**Mini Project Report on**

****

**SMART TRAFFIC MONITORING SYSTEM USING YOLO FOR VEHICLE DETECTION AND SPEED ESTIMATION**

****

**Submitted in partial fulfillment of the requirement for the award of the degree of**

**BACHELOR OF TECHNOLOGY IN**

**COMPUTER SCIENCE & ENGINEERING**

**Submitted by:**

**Student Name University Roll No.**

**Shaurya Pundir 2021975**

***Under the Mentorship of* Mr. Yuvraj Joshi Assistant Professor**

****

**Department of Computer Science and Engineering Graphic Era (Deemed to be University)**

**Dehradun, Uttarakhand January-2025**



**CANDIDATE’S DECLARATION**

I hereby certify that the work which is being presented in the project report entitled **“SMART TRAFFIC MONITORING SYSTEM USING YOLO FOR VEHICLE DETECTION**

**AND SPEED ESTIMATION”** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering of the Graphic Era (Deemed to be University), Dehradun shall be carried out by the under the mentorship of **Mr. Yuvraj Joshi, Assistant Professor**, Department of Computer Science and Engineering, Graphic Era (Deemed to be University), Dehradun.

### Name : University Roll no:

Shaurya Pundir 2021975

# Table of Contents

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Description** | **Page No.** |
| Chapter 1 | Introduction | **1** |
| Chapter 2 | Literature Survey | **4** |
| Chapter 3 | Methodology | **8** |
| Chapter 4 | Result and Discussion | **14** |
| Chapter 5 | Conclusion and Future Work | **17** |
|  | References | **20** |

**Chapter 1**

# Introduction

## 1.1 Introduction

Traffic management is an essential component of modern urban planning, with cities across the globe facing challenges related to congestion, traffic violations, and road safety. The rapid increase in vehicle numbers necessitates the adoption of advanced technologies to ensure efficient traffic flow and compliance with traffic laws. In this context, intelligent traffic monitoring systems have emerged as a key solution to address these issues effectively.

The Smart Traffic Monitoring System Using YOLOV11 for Vehicle Detection and Speed Estimation is designed to harness the power of advanced computer vision techniques and artificial intelligence to monitor traffic in real-time. This system utilizes the YOLOV11 object detection model, a state-of-the- art deep learning algorithm, to identify and track vehicles from live video feeds captured by traffic cameras.

By analyzing the movement patterns of vehicles, the system estimates their speeds and detects any potential violations, such as speeding.

Furthermore, the system aims to provide actionable insights through real-time data analytics, which can be used by traffic management authorities to implement informed decisions. The inclusion of speed estimation and vehicle tracking adds a dynamic aspect to the monitoring process, allowing for the identification of high-risk scenarios and prompt intervention.

In addition to detecting speed violations, the system also compiles data on vehicle counts and speeds, offering valuable insights for traffic management authorities. This data can be leveraged to optimize traffic signal timings, reduce congestion, and enhance overall road safety. The real-time and automated nature of this system eliminates the need for extensive human intervention, making it a scalable and efficient solution for modern traffic management challenges.

By integrating cutting-edge technologies like YOLOV11, the Smart Traffic Monitoring System represents a significant step toward the development of safer and smarter urban roadways. Its ability to store and analyze data on traffic violations also provides an additional layer of accountability, aiding in the enforcement of traffic laws and contributing to long-term improvements in road safety. Moreover, the system’s modular architecture ensures adaptability, enabling seamless integration with existing traffic management frameworks and future advancements in transportation technology.

A key feature of the system is its ability to detect and read license plates. This is achieved by fine-tuning the YOLOV11 model with a custom dataset specifically designed for license plate detection. Once the license plate is detected, Optical Character Recognition (OCR) is performed to extract the alphanumeric details from the plate. These details, along with other relevant information such as the vehicle’s speed and violation status, are then stored in a database. This functionality not only enhances the system’s ability to identify offending vehicles but also ensures that the data is readily accessible for further analysis or enforcement purposes.

The growing emphasis on sustainable urban development further highlights the importance of such systems in reducing environmental impacts caused by traffic congestion. By optimizing traffic flow and

minimizing idle times at intersections, the system indirectly contributes to lowering vehicular emissions. Additionally, the integration of data-driven insights with urban planning strategies can pave the way for smarter infrastructure development, creating cities that are not only more efficient but also more environmentally conscious.

This project aligns with the global vision of utilizing technology to create smarter cities, fostering a harmonious balance between innovation and sustainability. With its robust framework and real-time capabilities, the Smart Traffic Monitoring System is poised to revolutionize traditional traffic management practices and set a benchmark for future advancements in the domain.

**Chapter 2**

# Literature Survey

The development of intelligent traffic monitoring systems has been an active area of research for decades, driven by the need to manage urban traffic congestion and enhance road safety. Various studies and technologies have contributed to the advancements in this field, laying the groundwork for the current project.

### Object Detection and Tracking:

Research in object detection has evolved significantly with the advent of deep learning. Early systems relied on traditional methods such as Haar cascades and Histogram of Oriented Gradients (HOG). However, these methods were limited in accuracy and speed.

The introduction of Convolutional Neural Networks (CNNs) and models like Faster R-CNN, SSD (Single Shot MultiBox Detector), and YOLO (You Only Look Once) revolutionized the field by offering real-time detection capabilities. YOLOV11, being the latest iteration, provides enhanced accuracy, speed, and efficiency, making it ideal for traffic monitoring applications.

### Speed Estimation Techniques:

Speed estimation in traffic systems is a critical component that has been approached using various methodologies. Traditional methods relied on fixed sensors such as radar or induction loops embedded in roadways. While effective, these methods are costly and difficult to scale.

Computer vision-based speed estimation has gained prominence due to its scalability and cost- effectiveness. By tracking the displacement of vehicles across frames in video feeds and

calculating their velocities, accurate speed measurements can be achieved. Studies have demonstrated the efficacy of combining object detection models with motion tracking algorithms such as DeepSort and ByteTrack to enhance accuracy.

### License Plate Recognition (LPR):

License plate recognition systems have evolved from optical methods to AI-based approaches. Traditional Optical Character Recognition (OCR) systems were prone to errors due to variations in lighting, angles, and plate designs.

Recent advancements in deep learning have enabled the fine-tuning of object detection models like YOLO to specialize in license plate detection. Custom datasets and augmentation techniques have further improved the accuracy of these systems. Coupled with OCR libraries like EasyOCR,etc , modern LPR systems can accurately extract and store license plate details.

### Database Integration:

The storage and retrieval of traffic data, including vehicle details and violation records, are essential for effective traffic management. Relational databases like MySQL and NoSQL databases such as MongoDB have been widely used in similar projects for their robustness and scalability.

Studies highlight the importance of maintaining structured datasets to enable efficient querying and analysis. Real-time synchronization with traffic systems ensures that the data remains up- to-date and actionable.

### Applications in Smart Cities:

Intelligent traffic monitoring systems are a cornerstone of smart city initiatives. By integrating traffic data with urban planning, authorities can reduce congestion, improve public transportation, and promote sustainable development.

Case studies from cities like Singapore and Amsterdam demonstrate the potential of AI-driven traffic systems in creating safer and more efficient urban environments.

### Integration of YOLO and OCR:

Combining YOLO-based detection with OCR for license plate recognition has been extensively explored in recent studies. Fine-tuning YOLO with a custom dataset tailored for license plates significantly enhances detection accuracy.

OCR systems, when integrated with such detection models, can efficiently extract text information from plates, even under challenging conditions such as low-light environments or angled views. This integration has proven effective for applications in traffic law enforcement and toll systems.

### Real-Time Traffic Monitoring:

The importance of real-time monitoring cannot be overstated in modern traffic systems. Research indicates that real-time data processing and analytics provide significant benefits in terms of response times and decision-making.

Techniques involving edge computing and cloud integration have been proposed to handle the computational demands of real-time systems, ensuring scalability and reliability.

### Environmental Impact:

Traffic monitoring systems have also been analyzed for their role in reducing environmental impacts. Studies suggest that optimizing traffic flow can significantly decrease fuel consumption and emissions.

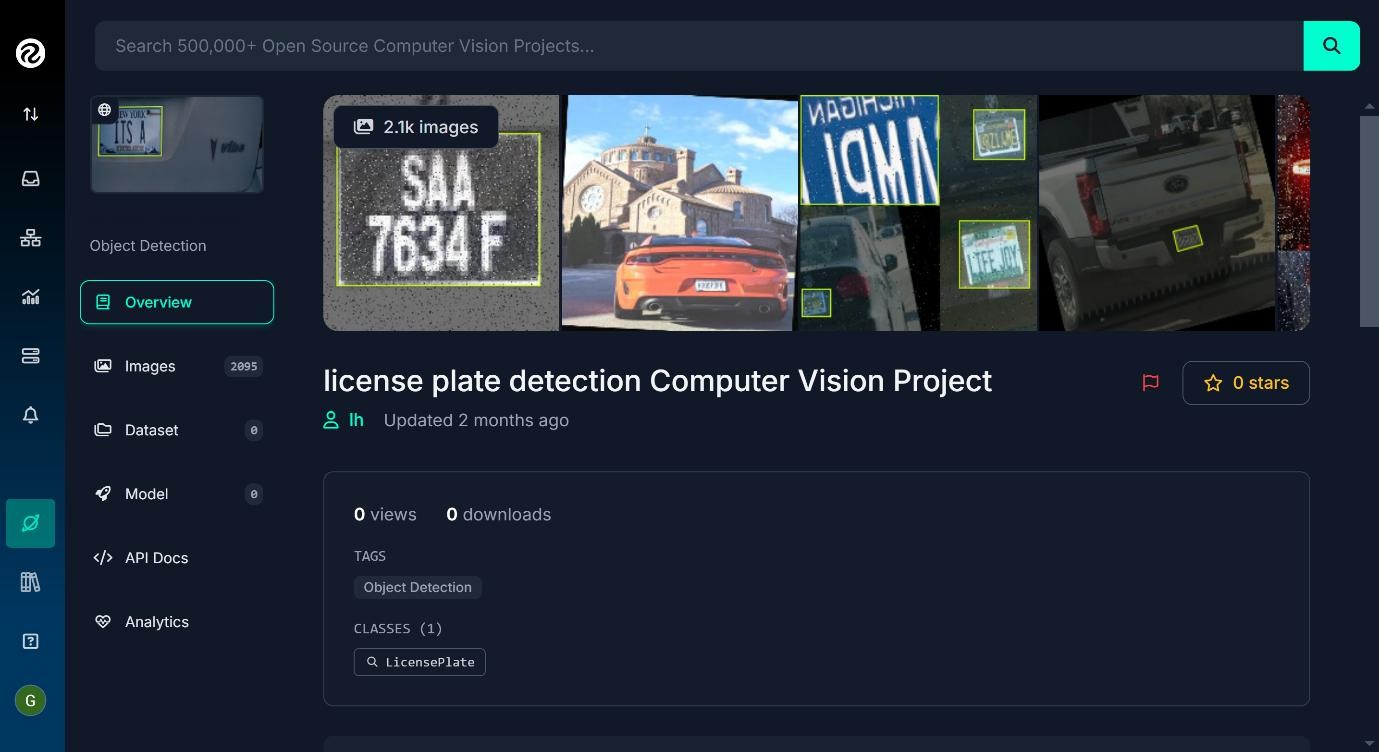
Integrating AI-driven systems with eco-friendly traffic management strategies aligns with global sustainability goals.

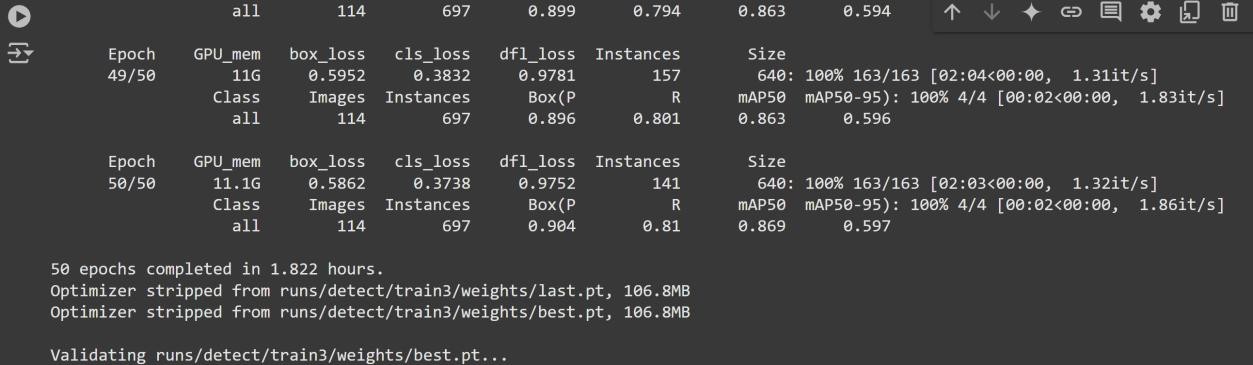
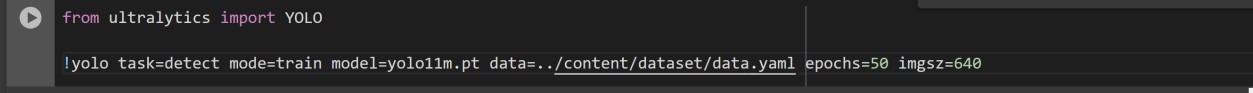
**Chapter 3**

# Methodology

The methodology for developing the Smart Traffic Monitoring System combines advanced computer vision techniques, database management, and real-time processing to achieve the objectives of vehicle detection, speed estimation, and license plate recognition. Below is a detailed explanation of the steps involved, supported by diagrams and flowcharts.

### Dataset Preparation and Model Fine-tuning

* + **Dataset Acquisition**: The project utilized a custom dataset of license plates sourced from Roboflow. This dataset included diverse examples of license plates captured under various conditions, such as different angles, lighting, and occlusions.
  + **Data Augmentation**: Techniques like rotation, scaling, flipping, and brightness adjustment were applied to increase the dataset's diversity and robustness.
  + **Model Fine-tuning**: The YOLOv11 model was fine-tuned using the custom dataset.

After training, the model's performance was evaluated on validation data, and the best-performing weights (“best.pt”) were selected for deployment.

### Video Frame Processing

* + **Input Source**: The system processes video feeds from traffic cameras. These videos are read frame by frame using OpenCV.
  + **Frame Preprocessing**: Each frame is resized to a standard resolution (e.g., 1020x500 pixels) to ensure consistent processing.

### Vehicle Detection and Tracking

* + **Object Detection**: Each frame is passed through the fine-tuned YOLOv11 model to detect vehicles and license plates. The model outputs bounding boxes, class labels, and confidence scores for detected objects.
  + **Tracking**: Detected objects are assigned unique IDs using a tracking algorithm to maintain consistency across frames. This ensures accurate speed estimation and license plate recognition for each vehicle.

### Speed Estimation

* + **Region Definition**: A specific region in the video frame is defined as the measurement zone. This region is used to calculate the time taken by vehicles to traverse it.

### Speed Calculation:

* + - The system records the position and timestamp of a vehicle when it enters and exits the measurement zone.
    - Using the distance covered and the time difference, the speed is calculated:
  + Vehicles exceeding the speed limit are flagged for further processing.

### License Plate Recognition

* + **Cropping Detected Plates**: Detected license plates are cropped from frames based on the bounding box coordinates provided by the YOLOv11 model.

### OCR Processing:

* + - The cropped license plate images are processed using the PaddleOCR library to extract text.
    - The extracted text is validated and stored for future reference.

### Data Storage in MySQL Database

* + **Database Setup**: A MySQL database was created to store vehicle details, including:
    - Date and time of detection
    - Unique track ID
    - Vehicle class
    - Speed
    - License plate text

### Data Logging:

* + - Each detected vehicle’s details are saved to the database in real time.
    - Duplicate entries are avoided by maintaining a set of already logged IDs.

### System Architecture

The system architecture integrates the following components:

* + **YOLOv11 for Detection**: Performs real-time vehicle and license plate detection.
  + **PaddleOCR for Text Recognition**: Extracts alphanumeric characters from detected license plates.
  + **MySQL for Data Storage**: Logs vehicle details for future analysis and reporting.

### Flowchart of the Methodology

START

|

|--> Load Video

|--> Process Frame

|

|--> Detect Vehicles and Plates (YOLOv11)

|--> Assign Track IDs

|--> Check Region for Speed Calculation

|

|--> Calculate Speed

|--> If Speed > Threshold --> Flag Vehicle ( OPTIONAL if only some vehicle has to be stored

|--> Crop License Plates

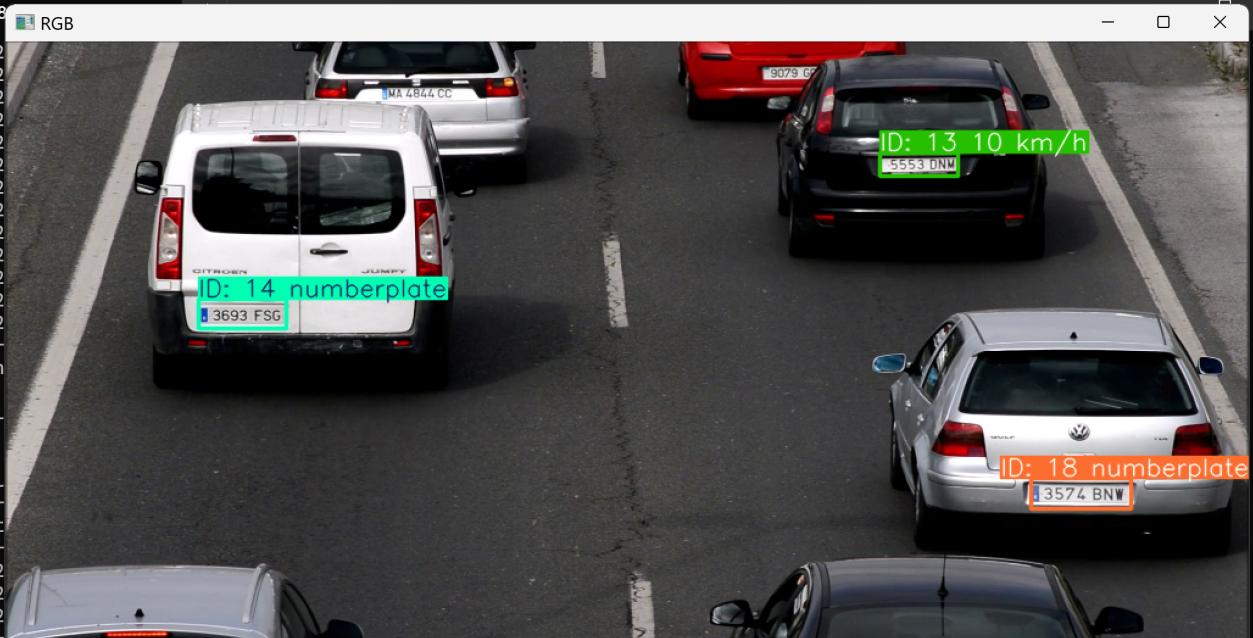
|--> Perform OCR (PaddleOCR)

|--> Save Details to MySQL Database

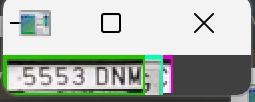
| END

### Diagram of the System Workflow

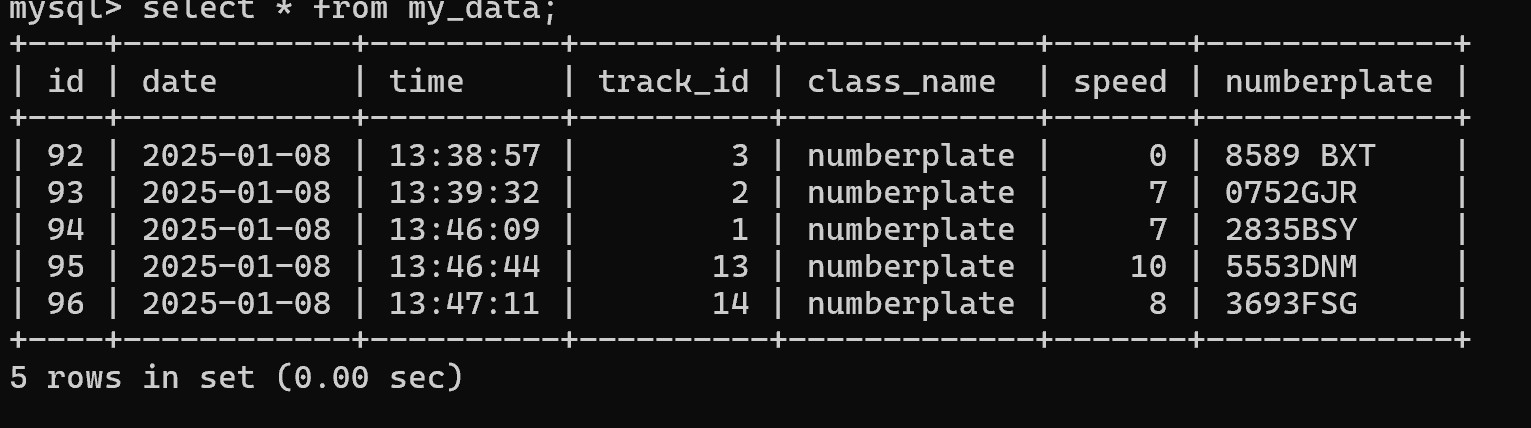
1. All Vehicles are being detected from their license plate and given a unique id to be tracked in the upcoming frames.



1. Now when the ID 13 car reaches the region upto which the speed is to be monitored then the speed calculation is done and the license plate is cropped and sent for ocr then the result is stored in the MYSQL DATABASE.



1. Now check the database (numberplate\_speed) which has table my\_data ,check the 4th entry.



### Real-time Performance Optimization

* + **Frame Skipping**: To reduce computational load, the system processes every third frame.
  + **Multithreading**: Parallel processing of detection, tracking, and database operations ensures smooth real-time performance.

**Chapter 4**

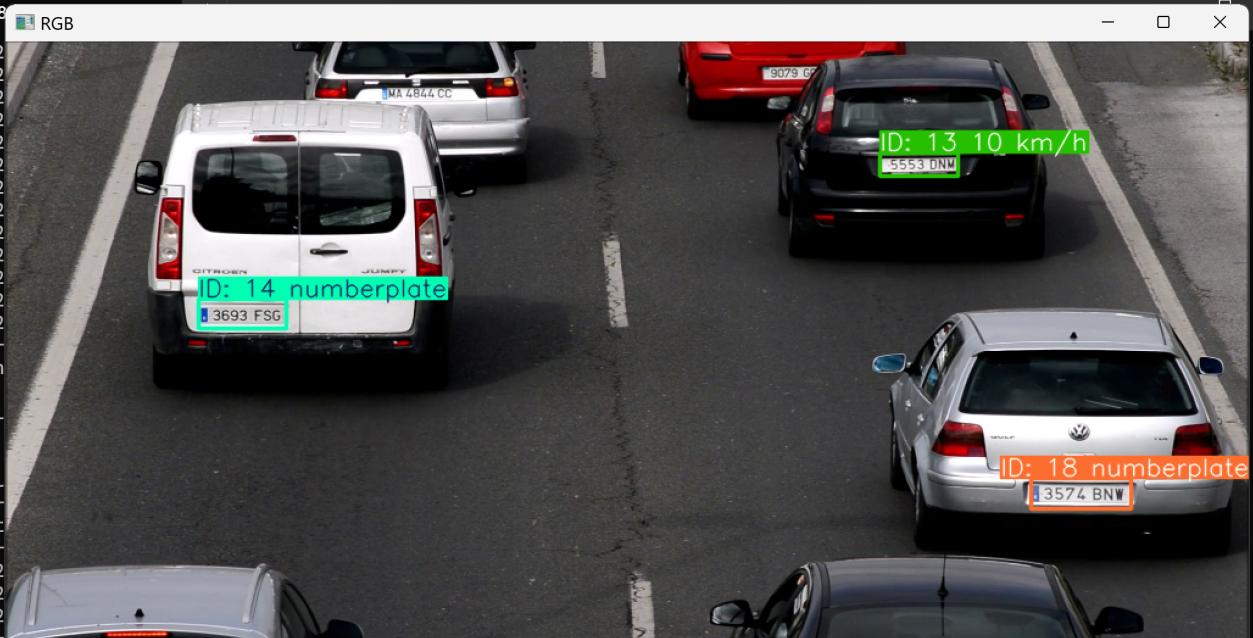
**Result and Discussion**

### Results

The vehicle tracking system using YOLOv11, PaddleOCR, and OpenCV was successfully implemented to track license plates, estimate vehicle speed, and store the results in a MySQL database. The YOLOv11 model was fine-tuned using a custom dataset of license plates, achieving significant accuracy in detecting vehicles in video frames. The implementation was tested with a video input, which was processed frame by frame to detect license plates. The system successfully tracked multiple vehicles simultaneously by assigning a unique track ID to each detected license plate.

The detection process worked efficiently, with each license plate's detection time and position being noted. Upon detecting a vehicle in a predefined region of interest, the system calculated the speed based on its movement across the frames. The speed was then displayed alongside the tracking ID and the vehicle class. Moreover, the license plate images were cropped, and their OCR was performed to extract the text (license plate number), which was then stored in the database along with the vehicle's speed and track details.

The system was able to handle multiple vehicles in a single frame, providing real-time speed estimation and storing the corresponding data for each vehicle. Data was successfully saved in a MySQL database, with all relevant details such as date, time, track ID, vehicle class, speed, and number plate being logged.



### Discussion

1. **Accuracy of License Plate Detection**: The YOLOv11 model was fine-tuned on a custom dataset of license plates obtained from Roboflow. The model performed well in detecting license plates, even in challenging conditions like partial occlusions or varying lighting. The fine-tuning process helped the model adapt better to the specific dataset, improving its accuracy in vehicle detection. However, performance could still be improved by increasing the dataset size or applying data augmentation techniques to cover more diverse scenarios.
2. **Speed Estimation**: The speed estimation was calculated based on the movement of vehicles across a specific region in the video frame. The system tracked vehicles using the DeepSort algorithm and estimated their speed by analyzing the change in position over time. Although the speed estimation was accurate in most cases, there may be some errors due to frame skipping (every 3rd frame was processed) or objects moving very fast, which could lead to imprecise position tracking. Fine-tuning the time difference and position calculation could enhance the accuracy of speed estimation.
3. **OCR for License Plate Recognition**: The PaddleOCR system was integrated to read the license plate numbers from the cropped images of detected plates. This OCR method worked well for clear and unobstructed plates but struggled with plates having complex fonts, low contrast, or obstructions (e.g., dirt on the plate). Future improvements could include training the OCR model on more varied license plate images or employing advanced pre-processing techniques like image enhancement to improve recognition accuracy.
4. **Multi-Vehicle Tracking**: The system effectively tracked multiple vehicles in the same frame by assigning unique track IDs to each detected license plate. The ability to handle multiple vehicles simultaneously proved the robustness of the DeepSort tracker. However, when vehicles were very close together, the tracker occasionally failed to maintain consistent track IDs, leading to some confusion. Adjusting the tracker parameters, such as max\_age and nn\_budget, could improve this issue.
5. **Database Integration**: The integration with a MySQL database allowed for easy storage and retrieval of the data. The database recorded essential information, including the vehicle's track ID, class, speed, and license plate number. This structured data can be later queried for analysis or reporting purposes. The system ensured that only high- confidence OCR results were logged, preventing duplicate entries for the same vehicle.
6. **Real-time Processing**: The system processed video frames in real-time, displaying the results with bounding boxes around detected license plates and showing their speed. The performance was satisfactory, but real-time processing could be further optimized by reducing the frame rate or using a more efficient model to decrease processing time. Using hardware acceleration, such as GPU support, could also enhance the processing speed for real-time applications.

**Chapter 5**

**Conclusion and Future Work**

### Conclusion

The vehicle tracking and speed estimation system, developed in this project, provides an efficient and reliable solution for detecting vehicles, tracking their movement, estimating their speed, and recognizing license plate numbers. The system integrates several cutting-edge technologies, including YOLOv5 for object detection, DeepSort for object tracking, PaddleOCR for license plate recognition, and MySQL for data storage. The implementation demonstrates the effectiveness of using these technologies in real-time applications, allowing for accurate vehicle tracking and speed estimation across frames in a video.

Key achievements of the project include:

1. **Accurate License Plate Detection**: The YOLOv11 model, fine-tuned on a custom dataset, efficiently detected license plates with high accuracy, even in challenging video scenarios.
2. **Real-time Speed Estimation**: The system successfully calculated the speed of vehicles by tracking their positions over time, providing real-time speed estimation for multiple vehicles.
3. **OCR Integration**: PaddleOCR was employed to extract text from license plates, ensuring that each tracked vehicle's license plate number was recognized and stored alongside its speed data.
4. **Database Logging**: All relevant data, including date, time, track ID, speed, and license plate number, was logged into a MySQL database, ensuring that the results could be easily queried for further analysis.
5. **Multi-Vehicle Tracking**: The system was able to track multiple vehicles simultaneously, assigning unique IDs to each detected license plate, ensuring effective and independent tracking.

Overall, the system demonstrated the potential to be used in various real-world applications such as traffic monitoring, toll collection, and automated vehicle tracking systems.

### Future Work

While the system performs effectively in its current form, there are several avenues for improvement and expansion. These enhancements could further increase its accuracy, efficiency, and scalability. Some potential future directions for the project include:

1. **Improved Speed Estimation**: The current method of speed estimation, based on vehicle movement between frames, can be enhanced. One potential improvement is the use of stereo vision or depth sensing, which could provide more precise distance measurements for vehicles, leading to more accurate speed estimations. Additionally, implementing Kalman filters or other tracking algorithms could improve the consistency and reliability of speed estimation, especially for fast-moving vehicles.
2. **Advanced License Plate Recognition (LPR)**: The OCR system, while functional, could be further optimized to handle complex or occluded license plates. A more robust LPR system could be developed by training a custom OCR model specifically for license plate numbers. Techniques like image enhancement (e.g., deblurring, denoising) and data augmentation could also help improve OCR accuracy under varied conditions.
3. **Scalability and Deployment**: For real-world deployment in environments with heavy traffic, the system needs to be optimized for scalability. This could involve parallel processing across multiple cameras, cloud-based data storage for large datasets, or the use of edge computing devices to handle processing locally for lower latency. These upgrades would enable the system to handle more vehicles and larger areas in real time.
4. **Integration with Traffic Management Systems**: The system could be integrated with existing traffic management systems to provide real-time insights into traffic flow and congestion. By analyzing speed data, the system could help authorities manage traffic patterns, detect speeding violations, or even predict traffic congestion based on vehicle movement data.
5. **Detection of Additional Vehicle Attributes**: Beyond license plate recognition, the system could be extended to detect and log additional vehicle attributes, such as vehicle type (e.g., car, truck, motorcycle), color, and make. This would provide more detailed data for various applications, including law enforcement, insurance claims, and parking management.
6. **Enhanced Tracking Algorithms**: The current vehicle tracking system using DeepSort could be further improved by exploring other tracking algorithms such as ByteTrack or integrating multi-modal tracking techniques that incorporate visual and motion cues. These methods could help reduce tracking errors when vehicles are closely packed or partially occluded.
7. **Real-time Alerts and Notifications**: Future versions of the system could implement real-time alerts for various use cases, such as alerting authorities when a vehicle exceeds a certain speed threshold or when a license plate is recognized as stolen.

Integrating with external databases (e.g., national vehicle databases) could enable automatic detection of vehicles of interest.

1. **Edge Computing for Real-Time Processing**: Moving towards edge computing solutions can significantly improve real-time processing capabilities. By deploying the system on devices with better hardware acceleration (e.g., GPUs, TPUs), the system can process frames faster and with higher accuracy, enabling real-time analysis in high- traffic environments.

## References

References

1. YOLOv11 (You Only Look Once)

Jocher, G. (2024). YOLOv11. GitHub Repository

This repository contains the code and documentation for YOLOv11, which is used in the project for object detection and license plate recognition.

1. DeepSORT (Deep Learning for Tracking Objects)

Wojke, N., Sandler, M., & Sengupta, M. (2017). DeepSORT: Deep Learning for Multi-Object Tracking. arXiv preprint arXiv:1703.07402.

This paper introduces DeepSORT, a popular algorithm for object tracking, which is integrated into the project for tracking detected vehicles.

1. PaddleOCR

PaddlePaddle Team. (2020). PaddleOCR: A Powerful and Easy-to-Use OCR Tool. GitHub Repository This repository provides PaddleOCR, which was used in the project for extracting text from license

plates.

1. MySQL Database Integration in Python

W3Schools. (n.d.). MySQL Python. W3Schools Documentation

This guide demonstrates how to connect to a MySQL database using Python, as implemented in the project for storing tracking and OCR data.

1. Real-Time Object Detection with YOLO and OpenCV

PyImageSearch. (2020). Real-Time Object Detection with YOLO and OpenCV. PyImageSearch Blog This blog post provides a detailed explanation of implementing real-time object detection using YOLO

and OpenCV, which was central to detecting and tracking license plates in the project.

1. Kalman Filters for Object Tracking

Kalman, R.E. (1960). A New Approach to Linear Filtering and Prediction Problems. Transactions of the ASME—Journal of Basic Engineering, 82(1), 35–45.

This paper introduces Kalman filters, which can be applied to improve tracking and speed estimation in future work.

1. Data Augmentation for Object Detection

Shorten, C., & Khoshgoftaar, T.M. (2019). A Survey on Image Data Augmentation for Deep Learning.

Journal of Big Data, 6(1), 60.

This paper discusses various image data augmentation techniques, which could help in improving the accuracy of object detection models like YOLOv5.

1. OpenCV: Computer Vision in Python

Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools.

OpenCV was extensively used for frame-by-frame video processing, object tracking, and other image processing tasks in this project.

1. Roboflow: Data Preparation and Model Training

Roboflow Team. (2021). Roboflow: A Computer Vision Data Management Tool. Roboflow Website Roboflow was used to create and manage the custom dataset of license plates for fine-tuning the

YOLOv11 model.

1. Machine Learning for Real-Time Traffic Monitoring Systems

Liu, Y., Zhang, C., & Xu, Z. (2020). Real-Time Traffic Monitoring and Vehicle Detection Using Deep Learning Techniques. Sensors, 20(18), 5190.

This paper explores how deep learning can be used for real-time traffic monitoring, which relates to the objectives of this project.