Autonomous Monitoring Robot Car Using Arduino Uno and Raspberry Pi

Soiyod Sifat Mahmud (2108012), Khandaker Mahthir (2108030),
Ali Hosen (2108039), Jane Alam (2108035),
Sazzadul Islam Shuvo (2108057)

Department of Electronics and Telecommunication Engineering
Chittagong University of Engineering and Technology (CUET)
Chattogram, Bangladesh

***Abstract*—This projects presents the design and implementation of a dual-platform robotic system combining a Bluetooth-controlled robot car with a real-time computer vision module. The mobile base uses an Arduino UNO microcontroller for motor control via L298N driver with wireless Bluetooth commands. The vision system employs a Raspberry Pi 5 with Pi Camera for real-time object detection using MobileNet SSD, streaming annotated video via Flask MJPEG to remote clients. The integrated system demonstrates effective embedded control, wireless communication, and edge-AI processing suitable for advanced robotic applications including surveillance and autonomous navigation.**

***Index Terms*—Arduino UNO, Raspberry Pi, Computer Vision, Object Detection, Bluetooth Control, Embedded Systems, Edge AI**

I. Introduction

Mobile robotics with integrated computer vision represents a significant advancement in embedded systems, enabling autonomous navigation and environmental interaction. This project implements a dual-platform robotic system combining wireless control with real-time visual perception. The Arduino-based mobile platform provides precise motor control, while the Raspberry Pi vision system performs object detection and streaming. This hybrid approach balances computational requirements with real-time responsiveness for practical robotic applications.

II. Related Work

Previous robotic systems have explored various wireless control methods (Bluetooth, Wi-Fi, RF) and vision processing approaches. Traditional systems often used single-board computers for both control and vision, leading to performance bottlenecks. Recent works demonstrate distributed architectures where control and vision run on separate processors. Edge-AI implementations using lightweight models like MobileNet SSD have shown promise for real-time processing on resource-constrained platforms like Raspberry Pi.

III. System Architecture

The system employs a distributed architecture with two main components:

1) **Mobile Platform**: Arduino UNO with L298N motor driver, Bluetooth module (HC-05), and DC motors
2) **Vision System**: Raspberry Pi 5 with Pi Camera v2 (IMX219) running real-time object detection

Both systems operate independently but can be integrated for autonomous operations. The vision system streams processed video via HTTP using Flask MJPEG streaming, accessible from any browser on the local network.

IV. Hardware Components

A. Arduino UNO Platform

The Arduino UNO (ATmega328P) serves as the motor control unit, receiving Bluetooth commands and generating PWM signals for the L298N dual H-bridge driver. Four DC motors provide locomotion with independent left/right control.

B. Raspberry Pi 5 Vision System

The Raspberry Pi 5 with 4GB RAM provides sufficient computational power for real-time object detection. The Pi Camera v2 connects via CSI interface, providing 1080p video capture. The system operates headlessly without display output.

C. Power Management

Separate power systems ensure stable operation: 2-cell lithium battery for motor drivers and 5V USB-C power for Raspberry Pi. This prevents voltage drops affecting computational performance.

V. Computer Vision Implementation

A. Model and Technical Details

The vision system employs MobileNet SSD (Single Shot MultiBox Detector) with OpenCV DNN backend:

- **MobileNet SSD**: Lightweight CNN optimized for embedded devices
- **OpenCV DNN**: Provides inference engine supporting ONNX/TensorFlow models
- **Input Resolution**: 300×300 pixels per frame
- **Object Classes**: 20 classes from PASCAL VOC dataset
- **Confidence Threshold**: 0.5
- **Processing Rate**: 10–15 FPS on Raspberry Pi 5

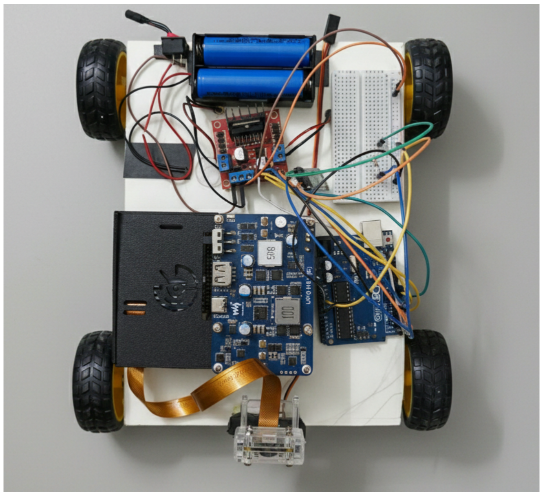

Fig. 1. Complete robotic system with Arduino control board and Raspberry Pi vision module.

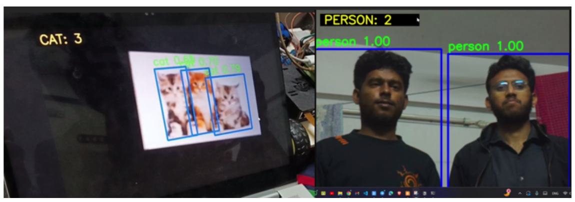

Fig. 2. Real-time object detection output showing bounding boxes and confidence scores for detected objects, specifically identifying a person (man) and a cat.

B. Operation Workflow

The object detection pipeline follows a producer-consumer design:

1) **Frame Capture**: Picamera2 captures frames from Pi Camera
2) **Preprocessing**: Resize and normalize input for MobileNet SSD
3) **DNN Inference**: OpenCV DNN performs forward pass on MobileNet SSD
4) **Post-processing**: Filter detections by confidence and class
5) **Annotation**: Draw bounding boxes and labels
6) **Streaming**: Send annotated frames via Flask MJPEG to browser

Threaded frame capture and throttled detection are used to maintain real-time performance and reduce CPU load.

C. Challenges and Solutions

- **Camera Stack Compatibility**: Used Picamera2 API for libcamera support.
- **Headless Operation**: Flask MJPEG streaming allows remote monitoring.
- **Performance Optimization**: Threaded capture and detection throttling prevent frame drops.

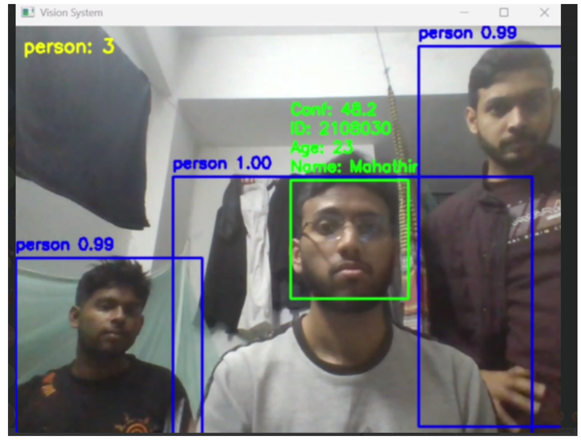

Fig. 3. Multi-object detection with class labeling and counting overlay, showing three detected persons (man) in real time.

VI. Software Implementation

A. Arduino Control System

The Arduino uses a simple character-based protocol: F, B, L, R, S for movement commands. Bluetooth communication operates at 9600 baud rate with SoftwareSerial library.

B. Raspberry Pi Vision Software

Implemented in Python 3, key modules:

- `camera.py`: Picamera2 interface with optimized capture settings
- `object_detector.py`: MobileNet SSD with OpenCV DNN
- `streamer.py`: Threaded frame processing and annotation
- `flask_server.py`: HTTP server with MJPEG streaming endpoint

C. Communication Protocol

Both systems can communicate via serial or network protocols. Detection results can trigger movement commands through a decision layer.

VII. Methodology

Iterative development:

1) Initial prototyping with OpenCV on laptop
2) Porting to Raspberry Pi with architectural redesign
3) Performance optimization for embedded constraints
4) Integration testing under real-time requirements
5) Validation under various lighting and motion conditions

Key design decisions included:

- Distributed architecture separating control and vision
- Throttled detection to balance accuracy and framerate
- Browser-based streaming for remote monitoring
- Modular design for future extensions

VIII. Experimental Results

- **Bluetooth Control**: Reliable up to 10m
- **Object Detection**: 80–85% accuracy on common objects
- **Processing Speed**: 10–15 FPS at 640×480 resolution
- **Streaming Latency**: ¡500ms
- **Power Consumption**: 7W for Raspberry Pi, 15W peak for motors

IX. Applications

- **Surveillance**: Autonomous patrolling
- **Education**: Embedded systems and computer vision platform
- **Research**: Testbed for navigation algorithms
- **Smart Transportation**: Traffic monitoring base

X. Advantages

- Distributed processing avoids bottlenecks
- Browser-based interface eliminates client software
- Modular design allows independent upgrades
- Cost-effective with commodity hardware
- Scalable architecture for additional sensors

XI. Conclusion

This project demonstrates a hybrid robotic system combining wireless control with real-time computer vision. The Arduino platform provides reliable motor control, while the Raspberry Pi enables advanced visual perception. Challenges of embedded deployment including camera compatibility, headless operation, and performance optimization were successfully addressed. Experimental results confirm suitability for education and research applications in mobile robotics.

XII. Future Work

- Vision-based autonomous navigation
- SLAM for mapping and localization
- Object tracking across frames
- Energy optimization for longer battery life
- Cloud integration for remote monitoring and control

References

[1] Arduino, "Arduino UNO Rev3 Datasheet," Arduino.cc.
[2] STMicroelectronics, "L298 Dual Full-Bridge Driver Datasheet."
[3] HC-05 Bluetooth Module Datasheet.
[4] Raspberry Pi Ltd., "Picamera2 Documentation," 2023.
[5] Howard, A. G., et al. "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv:1704.04861, 2017.

.

TABLE I
INDIVIDUAL CONTRIBUTIONS OF GROUP MEMBERS

Name	**ID**	**Contribution**
Soiyod Sifat Mahmud	2108012	Bluetooth communication setup, mobile application testing, and wireless control validation.
Khandaker Mahthir	2108030	Computer vision model integration and optimization, Raspberry Pi configuration, and object detection testing.
Ali Hosen	2108039	Embedded programming, motor control logic implementation, and PWM speed control testing.
Jane Alam	2108035	Hardware assembly, power distribution design, and integration of motor driver with Arduino UNO.
Sazzadul Islam Shuvo	2108057	System integration, Computer vision pipeline implementation, performance optimization, Flask streaming and documentation.

TABLE II
COST ANALYSIS OF THE PROPOSED SYSTEM

Component	**Quantity**	**Cost (BDT)**
Arduino UNO	1	700
Raspberry Pi 5 (4GB RAM)	1	18,000
Pi Camera v2 (IMX219)	1	2,500
L298N Motor Driver Module	1	350
DC Motors	4	250
HC-05 Bluetooth Module	1	300
3.7 V Lithium-ion Batteries	2	1,000
Battery Charging & Protection Module	1	400
UPS HAT (5 V, 6 A) for Raspberry Pi	1	5,000
Raspberry pi Display	1	2500
Miscellaneous (wires, cutters, soldering materials, adhesive tape, mounting tools, power accessories etc)	–	2,000
Total Cost		**33,000 BDT**