Register number:113323106005
Name: Arjun Rathinam R
Naan mudhalvan id: aut113323eca03
Department: II YEAR-ECE-A

Index

1.project demonstration……..1

2.project documentation……..1

3.feedback and final adjustment……..2

4.project handover and future works……..2

5.sample program……3

6.output……4

Abstract:

The Autonomous Vehicles and Robotics project focuses on developing intelligent, self-navigating vehicles and robotic systems that can interact autonomously within dynamic environments. In this final phase, the system demonstrates the integration of machine learning, sensor fusion, real-time control, and secure data communication. This document outlines the complete implementation, project demonstration, performance metrics, technical documentation, source code samples, and testing procedures. It serves as the final report summarizing project outcomes and future development paths.

1. Project Demonstration

Overview:

The Autonomous Vehicles and Robotics system will be demonstrated to stakeholders, showcasing autonomous navigation, sensor integration, robotic controls, and system response under varied conditions.

Demonstration Details:

- System Walkthrough: Live walkthrough from path selection to autonomous operation in a test environment.
- Sensor Feedback: Display and analysis of real-time sensor data including LiDAR, GPS, and camera feeds.
- Robotic Actuation: Demonstration of automated steering, acceleration, and braking.
- Performance Metrics: Real-time system monitoring for latency, precision, and obstacle detection.
- Safety Protocols: Demonstration of failsafe modes, emergency stops, and system overrides.

Outcome:

The demonstration will prove the system's capability to perform autonomous functions reliably and safely in controlled settings.

2. Project Documentation

Overview:

Comprehensive documentation is provided detailing system architecture, codebase, AI models, sensor processing units, and user/administrator guidelines.

Documentation Sections:

- System Architecture: Diagrams of autonomous stack including perception, planning, and control layers.
- Code Documentation: Explanation of motion planning algorithms, sensor interfaces, and control logic.
- User Guide: Instructions on initiating the vehicle system, monitoring, and logging operations.
- Administrator Guide: System calibration, firmware updates, and diagnostics tools.
- Testing Reports: Metrics on object detection accuracy, latency, and real-world testing outcomes.

Outcome:

The documentation will serve as a reference for future development and operational deployment.

3. Feedback and Final Adjustments

Overview:

Feedback will be gathered from project evaluators and test users to improve usability, performance, and safety.

Steps:

- Feedback Collection: Surveys and live observation from demonstration attendees.
- Refinement: Adjustments to navigation tuning, sensor calibration, and UX.
- Final Testing: Validation of improvements under final testing scenarios.

Outcome:

The final system will be optimized and validated for extended deployment and reliability.

4. Final Project Report Submission

Overview:

This report summarizes the entire project lifecycle, from concept to full demonstration, including test results and implementation insights.

Report Sections:

- Executive Summary: Project overview and highlights.
- Phase Breakdown: Technical progress in navigation systems, robotics, and integration.
- Challenges & Solutions: Sensor misreads, pathfinding issues, and power efficiency strategies.
- Outcomes: Deployment readiness and system benchmarks.

Outcome:

The submitted report will capture all achievements and prepare for further research or scaling.

5. Project Handover and Future Works

Overview:

Final notes on the system handover, future directions, and scalability considerations.

Handover Details:

• Next Steps: Multi-vehicle collaboration, enhanced AI for traffic interpretation, global mapping, and real-world deployments.

Outcome:

Autonomous Vehicles and Robotics system will be officially handed over with technical references and development roadmaps.

Include Screenshots of source code and Working final project.

Sample program:

```
import random
import time

class SensorModule:
    def __init__(self):
        self.lidar_range = 100  # in meters
        self.gps_coordinates = (0.0, 0.0)

    def read_lidar(self):
        # Simulate distance to obstacle (0 = collision imminent, 100 = clear)
        return random.randint(0, self.lidar_range)

    def read_gps(self):
        # Simulate GPS position update
        lat = self.gps_coordinates[0] + random.uniform(-0.0001, 0.0001)
        lon = self.gps_coordinates[1] + random.uniform(-0.0001, 0.0001)
        self.gps_coordinates = (lat, lon)
        return self.gps_coordinates

class ActuatorModule:
    def steer(self, direction):
        print(f"[Actuator] Steering {direction}")

    def accelerate(self, speed):
        print(f"[Actuator] Accelerating to {speed} km/h")

    def brake(self):
        print("[Actuator] Braking!")
```

```
class AutonomousVehicle:
    def __init__(self):
        self.sensor = SensorModule()
        self.actuator = ActuatorModule()
        self.speed = 0
        self.operational = True

    def make_decision(self, lidar_distance):
        if lidar_distance < 20:
            print("[System] Obstacle detected! Activating emergency brake.")
            self.actuator.brake()
            self.speed = 0
            return "STOP"
        elif lidar_distance < 50:
            print("[System] Obstacle ahead, slowing down.")
            self.speed = 10
            self.actuator.accelerate(self.speed)
            return "SLOW"
        else:
            print("[System] Path is clear, cruising.")
            self.speed = 30
            self.actuator.accelerate(self.speed)
            return "GO"

    def run(self, duration=10):
        print("[System] Starting autonomous navigation.")
        for _ in range(duration):
            if not self.operational:
                print("[System] Vehicle not operational. Halting.")
                break
```

```
            gps = self.sensor.read_gps()
            lidar = self.sensor.read_lidar()
            print(f"[Sensor] GPS: {gps}, LIDAR Distance: {lidar}m")
            action = self.make_decision(lidar)

            if action == "STOP":
                self.operational = False
                print("[Failsafe] System halted due to emergency stop.")
                break

            time.sleep(1)
        print("[System] Navigation session ended.")

if __name__ == "__main__":
    vehicle = AutonomousVehicle()
    vehicle.run()
```

Output:

```
[System] Starting autonomous navigation.
[Sensor] GPS: (0.000086, -6.4e-05), LIDAR Distance: 72m
[System] Path is clear, cruising.
[Actuator] Accelerating to 30 km/h
[Sensor] GPS: (0.000138, -0.000143), LIDAR Distance: 55m
[System] Path is clear, cruising.
[Actuator] Accelerating to 30 km/h
[Sensor] GPS: (0.000217, -0.000159), LIDAR Distance: 38m
[System] Obstacle ahead, slowing down.
[Actuator] Accelerating to 10 km/h
[Sensor] GPS: (0.000305, -0.000207), LIDAR Distance: 21m
[System] Obstacle ahead, slowing down.
[Actuator] Accelerating to 10 km/h
[Sensor] GPS: (0.000349, -0.000219), LIDAR Distance: 14m
[System] Obstacle detected! Activating emergency brake.
[Actuator] Braking!
[Failsafe] System halted due to emergency stop.
[System] Navigation session ended.
```