tasks amid static and

moving obstacles. The authors report robust performance and significant improvements over prior methods that relied on simpler perception models or linear control strategies. Overall, this work highlights the value of coupling adaptive perception with safety-aware controls for reliable mobile manipulation in unstructured, real-world scenarios.

In[4]"HPPS: A Hierarchical Progressive Perception System for Luggage Trolley Detection and Localization at Airports" The paper introduces HPPS, a streamlined perception system designed to bolster autonomous luggage trolley handling by robots, particularly in cluttered or partially occluded airport environments. Recognizing that conventional methods often falter due to dependency on full object visibility and extensive 3D data, the authors split the perception task into two distinct stages. First, a hierarchical keypoint-based detection identifies a single robust keypoint in 2D RGB images, enabling accurate trolley positioning even when the object is occluded. Next, orientation estimation is separately inferred once sufficient visual cues are available. This hierarchical breakdown allows for lighter training requirements and labels based purely on 2D keypoints sidestepping the need for complex 3D annotations. Once the robot moves closer, the system applies progressive refinement, continuously updating pose estimates to support precise manipulation and docking. In both simulated and real-world field tests, HPPS significantly outperforms traditional 2D-plus-3D methods, demonstrating enhanced recall under occlusion and improved localization accuracy.

In [5], This paper presents HFC as a low-cost AGV for carrying goods in airports, malls, and homes. It follows users via GPS and Bluetooth, identifying their smartphone among multiple devices. Designed for contactless transport, HFC enhances efficiency in dynamic environments. The paper details its design, sensors, and actuators, highlighting its adaptability and real-world applications.

In [6], Paper described Automated Guided Vehicles (AGVs) improve industrial automation using RFID for real-time tracking and navigation. RFID-based localization enhances path planning and inventory management while reducing reliance on costly sensors. Researchers have integrated RFID with vision systems, LiDAR, and IoT for better accuracy. However, issues like signal interference and limited range require optimization. Recent studies explore AI and machine learning for predictive routing and collision avoidance. Hybrid models combining RFID and AI improve AGV performance in dynamic environments, making them more efficient and reliable.

Chapter 3

Proposed Work

3.1 Problem Statement

Manual transportation of materials in industrial and institutional settings is inefficient, labor-intensive, and prone to human error. Existing autonomous systems are often complex and expensive, making them unsuitable for small-scale or budget-conscious applications. There is a need for a simple, low-cost, and intelligent trolley system that can autonomously follow a path, detect obstacles, and reroute when necessary to ensure uninterrupted movement.

3.2 Proposed Work

The proposed work focuses on designing and implementing an Autonomous Trolley capable of navigating a predefined path using time-based control, while dynamically detecting and avoiding obstacles using a LiDAR sensor. The goal is to provide a simple, low-cost, and intelligent solution for autonomous transport of goods or materials in controlled environments such as campuses, hospitals, warehouses, and manufacturing units.

The trolley is built using the following key hardware components:

ESP8266 Microcontroller: Acts as the brain of the system, controlling motor operations, processing sensor data, managing rerouting logic, and hosting a web-based control interface.

VL53L0X LiDAR Sensor: Used to continuously measure the distance in front of the trolley to detect the presence of obstacles in real-time.

L298 Motor Driver: Controls two DC motors to facilitate movement in forward, backward, left, and right directions.

Buzzer: Provides audible feedback when obstacles are detected or rerouting is initiated.

Battery: Supplies portable power to all electronic components for untethered operation.

The operational logic of the trolley includes the following stages:

Initialization:
The ESP8266 initializes all hardware components, connects to a local Wi-Fi network, and launches a basic web server for manual testing and control.

Path Execution:

The trolley begins moving forward along a predefined route. Instead of relying on sensors for navigation, the movement is controlled based on time durations associated with each direction (e.g., move forward for 5 seconds, turn left for 2.5 seconds, etc.).

Obstacle Detection and Waiting:

During motion, the VL53L0X LiDAR sensor constantly checks for obstacles within a specific range (e.g., 250 mm). If an object is detected, the trolley immediately halts and activates the buzzer. It then enters a 5-second waiting period to allow the object to clear.

Conditional Path Resumption:

If the obstacle moves within 5 seconds, the buzzer stops, and the trolley resumes its original motion from where it left off.

If the obstacle remains, the trolley initiates a predefined rerouting path composed of directional steps (left, forward, right, etc.) executed using specific time durations.

Rerouting Logic:

The reroute sequence is designed to circumvent the obstacle and reconnect to the main path. During this reroute, the LiDAR continues to monitor the path. If another obstacle is encountered, the trolley pauses again and waits until it clears before resuming.

Path Completion:

The trolley continues this process until it completes the total forward movement time originally set (e.g., 20 seconds). Once complete, it stops automatically.

Web Interface:

A simple HTML-based user interface is hosted on the ESP32, allowing the user to manually control the trolley’s movement (Forward, Backward, Left, Right, Stop) during testing or emergencies.

Chapter 4

Hardware & Software Implementation

4.1 Block Diagram

4.2 Design and Components

4.2.1) ESP8266 Microcontroller:

The ESP8266 is a low-cost Wi-Fi-enabled microcontroller developed by Espressif Systems. It is widely used in IoT (Internet of Things) and automation projects due to its compact size, built-in Wi-Fi, and support for digital and analog interfacing. In this project, the ESP8266 acts as the brain of the Autonomous Trolley, controlling the motors, reading the LiDAR sensor data, managing obstacle detection and rerouting logic, and hosting a web interface for manual control.

- **Features of a L298N:**
  - Built-in Wi-Fi Module: Enables wireless control and web server hosting without external modules.
  - 80 MHz (or 160 MHz) CPU: Sufficient processing power for real-time control and sensor interfacing.
  - Flash Memory: Typically comes with 512 KB to 4 MB flash storage for code and data.
  - GPIO Pins: Provides multiple General Purpose Input/Output (GPIO) pins for connecting sensors, motors, and peripherals.
  - Supports I2C, SPI, and UART: Allows interfacing with various digital sensors like the VL53L0X.
  - Supports PWM: Used for controlling motor speed via the L298N driver.
  - Operates at 3.3V Logic: Requires level shifting when interfacing with 5V components.
  - Compact Form Factor: Ideal for embedding in small robotic platforms like trolleys.
  - Low Power Consumption: Efficient for battery-powered systems.
  - Open Source Support: Large community with extensive libraries (Arduino IDE, NodeMCU, MicroPython).

Fig. 2 – ESP8266 Microcontroller

4.2.2) L298N motor driver:

The L298N motor driver is a dual H-bridge IC module that controls the direction and speed of DC motors. In the grass cutter, this module is essential for managing the movement of the wheels and the cutting blade. It receives control signals from the ESP32 and converts them into higher current outputs needed by the motors. Its ability to manage two motors independently or one motor in both forward and reverse makes it suitable for differential drive movement in robotic applications.

- **Features of a L298N:**
  - Dual H-Bridge: The L298N incorporates two H-bridge circuits, allowing it to control two DC motors independently.
  - High Current Capability: It can handle rclativcly high currcnts, making it suitablc for powering various motors.
  - Input Voltage Range: The chip can operate with input voltages ranging from 4.5V to 36V.
  - Logic Voltage Input: It accepts logic-level inputs (typically 5V) from microcontrollers or other control circuits.
  - Enable Pins: Individual enable pins for each H-bridge allow for selective motor control.

Fig.3 – L298N Motor Driver

4.2.3) DC Motors (4 for Wheels)

DC motors drive the wheels of the grass cutter, allowing it to move in different directions. These motors are easy to control, dependable, and affordable. You can manage their speed and rotation direction using the L298N motor driver with PWM signals from the ESP32. Using four motors provides stability, improves load handling, and allows for smoother movement on uneven surfaces.

A high-speed DC motor rotates the cutting blade. This motor carries out the cutting task in the grass cutter. It usually connects through a relay module, which enables the microcontroller to safely turn the blade on or off. The motor needs to be powerful enough to deal with light to medium grass while staying energy efficient.

- **Features of a DC motor:**
    1. Converts electrical energy into mechanical rotation.
    2. Controlled via PWM for speed variation.
    3. Direction controlled using motor driver.
    4. Four-wheel drive improves mobility and traction.
    5. Suitable for medium-torque applications like grass cutting.
    6. Drives the rotating cutting blade.
    7. Connects via relay for switching control.
    8. Requires high RPM and torque.

.

4.2.4) VL53L0X Sensor

The VL53L0X is a high-accuracy Time-of-Flight (ToF) distance sensor developed by STMicroelectronics. Unlike traditional IR sensors that measure reflected light intensity, the VL53L0X calculates the time it takes for a laser pulse to travel to an object and back, providing precise distance measurements regardless of the object's color or ambient lighting conditions. This sensor is ideal for applications that require reliable and fast obstacle detection, such as autonomous robots and trolleys.

- **Features of a VL53L0X Sensor:**
  - High Accuracy: Measures absolute distance up to 2 meters with millimeter-level precision.
  - Fast Response Time: Capable of reading distance measurements at rates up to 50Hz (20ms intervals).
  - Laser-Based Measurement: Uses a 940 nm Class 1 laser emitter, invisible to the human eye and safe for general use.
  - Time-of-Flight Technology: Measures the actual time taken for light to reflect, offering more consistent readings than analog IR sensors.
  - Compact Size: Small PCB footprint (as small as 5mm x 5mm module), ideal for embedded and space-constrained projects
  - Low Light Performance: Provides reliable distance measurement even in low or no ambient light conditions
  - Continuous and Single-Shot Modes: Supports both real-time continuous reading and single-measurement triggering.
  - I2C Communication: Easily interfaces with microcontrollers like ESP32 via the I2C protocol.
  - Adjustable Ranging Profiles: Allows tuning for longer range, higher speed, or higher accuracy based on application needs.
  - Low Power Consumption: Operates efficiently with minimal current draw, suitable for battery-powered systems.

4.2.5) Battery

The 6V 4.5Ah rechargeable sealed lead-acid (SLA) battery is a commonly used power source in small electronics, robotic systems, emergency lighting, and backup systems. In the Autonomous Trolley project, this battery supplies stable power to the ESP32 microcontroller, motor driver, sensors, and DC motors. Its relatively compact size, high discharge capacity, and reusability make it ideal for mobile and battery-operated applications.

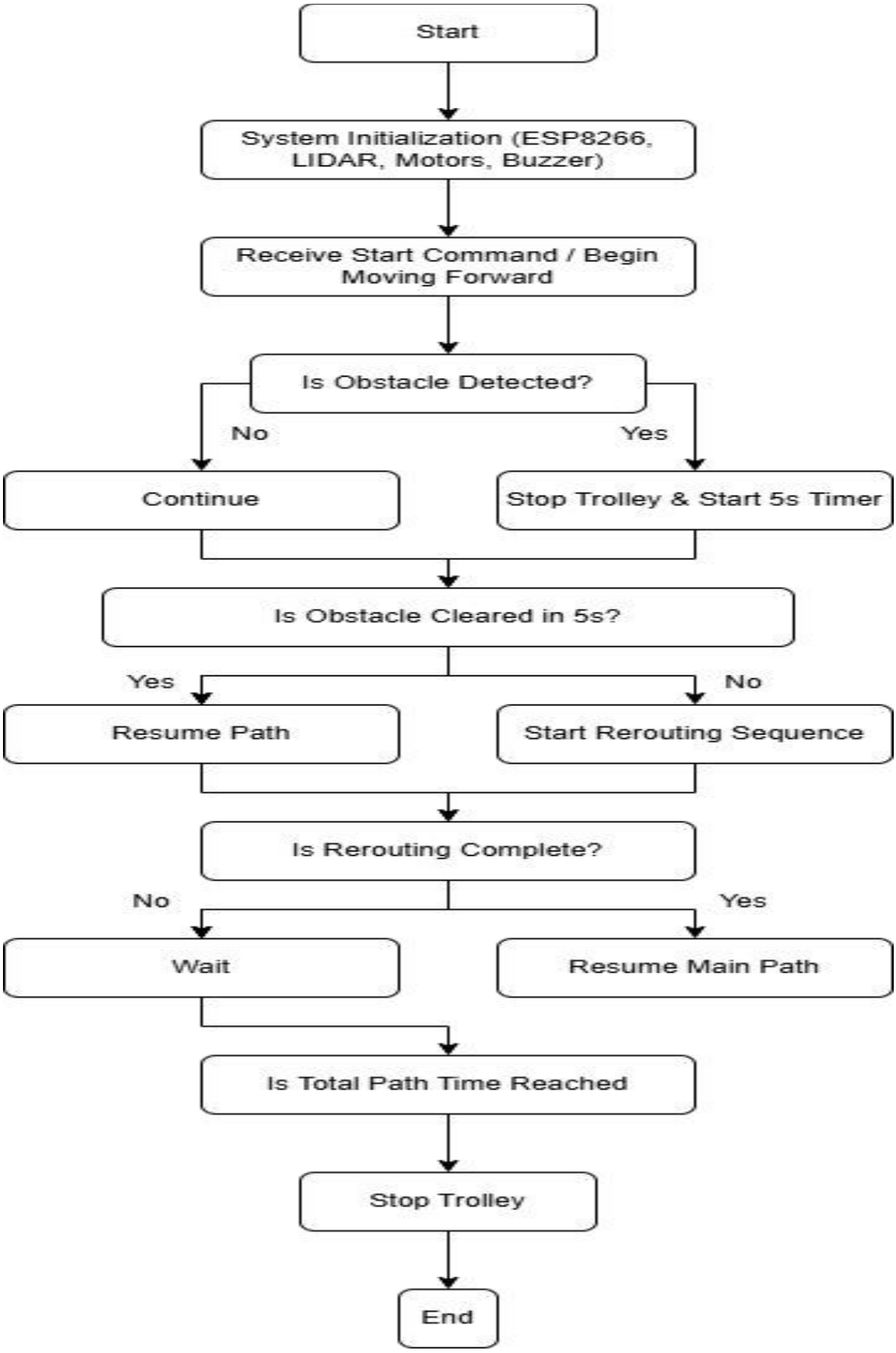
- **Features of a Battery :**
  - Rated Voltage: 6 Volts – suitable for low-voltage embedded systems and motors.
  - Rated Capacity: 4.5 Ampere-hours (Ah) – capable of delivering 4.5 amps for 1 hour or 0.45 amps for 10 hours.
  - Rechargeable: Can be recharged hundreds of times using a compatible DC charger.
  - Sealed Lead-Acid (SLA) Type: Maintenance-free, spill-proof, and safe for horizontal or vertical mounting.
  - Deep Discharge Recovery: Provides stable performance even after partial or full discharge cycles.
  - Compact Design: Fits well in small robotic platforms and mobile electronics.
  - Durable Construction: Enclosed in a hard plastic casing that resists vibration and physical shock.
  - Overcharge Protection (with charger): When used with appropriate smart chargers, it prevents overcharging and overheating.
  - Stable Output: Provides consistent voltage over a large part of the discharge cycle, ensuring reliable operation of sensors and controllers.
  - Application: Widely used in robotics, UPS systems, emergency lighting, medical devices, and mobility equipment.

4.2.6) Buzzer

A buzzer is an electronic sound-emitting device that produces a tone or alert signal. In this project, the buzzer is used to audibly indicate when the trolley encounters an obstacle. It helps in alerting users or bystanders that the trolley is in a paused or rerouting state, enhancing safety and awareness.

- **Features of a Buzzer :**
  - Audio Feedback: Emits a beep or tone during obstacle detection or rerouting.
  - Low Power Consumption: Typically operates on 3V–5V, ideal for microcontroller-based systems.
  - Compact Size: Easy to integrate into small enclosures or trolley chassis.

4.) Flowchart :


Start
System Initialization (ESP8266, LIDAR, Motors, Buzzer)
Receive Start Command / Begin Moving Forward
Is Obstacle Detected?
No
Yes
Continue
Stop Trolley & Start 5s Timer
Is Obstacle Cleared in 5s?
Yes
No
Resume Path
Start Rerouting Sequence
Is Rerouting Complete?
No
Yes
Wait
Resume Main Path
Is Total Path Time Reached
Stop Trolley
End

4.) Algorithm:

Step 1: Start the system.

Step 2: Initialize components:

- ESP8266 microcontroller
- VL53L0X LiDAR sensor
- L298N motor driver and motors
- Buzzer

Step 3: Receive start command or activate forward motion.

Step 4: While the total path time is not completed, repeat steps below:

Step 4.1: Check for obstacle using LiDAR sensor.

Step 4.2: If no obstacle is detected:

→ Continue moving forward.

Step 4.3: If obstacle is detected:

→ Stop the trolley.

→ Start a 5-second timer.

Step 4.4: If the obstacle clears within 5 seconds:

→ Resume forward movement from paused location.

Step 4.5: If the obstacle does not clear:

→ Start rerouting sequence (timed left/right/forward movements).

Step 4.6: During rerouting, if another obstacle is detected:

→ Wait until the new obstacle is cleared.

→ Resume reroute step after clearance.

Step 4.7: After rerouting is complete:

→ Rejoin main path and continue movement.

Step 5: If total forward movement time (e.g., 20 seconds) is reached:

→ Stop the trolley.

Step 6: End the process.

4.) Project code

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include <Wire.h>
#include <Adafruit_VL53L0X.h>

// WiFi credentials
const char* ssid = "Shubham";
const char* password = "12345678";

// Motor pins
#define IN1 D5
#define IN2 D6
#define IN3 D7
#define IN4 D8

// Buzzer pin
#define BUZZER_PIN D4

ESP8266WebServer server(80);
Adafruit_VL53L0X lox = Adafruit_VL53L0X();

bool isMoving = false;
bool rerouting = false;
bool waitingInitial = false;
bool pausedDuringReroute = false;
bool lidarDisabledDuringReroute = false;
bool countingForward = false;
unsigned long lastSensorCheck = 0;
```

```
unsigned long initialWaitStart = 0;
unsigned long rerouteStartTime = 0;
unsigned long initialForwardStartTime = 0;
unsigned long initialForwardDuration = 0;
unsigned long lidarDisableStartTime = 0;
int rerouteStep = 0;

const unsigned long sensorInterval = 200;
const unsigned long targetForwardTime = 20000;
const unsigned long initialWaitTime = 5000;
const unsigned long lidarIgnoreTime = 5000;

// Updated durations (8 steps)
unsigned long stepDurations[] = {
  3000, // Placeholder
  3000, // 1: left()
  3000, // 2: forward()
  4200, // 3: right()
  5000, // 4: forward()
  4200, // 5: right()
  3000, // 6: forward() ← added
  3000, // 7: left()
  3000, // 8: forward()
  0     // End
};

// Motor control
void forward() {
  digitalWrite(IN1, HIGH); digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH); digitalWrite(IN4, LOW);
}
void backward() {
  digitalWrite(IN1, LOW); digitalWrite(IN2, HIGH);
  digitalWrite(IN3, LOW); digitalWrite(IN4, HIGH);
```

```
}
void left() {
  digitalWrite(IN1, HIGH); digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW); digitalWrite(IN4, HIGH);
}
void right() {
  digitalWrite(IN1, LOW); digitalWrite(IN2, HIGH);
  digitalWrite(IN3, HIGH); digitalWrite(IN4, LOW);
}
void stopMotors() {
  digitalWrite(IN1, LOW); digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW); digitalWrite(IN4, LOW);
}

void resumeCurrentRerouteStep() {
  switch (rerouteStep) {
    case 1: left(); break;
    case 2: forward(); break;
    case 3: right(); break;
    case 4: forward(); break;
    case 5: right(); break;
    case 6: forward(); break; // ← added
    case 7: left(); break;
    case 8: forward(); break;
    default: stopMotors(); break;
  }
}

String webPage = R"rawliteral(
<!DOCTYPE html><html>
<head><title>Robot Control</title></head>
<body style='text-align:center; font-family: Arial;'>
<h1>NodeMCU Robot Control</h1>
```

```
<button onclick="location.href='/forward'" style='width:100px;height:50px;'>Forward</button><br><br>
<button onclick="location.href='/left'" style='width:100px;height:50px;'>Left</button>
<button onclick="location.href='/stop'" style='width:100px;height:50px;'>Stop</button>
<button onclick="location.href='/right'" style='width:100px;height:50px;'>Right</button><br><br>
<button onclick="location.href='/backward'" style='width:100px;height:50px;'>Backward</button>
</body></html>
)rawliteral";

void setup() {
  Serial.begin(115200);
  pinMode(IN1, OUTPUT); pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT); pinMode(IN4, OUTPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  digitalWrite(BUZZER_PIN, LOW);
  stopMotors();

  Wire.begin(D2, D1);
  if (!lox.begin()) {
    Serial.println("VL53L0X failed!");
    while (1);
  }

  WiFi.begin(ssid, password);
  Serial.print("Connecting");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500); Serial.print(".");
  }
  Serial.println("\nConnected. IP:");
  Serial.println(WiFi.localIP());

  server.on("/", []() {
```

```
  server.send(200, "text/html", webPage);
});
server.on("/forward", []() {
  isMoving = true;
  rerouting = false;
  waitingInitial = false;
  rerouteStep = 0;
  pausedDuringReroute = false;
  countingForward = true;
  initialForwardStartTime = millis();
  forward();
  server.send(200, "text/html", webPage);
});
server.on("/stop", []() {
  isMoving = false;
  rerouting = false;
  waitingInitial = false;
  pausedDuringReroute = false;
  stopMotors();
  countingForward = false;
  initialForwardDuration = 0;
  initialForwardStartTime = 0;
  digitalWrite(BUZZER_PIN, LOW);
  server.send(200, "text/html", webPage);
});
server.on("/backward", []() {
  isMoving = false; backward();
  server.send(200, "text/html", webPage);
});
server.on("/left", []() {
  isMoving = false; left();
  server.send(200, "text/html", webPage);
});
server.on("/right", []() {
```

```
    isMoving = false; right();
    server.send(200, "text/html", webPage);
  });

  server.begin();
}

void loop() {
  server.handleClient();
  if (!isMoving) return;

  unsigned long now = millis();

  if (countingForward) {
    initialForwardDuration += now - initialForwardStartTime;
    initialForwardStartTime = now;
  }

  if (pausedDuringReroute) {
    VL53L0X_RangingMeasurementData_t measure;
    lox.rangingTest(&measure, false);
    if (measure.RangeStatus != 4 && measure.RangeMilliMeter > 250) {
      Serial.println("Obstacle cleared, resuming reroute...");
      pausedDuringReroute = false;
      digitalWrite(BUZZER_PIN, LOW);
      rerouteStartTime = millis();
      resumeCurrentRerouteStep();
    } else {
      stopMotors();
      return;
    }
  }

  if (rerouting) {
```

```
if (lidarDisabledDuringReroute && now - lidarDisableStartTime >= lidarIgnoreTime) {
  lidarDisabledDuringReroute = false;
}

if (!pausedDuringReroute) {
  if (now - rerouteStartTime >= stepDurations[rerouteStep]) {
    rerouteStep++;
    rerouteStartTime = now;
    countingForward = false;

    switch (rerouteStep) {
      case 1: left(); break;
      case 2: forward(); break;
      case 3: right(); break;
      case 4: forward(); countingForward = true; initialForwardStartTime = now; break;
      case 5: right(); break;
      case 6: forward();break;
      case 7: left(); break;
      case 8: forward(); countingForward = true; initialForwardStartTime = now; break;
      default:
        rerouting = false;
        rerouteStep = 0;
        forward();
        countingForward = true;
        initialForwardStartTime = now;
        Serial.println("Rerouting complete.");
        return;
    }
  }

  if (!lidarDisabledDuringReroute && now - lastSensorCheck >= sensorInterval) {
    lastSensorCheck = now;
    VL53L0X_RangingMeasurementData_t measure;
    lox.rangingTest(&measure, false);
```

```
      if (measure.RangeStatus != 4 && measure.RangeMilliMeter < 250) {
        Serial.println("Obstacle in reroute path, pausing...");
        pausedDuringReroute = true;
        stopMotors();
        digitalWrite(BUZZER_PIN, HIGH);
        return;
      }
    }
  }
  return;
}

if (waitingInitial) {
  if (now - initialWaitStart >= initialWaitTime) {
    VL53L0X_RangingMeasurementData_t measure;
    lox.rangingTest(&measure, false);
    if (measure.RangeStatus != 4 && measure.RangeMilliMeter < 250) {
      Serial.println("Object still present after 5s, start rerouting...");
      rerouting = true;
      waitingInitial = false;
      lidarDisabledDuringReroute = true;
      lidarDisableStartTime = millis();
      rerouteStartTime = now;
      rerouteStep = 0;
      stopMotors();
      digitalWrite(BUZZER_PIN, LOW);
    } else {
      Serial.println("Object moved, resuming forward...");
      forward();
      initialForwardStartTime = now;
      waitingInitial = false;
      countingForward = true;
      digitalWrite(BUZZER_PIN, LOW);
    }
```

```
    }
    return;
  }

  if (now - lastSensorCheck >= sensorInterval) {
    lastSensorCheck = now;
    VL53L0X_RangingMeasurementData_t measure;
    lox.rangingTest(&measure, false);
    if (measure.RangeStatus != 4 && measure.RangeMilliMeter < 250) {
      Serial.println("Obstacle detected, waiting 5s...");
      stopMotors();
      countingForward = false;
      initialForwardDuration += now - initialForwardStartTime;
      waitingInitial = true;
      initialWaitStart = now;
      digitalWrite(BUZZER_PIN, HIGH);
    }
  }

  if (!rerouting) {
    if (initialForwardDuration >= targetForwardTime) {
      Serial.println("Target reached.");
      stopMotors();
      isMoving = false;
      countingForward = false;
      digitalWrite(BUZZER_PIN, LOW);
    }
  }
}
```

4.) Working

The Autonomous Trolley is designed to navigate along a predefined path using a time-based movement logic. The core of the system is an ESP8266 (NodeMCU) microcontroller which receives input from a VL53L0X LiDAR sensor and controls the direction and motion of the DC motors via the L298N motor driver module. The trolley can detect obstacles in its path and make intelligent decisions to wait, reroute, or continue based on sensor input.

The working can be broken down into the following steps:

1. Initialization:
   • The system powers on and initializes the sensor, motor driver, buzzer, and Wi-Fi (for optional web-based control).
   • The trolley is in idle state, waiting for a "start" command either from the onboard code or web interface.

2. Forward Motion:
   • When the "forward" command is given, the trolley starts moving forward.
   • A timer counts the forward motion until a total of 20 seconds of movement is completed (excluding pause time).
   • The VL53L0X sensor continuously checks for obstacles in front.

3. Obstacle Detection and Wait:
   • If an object is detected within 250 mm:
      – The trolley stops immediately.
      – The buzzer turns ON to indicate obstruction.
      – It waits for up to 5 seconds to check if the object moves away.
   • If the obstacle clears during this time, the trolley resumes movement from where it left off.

4. Rerouting:
   • If the obstacle does not clear after 5 seconds:
      – The buzzer turns OFF.
      – The trolley initiates a predefined reroute sequence consisting of left/right/forward movements using step timing.
      – The route is designed to bypass the obstacle and return to the main path.

5. Dynamic Reroute Management:

• If another obstacle appears during the reroute:

– The trolley pauses the current reroute step.

– The buzzer turns ON again.

– Once the path is clear, the reroute step resumes from where it was paused.

6. Completion:

• After completing the total 20 seconds of forward movement (including counted segments during reroute), the trolley stops completely.

• Motors are turned off and buzzer is silent.

7. Web Interface (Optional):

• The ESP8266 hosts a simple web page through which users can manually control the trolley (Forward, Left, Right, Stop, Backward).

• This is useful for testing, calibration, or manual override.

This working mechanism enables the trolley to move autonomously while intelligently handling dynamic obstacles, making it suitable for indoor transport in smart environments like hospitals, warehouses, or campuses.

Chapter 5

RESULT AND DISCUSSION

The Autonomous Trolley successfully followed a predefined path using time-based logic and handled obstacles using a VL53L0X LiDAR sensor. When an object was detected, the trolley paused and resumed if the path cleared within 5 seconds; otherwise, it rerouted using a timed directional sequence. The system was able to return to its main path and complete the journey within the intended duration.

Key observations include:

- Accurate obstacle detection and response using LiDAR.
- Effective rerouting and resumption logic.
- Buzzer feedback helped in alerting during pauses or rerouting.
- Web interface enabled easy manual control and debugging.

Challenges included terrain sensitivity, time-based movement inaccuracy, and limited sensing direction. Despite this, the trolley performed reliably and is suitable for indoor applications like warehouses, hospitals, and smart campuses. The project proved that low-cost components can achieve basic autonomous navigation effectively..

Chapter 6
CONCLUSION

The Autonomous Trolley project successfully demonstrated a low-cost, intelligent transport solution capable of following a predefined path while handling real-time obstacles using a VL53L0X LiDAR sensor. By employing time-based logic and a rerouting mechanism, the trolley was able to pause, wait, and bypass obstacles efficiently, then return to its original path. The use of the ESP8266 microcontroller, L298N motor driver, and buzzer enabled smooth motion control and effective user alerts. The project achieved its objective of developing an autonomous system without relying on complex navigation methods like GPS or visual tracking. Although certain limitations such as terrain sensitivity and forward-only sensing were observed, the system performed reliably in controlled indoor environments.

Overall, this project highlights the potential of simple embedded systems to solve real-world automation challenges and can serve as a foundation for future enhancements like multi-directional sensing, adaptive routing, and load-carrying capability.

4.) References

[1] Anxing Xiao, Hao Luan, Ziqi Zhao, Yue Hong, Jieting Zhao, Weinan Chen, Jiankun Wang, Max Q.-H. Meng, “Robotic Autonomous Trolley Collection with Progressive Perception and Nonlinear Model Predictive Control” *2022 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4480–4486.

[2] H. Lim, Y. Kang, C. Kim, J. Kim, B.-J. You “Nonlinear Model Predictive Controller Design with Obstacle Avoidance for a Mobile Robot” *2008 IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA).*

[3] I. Sánchez, A. D’Jorge, G. Raffo, A. H. González, A. Ferramosca, “Nonlinear Model Predictive Path-Following Controller with Obstacle Avoidance” *Journal of Intelligent & Robotic Systems, Vol. 102, Article 16.*

[4] Z. Sun, Z. Zhang, J. Zhao, H. Ye, J. Wang, “HPPS: A Hierarchical Progressive Perception System for Luggage Trolley Detection and Localization at Airports” *CoRR, arXiv:2405.05514*, May 2024.

{5] Weinan Chen, G. Raffo,“AGV Localization and Navigation System Using Bluetooth GPS” *Presented in a robotics/AGV conference* – ~2014.

[6] S. Lu, C. Xu, R. Y. Zhong, L. Wang, “A RFID-enabled Positioning System in Automated Guided Vehicle for Smart Factories” *J. Manufacturing Systems*, Vol. 44, July 2017, pp. 179–190.