Contents

1 List of Figures

List of Figures

2 Abstract

The burgeoning necessity for intelligent traffic management and surveillance in the development of smart cities, especially in India, is the driving force behind the project "Speed-Tech." This initiative addresses the critical challenges posed by overspeeding vehicles, emphasizing the detection of moving vehicles, speed estimation, and the identification of speed limit violations with registration numbers. The proposed system employs a sophisticated and efficient approach utilizing a single camera in well-lit environments or daylight conditions. It detects and tracks vehicles within the surveillance area, recording their positions over time. Vehicle tracking relies on the relative positions of vehicles in consecutive frames, providing valuable data for the Automatic Number Plate Recognition (ANPR) System. The integration of cutting-edge technologies, including YOLOv8, Centroid Tracking, Flask, and OCR, enables real-time insights into vehicle speed and identification. Speed-Tech aims to achieve a significant leap in road safety, traffic management optimization, and overall urban transportation efficiency, boasting an average detection accuracy of approximately 87 percent. This project showcases the transformative potential of technology-driven solutions in addressing pressing issues in modern traffic management.

CHAPTER 1
INTRODUCTION

3 INTRODUCTION

3.1 Problem Statement

Modern urban traffic management faces significant challenges, particularly in the detection and control of overspeeding vehicles. The absence of an efficient, automated system leads to increased road safety hazards, traffic congestion, and potential accidents. Traditional methods of speed monitoring and law enforcement are often manual, time-consuming, and lack real-time capabilities.

3.2 Proposed Solution

The project addresses the identified problems by proposing an automated overspeeding detection system using computer vision, machine learning, deep learning, and real-time analytics. Leveraging Haar Cascade Classifier for accurate vehicle detection and Centroid Tracking for object tracking, the system creates a robust foundation for monitoring traffic scenarios. It incorporates YOLOV8 and OCR for license plate recognition, enhancing vehicle identification capabilities.The integration of Flask establishes a user-friendly web application for real-time video streaming, providing users with instant access to traffic data.

The solution includes dynamic vehicle counting, ID annotation, and movement tracking. Entry and exit times are recorded, allowing for comprehensive traffic flow analysis. By calculating vehicle speed and issuing alerts for overspeeding, Speed-Tech aims to contribute to improved traffic management and enhanced road safety. The intuitive web interface ensures accessibility and ease of use, making it a holistic solution for urban traffic challenges.

CHAPTER 2
LITERATURE REVIEW

4 LITERATURE REVIEW

4.1 Introduction

In contemporary transportation systems, the growing concerns about road safety and the rising incidence of vehicular accidents underscore the need for advanced technologies to address specific challenges. A significant contributor to accidents is overspeeding, where vehicles surpass speed limits, jeopardizing safety. Existing speed monitoring methods relying on manual enforcement lack the real-time and comprehensive overspeeding detection necessary for effective safety measures.

4.2 Speed-Tech: Automated Overspeeding Detection System

This project addresses overspeeding challenges through the "Speed-Tech, Automated Overspeeding Detection System." Filling the gap in current detection systems, this solution integrates computer vision, machine learning, deep learning, and real-time data analytics to accurately identify overspeeding instances, provide timely alerts, and enhance road safety.

4.3 Automated Number Plate Detection

The absence of robust overspeeding detection, coupled with a lack of automated number plate detection, elevates accident risks. This project introduces a technology-driven solution capable of real-time overspeeding detection, facilitating prompt corrective measures, and fostering a safer driving culture.

4.4 Moving Object Detection and Background Modeling

Moving object detection involves image acquisition, pre-processing, background modeling, and object detection. While various techniques exist, issues like false positives persist. Proposed solutions include feature extraction, template matching, and contour processing. This work proposes a memory and time-efficient tracking algorithm to address these challenges.

4.5 Morphological Operations in Image Processing

Morphological operations, including erosion and dilation, have been explored to improve image appearance and reduce noise. This literature review discusses the experimental use of four morphological operations in reducing noise from grayscale images, contributing to image quality enhancement.

4.6 Adaptive Background Subtraction for Speed Detection Radar

Authors introduce "adaptive background subtraction" as a novel object detection technique, serving as a foundational step toward developing the Speed Detection Radar. This approach offers promise for accurate and adaptive speed detection.

4.7 Speed Estimation Techniques

Various speed estimation techniques are reviewed. Utilizing video and image processing toolbox, Rad et al. achieve vehicle speed calculation with an average error within an acceptable range. Other techniques include extended Kalman filter applications, such as Shedbalkar's speed estimation for synchronous motors, and RF-based speed estimation proposed by Kassen, providing accurate results in typical streets.

In conclusion, the diverse methodologies explored in the literature lay the groundwork for the development of the "Speed-Tech" system, poised to address critical overspeeding issues in contemporary transportation systems.

CHAPTER 3
SYSTEM REQUIREMENTS

5 SYSTEM REQUIREMENTS

5.1 HARDWARE REQUIREMENTS

5.1.1 Raspberry Pi 4 Model B

Raspberry Pi is a low-cost credit card-sized computer that plugs into a computer monitor or TV and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing and to learn how to program in languages like Python. It is capable of doing everything you would expect a desktop computer to do, from browsing the internet and playing high definition video to making spreadsheets word-processing, and playing games.

Figure 1: Raspberry PI

GPU	Broadcom Video Core VI
CPU	Broadcom BCM2711 SoC with a 1.5 GHz (later models: 1.8 GHz) 64-bit quad-core ARM Cortex-A72 processor
MEMORY	4 GB DDR4
CONNECTIVITY	2.4-GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE, Gigabit Ethernet

Figure 2: Raspberry PI Specifications

5.1.2 Jetson Nano

The Jetson Nano is a small, powerful single-board computer developed by NVIDIA, primarily designed for edge computing and embedded systems. It is part of the Jetson family of devices, which are specialized hardware platforms for running artificial intelligence (AI) workloads at the edge.

Figure 3: Jetson Nano

GPU	NVIDIA Maxwell architecture with 128 NVIDIA CUDA cores
CPU	Quad-core ARM Cortex-A57 MP Core processor
MEMORY	4 GB 64-bit LPDDR4, 1600MHz 256 GB/s
STORAGE	16 GB eMMC 5.1

Figure 4: Jetson Nano Specifications

5.1.3 HP Webcam W100

We have used the HP WEBCAM W100 to get real-time visual information. The HP WEBCAM W100 Series is made to provide gadgets the capacity to perceive, understand, engage with, and learn from their surroundings. It consists of ready-to-use cameras that can be quickly added to existing prototypes via USB.

Figure 5: Camera

Easy Plug-and-Play via USB 2.0 connectivity
30fps (frames per second) with a high resolution of 480p
60° wide-angle view for a stunning visual experience
Swivel 270°adjustable, easy clip-on laptop/monitor that also supports tripod stand
Digital built-in noise-isolating microphone
Dimensions (L x W x H): 5.1 x 6.8 x 3.3
Weight:92 g

Figure 6: Camera Specifications

5.2 SOFTWARE REQUIREMENTS

The success of an automated speed detection project relies heavily on robust and well designed software components. To develop this project we used two main technologies i.e., Computer Vision and Machine Learning.

5.2.1 Python Libraries used in Computer Vision Module

- OpenCV
- Numpy
- Pytesseract
- Time
- Scipy

5.2.2 Python Libraries used in Machine Learning

- Ultralytics
- Tensorflow
- Keras

5.2.3 Libraries used in Web App

- HTML
- Flask

5.3 Summary

The "Speed-Tech" overspeeding detection system navigates the delicate balance between model accuracy, inference speed, and hardware constraints. Initially exploring YOLO models like YOLOv3, YOLOv4, and YOLOv8, the computational intensity, especially for real-time object detection, posed challenges on Raspberry Pi. Acknowledging these limitations, a strategic shift to the NVIDIA Jetson Nano emerged as a solution. Jetson Nano's superior computational capabilities offered a more optimized platform, adept at handling the demands of real-time overspeeding detection.

CHAPTER 4
SYSTEM DESIGN

6 SYSTEM DESIGN

The following diagrams were developed through the course of this project:

6.1 Use Case Diagram

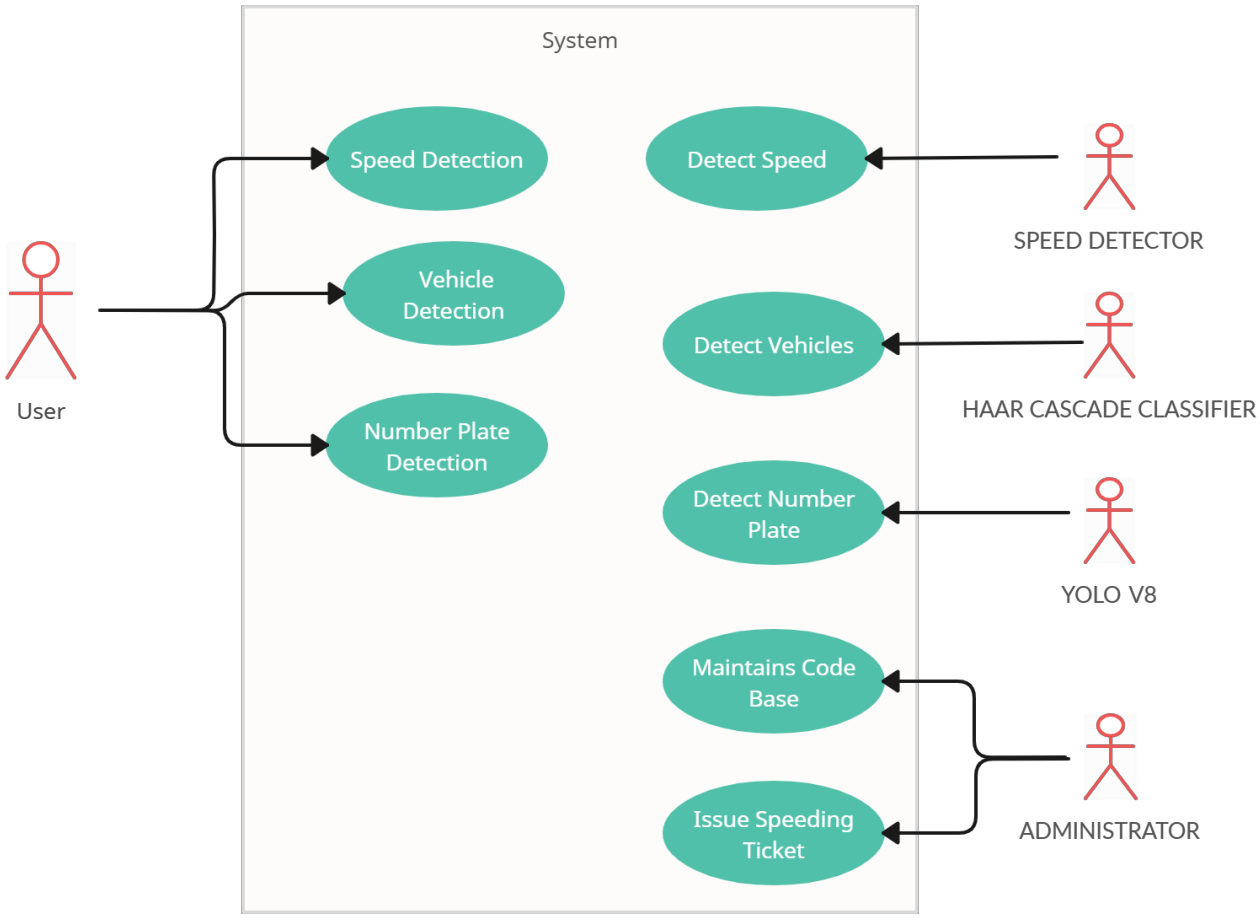

Figure 7: Use Case Diagram

6.2 Activity Diagram

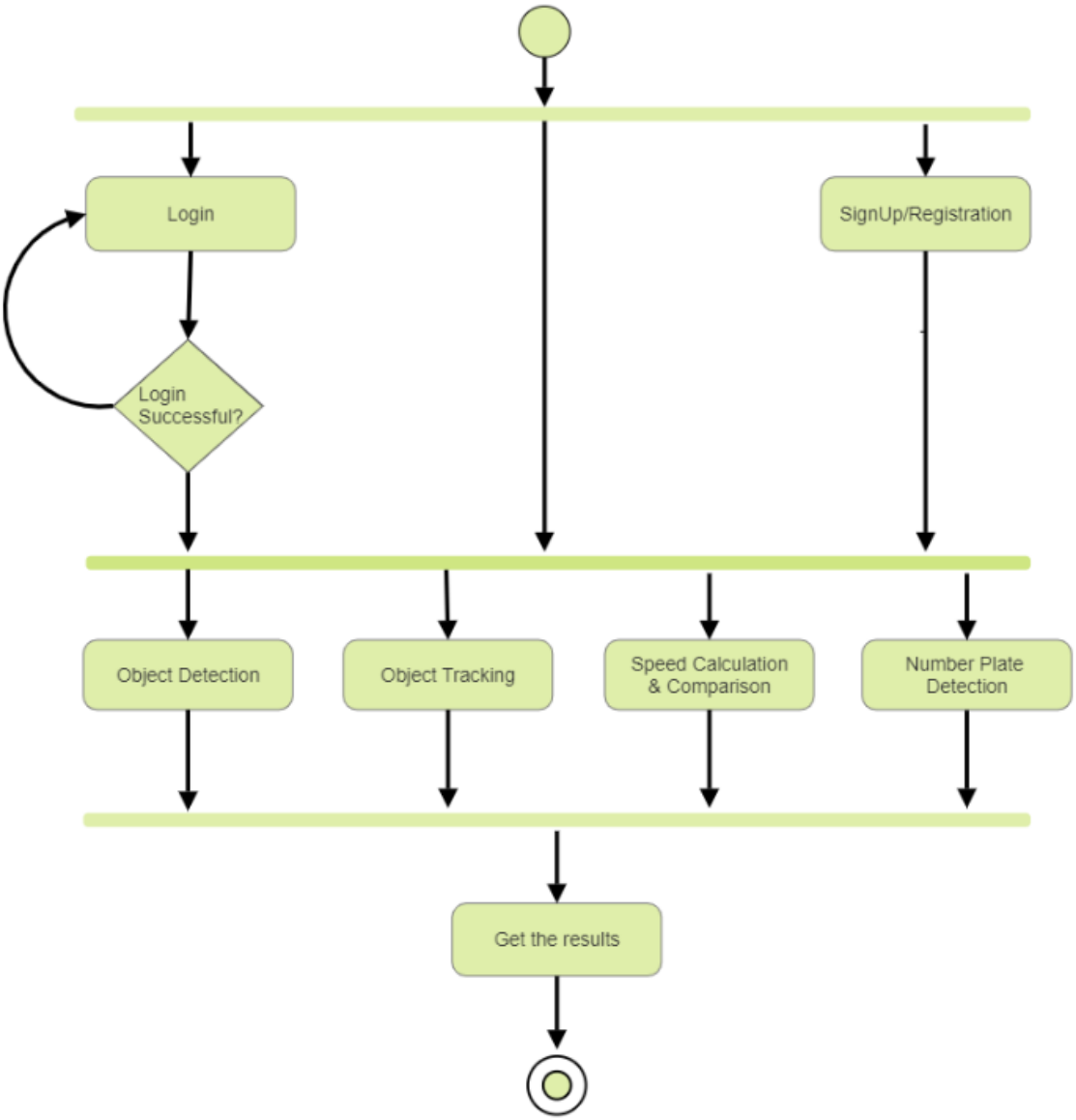

Figure 8: Activity Diagram

CHAPTER 5
SYSTEM IMPLEMENTATION

7 SYSTEM IMPLEMENTATION

7.1 DATA COLLECTION

7.1.1 Camera Caliberation

Camera calibration is an important aspect of speed calculation. It is the real-world distance to pixel ratio. The images captured by the camera are in 2-D (Dimensions) but in real world it is 3-D (Dimensions). However the vehicle can't move above the ground surface (fly into air) so it can be considered as 2D to 2D conversion [7]. Calibration factor can be calculated by knowing the actual length of any object and dividing it by length in pixels of the same object in image. In figure 2 a cross sectional view of the camera, its field of view and the road. c = length of the object in real world (cms) / length of the object in frame (pixels)

7.1.2 Capturing of Video Stream

The video will be captured by camera fixed at a pole on the road, which to be analyzed by the system. The video frames will be taken into consideration when the speed is calculated within stipulated time frame.

7.2 OBJECT DETECTION

Object Detection is a computer vision mechanism that involves identifying and localizing objects of interest within an image or a video. It is a challenging task as it involves not only recognizing the presence of an object but also detecting its precise location and size within the image or video.

7.2.1 Methodology Used: Haar Cascase Classifier

We used Haar Cascade Classifier, a machine learning based object detection method to detect vehicles in video frames. This technique is particularly efficient for real time object detection.

Haar Cascade Classifiers are efficient in terms of both computation and memory usage, making them suitable for devices with limited resources, such as embedded systems like Jetson Nano. It remain a valuable tool for certain tasks and resource-constrained environments, and they are relatively easy to train and implement compared to more complex methods.

Figure 9: Vehicle Detection

7.3 OBJECT TRACKING

Object tracking is a computer vision and image processing technique used to follow the movement of objects within a sequence of frames in a video or series of images.

7.3.1 Methodology Used: Centroid Tracker

We designed a Centroid Tracker algorithm for tracking objects in video frames, with a specific application for vehicle tracking. It employs a set of centroids representing object locations and uses the Euclidean distance to match and update centroids across frames. The tracker registers and deregisters objects based on their movement, effectively handling appearance and disappearance in the video stream. Additionally, the algorithm incorporates a mechanism to set and update the time a car has been tracked. This Centroid Tracker is well-suited for applications like traffic monitoring, where efficient object tracking is crucial for analyzing vehicle movement over time.

7.4 SPEED DETECTION AND COMPARISON

Speed Calculation is achieved by recording the time each vehicle enters the frame and calculating the distance it travels between consecutive frames.

7.4.1 Methodology Used: Computer Vision

First we have to identify each vehicle in the video and then find their corresponding distance covered in consecutive frames. The proposed method tracks each vehicle and traces their centroid in upcoming frames to get the distance travelled by that vehicle. When vehicles arrive into the region of interest, in the video, their corresponding bounding boxes are created and then centroid of the each bounding box is generated. For every new vehicle arrival, we store its centroid value to the track. In addition, we update its centroid value in upcoming frames, and keep on updating its tracked value of centroid until that vehicle passes out by the video. We track all vehicles in a video simultaneously. Moreover, we store their centroid values into the track structure that we have created. The speed of each identified vehicle is calculated as: $Speed = Distance/Time$. Aditionally, there is a conditional check to see if the calculated speed exceeds 90 Km/hr. If the speed is above the threshold, a message is printed indicating that the vehicle is overspeeding and has been submitted for a traffic violation.

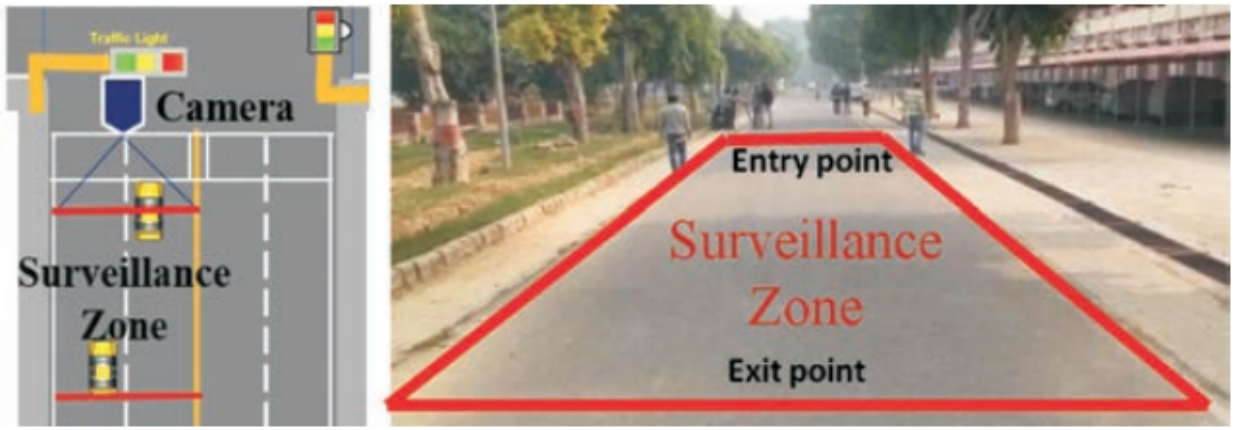

Figure 10: Speed Calculation

7.5 LICENSE PLATE DETECTION

Automatic Number Plate Detection is a technology that combines computer vision and image processing techniques to automatically identify and recognize license plates on vehicles. This technology has various applications, with speed detection being one of them.

7.5.1 Dataset

The dataset that we used is Licence Plate Recognition dataset, we performed 2 pre-processing techniques over each image, it contains 10k auto oriented images of resolution (640*640). The dataset contains images divided into a single classes (LICENSE_PLATE).

Figure 11: Dataset

7.5.2 Methodology Used: YOLOv8 and OCR

YOLOv8 (You Only Look Once)

For YOLOv8 with a single class detected, the model utilizes the YOLOv8 architecture. The neural network is trained with ImageNet weights, and additional training is conducted through transfer learning. This enables the model to detect and classify objects into a single class.

This model is versatile, being applied not only for object detection. The input image is initially resized to a 224x224 dimension before being input into the YOLOv8 model.

To ensure effective training, the dataset is partitioned into an 80/20 ratio for training and validation purposes. This division helps in assessing the model's performance and generalization on unseen data during the training process.

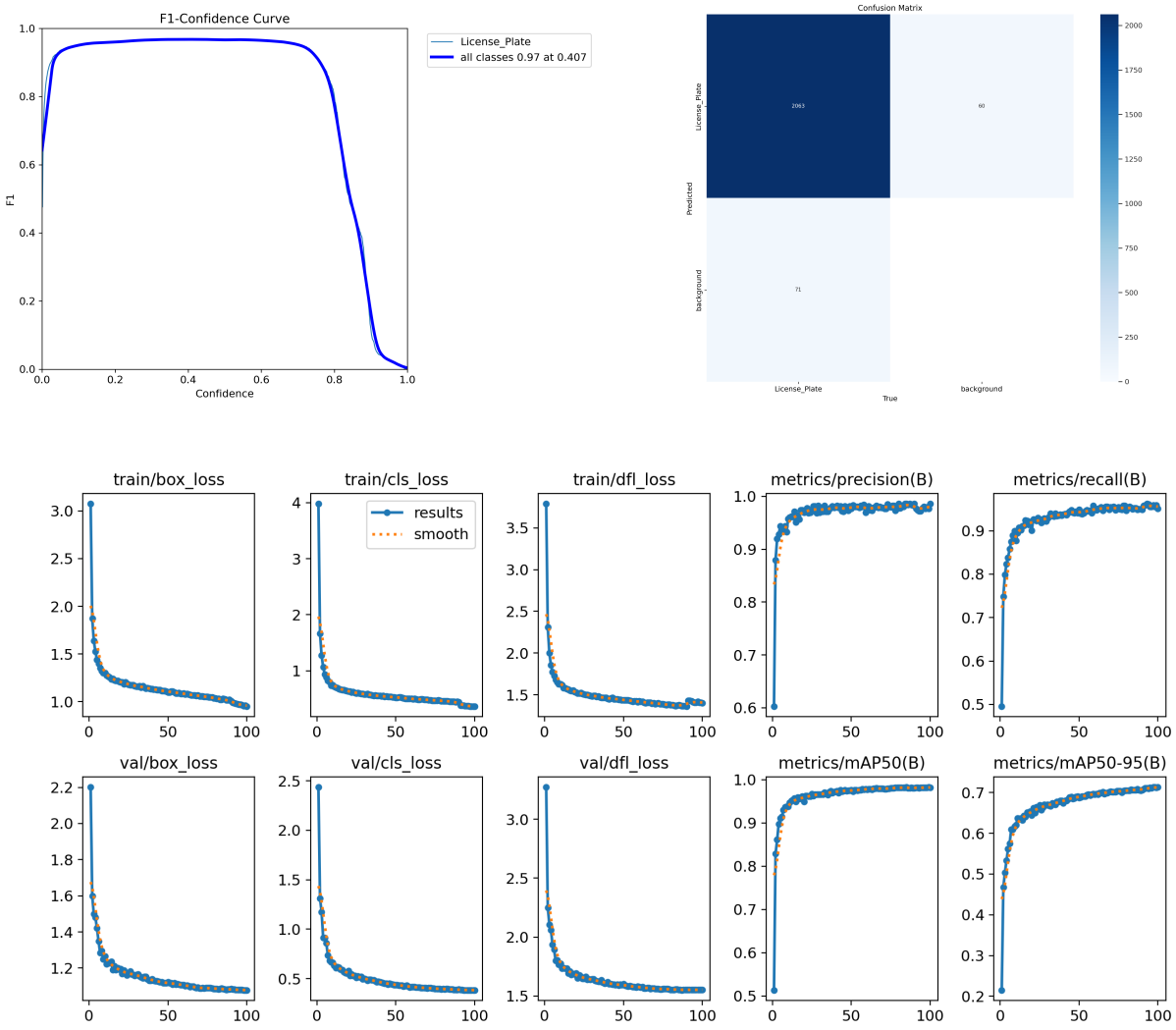

Figure 13: YOLOv8 Results

OCR (Optical Character Recognition)

OCR, or Optical Character Recognition is a technology that is used to recognize and extract text from images, and other visual representations of text. The primary goal of OCR is to convert printed or handwritten text into machine readable text that can be edited, searched, and processed by computers. We have implemented YOLOv8, an object detection algorithm, for identifying regions of interest within images, specifically focusing on number plates of vehicles. OCR is applied to this identified area. OCR algorithms, such as Tesseract or other proprietary solutions, analyze the visual representation of text on the number plate. This process involves recognizing individual characters and reconstructing them into a coherent textual format.

Figure 14: OCR Results

CHAPTER 6
RESULTS

8 RESULTS

In the below example the speed of the vehicles have been successfully detected as well as the details of the vehicles overspeeding are provided.

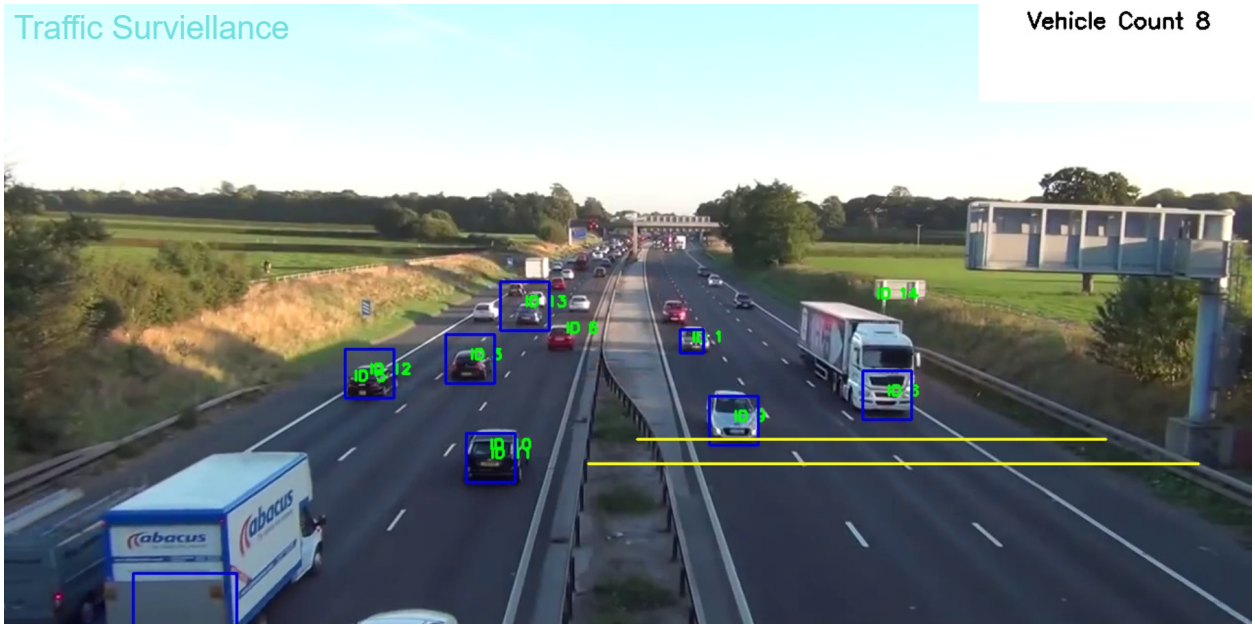

Figure 15: Final Results

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

ID -->  128  -> Speed in (Km/h) is: 60
Car Entered.
Car Entered.
Car Entered.
Car Entered.
Car Left.
ID -->  130  -> Speed in (Km/h) is: 38
Car Left.
ID -->  129  -> Speed in (Km/h) is: 45
Car Entered.
Car Entered.
Car Left.
ID -->  130  -> Speed in (Km/h) is: 42
Car Entered.
Car Left.
ID -->  129  -> Speed in (Km/h) is: 40
Car Entered.
Car Left.
ID -->  145  -> Speed in (Km/h) is: 43
Car Entered.
Car Entered.
Car Left.
ID -->  147  -> Speed in (Km/h) is: 47
Car Left.
ID -->  151  -> Speed in (Km/h) is: 37
Car Entered.
```

Figure 16: Terminal Output

CHAPTER 7
CONCLUSION AND FUTURE WORK

9 CONCLUSION AND FUTURE SCOPE

9.1 Conclusion

In conclusion, "Speed-Tech" is a groundbreaking solution addressing overspeeding challenges and modern urban traffic management. Focused on smart city development, particularly in India, the project tackles the inefficiencies of manual speed monitoring through advanced technologies— Haar Cascade Classifier, YOLOv8, Centroid Tracking, and OCR. The system's adaptability to diverse conditions, emphasizes practicality. Integration of Flask ensures seamless video streaming and a user-friendly web app for instant traffic data access.

Beyond detection, "Speed-Tech" includes dynamic vehicle counting, ID annotation, and movement tracking, recording entry/exit times for traffic flow analysis. Calculating speeds and issuing overspeeding alerts significantly contributes to traffic management and road safety. The intuitive web interface enhances accessibility, solidifying "Speed-Tech" as a holistic solution. Achieving an 87% detection accuracy, the project exemplifies transformative technology in reshaping modern traffic management, promising safer roads and streamlined urban transportation.

9.2 Future Work

9.2.1 Improving Accuracy:

The implemented model is low cost and has low computational power. Therefore, it's accuracy can be improved through feedback loop and by training a model to recognize the exact model of the vehicle whenever an overspeeding violation is detected.

9.2.2 Mobile Application for Real-Time Alerts and Speeding Ticket:

Develop a mobile application that allows users, including law enforcement and traffic management authorities, to receive real-time alerts and notifications. This application can maintain a comprehensive database of registered vehicles. Upon detecting an overspeeding violation, the system could seamlessly integrate with this app to generate automated speeding tickets or challans. This feature ensures swift responses to traffic incidents and facilitates timely interventions.

References

1. Ramasamy, Balasubramani. (2017). A Review on Vehicle Speed Detection using Image Processing. 4.

2. Tarun Kumar, Dharmender Singh Kushwaha, An Efficient Approach for Detection and Speed Estimation of Moving Vehicles, Procedia Computer Science, Volume 89,2016.

3. P. K. Thadagoppula and V. Upadhyaya, "Speed detection using image processing," 2016 International Conference on Computer, Control, Informatics and its Applications (IC3INA), Tangerang, Indonesia, 2016, pp. 11-16, doi: 10.1109/IC3INA.2016.7863015.

4. Wicaksono, Danang Setiyono, Budi. (2017). Speed Estimation On Moving Vehicle Based On Digital Image Processing. International Journal of Computing Science and Applied Mathematics. 3. 21. 10.12962/j24775401.v3i1.2117.

5. B. Suresh, K. Triveni, Y. V. Lakshmi, P. Saritha, K. Sriharsha, D. Srinivas Reddy, 2016, Determination of Moving Vehicle Speed using Image Processing, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH TECHNOLOGY (IJERT) NCACSPV – 2016 (Volume 4 – Issue 18).

6. Traffic-Net: 3D Traffic Monitoring Using a Single Camera, arXiv:2109.09165.

References