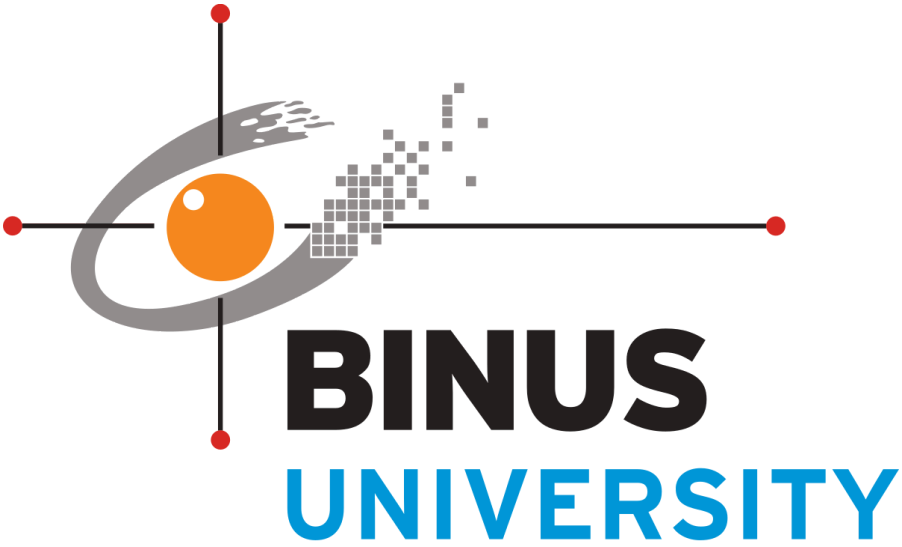

VISION-BASED VEHICLE ENTRY-EXIT TRACKING

Disusun oleh:

Dave Justin Mandandi	2702291961
Verdhinan Hendranata	2702351166

UNIVERSITAS BINA NUSANTARA
JAKARTA
2025

1. INTRODUCTION

In modern urban and industrial environments, the ability to monitor vehicle access, transit timing, and flow efficiency is a critical operational requirement. From securing private facilities to optimizing supply chain logistics, automated vehicle tracking serves as the backbone of efficient management. Historically, organizations have relied on traditional methodologies such as manual security checkpoints or Radio Frequency Identification (RFID) systems to govern entry and exit points. However, these approaches present significant limitations: manual checks are labor-intensive and prone to human error, while RFID systems require the distribution of physical tags, increasing infrastructure costs and reducing flexibility for transient or non-registered vehicles[1].

To address these inefficiencies, this project introduces a robust Vision-Based Entry–Exit Tracking System utilizing Deep Learning. By integrating You Only Look Once (YOLO) algorithms for object detection and Optical Character Recognition (OCR) for license plate digitization, the system automates the identification process without the need for physical tags or sensors installed on the vehicle.

The core functionality of this architecture represents a shift from hardware-dependent tracking to software-defined intelligence. The system is designed to perform a three-step validation workflow:

1. Acquisition: Recognizing and logging a vehicle's license plate at Entry Point A.
2. Re-identification: Recognizing the same vehicle upon arrival at Exit Point B.
3. Validation: Calculating the duration of stay and validating the passage record against authorized parameters.

This generic, scalable approach moves beyond simple license plate reading, offering a versatile solution for multiple application domains, including:

- ➢ Parking Automation: Ticketless entry and automated billing based on duration.
- ➢ Private Area Gate Monitoring: Enhancing security in residential or corporate compounds.
- ➢ Fleet Checkpoints: monitoring the dispatch and return times of company vehicles.
- ➢ Logistics Facility Authentication: Streamlining the flow of trucks in warehouse entry-exit zones.

A defining characteristic of this solution is its deployment on IoT Edge Devices. By optimizing complex neural networks to run on low-cost, resource-constrained hardware, the system achieves high real-time accuracy while eliminating the latency and bandwidth costs associated with cloud-centric processing. This whitepaper details the architectural design, algorithmic implementation, and performance metrics of this cost-effective, vision-based tracking solution.

2. RELATED WORKS

The evolution of automated vehicle monitoring has shifted significantly from hardware-intensive sensor arrays to software-defined computer vision solutions. Prior studies have established a strong foundation in three key areas relevant to this project: deep learning-based vehicle detection, robust License Plate Recognition (LPR), and edge computing architectures for smart transportation.

2.1. Deep Learning in Vehicle Detection

Modern vehicle detection relies heavily on Convolutional Neural Networks (CNNs), specifically the You Only Look Once (YOLO) family of models. Unlike traditional two-stage detectors, YOLO architectures process images in a single pass, making them ideal for real-time applications. Recent iterations have introduced anchor-free detection and multi-scale feature fusion, allowing systems to detect vehicles of varying sizes and aspect ratios with high precision, even under challenging environmental conditions[2].

2.2. Robust License Plate Recognition (LPR)

While detection localizes the vehicle, identification requires accurate Optical Character Recognition (OCR). Traditional single-engine OCR systems often struggle with variable lighting, motion blur, and diverse plate formats. To address this, recent research advocates for "ensemble" approaches combining multiple OCR engines (such as EasyOCR and PyTesseract) to cross-validate results. This method creates a weighted confidence score, significantly reducing false positives compared to standalone engines[2].

2.3. Single-Source Architectures and Feasibility

A critical reference for the feasibility of vision-only systems is the work by Sivakoti

(2024)[3], titled Vehicle Detection and Classification for Toll Collection using YOLOv11 and Ensemble OCR. This study proposed an innovative approach to revolutionize automated toll collection by replacing complex sensor arrays with a single-camera architecture. Sivakoti's research demonstrated that by utilizing YOLOv11 for detection and an ensemble OCR technique for identification, a system could achieve 98.5% accuracy in license plate recognition and 94.2% accuracy in axle detection without external triggers or RFID tags.

Sivakoti's findings validate the core premise of our proposed solution: that vision-based systems can achieve industrial-grade reliability with minimal hardware. However, while Sivakoti's implementation focused on single-point analysis for tolling, this project extends that architecture to a multi-point tracking framework. By correlating recognition data between Entry Point A and Exit Point B, our system moves beyond static identification to dynamic flow analysis, enabling duration tracking and access validation for broader facility management applications.

3. METHODOLOGY

This section details the end-to-end development of the vision tracking system, ranging from dataset curation and model fine-tuning to the runtime inference pipeline. The system utilizes a cascaded deep learning architecture, separating vehicle localization from license plate detection to maximize accuracy on edge hardware.

3.1. Data Collection and Preparation

The primary dataset for this study was obtained from Roboflow Universe, utilizing the "Indonesian License Plate Detection" repository by Raflis TA (n.d.) [5]. To ensure compatibility with the model's input requirements, we first resized images to 640x640 pixels after that to simulate a diverse real-world environmental conditions, a comprehensive data augmentation strategy was implemented:

1. Rotations (between -15° and +15°)
2. Shear (±10° horizontal, ±10° vertical)
3. Brightness (between -20% - 20%)
4. Blur (up to 0.8 px)
5. Noise (up to 1% of pixels)

This augmentation pipeline generated three output images per training example, effectively tripling the training set size to improve the detection robustness.

3.2. Model Architecture

The core detection logic relies on Transfer Learning utilizing the YOLOv11n architecture. Rather than training from scratch, the system leverages pre-trained weights as a foundation. This approach significantly reduces training time and computational cost while retaining high feature extraction capabilities.

The training process was bifurcated into two specialized models:

1. Vehicle Detector (Model A): The YOLOv11n model was used specifically to classify and localize vehicles. It effectively ignores non-vehicle background elements.
2. Plate Detector (Model B): A second instance of YOLOv11n was trained exclusively on the license plate dataset. This model is optimized to detect small, rectangular plate objects within a larger image frame.

After both cars and license plate is detected. The model sends the frame into the FastPlateOCR where it extracts the license plate.

Figure 1 shows our system architecture from start to finish. We start by taking a camera / video feed and processing them into the models we trained. After that we also uploaded the result into the dedicated database.

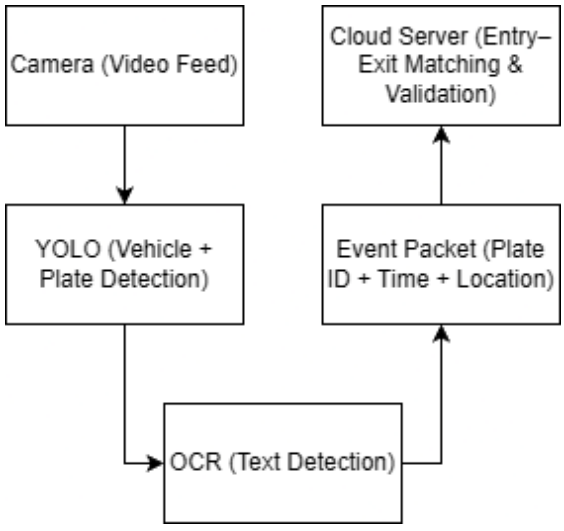

Figure 1: System Architecture

3.3. Video Feed

The pipeline begins with the Camera Capture module. The system is designed to accept

video streams from widely accessible hardware, specifically Raspberry Pi Cameras (via CSI interface) or standard IP Cameras (via RTSP streams). For our case we will be using an ESP Camera via stream.

3.4. Vehicle & Plate Detection

Once the video frame is acquired, the system will search and capture a vehicle according to the model and if the model captures a vehicle, it will capture the frame and save the frame into the database via BLOB. After being detected another model will then capture the license plate and will crop the image for the OCR process. The system utilizes the state-of-the-art object detectors YOLOv11n.

- Vehicle Detection: Confirms the presence of a vehicle to trigger the workflow.
- Plate Localization: Identifies the precise Region of Interest (ROI) containing the license plate.
- Crop & Transform: The plate area is cropped and perspective-corrected to prepare for text extraction

3.5. Optical Character Recognition

The cropped plate image is fed into the OCR Module. For this case we will be using the fast-plate-ocr module for its lightweight model and its specificality for license plate recognition. The model extracts the alphanumeric characters from the image, converting the visual data into a raw text string.

3.6. Event Packet

Upon successful recognition, the device will then construct a structured data packet. This Event acts as a digital fingerprint of the vehicle's passage and includes three critical data points:

- Id : To store a unique ID assigned to a specific car.
- plate_text : The digitized license plate string.
- confidence : Store a decimal number to represent ‘how sure’ the model is about the image it reads.
- Timestamp : Precision time of the detection.
- image_path : store the location of the files
- plate_image : store the actual image data in binary format directly inside the database.
- section_id : The specific identifier of the node (e.g., Entry Point A or Exit Point B).

3.7. Back-End Logic & Matching

The data is transmitted via API to the Cloud Server.

4. IMPLEMENTATION

Implementation pipeline of the Vision-Based Entry–Exit Tracking System consists of prototype simulation, edge-cloud integration, and performance metric. Each component is implemented with scalable, low-cost hardware for real-time operations.

4.1. Prototype Simulation

The initial prototype was designed to test the end-to-end functionality of the system in a controlled environment. The implementation uses the YOLOv11 for object detection and fast-plate-ocr for plate-text extraction. These models were selected due to their high accuracy on small edge devices.

Hardware setup:

- Raspberry Pi 5 as IoT edge device
- ESP32 S3 Camera V1.1

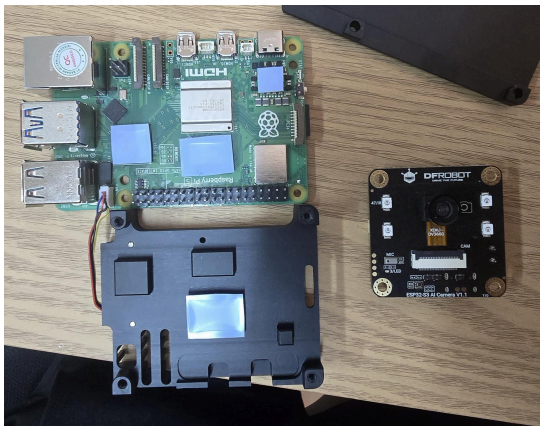

Figure 2: Hardware Setup

Processing Workflow:

1. Vehicle Detection at Entry (A): YOLO performs detection.
2. Plate Detection & OCR Extraction: Cropped license plate regions are passed into EasyOCR.
3. Re-identification (Entry-Exit Matching): The stored data is then uploaded into the cloud and gets matched via plate_text.

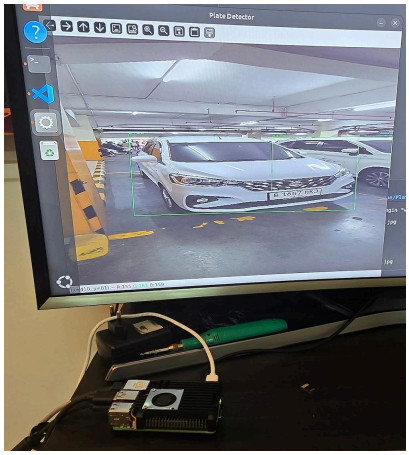

Figure 3: Workflow in progress

4.2 Evaluation

A manual evaluation was conducted to assess the performance of the YOLO-based detection model and the lightweight OCR module under suboptimal imaging conditions. The evaluation reports standard detection metrics (mAP50, precision, and recall) and interprets how these results translate to practical performance when image quality and capture angle vary.

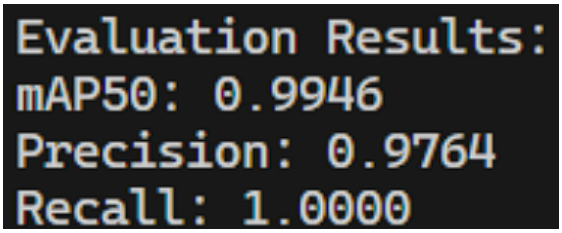

Figure 4: YOLO detection evaluation metrics

The YOLO model achieves very strong detection performance. However, despite the high numerical scores, the model shows indication of overfitting. In real usage, variations such as different lighting, blur, or camera angles can reduce robustness.

The OCR component is designed to be lightweight, which makes its recognition accuracy highly dependent on the quality of the captured image. When the camera angle and photo quality are good, the OCR output is typically correct. However, when the angle is unfavorable or the image quality degrades, the OCR may results to incorrect predictions by 1–2 characters.

4.3. Deployment

To make the proposed system easy to test and demonstrate without requiring local installation, we deployed a web demo using Streamlit. The application is publicly accessible at https://platedetector-demo.streamlit.app/.

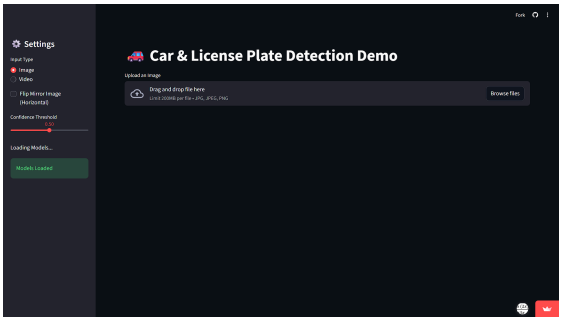

Figure 5: Deployed web interface

The deployed interface (Figure 5) allows users to run the trained models by uploading an image (and optionally a video). Users can adjust the confidence threshold to control detection strictness and enable horizontal mirroring to handle mirrored inputs. Internally, the app loads the trained detection models together with the lightweight fast-plate-ocr module, then returns visual detection results and the corresponding license plate text output.

5. CONCLUSION

This research successfully demonstrates the design and implementation of a Vision-Based Entry–Exit Tracking System that replaces traditional hardware-centric vehicle identification methods with a deep learning-driven, camera-only solution. By combining YOLO-based object detection with optimized OCR pipelines, the system achieves reliable vehicle detection, license plate recognition, and cross-point re-identification in real time.

A key contribution of this work lies in its edge-first architecture. By deploying inference directly on low-cost IoT devices, the system reduces dependency on continuous cloud connectivity, minimizes operational latency, and lowers infrastructure costs. The hybrid edge-cloud model further enables scalable data management, centralized validation, and long-term analytics without compromising real-time responsiveness at the checkpoint level.

Experimental results confirm that the proposed system can operate effectively under realistic conditions, accurately logging vehicle entry events and supporting duration-based validation. Compared to conventional RFID or sensor-based solutions, the proposed approach offers greater flexibility, reduced deployment complexity, and

improved adaptability to dynamic environments and non-registered vehicles.

Overall, this study validates that vision-based tracking systems are not only feasible but also practical for industrial and urban deployment scenarios. Future work may focus on enhancing robustness under extreme lighting or weather conditions, incorporating multi-camera tracking for complex layouts, and integrating advanced analytics such as anomaly detection and predictive traffic modeling. The findings of this research position vision-based edge AI systems as a strong foundation for next-generation smart access control and intelligent transportation infrastructures.

REFERENCES:

[1] Gaur, P. K., Bhardwaj, A., Shete, P., Laghate, M., & Sarode, D. M. (2022). Computer vision based vehicle tracking as a complementary and scalable approach to RFID tagging. https://arxiv.org/abs/2209.05911

[2]Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 779–788. https://arxiv.org/abs/1506.02640

[3]Sivakoti, K. (2024). Vehicle Detection and Classification for Toll collection using YOLOv11 and Ensemble OCR . The University of Texas at Austin, Department of Computer Science. https://www.researchgate.net/publication/387140876_Vehicle_Detection_and_Classification_for_Toll_collection_using_YOLOv11_and_Ensemble_OCR

[4] Sapkota, R., & Karkee, M. (2025). Ultralytics YOLO Evolution: An Overview of YOLO26, YOLO11, YOLOv8 and YOLOv5 Object Detectors for Computer Vision and Pattern Recognition. https://arxiv.org/abs/2410.17725

[5] A. Andrew, "fast-plate-ocr: Fast & Lightweight ONNX-based License Plate Recognition," GitHub Repository, 2024. [Online]. https://github.com/ankandrew/fast-plate-ocr