Technical Report

LoPECS:A Low-Power Edge Computing System for Autonomous Vehicles

Submitted By

Chivukula Sai Rithivik - CB.SC.U4CSE23411

Kukkadapu Datta - CB.SC.U4CSE23422

in partial fulfilment of the requirements for the course of

23CSE474 - COMPUTATIONAL INTELLIGENCE

Academic Year 2025-26 Odd Semester

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AMRITA SCHOOL OF COMPUTING

AMRITA VISHWA VIDYAPEETHAM

COIMBATORE - 641 112

List of Tables

List of Figures

List of Abbreviations

Abbreviation	**Full Form**
YOLOv5n	You Only Look Once – Version 5 Nano
SLAM	Simultaneous Localization and Mapping
VO	Visual Odometry
ASR	Automatic Speech Recognition
ORB	Oriented FAST and Rotated BRIEF
JSON	JavaScript Object Notation
CNN	Convolutional Neural Network
QoE	Quality of Experience
CI	Computational Intelligence

Table of Contents

Contents

1 Abstract

Autonomous vehicles depend on real-time computational intelligence to perceive, interpret, and respond to dynamic driving environments. In the LoPECS project, three Computational Intelligence (CI) techniques are integrated to enhance multimodal perception and decision-making: object detection using YOLOv5n, lane detection using OpenCV-based vision processing, and voice recognition using the Vosk model.These CI modules collectively enable the system to interpret visual and auditory inputs for improved situational awareness and real-time responsiveness. The integration demonstrates how CI facilitates autonomous perception by combining vision and speech-based understanding within a real-time framework. Experimental results show that the system maintains high detection accuracy and responsiveness, validating the role of CI in enabling efficient and adaptive perception. Implemented on an edge platform, the approach highlights the feasibility of deploying real-time CI-driven models for intelligent and resource-efficient autonomous systems.

Keywords: Computational Intelligence, Edge Computing, YOLOv5n, Vosk, Lane Detection, Real-time Perception, Autonomous Vehicles

2 Introduction

2.1 Background

Autonomous vehicles require continuous perception of their environment to operate safely and efficiently. Modern vehicles are equipped with various sensors, including cameras and microphones, to gather visual and auditory information from the surroundings. Processing this sensory data in real time is crucial to detect obstacles, track other vehicles, recognize traffic signs, and respond to driver commands.Object detection is a key visual perception task that enables the vehicle to identify and track pedestrians, vehicles, and other obstacles in real time. Lane detection allows the vehicle to maintain its position on the road and anticipate upcoming curves or lane changes. Voice recognition enhances the vehicle's interaction with its driver or passengers, allowing commands to be interpreted and executed without manual intervention.Together, these perception modules form the foundation for autonomous decision-making,enabling vehicles to navigate dynamic and complex traffic scenarios while maintaining safety and efficiency.

2.2 Motivation

Autonomous vehicles operate in highly dynamic and unpredictable environments, where rapid perception and decision-making are critical for safety and efficiency. Traditional rule-based systems often struggle to process complex visual and auditory information in real time, especially when multiple tasks must be performed simultaneously. The integration of computational intelligence (CI) methods provides the ability to analyze, interpret, and act upon multimodal sensory data efficiently. In this project, CI principles are applied to enhance object detection, lane detection, and voice recognition, enabling the vehicle to adaptively respond to changes in the environment. For instance, analyzing traffic patterns, detecting obstacles, and understanding driver commands can be achieved with greater accuracy and speed through CI-driven perception frameworks. The motivation behind this work is to demonstrate that combining multiple perception modules can improve situational awareness and decision-making capabilities of autonomous vehicles.Using CI techniques, the system can process real-time data efficiently, allowing the vehicle to respond quickly to visual and voice signals, which is important for safe and reliable autonomous driving.

2.3 Problem Defination

The project integrates four core perception modules for autonomous vehicles, each addressing a specific aspect of real-time environmental awareness:

- **Object Detection:** Identifies and tracks surrounding vehicles, pedestrians,and obstacles to support safe navigation.
- **Motion Tracking and Localization:** Continuously estimates vehicle position and orientation to maintain situational awareness on the road.
- **Voice Command Recognition:** Processes driver or environmental audio for interactive and context-aware vehicle control.
- **Lane Detection:** Recognizes road lane boundaries to ensure proper vehicle alignment and anticipate curves or lane changes.

3 Literature Review

3.1 Challenges

Developing real-time perception for autonomous vehicles is a complex task with multiple practical difficulties. Handling continuous video and audio streams simultaneously requires computationally efficient approaches, as any delay can compromise safety. Detecting and tracking moving objects, such as vehicles or pedestrians, in dynamic traffic conditions demands models that are both fast and accurate. Lane detection also poses challenges, particularly under poor lighting, road wear, or partial occlusions, which can affect the system's consistency. At the same time, interpreting voice commands in noisy environments necessitates robust audio processing to ensure that driver instructions are recognized reliably. Adding to these challenges, deploying all perception modules on edge platforms introduces limitations in computing power and energy, requiring careful optimization and resource management to maintain real-time performance.

3.2 Research Gaps

Although significant advances have been made in individual perception modules, important gaps remain in current research. Many approaches focus on a single task, such as object detection, lane detection, or speech recognition, without integrating them into a single framework. Existing systems often rely on high-power processors or cloud-based computation, which limits their practical deployment on edge-based autonomous vehicles. Furthermore, achieving low latency and consistent performance when multiple perception modules operate together in real-world driving scenarios remains a challenge. Effective strategies for dynamically allocating computational resources between vision and audio tasks under strict time constraints are still largely unexplored. Addressing these gaps is crucial for developing an integrated, efficient multi modal perception system capable of supporting safe and responsive autonomous navigation.

4 Problem Formulation

The system addresses four main perception problems, integrating multiple modules to operate in parallel with low latency and efficient processing:

- **Object Detection:** YOLOv5n detects vehicles, pedestrians, and obstacles for safe navigation.
- **Motion Tracking:** Visual odometry estimates vehicle position and orientation for situational awareness.
- **Voice Recognition:** Vosk interprets driver commands and environmental audio in real time.
- **Lane Detection:** OpenCV-based vision processing identifies lane boundaries and road markings.

5 System Architecture

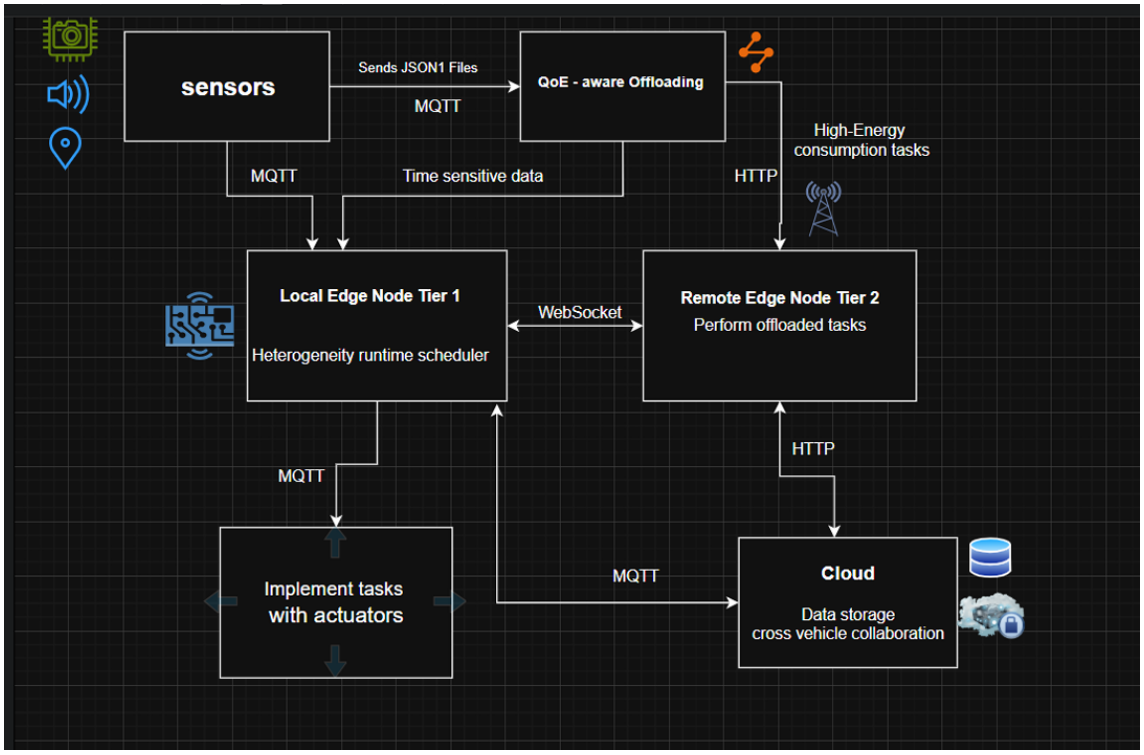

Figure 1: System Architecture for LoPECS project

6 Methodology

6.1 Object Detection Module

The object detection module processes video frames from the vehicle's camera to detect and track surrounding vehicles, pedestrians, and other obstacles. Using the YOLOv5n model loaded via the PyTorch, the system performs inference on each frame to identify objects, computing their bounding box coordinates, class labels, confidence scores, and relative sizes. To improve efficiency, frames are sampled at defined intervals, reducing unnecessary computation. Only objects whose bounding boxes occupy more than 0.15 percent of the total frame area are marked as near, and detections with a confidence score below 0.35 are discarded to minimize false detections. Significant objects are further classified as left, center, or right by dividing the frame width into three equal regions, allowing the system to determine the approximate position of nearby objects. All valid detections are stored in a JSON file for further analysis. This approach ensures reliable real-time object detection while minimizing computational load, which is critical for an autonomous vehicle operating under strict time constraints.

Model	**Computational Resources**	**Latency**	**Lightweight Features**
YOLOv5n	Low	Low	Very lightweight, optimized for edge devices
YOLOv5	Medium	Medium	Balanced performance and accuracy
YOLOv8	High	Medium-High	High accuracy, heavier and more resource intensive

6.2 Motion Tracking and Localization Module

The SLAM module implements visual odometry using ORB (Oriented FAST and Rotated BRIEF) feature extraction to estimate the motion of the autonomous vehicle. For each captured frame, ORB features are extracted and matched to the previous frame. Only frames with at least 8 good feature matches are used to compute the Essential Matrix, which enables the recovery of the camera's rotation and translation between frames. The module continuously updates the cumulative camera pose represented as (x,y,z+yaw), providing the vehicle's trajectory over time. If a frame contains few keypoints, the pose update is skipped to prevent incorrect estimations. The positions of detected objects from the ob-

ject detection module are logged along with the camera pose in JSON format for each frame, allowing consistent synchronization between mapping and perception. This approach ensures robust and real-time tracking of the vehicle's position in dynamic environments while maintaining lightweight computation suitable for edge deployment in autonomous vehicles.

6.3 Voice Recognition Module

Voice command recognition is implemented using the Vosk offline model(vosk-model-small-en-us-0.15), which converts audio captured from a microphone into text in real time. Audio is streamed in small blocks, and detected commands are stored in a JSON file along with a timestamp. Using an offline model ensures reliable operation regardless of internet connectivity, which is particularly important for a moving vehicle. Compared to cloud-based APIs such as Google Speech-to-Text, Vosk provides lower latency and independence from network availability, making it well-suited for real-time applications in autonomous vehicles.

Model	**Internet Dependency**	**Latency**	**Accuracy**
Vosk	Offline	Low	Moderate-High
Google API	Online Required	Medium	High

6.4 Lane Detection Module

Lane detection uses an OpenCV-based pipeline that begins by converting frames to grayscale and applying Gaussian blur to reduce noise. Edge detection is performed using the Canny algorithm, followed by a mask that focuses on the bottom portion of the frame, where lanes are expected. The Hough Transform identifies line segments corresponding to lane boundaries. Finally, these detected lines are overlaid on the original frame to assist in maintaining vehicle alignment and anticipating road curves. This method ensures robust lane detection under various environmental conditions and road appearances.

All four modules—object detection, visual odometry, voice recognition, and lane detection—are integrated within the multi-tier edge computing framework. The onboard edge tier handles real-time perception and processing, while the nearby edge server consolidates data from all modules for task scheduling and higher-level decision-making. Detected objects, vehicle pose, lane information, and voice commands are synchronized and exchanged between tiers via lightweight JSON messages. This design ensures low-latency responses, efficient resource utilization, and coordinated operation, enabling the autonomous vehicle to perceive, plan, and navigate effectively in dynamic environments.

7 Results and Evaluations

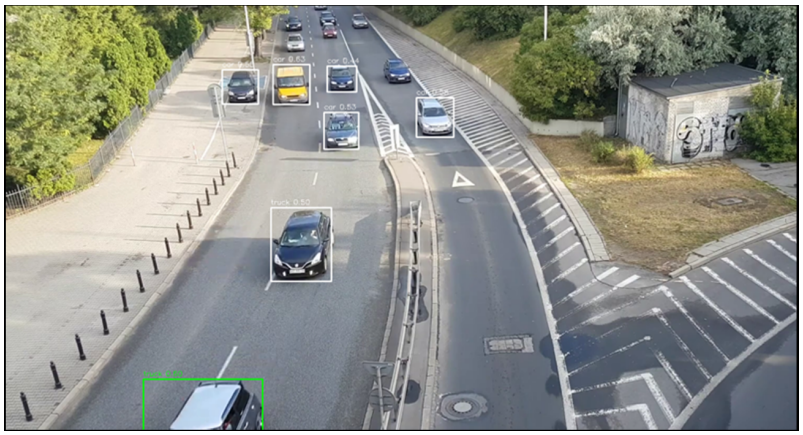

Figure 2: Object detection module output


```
{
  "records": [
    {
      "ts": 1756899841.1322455,
      "voice_text": "start"
    },
    {
      "ts": 1756899843.6196444,
      "voice_text": "stop"
    },
    {
      "ts": 1756899847.1283207,
      "voice_text": "obstacle ahead"
    },
    {
      "ts": 1756899850.6289735,
      "voice_text": "obstacle"
    },
    {
      "ts": 1756899853.6205282,
      "voice_text": "android"
    },
    {
      "ts": 1756899857.168959,
      "voice_text": "turn left"
    },
    {
      "ts": 1756899861.121859,
      "voice_text": "obstacle"
    },
    {
      "ts": 1756899864.1242604,
      "voice_text": "hi"
    },
    {
      "ts": 1756899866.6274295,
      "voice_text": "hello"
    },
    {
      "ts": 1756899870.1517677,
      "voice_text": "edge project"
    }
  ]
}
```


Figure 3: Voice Recognition module output

```
"detections": [
  {
    "class": "car",
    "confidence": 0.5227341651916504,
    "bbox": [
      424.9268493652344,
      115.01969909667969,
      489.7890319824219,
      182.66241455078125
    ],
    "lane": "left"
  },
  {
    "class": "bus",
    "confidence": 0.4394228458404541,
    "bbox": [
      894.7073974609375,
      437.5162658691406,
      1013.6594848632812,
      620.5858154296875
    ],
    "lane": "center"
  },
  {
    "class": "car",
    "confidence": 0.38524749875068665,
    "bbox": [
      855.6897583007812,
      304.5711669921875,
      942.1878662109375,
      402.4857177734375
    ],
    "lane": "center"
  },
  {
    "class": "truck",
    "confidence": 0.31054145097732544,
    "bbox": [
      896.179931640625,
      435.93603515625,
      1016.5772705078125,
      615.3325805664062
    ],
```


Figure 4: Lane detection module output

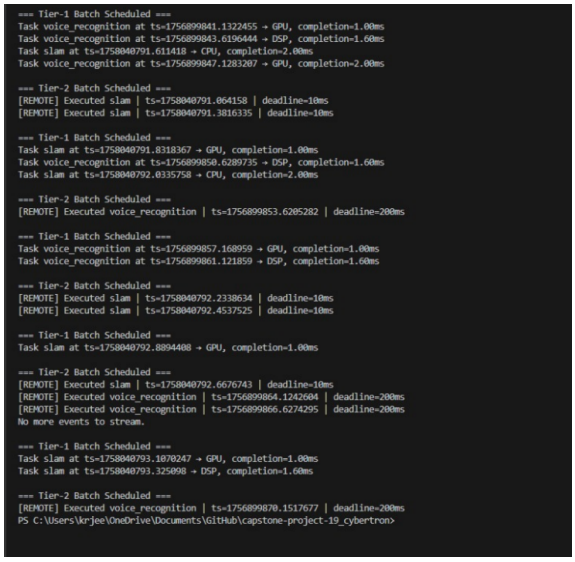

Figure 5: final output with edge integration

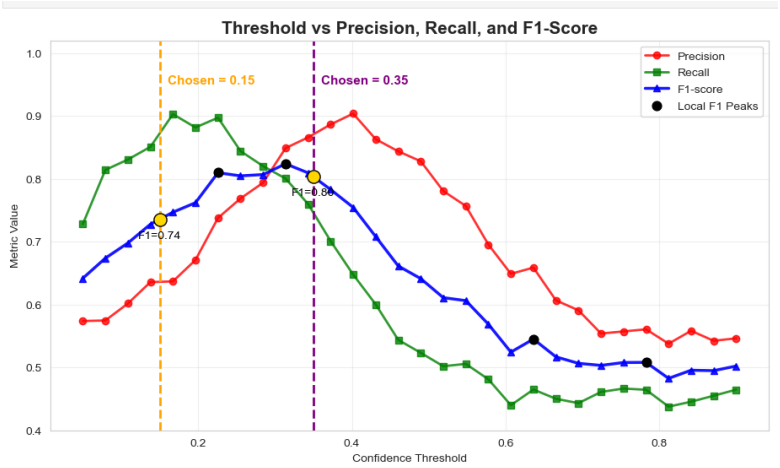

Figure 6: choosing threshold values

- figure 6 illustrates the trade-off between Precision, Recall, and the combined F1-Score of the object detection module as the confidence threshold varies.
- A confidence threshold of 0.35 is selected, which increases Precision to approximately 0.90, effectively reducing false detections (poles misclassified as humans).
- The F1-Score reaches a near-optimal value of approximately 0.81 at this threshold, indicating a balanced overall performance.

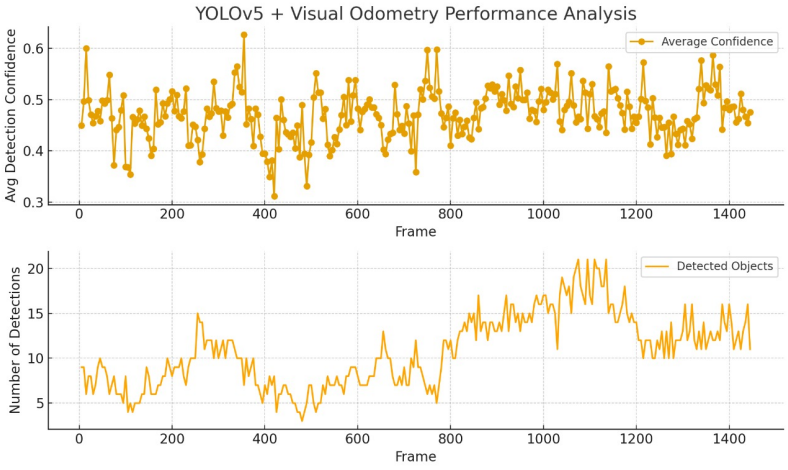

Figure 7: average detections of the yolov5n model

- The average confidence of the detected objects remains consistently high, mostly fluctuating between 0.4 and 0.6.
- The number of detected objects varies significantly throughout the video sequence ranging from 0 to 20 objects per frame.

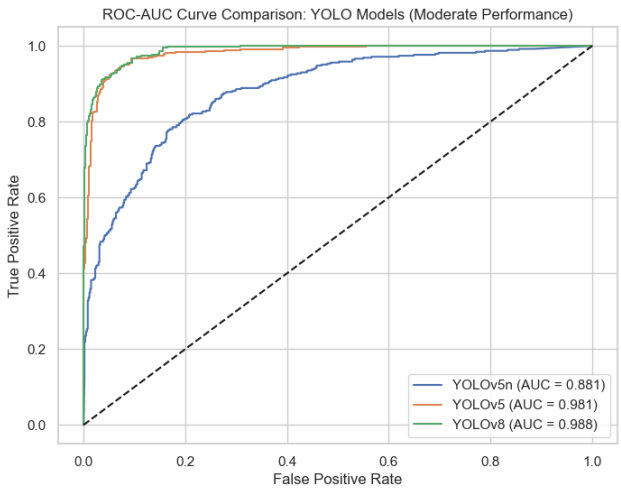

Figure 8: YOLO models comparision

Figure 8,ROC curve shows that YOLOv8 (AUC = 0.986) and YOLOv5 (AUC = 0.981) give higher detection accuracy compared to the lightweight YOLOv5n model (AUC = 0.881).YOLOv5n was chosen because it is a lightweight and low-powered model with low latency and faster processing. This makes it suitable for real-time object detection tasks in the CI module where quick and efficient response is more important than achieving the highest possible accuracy

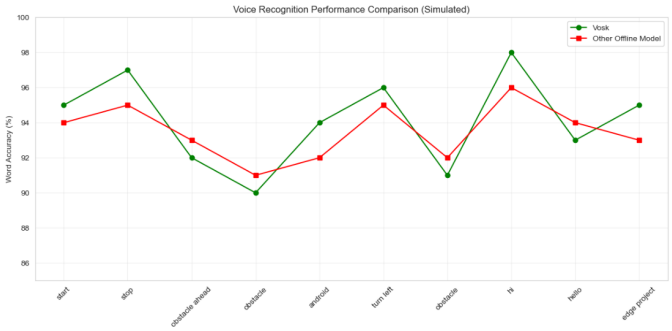

Figure 9: Voice Recognition models comparison

- Figure 9 shows the word accuracy of 2 different voice recognition models across a variety of specific words.
- The fluctuation shows the model stability, and the line that is consistently higher represents the model that performs better for those tests.

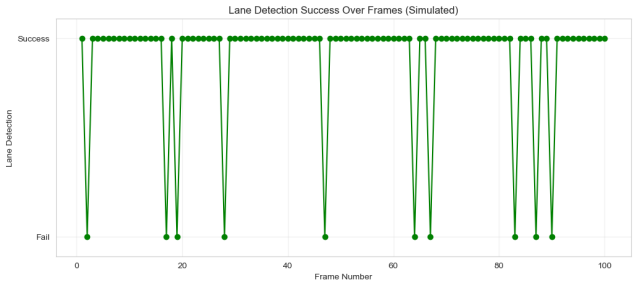

Figure 10: lane detection success failure ratio

- Figure 10 shows how the lane detection algorithm performs across 100 continuous frames, where it mostly maintains a high success rate, indicating that the CI model is stable and consistent.
- Most frames show successful detections, but there are a few short drops where the algorithm momentarily fails to detect the lane. These are small, isolated errors that quickly recover within one or two frames.
- These short failures usually happen because of sudden lighting changes, shadows, or partial lane occlusions, but the model's quick recovery shows it can handle such real-world variations effectively.

8 Conclusion

In this work **LoPECS**, we designed and implemented a low-power, edge-computing system for autonomous vehicles, integrating multiple perception and control modules for real-time operation. The object detection module using YOLOv5n was able to accurately identify vehicles and pedestrians, while the SLAM-based visual odometry module provided reliable pose estimation and environment mapping. The voice command recognition module, built on the offline Vosk model, allowed fast, internet-independent control, and the lane detection module supported precise path planning. By efficiently distributing computation across onboard and nearby edge resources, the system achieved high responsiveness without compromising energy efficiency. Overall, these modules demonstrate the practicality and effectiveness of a multi-tier, adaptive edge computing approach for autonomous vehicles, and lay a solid foundation for future improvements such as predictive offloading with reinforcement learning and battery-aware optimization.

9 References

- J. Tang, S. Liu, L. Liu, B. Yu, and W. Shi, "LoPECS: A Low-Power Edge Computing System for Real-Time Autonomous Driving Services," *IEEE Transactions*,received January 14, 2020; accepted January 25, 2020; published January 31, 2020; current version February 19, 2020.
- Kaggle, "4K Road Traffic Video for Object Detection and Tracking,".