COLLEGE CODE : 1128

COLLEGE NAME : TJS ENGINEERING COLLEGE

DEPARTMENT : ELECTRONIC AND COMMUNICATION ENGINEERING

STUDENT NM-ID : aut112823106305,
aut23ece39a,
aut23ece41a,
autece43a,
aut112823106302

ROLL NUMBER : 112823106305,
112823106039,
112823106041,
112823106043,
112823106302

DATE : 9/05/2025

COMPLETE THE PROJECT NAME AS

AI-Powered Autonomous Vehicle and Robotics System

SUBMITTED BY

NAME : **Rajesh B,**
Karthick G,
Samuel kirubagaran.J,
Sanjay krishnan.c,
Dinesh.K

MOBILE NUMBER : **9566336823,**
9600861301,
8015464817,
63856 59197,

Phase 4:Performance of the Project
Title: AI-Powered Autonomous Vehicle and Robotics System

Objective:
The focus of Phase 4 is to evaluate and enhance the performance of the AI-Powered Autonomous Vehicle and Robotics System developed in Phase 3. Key goals include improving AI model accuracy, ensuring stable sensor data processing, strengthening cybersecurity under load, optimizing system efficiency, and preparing the system for real-world deployment.

1. AI Model Performance Enhancement

Overview:
The navigation and perception AI models from Phase 3 will be refined using extended datasets and real-world driving data.

Enhancements:

Model Fine-Tuning: Use feedback from simulation and physical tests to retrain the object detection and path-planning models.

Latency Reduction: Optimize inference speed to improve real-time decision-making under operational constraints.

Generalization Improvements: Introduce synthetic and domain-randomized data to improve model robustness in unfamiliar environments.

Outcome:
Improved accuracy and responsiveness of AI navigation decisions in real-time with reduced false detections.

---

2. Control System and Actuation Performance

Overview:
Evaluate and improve the responsiveness and precision of the robotic/vehicle control interface.

Key Enhancements:

Response Tuning: Adjust PID control parameters for smoother actuation.

Redundancy Handling: Ensure seamless transition between AI and manual override modes.

Interface Stability: Monitor actuator feedback loops to ensure accurate translation of AI commands into motion.

Outcome:
Enhanced stability and control precision of autonomous maneuvers across various test scenarios.

---

3. Sensor Data Handling Optimization

Overview:
Improve the performance of the sensor fusion system under real-time and noisy conditions.

Enhancements:

Real-Time Processing: Upgrade sensor drivers and middleware (e.g., ROS) for faster and more reliable data streaming.

Sensor Validation: Run stress tests to assess failure rates, and implement fallback protocols.

Environmental Adaptation: Improve fusion logic to handle adverse conditions (rain, low light, etc.).

Outcome:
Accurate, real-time perception from multi-sensor input, even in variable environments.

---

4. Cybersecurity and Safety Assurance

Overview:
Reinforce data security and operational safety as the system scales for broader deployment.

Key Enhancements:

Advanced Encryption: Implement TLS-based secure communication between onboard and remote modules.

Attack Simulations: Conduct penetration testing and network attack simulations to assess system vulnerabilities.

Fail-Safe Upgrades: Enhance redundancy protocols to handle hardware and communication failures gracefully.

Outcome:
A resilient and secure autonomous system capable of maintaining safe operation under attack or failure.

---

5. System Performance Testing and Evaluation

Overview:
Thorough testing under controlled and semi-controlled environments to assess scalability and readiness for real-world deployment.

Implementation:

Load Testing: Simulate dense traffic and complex navigation scenarios using platforms like CARLA and Gazebo.

Performance Metrics: Collect KPIs such as decision latency, control actuation delay, and navigation success rate.

Feedback Analysis: Gather feedback from beta users and test drivers to refine the system further.

Outcome:
Validated performance under stress conditions, with data supporting improvements in reliability, efficiency, and user experience.

Key Challenges in Phase 4

1. Scalability of AI and Sensor Fusion

Challenge: Maintaining performance under increased data throughput.

Solution: Optimize code, parallelize data handling, and use lightweight models.

2. Cybersecurity Under Load

Challenge: Ensuring secure operation with increased remote interactions.

Solution: Employ real-time encryption and decentralized authentication mechanisms.

3. Control Feedback Consistency

Challenge: Actuator feedback delays under stress.

Solution: Introduce predictive control and dynamic feedback prioritization.

Outcomes of Phase 4

1. Enhanced AI Navigation Accuracy: Smarter and faster route decisions with fewer errors.

2. Stable Robotic Control: Improved actuation fidelity and system responsiveness.

3. Optimized Sensor Processing: Real-time, noise-resilient environmental awareness.

4. Strengthened Security Infrastructure: Operational integrity under real-world stress validated.

5. Deployment-Ready Testing Results: System performance metrics support readiness for full-scale implementation.

```python
import cv2
import numpy as np
import hashlib
import time
from ultralytics import YOLO
from statistics import mean

# Load object detection model
model = YOLO('yolov5s.pt')  # Replace with a fine-tuned model for your use

# Simulated real-time sensor readings
def get_mock_sensor_data():
    return {
        "LiDAR": np.random.rand(5),
        "GPS": (12.9716 + np.random.normal(0, 0.0001), 77.5946 + np.random
        "IMU": {"acceleration": np.random.randn(3), "gyro": np.random.randn
        "CameraFeed": 'test_image.jpg'
    }

# Cybersecurity: Enhanced data hash check with timestamping
def verify_data_integrity(data):
    data_str = str(data) + str(time.time())
    return hashlib.sha512(data_str.encode()).hexdigest()

# AI decision logic
def navigation_decision(objects_detected):
    actions = []
    for obj in objects_detected:
        if obj['name'] in ['car', 'person', 'traffic light']:
            actions.append("Slow Down or Stop")
        else:
            actions.append("Proceed")
    return "Slow Down or Stop" if "Slow Down or Stop" in actions else "Pro

# Performance Tracker
class PerformanceLogger:
    def __init__(self):
        self.decision_times = []
        self.frame_rates = []

    def log(self, decision_time, frame_time):
        self.decision_times.append(decision_time)
        self.frame_rates.append(1.0 / frame_time if frame_time > 0 else 0)

    def report(self):
        return {
            "Avg Decision Time (s)": round(mean(self.decision_times), 4),
            "Avg Frame Rate (fps)": round(mean(self.frame_rates), 2)
        }
```

```
logger = PerformanceLogger()

# Run simulated loop
for _ in range(10):  # Simulate 10 cycles
    start = time.time()

    # Sensor input
    sensor_data = get_mock_sensor_data()
    img = cv2.imread(sensor_data["CameraFeed"])

    # Object detection
    detection_start = time.time()
    results = model(img)
    detection_end = time.time()

    detections = results[0].boxes.data.cpu().numpy()
    names = results[0].names
    objects = [{"name": names[int(det[5])]} for det in detections]

    # Navigation decision
    decision = navigation_decision(objects)

    # Cybersecurity validation
    data_hash = verify_data_integrity(sensor_data)

    # Log performance
    frame_time = time.time() - start
    decision_time = detection_end - detection_start
    logger.log(decision_time, frame_time)

    # Output
    print("\nDetected Objects:", objects)
    print("Decision:", decision)
    print("Sensor Integrity Hash:", data_hash[:10] + "...")
    print("Frame Time:", round(frame_time, 3), "s")

# Report
print("\n--- Performance Report ---")
report = logger.report()
for k, v in report.items():
    print(f"{k}: {v}")
```

Phase 4 Performance Metrics