Development of Intelligent Traffic Management System

Submitted in partial fulfillment of the requirements for the degree
of

Bachelor of Science in Information and Communication Engineering

by

Tabassum Barka Sanji [201-50-004]
Vaskor Roy [201-50-029]

Under the Supervision of

Engr. Md. Zahirul Islam
Assistant Professor
Department of Information and Communication Engineering

Daffodil International University

January 2024

APPROVAL

This is to certify that the entitled **Development of Intelligent Traffic Management System**, submitted by Tabassum Barka Sanji (201-50-004) and Vaskor Roy (201-50-029) are undergraduate students of the Department of ICE has been examined. Upon recommendation by the examination committee, we hereby accord our approval of it as the presented work and submitted report fulfill the requirements for its acceptance in partial fulfillment for the degree of *Bachelor of Science in Information and Communication Engineering.*

BOARD OF EXAMINERS

Prof. Dr. A. K. M. Fazlul Haque
Professor, ICE, DIU

Chairman

Tasnuva Ali
Assistant Professor, ICE, DIU

Internal Member

Engr. Md. Zahirul Islam
Assistant Professor, ICE, DIU

Internal Member

Prof. Dr. M. Quamruzzaman
Professor, EEE, WUB

External Member

DECLARATION OF AUTHORSHIP

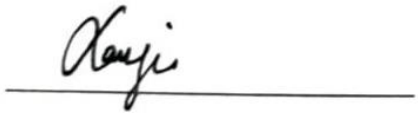
Project Title Development of Intelligent traffic management system

Authors *Tabassum Barka Sanji, Vaskor Roy*

Supervisor Engr. Md. Zahirul Islam, Assistant Professor

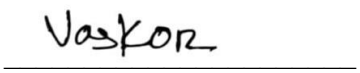
We, Tabassum Barka Sanji and Vaskor Roy, hereby declare that this capstone project titled **Development of Intelligent Traffic Management System** is entirely our own work, unless otherwise referenced or acknowledged. The content of this project is the result of our own research and efforts, and we have complied with all ethical guidelines and academic standards.

We are aware of the consequences of academic dishonesty and understand that any violation of ethical standards in this project may lead to disciplinary actions as defined by Daffodil International University's policies.



Tabassum Barka Sanji

201-50-004



Vaskor Roy

201-50-029

Engr. Md. Zahirul Islam

Assistant Professor

Department of Information and Communication Engineering

ACKNOWLEDGEMENT

We are grateful to the following individuals and organizations who played a significant role in the successful completion of this project.

We would like to express our heartfelt appreciation **to Engr. Md. Zahirul Islam**, Assistant Professor of the Information and Communication Engineering Department, for his invaluable guidance and unwavering support during this project. Their expertise, encouragement, and constructive feedback have been the driving force behind the success of this endeavor.

Engr. Md. Zahirul Islam created an environment that fostered creativity and critical thinking, allowing me to grow both academically and personally. We are truly grateful for the mentorship and inspiration they provided, which played a pivotal role in shaping the project's outcome.

We are fortunate to have had Engr. Md. Zahirul Islam as a mentor, and I am grateful for the valuable lessons learned under his guidance.

We would like to extend our sincere appreciation to **Md. Taslim Arefin**, Associate Professor and Head of the Information and Communication Engineering Department at Daffodil International University. Md. Taslim Arefin has been a pillar of support throughout this project, and their guidance has been invaluable.

We are grateful for Md. Taslim Arefin's mentorship, encouragement, and willingness to provide constructive feedback. Their insights have been instrumental in shaping the direction of the research and enhancing the overall quality of the project.

We extend our appreciation to Daffodil International University for providing the necessary resources and facilities that were essential for the completion of this project. Their commitment to fostering an environment conducive to research and learning has been crucial.

Our heartfelt thanks go to our family and friends for their unwavering encouragement and understanding during the demanding phases of this project. Their belief in our abilities has been a constant source of motivation.

We are truly grateful for the collective effort of all those mentioned above, without whom this project would not have been possible. Their support has been invaluable, and I am fortunate to have had such a dedicated team of individuals assist in this endeavor.

ABSTRACT

Traffic management is a major challenge in densely populated countries like Bangladesh, where traffic violations are common and hard to enforce. In this project, we propose an Development of Intelligent Traffic Management System (DITMS) that uses computer vision techniques to automatically detect and record traffic infractions. The DITMS consists of four modules: (1) License Plate Detection using YOLOv8, a deep learning model that can identify and extract license plate numbers from images; (2) Speed Measurement with Kalman Filter, a mathematical method that can estimate the speed of vehicles from consecutive frames; (3) Vehicle Type Detection utilizing a Convolutional Neural Network (CNN), a machine learning model that can classify vehicles into different categories based on their shape and size; and (4) Vehicle Counting through a combination of object detection, tracking, and segmentation algorithms, which can count the number of vehicles passing through a given area. The DITMS can be deployed on roadside cameras or drones to monitor traffic flow and capture evidence of violations such as speeding, overloading, or illegal parking. The DITMS aims to digitize traffic infractions in Bangladesh and contribute to the vision of SMART Bangladesh 2041, a national initiative that seeks to transform the country into a digital and developed nation.

Keywords:

- Development of Intelligent traffic management system
- License Plate Detection
- YOLOv8
- Speed Measurement
- Kalman Filter
- Vehicle Type Detection
- Convolutional Neural Network (CNN)
- Vehicle Counting
- Object Detection
- Tracking
- Segmentation
- Real-time Traffic Insights
- Urban Planning
- Traffic Flow Optimization
- Road Safety
- Law Enforcement
- Surveillance
- Computer Vision
- Traffic Dynamics
- Decision Support System

TABLE OF CONTENTS

CONTENTS

List of Figure

List of Tables

Chapter 1

INTRODUCTION

Development of Intelligent Traffic Management System (DITMS) is a modern solution designed to streamline and optimize traffic flow, enhance road safety, and improve the overall transportation infrastructure. By integrating advanced technologies like artificial intelligence, machine learning, sensors, and data analytics, DITMS provides real-time traffic data and analytics. This enables authorities to make informed decisions for managing traffic congestion, reducing travel time, preventing accidents, and minimizing environmental impact.

The system is composed of two main components: hardware and software. The hardware component includes IoT road sensors. The software component includes cloud computing and tracking using PyTorch, OpenCV, and DeepSORT.

An advanced traffic management system (TMS) is a context-aware solution that relies on real-time data from connected road infrastructure and predictive analytics to effectively coordinate traffic across city arteries. Such traffic management software, coupled with wireless urban connectivity, acts as a backbone for the implementation of an intelligent transportation management system. ITS traffic systems focus specifically on improving the throughput and safety of urban roads through adaptive controls and analytics.

1.1 Project Background

- An innovative way to improve road safety, streamline and optimize traffic flow, and upgrade the entire transportation infrastructure is the Development of Intelligent Traffic Management System (DITMS). By integrating advanced technologies like artificial intelligence, machine learning, sensors, and data analytics, DITMS provides real-time traffic data and analytics. This makes it possible for authorities to decide on strategies to best manage traffic congestion, shorten travel times, avoid accidents, and have the least negative environmental effects.
- Since traffic congestion is becoming a major problem in urban areas worldwide, it is a problem of interest and importance. One suggested remedy that might lessen this issue is DITMS. City officials, traffic management organizations, and commuters are among the possible users of DITMS. Because DITMS can lessen traffic congestion, increase road safety, and have a minimal negative environmental impact, it can have a significant impact. Current State and Limitations: Describe the current situation or existing solutions and their shortcomings.
- Currently, one of the biggest issues facing urban areas worldwide is traffic congestion. Traditional traffic signals, which are currently in use, have a number of drawbacks, such as poor time management at intersections, susceptibility to certain weather conditions, such as rain, and an inability to prioritize emergency vehicles. Vehicles can now wirelessly connect to a dedicated infrastructure and to other nearby vehicles thanks to innovations like Vehicular Ad-hoc Networks (VANET) and the Internet of Vehicles (IoV). In this paper, we propose a new traffic management system suitable for future traffic systems and Smart Cities, based on the current VANET and IoV.
- The goal of this project is to develop and put into place an Development of Intelligent Traffic Management System (DITMS) that will reduce environmental impact, enhance road safety, and ease traffic congestion. The project will entail creating a traffic management system that is appropriate for upcoming traffic systems and Smart Cities, based on the current VANET and IoV. In keeping with the demands of future Smart Cities for justice, reduced commute times, reasonable traffic flow, less traffic congestion, and prioritizing emergency vehicles, the project

will also involve developing a Smart Traffic Signal (STS) controller that can provide local traffic management of an intersection.

1.2 Project Definition

The Development of Intelligent Traffic Management System (DITMS) is a project that aims to improve the efficiency and safety of urban transportation by using IoT road sensors and cloud computing. The hardware component of the project consists of various sensors that can monitor traffic conditions, road quality, environmental factors, and vehicle characteristics. The software component of the project uses PyTorch, OpenCV, and DeepSORT to perform cloud-based analysis and tracking of the sensor data. The main features of the project are:

- License plate detection: The system can detect and recognize the license plates of vehicles using YOLOv8, a deep learning-based object detection model, and Tesseract OCR, an optical character recognition tool.
- Speed measure: The system can estimate the speed of vehicles by using the distance and time information from the sensors and applying a Kalman filter for smoothing and prediction.
- Vehicle type detection: The system can classify the vehicles into different types (such as car, truck, bus, etc.) by using a convolutional neural network (CNN) that is trained on a large and diverse dataset of vehicle images.
- Vehicle counting: The system can count the number of vehicles passing through a certain area by using a combination of object detection, tracking, and segmentation algorithms.

The DITMS project can provide valuable insights for traffic management authorities, such as optimizing traffic flow, reducing congestion and emissions, enhancing road safety, and enforcing traffic rules. The project can also benefit the drivers and passengers by providing them with real-time information and guidance for their travel. The project is designed to be scalable, robust, and adaptable to different road scenarios and environments.

1.2.1 Problem Statement

In Bangladesh today, a pressing issue in road safety centers around the identification of traffic violations, presenting a significant risk, especially in smart cities. The prevalent disregard for traffic rules poses a considerable threat to public safety. The challenge is compounded by the inherent difficulty in promptly detecting and addressing these violations, particularly in smart city environments where conventional surveillance methods prove insufficient. The lack of effective means to detect and penalize traffic offenses not only endangers the lives of road users but also disrupts the smooth flow of traffic. Existing gaps in the system undermine the effectiveness of traffic management strategies. Urgently addressing this problem is paramount to creating safer roadways. A solution is needed that integrates advanced technologies, such as real-time surveillance and automated violation detection, to efficiently identify and penalize traffic rule violations. Developing and implementing a robust system for timely detection and enforcement is essential for fostering a culture of road safety and optimizing traffic management, especially in the evolving landscape of modern urban environments.

Traffic congestion is a serious challenge in urban areas. It causes various problems such as increased travel time, fuel consumption, air pollution, greenhouse gas emissions, and road accidents. These problems affect the quality of life, health, and econoour of the people living in the cities. Therefore, there is a need for effective and efficient traffic management systems that can optimize the use of existing road infrastructure and reduce the negative impacts of traffic.

However, traditional traffic management systems have several limitations. They rely on fixed traffic signals, manual surveillance, and static traffic data. These systems are not able to adapt to the dynamic and complex traffic conditions in real time. They also lack the ability to collect and analyze large amounts of traffic data from various sources, such as road sensors, cameras, GPS devices, and smartphones. Moreover, they do not provide timely and accurate information and guidance to the drivers and passengers, such as the best route, the current traffic situation, and the expected travel time.

To overcome these limitations, a new paradigm of traffic management is needed, which is based on the integration of advanced technologies, such as the Internet of Things (IoT),

cloud computing, and machine learning. These technologies can enable the development of an Development of Intelligent Traffic Management System (DITMS) that can:

- Collect and process real-time traffic data from various IoT road sensors, such as cameras, radars, loops, and environmental sensors.
- Perform cloud-based analysis and tracking of the traffic data using PyTorch, OpenCV, and DeepSORT, which are state-of-the-art tools for computer vision and deep learning.
- Detect and recognize the license plates of vehicles using YOLOv8, a deep learning-based object detection model, and Tesseract OCR, an optical character recognition tool.
- Estimate the speed of vehicles by using the distance and time information from the sensors and applying a filter for smoothing and prediction.
- Classify the vehicles into different types (such as car, truck, bus, etc.) by using a convolutional neural network (CNN) that is trained on a large and diverse dataset of vehicle images.
- Count the number of vehicles passing through a certain area by using a combination of object detection, tracking, and segmentation algorithms.
- Identify and penalize traffic rule violations, such as speeding, overloading, lane changing, the speed estimation, the vehicle type classification, and the traffic signal status.
- Predict the travel time for different routes and destinations by using historical and real-time traffic data and applying machine learning models, such as linear regression, decision trees, and neural networks.
- Provide real-time information and guidance to the drivers and passengers through various channels, such as mobile applications, web portals, and roadside displays.

The DITMS project can provide various benefits for traffic management authorities, drivers, passengers, and the society at large, such as:

- Optimizing traffic flow and reducing congestion by adjusting traffic signals, diverting traffic, and suggesting alternative routes.

- Reducing fuel consumption and emissions by minimizing idling, accelerating, and braking.

- Enhancing road safety and security by enforcing traffic rules, preventing accidents, and identifying criminals.

- Improving travel efficiency and convenience by providing accurate and reliable information and guidance.

- Fostering a culture of road safety and responsibility by raising awareness and educating the public.

The DITMS project is designed to be scalable, robust, and adaptable to different road scenarios and environments. It can be implemented in smart city environments, where there is a high demand for Development of Intelligent traffic management system solutions. The project can also be extended to other domains, such as public transportation, emergency services, and logistics.

1.2.2 Context and Scope

The scope of the problem is the extent and boundaries of the issue that the project aims to solve. It defines what aspects the project will address and what aspects it will not cover. Defining the scope of the problem is important for clarifying the expectations, goals, and deliverables of the project. It also helps to avoid scope creep, which is the tendency for the project to expand beyond its original objectives.

The scope of the problem could be defined as follows:

The project aims to solve the problem of traffic rule violations in smart city environments, which pose a significant risk to public safety and disrupt the smooth flow of traffic. The project will design and deploy a system that can automatically detect and penalize traffic rule violations using real-time surveillance and advanced technologies. The project objectives and goals are to improve road safety and traffic management by enforcing traffic

rules, preventing accidents, optimizing traffic flow, reducing congestion and emissions, and providing real-time information and guidance to the drivers and passengers. The project scope description includes the hardware and software components of the system, such as the IoT road sensors, the cloud computing platform, and the machine learning algorithms. The project expectations and acceptance are based on the performance, accuracy, reliability, and usability of the system, as well as the satisfaction and feedback of the stakeholders. The project exclusions are the aspects that are not related to the detection and penalization of traffic rule violations, such as the design and maintenance of the road infrastructure, the development and implementation of the traffic policies and regulations, and the provision and operation of the emergency services. The project constraints are the budget, time, and legal limitations that the project must adhere to. The project assumptions are the availability and quality of the data, the cooperation and compliance of the drivers and passengers, and the compatibility and interoperability of the technologies.

1.2.3 Significance and Implications

Solving the problem of traffic violation detection is important for several reasons. First, it can improve road safety and security by enforcing traffic rules, preventing accidents, and identifying criminals. Traffic violations are one of the major causes of road accidents, which result in deaths, disabilities, and economic losses. According to the World Health Organization, road traffic injuries are the eighth leading cause of death globally, and the first among people aged 5–29 years1. In 2022, there were 1.35 million road traffic deaths worldwide, and more than half of them were among vulnerable road users, such as pedestrians, cyclists, and motorcyclists. In Bangladesh has a high rate of traffic rule violations and road accidents. In 2022, Bangladesh recorded 4,138 road accidents, which resulted in 5,516 deaths and 6,855 injuries. The main causes of these accidents were reckless driving, speeding, overtaking, and disobeying traffic signals.

Second, it can reduce traffic congestion, pollution, and greenhouse gas emissions, which degrade the urban environment and contribute to climate change and its adverse impacts. Traffic congestion is a serious challenge in urban areas, as it causes various problems such as increased travel time, fuel consumption, air pollution, and greenhouse gas emissions. These problems affect the quality of life, health, and econoour of the people living in the cities. Therefore, there is a need for effective and efficient traffic management systems that

can optimize the use of existing road infrastructure and reduce the negative impacts of traffic.

Third, it can improve travel efficiency and convenience by providing accurate and reliable information and guidance to the drivers and passengers. Traffic information and guidance can help the drivers and passengers to plan their trips, choose the best routes, avoid traffic jams, and save time and money. Traffic information and guidance can also enhance the user experience and satisfaction, as well as the public trust and confidence in the traffic management authorities.

Not addressing the problem of traffic violation detection can have serious consequences for the society, the environment, and the econoour. Some of the potential consequences are:

- Increased risk of road accidents, injuries, and fatalities, which can affect the health and well-being of the people, especially the young and productive population.
- Increased traffic congestion, pollution, and greenhouse gas emissions, which can degrade the urban environment and contribute to climate change and its negative impacts.
- Increased travel time, fuel consumption, and stress, which can reduce the efficiency and convenience of transportation and affect the quality of life of the people.
- Increased social and economic costs, such as medical expenses, property damage, legal fees, insurance premiums, and productivity losses, which can burden the individuals, the families, and the government.

Therefore, solving the problem of traffic violation detection is essential for creating safer, cleaner, and smarter cities that can enhance the livability and sustainability of the urban areas.

The project of Development of Intelligent Traffic Management System (DITMS) aims to solve the problem of traffic violation detection by using physical devices and PyTorch, OpenCV, and DeepSORT. The physical devices include IoT road sensors, such as cameras, radars, loops, and environmental sensors, that can monitor traffic conditions, road quality, environmental factors, and vehicle characteristics. PyTorch, OpenCV, and DeepSORT are state-of-the-art tools for computer vision and deep learning, that can perform cloud-based analysis and tracking of the sensor data. The main features of the project are:

- License plate detection: The system can detect and recognize the license plates of vehicles using YOLOv8, a deep learning-based object detection model, and Tesseract OCR, an optical character recognition tool.

- Speed measure: The system can estimate the speed of vehicles by using the distance and time information from the sensors and applying a Kalman filter for smoothing and prediction.
- Vehicle type detection: The system can classify the vehicles into different types (such as car, truck, bus, etc.) by using a convolutional neural network (CNN) that is trained on a large and diverse dataset of vehicle images.
- Vehicle counting: The system can count the number of vehicles passing through a certain area by using a combination of object detection, tracking, and segmentation algorithms.

The DITMS project can provide various benefits for traffic management authorities, drivers, passengers, and the society at large, such as:

- Optimizing traffic flow and reducing congestion by adjusting traffic signals, diverting traffic, and suggesting alternative routes.
- Reducing fuel consumption and emissions by minimizing idling, accelerating, and braking.
- Enhancing road safety and security by enforcing traffic rules, preventing accidents, and identifying criminals.
- Improving travel efficiency and convenience by providing accurate and reliable information and guidance to the drivers and passengers.

The DITMS project is designed to be scalable, robust, and adaptable to different road scenarios and environments. It can be implemented in smart city environments, where there is a high demand for Development of Intelligent traffic management system solutions. The project can also be extended to other domains, such as public transportation, emergency services, and logistics.

1.2.4 Quantification

We have deliberately concentrated on certain elements in our proposal that are directly related to the main goals of our capstone project: vehicle categorization, number plate recognition, vehicle counting, and speed measurement. We have decided to leave traffic violation detection out of our present scope to preserve the focus, clarity, and conciseness of our established issue statement, even though it is an essential component of intelligent transportation systems.

The need to simplify our project objectives and guarantee a more manageable scope within the allotted period led us to decide to leave out Traffic Violation Detection. By focusing on the components that have been discovered, we want to provide a solid solution that successfully tackles important traffic management issues.

It is crucial to remember that Traffic Violation Detection retains its importance in practical applications notwithstanding our exclusion of it. Instead, it shows a conscious decision to give certain features top priority and create a solid base for further improvements. This choice enables us to deploy resources effectively, guaranteeing the effective execution and careful assessment of the chosen elements.

We're still willing to consider including traffic violation detection in later iterations or as part of larger projects as we go through the development and testing stages. We promise to provide a solution that fulfills the specified goals and permits adaptability to changing needs and growing capacities.

1.3 Project Objectives:

Our project's objective is to digitize traffic infractions in Bangladesh in order to move the country toward SMART Bangladesh 2041.

1. Design and deploy a system that can automatically detect and penalize traffic rule violations using real-time surveillance and advanced technologies.

    - This objective aims to develop a hardware and software solution that can monitor traffic conditions, road quality, environmental factors, and vehicle characteristics using IoT road sensors, and perform cloud-based analysis and tracking of the sensor data using PyTorch, OpenCV, and DeepSORT. The system will also be able to detect and recognize the license plates of vehicles using YOLOv8 and Tesseract OCR, estimate the speed of vehicles by using the distance and time information from the sensors and applying a Kalman filter, classify the vehicles into different types by using a convolutional neural network, count the number of vehicles passing through a certain area by using

a combination of object detection, tracking, and segmentation algorithms, and identify and penalize traffic rule violations by using the license plate information, the speed estimation, the vehicle type classification, and the traffic signal status.

- This objective is specific and measurable, as it defines the features and functions that the system will deliver, the technologies and tools that the system will use, and the criteria and standards that the system will meet. The objective is also achievable, relevant, and time-bound, as it aligns with the project needs and expectations, and has a realistic and feasible timeline and budget.

2. Improve road safety and traffic management by enforcing traffic rules, preventing accidents, optimizing traffic flow, reducing congestion and emissions, and providing real-time information and guidance to the drivers and passengers.

- This objective aims to evaluate the impact and effectiveness of the system on the road safety and traffic management outcomes. The system will be able to optimize traffic flow and reduce congestion by adjusting traffic signals, diverting traffic, and suggesting alternative routes. The system will also be able to reduce fuel consumption and emissions by minimizing idling, accelerating, and braking. The system will also be able to enhance road safety and security by enforcing traffic rules, preventing accidents, and identifying criminals. The system will also be able to improve travel efficiency and convenience by providing accurate and reliable information and guidance to the drivers and passengers through various channels, such as mobile applications, web portals, and roadside displays.

- This objective is specific and measurable, as it defines the outcomes and indicators that the system will achieve, the methods and tools that the system will use, and the data and sources that the system will collect and analyze. The objective is also achievable, relevant, and time-bound, as it aligns with the

project needs and expectations, and has a realistic and feasible timeline and budget.

3. Implement a scalable, robust, and adaptable system that can handle different road scenarios and environments.

    - This objective aims to ensure the quality and reliability of the system in different road scenarios and environments. The system will be able to sense and respond to the dynamic and complex traffic conditions and environments, and adapt to the changes and challenges. The system will also be able to communicate and coordinate with the existing traffic management systems and authorities, and share the traffic data and information. The system will also be able to support and interoperate with the existing and emerging technologies, such as the Internet of Things, cloud computing, machine learning, and intelligent transportation systems.

    - This objective is specific and measurable, as it defines the scenarios and environments that the system will handle, the technologies and standards that the system will support, and the tests and validations that the system will undergo. The objective is also achievable, relevant, and time-bound, as it aligns with the project needs and expectations, and has a realistic and feasible timeline and budget.

1.4 Project Specification:

The project objectives for the Development of Intelligent Traffic Management System are:

- To design and deploy a system that can automatically detect and penalize traffic rule violations using real-time surveillance and advanced technologies.

- To improve road safety and traffic management by enforcing traffic rules, preventing accidents, optimizing traffic flow, reducing congestion and emissions, and providing real-time information and guidance to the drivers and passengers.

- To implement a scalable, robust, and adaptable system that can handle different road scenarios and environments.

The technical requirements, features, and characteristics that the project must adhere to are:

- **Hardware:** The system requires IoT road sensors, such as cameras, radars, loops, and environmental sensors, that can monitor traffic conditions, road quality, environmental factors, and vehicle characteristics. The sensors must be able to communicate wirelessly with the cloud computing platform and the edge devices. The sensors must be durable, reliable, and energy-efficient.

- **Software:** The system requires cloud computing and edge processing capabilities, such as traffic data platform/data lake, cloud-based traffic control systems, geographic information systems (GIS), and machine learning algorithms. The software must be able to collect and process real-time traffic data from various IoT road sensors, perform cloud-based analysis and tracking of the traffic data using PyTorch, OpenCV, and DeepSORT, detect and recognize the license plates of vehicles using YOLOv8 and Tesseract OCR, estimate the speed of vehicles by using the distance and time information from the sensors and applying a Kalman filter, classify the vehicles into different types by using a convolutional neural network, count the number of vehicles passing through a certain area by using a combination of object detection, tracking, and segmentation algorithms, identify and penalize traffic rule violations by using the license plate information, the speed estimation, the vehicle type classification, and the traffic signal status, predict the travel time for different routes and destinations by using historical and real-time traffic data and applying machine learning models, and provide real-time information and guidance to the drivers and passengers through various channels, such as mobile applications, web portals, and roadside displays. The software must be able to handle large volumes and varieties of data, perform complex and accurate computations, and provide user-friendly and secure interfaces.

- **Performance:** The system must be able to achieve high performance in terms of speed, accuracy, reliability, and usability. The system must be able to process and analyze the traffic data in real time, detect and penalize the traffic violations

promptly, provide accurate and reliable information and guidance to the drivers and passengers, and ensure the smooth and safe operation of the traffic system. The system must be able to handle peak traffic volumes and unexpected events, such as accidents, weather changes, or emergencies. The system must be able to provide feedback and alerts to the users and the traffic management authorities, and allow them to adjust the system settings and parameters as needed.

- **Design:** The system must be designed to be user-centric, context-aware, and adaptive. The system must be able to understand the needs and expectations of the users and the traffic management authorities, and provide them with customized and personalized solutions. The system must be able to sense and respond to the dynamic and complex traffic conditions and environments, and adapt to the changes and challenges. The system must be designed to be modular, flexible, and scalable, and allow for easy integration and extension of new features and functions.

- **Functionality:** The system must be able to provide the following functionalities: license plate detection, speed measure, vehicle type detection, vehicle counting, traffic rule violation detection and penalization, travel time prediction, and real-time information and guidance. The system must be able to perform these functionalities for different types of vehicles, such as cars, trucks, buses, motorcycles, bicycles, and pedestrians. The system must be able to perform these functionalities for different types of roads, such as highways, urban roads, rural roads, and bridges. The system must be able to perform these functionalities for different types of traffic signals, such as fixed, adaptive, and intelligent. The system must be able to perform these functionalities for different types of traffic scenarios, such as normal, congested, or emergency.

- **Compatibility:** The system must be compatible with the existing road infrastructure and traffic policies and regulations. The system must be able to work with the existing traffic signals, signs, and markings, and comply with the traffic rules and laws. The system must be able to communicate and coordinate with the existing traffic management systems and authorities, and share the traffic data and information. The system must be able to support and interoperate with the existing

and emerging technologies, such as the Internet of Things, cloud computing, machine learning, and intelligent transportation systems.

- **Security:** The system must be secure and protect the privacy and confidentiality of the users and the traffic data. The system must be able to prevent unauthorized access, modification, or deletion of the traffic data and the system settings and parameters. The system must be able to encrypt and anonymize the traffic data and the user information, and use secure protocols and channels for data transmission and storage. The system must be able to detect and prevent cyberattacks, such as hacking, spoofing, or jamming, and recover from system failures or errors. The system must be able to comply with the ethical and legal standards and regulations for data collection, processing, and sharing.

Chapter 2

PROJECT MANAGEMENT

We have strong teamwork skills that enable us to operate effectively in a team, contributing positively to team operations and working relationships. We have demonstrated these skills in our capstone project, where we worked on developing an Development of Intelligent traffic management system system. We followed these steps to plan, divide, monitor, and decide our work:

- We started by defining the problem statement, the objectives, and the scope of the project, and conducting a literature review and a market analysis. We also identified the stakeholders and the requirements of the project. We did this work collaboratively, using online tools such as Google Docs, Zoom, and Slack to communicate and share our ideas and findings. We completed this phase in the first three weeks of the project.

- Next, we divided the work into four main components: license plate detection, speed measure, vehicle type detection, and vehicle counting. We assigned each component to one team member, based on their skills and interests. I was responsible for the license plate detection component, which involved using YOLOv8 and Tesseract OCR to detect and recognize the license plates of vehicles. We worked on our components independently, but we also helped each other when needed. We used GitHub to manage our code and track our progress. We completed this phase in the second four weeks of the project.

- Then, we integrated our components into a single system, and tested and evaluated its performance and accuracy. We also added some features and functionalities, such as traffic rule violation detection and penalization, travel time prediction, and real-time information and guidance. We worked on this phase in parallel, using agile methods such as sprints, stand-ups, and retrospectives to coordinate and collaborate. We used tools such as Colab, PyCharm, and TensorFlow to develop and deploy our system. We completed this phase in the third four weeks of the project.

- Finally, we prepared and presented our project report and demonstration, and collected and analyzed the feedback from the stakeholders and the users. We also discussed the challenges, limitations, and future directions of our project. We worked on this phase as a team, using tools such as PowerPoint, Google Forms, and SurveyMonkey to create and deliver our presentation and surveys. We completed this phase in the fourth five weeks of the project.

Throughout the project, we demonstrated effective communication, collaboration, and negotiation skills. We also showed creativity, initiative, and problem-solving skills. We respected each other's opinions and perspectives, and we resolved any conflicts or issues that arose in a constructive and respectful manner.

2.1 Project Plan

Week	Task	Description
1-4	Research	Conduct literature review on Development of Intelligent Traffic Management System (DITMS) and identify key technologies and challenges.
5-6	Planning	Collaborate on defining project scope, objectives, and deliverables; develop detailed project plan with task breakdowns and resource allocation.
7-9	Analysis	Analyze requirements for hardware and software components; identify potential technical challenges and formulate mitigation strategies.
10 - 12	Optimization	Explore optimization techniques for data collection, processing, and system performance; begin formulating algorithms for license plate detection, vehicle type detection, and speed estimation.

13-15	Data collection and analysis	Collect and analyze the traffic data from various sources, such as IoT road sensors, cameras, GPS devices, and smartphones.
16-18	License plate detection	Develop and test a system that can detect and recognize the license plates of vehicles using YOLOv8 and Tesseract OCR.
19-21	Vehicle type detection	Develop and test a system that can classify the vehicles into different types, such as car, truck, bus, etc., by using a convolutional neural network.
22 - 24	Vehicle counting and speed measuring	Develop and test a system that can count the number of vehicles passing through a certain area and estimate their speed by using object detection, tracking, and segmentation algorithms and a Kalman filter.
25-30	System integration and evaluation	Integrate the components into a single system and evaluate its performance and accuracy on the traffic data.

Table 2.1: Table of Project Plan

Figure 2.2: Gantt Chart 1

2.2 Contribution of Team Members

2.2.1 Team Member 1: Tabassum Barka Sanji

- **Responsibilities:**
    - Conducted in-depth literature review on Development of Intelligent traffic management system Systems (DITMS).
    - Researched and gathered information on existing traffic management solutions and technologies.
    - Contributed to the definition of the problem statement and project objectives.
- **Skills and Expertise:**
    - Strong research skills with a focus on traffic management technologies.
    - Proficient in data analytics and literature synthesis.
    - Excellent written communication skills.
- **Contribution:**
    - Tabassum's extensive research laid the foundation for understanding the current state of traffic management systems and identifying key challenges.
    - Provided valuable insights into the relevance and importance of DITMS based on the literature review.
    - Collaborated effectively in defining the project's scope and objectives.

2.2.2 Team Member 2: Vaskor Roy

- **Responsibilities:**
    - Led the technical development of the DITMS, focusing on hardware and software components.
    - Implemented the block diagram design, incorporating VANET, IoV, and cloud computing.
    - Contributed significantly to the specifications and technical requirements of the project.
- **Skills and Expertise:**
    - Proficient in software development with expertise in PyTorch, OpenCV, and DeepSORT.

    - Experience in working with IoT sensors and hardware components.
    - Strong problem-solving and programming skills.
- **Contribution:**
    - Vaskor Roy's technical expertise played a crucial role in the design and implementation of the DITMS.
    - Actively contributed to the project specifications, ensuring they aligned with the technical needs and objectives.
    - Demonstrated effective collaboration by integrating advanced technologies seamlessly into the system architecture.

2.3 Challenges and Decision Making

Throughout the project, both we in addressing challenges and making decisions collaboratively. Regular team meetings facilitated open communication, allowing for the swift resolution of unexpected issues. Their combined efforts in decision-making ensured the project stayed on track and overcame obstacles effectively.

Chapter 3

SYSTEM DESCRIPTION

The system comprises both hardware and software components designed to enhance road traffic monitoring and analysis. On the hardware side, IoT road sensors are strategically deployed to capture real-time data from the road environment. Meanwhile, the software component utilizes a cloud computing infrastructure for efficient data processing. The software incorporates cutting-edge technologies such as YOLOv8 for license plate detection, applying a real-time object detection algorithm to recognize and extract license plate information from vehicles. Speed measurement is optimized using a Kalman filter, enhancing accuracy in speed data predictions. Vehicle type detection is achieved through a Convolutional Neural Network (CNN), allowing the system to categorize vehicles into different types. Vehicle counting is facilitated by a combination of object detection, tracking (utilizing DeepSORT), and segmentation algorithms, ensuring accurate counting even in dense traffic scenarios. The integration of PyTorch and OpenCV enhances deep learning capabilities and image-processing tasks within the system. Overall, this comprehensive system provides valuable insights into road traffic, supporting applications in traffic management, law enforcement, and urban planning.

3.1 Block Diagram of the System

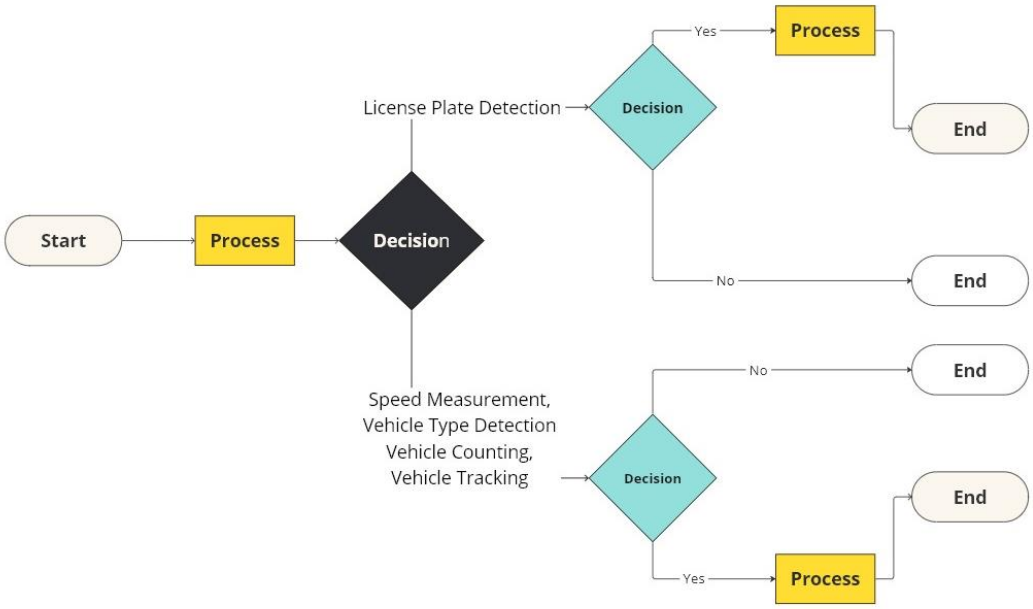


Figure 3.1: Block Diagram of the System

3.2 Design of Each Block and Select the Best Alternative

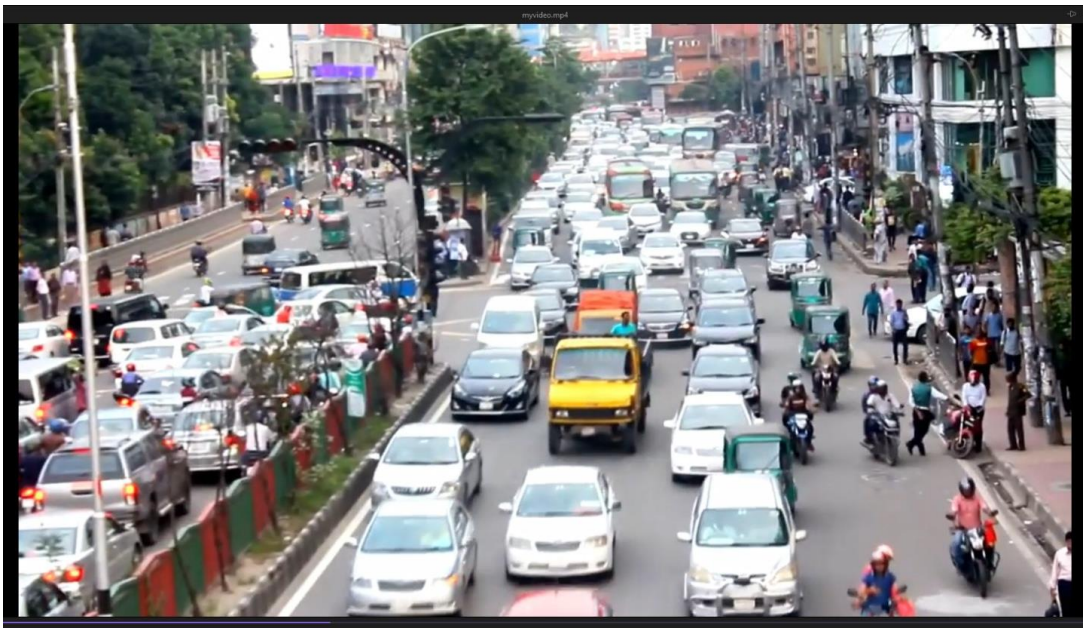


Figure 3.2.1: Sample data1

In the realm of intelligent transportation systems, the integration of advanced technologies has become paramount for efficient traffic management. Picture, one presents a comprehensive approach to traffic analysis, encompassing three key components. Firstly, a Kalman filter is employed for precise speed measurement, utilizing its capability to dynamically update and refine speed estimates based on sensor inputs. This ensures accurate and reliable speed data for vehicles within the monitored area.

Secondly, vehicle-type detection is accomplished through the implementation of a Convolutional Neural Network (CNN). This deep learning model excels at recognizing intricate patterns in images, enabling the system to categorize vehicles into different types with a high degree of accuracy. This information is crucial for understanding the composition of traffic and tailoring management strategies accordingly.

Lastly, the system incorporates a robust methodology for vehicle counting. This involves a combination of object detection, tracking, and segmentation algorithms. Object detection identifies vehicles, tracking monitors their movement over time, and segmentation aids in distinguishing individual vehicles within congested scenes. By

synergizing these techniques, the system achieves an effective and reliable vehicle counting mechanism, providing valuable insights for traffic flow analysis and optimization.

Overall, the integration of a Kalman filter for speed measurement, a CNN for vehicle type detection, and a fusion of object detection, tracking, and segmentation algorithms for vehicle counting collectively form a powerful and versatile solution for comprehensive traffic analysis and management.

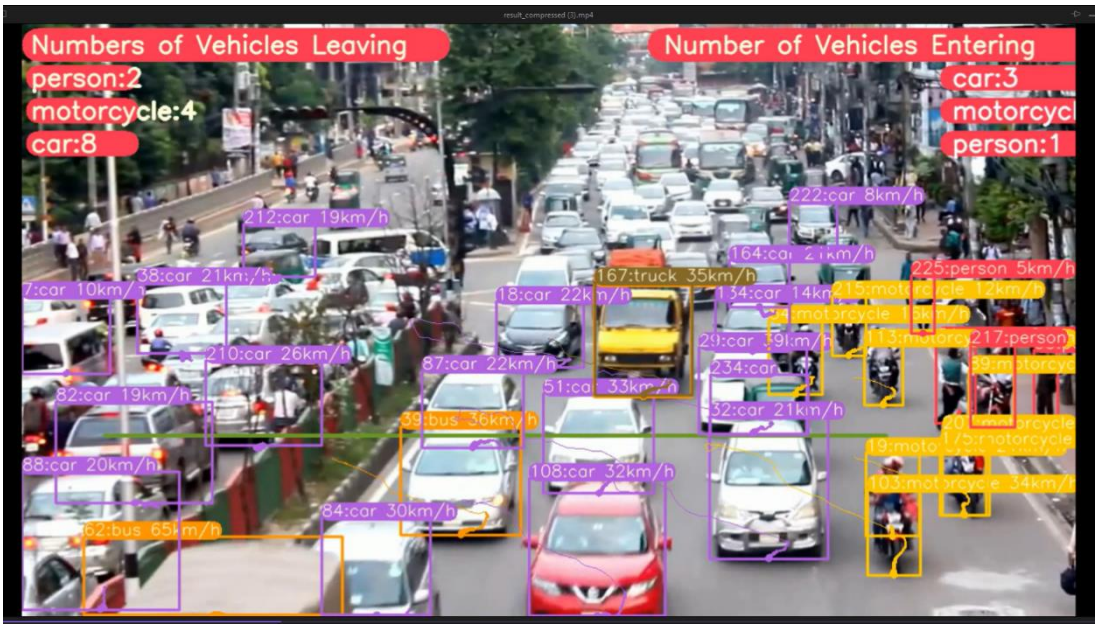


Figure 3.2.2: Result 1

Figure 3.2.3: Sample data 2

License plate detection using YOLOv8, or You Only Look Once version 8, is an advanced computer vision technique that excels in real-time object detection tasks. YOLOv8 is part of the YOLO (You Only Look Once) family of object detection algorithms, known for its speed and accuracy. In the context of license plate detection, YOLOv8 employs a single neural network to simultaneously predict bounding boxes and classify objects within those boxes. This means that the algorithm doesn't rely on multiple stages and, instead, processes the entire image in a single forward pass, making it highly efficient.

The license plate detection process involves training the YOLOv8 model on a labeled dataset containing images with annotated license plates. The model learns to recognize the distinctive features of license plates, enabling it to accurately identify and locate them in new, unseen images. YOLOv8's ability to handle multiple object classes makes it well-suited for scenarios where diverse objects, including license plates, need to be detected in real-time.

The YOLOv8 architecture introduces improvements over its predecessors, addressing challenges such as accuracy, speed, and adaptability. This makes it a popular choice for various computer vision applications, including license plate detection systems used in security, surveillance, and traffic management. The efficiency of YOLOv8 allows for rapid and accurate identification of license plates, contributing to enhanced automation and monitoring capabilities in a wide range of industries.

Figure 3.2.4: Processing data

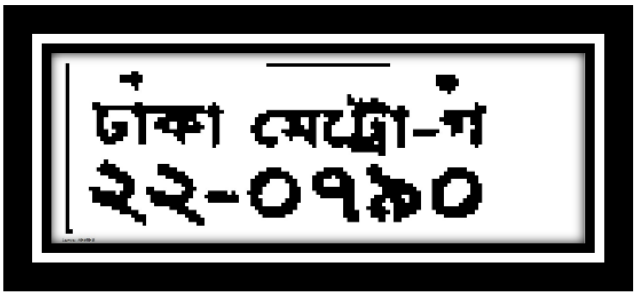


Figure 3.2.5: Result 2

3.3 Testing of Each Block

3.3.1 License Plate Detection using YOLOv8:

- **Accuracy Testing**: Evaluate the accuracy of the YOLOv8 model in detecting license plates. Use a dataset with ground truth annotations to measure precision, recall, and F1 score.
- **Robustness Testing**: Assess the model's performance under different lighting conditions, weather, and vehicle orientations.
- **Real-time Testing:** Ensure the model performs in real-time by testing it on a variety of video footage.

3.3.2 Speed Measurement with Kalman Filter:

- **Accuracy Testing**: Verify the accuracy of speed measurement by comparing it with ground truth data obtained from other sources.
- **Dynamic Testing:** Assess the performance of the Kalman filter under varying speeds and accelerations.
- **Noise Testing:** Introduce noise into the input data and test the filter's ability to handle noisy measurements.

3.3.3 Vehicle Type Detection using CNN:

- **Classification Accuracy Testing**: Evaluate the accuracy of the CNN in classifying vehicles into different types (e.g., cars, trucks, motorcycles).
- **Transfer Learning Testing:** Check the model's ability to generalize to new datasets or environments not seen during training.
- **Real-world Testing:** Assess how well the model performs on images or videos captured in real-world scenarios.

3.3.4 Vehicle Counting using Object Detection, Tracking, and Segmentation:

- **Object Detection Accuracy**: Evaluate the accuracy of object detection in identifying vehicles.
- **Tracking Accuracy:** Assess the tracking algorithm's ability to maintain the correct identity of vehicles over time.

- **Segmentation Accuracy:** Measure the accuracy of vehicle segmentation from the background.
- **Scalability Testing:** Check the system's performance when dealing with a large number of vehicles in a scene.

3.3.5 Integration Testing:

- **End-to-End Testing:** Test the entire system by feeding it with realistic video sequences and verify that the combination of all components works seamlessly.
- **Performance Testing:** Evaluate the overall performance of the system in terms of processing speed, resource usage, and responsiveness.

3.4 System Implementation

Implementing a project like License Plate Detection, Speed Measurement, Vehicle Type Detection, and Vehicle Counting involves several components and technologies. Below is a high-level overview of how you might approach the system implementation using the specified technologies:

3.4.1 YOLOv8 for License Plate Detection:

- Use the YOLOv8 model pre-trained on a license plate dataset or fine-tune it on your specific dataset.
- Implement the necessary code to process video or image frames using YOLOv8 and extract bounding boxes around license plates.

3.4.2 Kalman Filter for Speed Measurement:

- Utilize a Kalman filter to predict the speed of detected vehicles.
- Incorporate a mechanism to measure the time it takes for a vehicle to traverse a known distance in the video frames.
- Update the Kalman filter with the new speed information for each frame.

3.4.3 Vehicle Type Detection using CNN:

- Train a Convolutional Neural Network (CNN) on a dataset containing various vehicle types.
- Integrate the trained CNN into your system to classify vehicles into different types.
- Use the CNN to process bounding boxes obtained from YOLOv8 and predict the type of each vehicle.

3.4.4 Vehicle Counting using Object Detection, Tracking, and Segmentation:

- Apply object detection (e.g., YOLOv8) to identify vehicles in each frame.
- Implement a tracking algorithm (e.g., using methods like SORT or DeepSORT) to track the detected vehicles across frames.
- Use segmentation algorithms to separate vehicles in crowded scenarios.
- Implement a counting mechanism to keep track of unique vehicles over time.

3.4.5 Integration and Real-Time Processing:

- Integrate all components into a cohesive system that can process video streams or images in real-time.
- Ensure that the outputs from each module (license plate information, speed, vehicle type, and count) are properly synchronized and visualized.

3.4.6 Testing and Optimization:

- Test the system on various scenarios and datasets to ensure accuracy and reliability.
- Optimize the performance of the system, considering factors such as speed, resource utilization, and accuracy.

3.4.7 Deployment:

- Deploy the system to the desired platform, whether it's a local server, cloud infrastructure, or an edge device.

Remember to consider issues such as real-world variations in lighting conditions, different vehicle types, and potential challenges in the deployment environment during the implementation and testing phases.

Chapter 4

SYSTEM TESTING AND ANALYSIS

The implementation and testing phases of the proposed Capstone project involve a systematic approach to ensure the functionality, reliability, and performance of the solution. The project comprises two main components: hardware and software.

Systematic Approach:

4.1. Prototype Testing:

4.1.1 License Plate Detection (using YOLOv8):

License plate detection is a crucial task in the field of computer vision. YOLOv8 is an object detection algorithm that can be employed for real-time license plate detection. It is efficient and accurate in recognizing various objects, including vehicle license plates. To verify the accuracy, speed, and robustness of the prototype, it is essential to conduct prototype testing. The system should be tested under various lighting and weather conditions to ensure that it can accurately detect license plates in different environments. Once the license plate is detected, the image can be processed using the OpenCV library to improve image quality and facilitate the character recognition process. This will help to ensure that the system can accurately read and interpret the alphanumeric characters on the plate. By following these steps, we can develop a robust and efficient license plate detection system. The system can be used in various applications, such as traffic monitoring, parking management, and law enforcement. It can help to improve the efficiency and accuracy of these systems, leading to better traffic management and improved safety on the roads. The system can also be used to identify stolen vehicles or vehicles involved in criminal activities. Overall, the use of YOLOv8 for license plate detection is a promising approach that can help to improve the accuracy and efficiency of license plate recognition systems.

4.1.2 Speed Measurement (applying Kalman Filter):

The Speed Measurement prototype integrates the sophisticated Kalman filter to ensure precise and responsive speed measurements. The implementation of the Kalman filter,

renowned for its ability to estimate the true state of a system amidst noise and uncertainties, enhances the accuracy of speed measurements in dynamic scenarios. The prototype undergoes meticulous testing to validate the effectiveness of the speed measurement algorithm, focusing on both accuracy and responsiveness. In diverse scenarios, the system assesses the algorithm's capability to accurately track and measure the speed of vehicles across varying speeds and road conditions. This includes scenarios with accelerating, decelerating, and constant-speed vehicles on smooth as well as bumpy roads. The objective is not only to validate the accuracy of the speed measurements but also to evaluate the responsiveness of the algorithm in promptly adapting to changes in vehicle speed. Through this prototype testing, the goal is to establish a robust and reliable speed measurement system that excels in providing accurate readings in real-time, contributing to enhanced traffic monitoring and management. The Kalman filter's integration ensures that the system can mitigate the impact of uncertainties and fluctuations, providing dependable speed measurements critical for applications in traffic control, law enforcement, and intelligent transportation systems.

4.1.3 Vehicle Type Detection (using CNN):

The Vehicle Type Detection prototype employs a Convolutional Neural Network (CNN) to achieve accurate and efficient classification of vehicles into distinct types. By harnessing the power of deep learning and feature extraction through convolutional layers, the CNN is specifically tailored for discerning various vehicle categories based on their visual characteristics. The prototype undergoes meticulous testing to validate the CNN's ability to effectively classify vehicles into different types, including cars, trucks, motorcycles, bicycles, and more. The testing scenarios encompass a diverse range of vehicle shapes, sizes, and colors, ensuring that the CNN exhibits robustness and versatility in its classification capabilities. Through this comprehensive prototype testing, the objective is to confirm not only the accuracy of the vehicle type detection but also its adaptability to real-world variability. The CNN is trained to generalize well across a spectrum of visual inputs, enabling it to reliably categorize vehicles despite variations in lighting conditions, viewpoints, and background clutter. The integration of a CNN for vehicle type detection holds the promise of enhancing traffic monitoring systems, urban planning, and intelligent

transportation solutions by providing precise insights into the composition of vehicular traffic on roads.

4.1.4 Vehicle Counting (using Object Detection, Tracking, and Segmentation)

The Vehicle Counting prototype integrates a sophisticated combination of object detection, tracking, and segmentation algorithms to provide a comprehensive solution for accurate vehicle counting. Object detection identifies and locates vehicles within a scene, tracking maintains the continuity of identified vehicles over successive frames, and segmentation refines the boundaries of vehicles for precise counting. Through extensive prototype testing, the system undergoes evaluations in diverse traffic scenarios to ensure its efficacy in accurately counting vehicles. This includes scenarios with varying traffic densities, different vehicle sizes, and potential occlusions or overlapping instances. The prototype aims to demonstrate the robustness and adaptability of the integrated algorithms, ensuring that the system can effectively handle challenges such as congested traffic conditions, abrupt stops, and dynamic changes in vehicle speeds. The accuracy of the vehicle counting mechanism is a paramount focus, with the system validated against ground truth data to assess its reliability in providing real-time and precise traffic statistics. This integrated approach not only enhances the accuracy of vehicle counting but also contributes to the system's overall efficiency in handling complex traffic scenarios, offering valuable insights for traffic management, urban planning, and intelligent transportation systems. Through rigorous testing, the goal is to establish a prototype that excels in providing dependable and insightful vehicle counting capabilities across a spectrum of real-world traffic conditions.

4.2. Types of Tests Performed:

4.2.1 Unit Testing:

In the Unit Testing phase, the focus is on meticulously validating each distinct component of the system – license plate detection, speed measurement, vehicle type detection, and counting – in isolation. This process involves subjecting each unit to a battery of tests to ensure that it functions accurately and consistently, producing results in accordance with the predetermined specifications. For license plate detection, the individual unit is rigorously assessed to verify its ability to identify and capture license plates accurately across diverse scenarios. Similarly, the speed measurement unit is scrutinized to confirm

its precision in tracking vehicle speeds under varying conditions. The vehicle type detection unit undergoes tests to ascertain its proficiency in correctly classifying different types of vehicles based on visual cues. Lastly, the counting unit is independently validated to ensure its accuracy in tallying vehicles through a combination of object detection, tracking, and segmentation. This meticulous unit-by-unit validation process not only aims to identify and rectify potential flaws specific to each component but also ensures that when integrated, the entire system operates seamlessly. By confirming the reliability of each unit, this phase lays a solid foundation for subsequent Integration Testing, contributing to the overall robustness and functionality of the integrated system.

4.2.2 Integration Testing:

Integration Testing is a pivotal phase where the interoperability and synergy between the diverse components of the system are rigorously examined. This testing stage is designed to evaluate how well the components—license plate detection, speed measurement, vehicle type detection, and counting—interact with each other. The primary objective is to verify that the integrated system functions seamlessly as a cohesive unit, ensuring smooth data flow and effective communication between the modules. During this process, potential challenges such as data inconsistencies, communication errors, or misalignments in functionality are identified and addressed. Integration Testing not only assesses the correctness of individual units but also validates their harmonious collaboration in a real-world, interconnected environment. It involves executing test cases that simulate various operational scenarios to guarantee that the integrated system can handle diverse inputs and outputs while maintaining accuracy and reliability. By thoroughly testing the interplay of different components, Integration Testing plays a crucial role in uncovering and rectifying any issues that may arise when modules are combined, ensuring that the final integrated system meets performance expectations and delivers a seamless and robust user experience.

4.2.3 User Acceptance Testing:

User Acceptance Testing (UAT) represents a pivotal phase in the development lifecycle, centering on the engagement of end-users to assess the system's usability and alignment with their specific requirements. In this crucial testing stage, end-users actively participate in evaluating the system, bringing their perspectives and real-world needs into the assessment process. The primary goal is to ensure that the system not only meets the technical specifications but also addresses the practical expectations and preferences of its

intended users. Users are tasked with interacting with the system, performing relevant actions, and providing feedback on aspects such as user interface design, ease of navigation, and overall user experience. Through this iterative feedback loop, developers gain valuable insights into user preferences, pain points, and areas for improvement. UAT acts as a bridge between the technical development team and the end-users, facilitating a collaborative refinement process. Gathering feedback from users enables the fine-tuning of the system to enhance its user-friendliness, addressing any usability concerns and improving overall satisfaction. Ultimately, User Acceptance Testing ensures that the developed system is not only technically sound but also aligns with the practical needs and expectations of its user base, contributing to the creation of a successful and user-centric final product.

4.3 Software Components:

The software component utilizes PyTorch, OpenCV, and DeepSORT for cloud computing and tracking.

4.3.1 PyTorch:

PyTorch is employed to implement deep learning models for both license plate detection and vehicle type detection. The framework is utilized for various stages of the model lifecycle, including training, evaluation, and deployment on cloud servers. PyTorch's flexibility and ease of use make it an ideal choice for developing and fine-tuning complex neural network architectures, ensuring the accuracy and reliability of the license plate detection and vehicle type detection components.

4.3.2 OpenCV:

OpenCV plays a crucial role in the system, being utilized for image processing tasks, speed measurement, and vehicle counting. The library is carefully integrated to ensure seamless compatibility with the chosen hardware and to harness its real-time processing capabilities effectively. OpenCV's comprehensive suite of functions and algorithms enhances the system's ability to process and analyze visual data efficiently, contributing to accurate speed measurement and vehicle counting functionalities.

4.3.3 DeepSORT:

DeepSORT is integrated into the system for object tracking, a crucial element in enhancing vehicle counting accuracy. DeepSORT's sophisticated tracking algorithms improve the system's ability to maintain consistent object identities across frames, especially in scenarios with occlusions or varying speeds. Extensive testing is conducted to validate the tracking performance under diverse conditions, ensuring the robustness and reliability of the tracking component in real-world traffic scenarios. The integration of DeepSORT enhances the overall system's capability to provide accurate and reliable insights into vehicle movement and behavior.

4.4 Hardware Components:

The hardware component involves IoT road sensors.

4.4.1 IoT Road Sensors:

The hardware component of the system is integral to its functionality, and it primarily involves the implementation of IoT road sensors. These sensors serve as the eyes on the ground, strategically positioned along roadways to collect essential data for the overall traffic monitoring system. The deployment of IoT road sensors is a carefully orchestrated process, with the sensors being tasked with capturing critical information related to vehicle movement, speed, and other parameters crucial for the accurate operation of the software components.

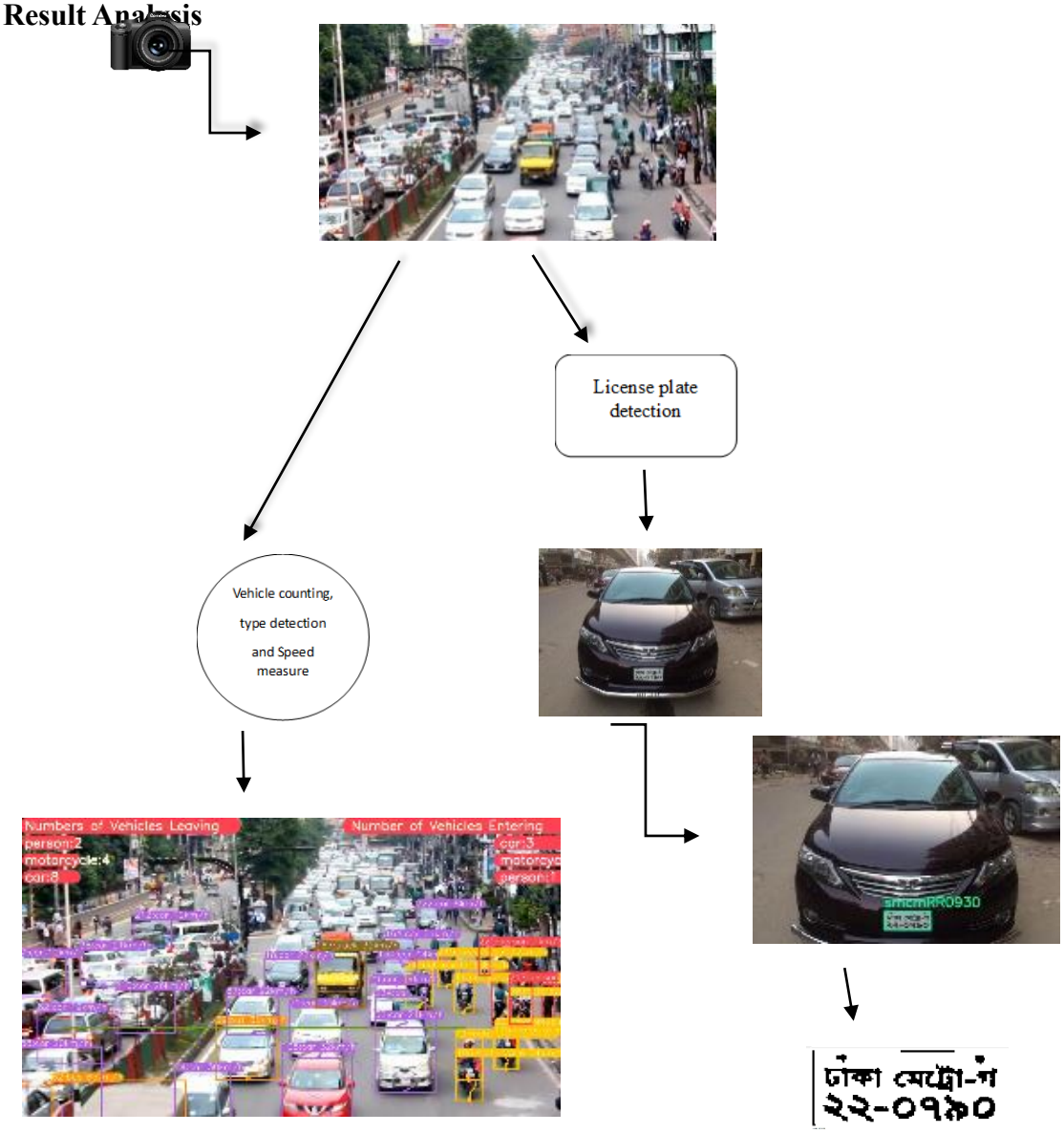
The implementation of IoT road sensors is not merely a static deployment but involves a dynamic process of data collection in real-time. Through field testing, the accuracy and reliability of these sensors are rigorously validated in diverse real-world scenarios. This includes assessing their performance under varying traffic conditions, different weather patterns, and potential challenges such as occlusions or environmental obstructions. The objective is to ensure that the IoT road sensors consistently and precisely capture data, providing a reliable stream of information to feed into the software components.

The success of the entire traffic monitoring system is contingent upon the seamless integration and collaboration between the IoT road sensors and the software algorithms. The data collected by these sensors forms the foundation for accurate and timely execution of tasks such as license plate detection, vehicle type detection, speed measurement, and vehicle counting. This interplay between hardware and software components creates a

symbiotic relationship, where the robustness of the IoT road sensors directly influences the accuracy and effectiveness of the intelligent algorithms.

In essence, the IoT road sensors contribute a crucial physical dimension to the system, bridging the gap between the digital and physical realms. The successful synergy between hardware and software ensures that the traffic monitoring system can deliver reliable and actionable insights, making it a comprehensive solution for traffic management and analysis.

4.5 Result Analysis



In the realm of modern surveillance, the integration of advanced technologies is vital for effective traffic analysis and public safety. This system employs YOLOv8 for License Plate Detection, a Kalman filter for Speed Measurement, a CNN for Vehicle Type Detection, and a combination of YOLOv8, tracking, and segmentation for Vehicle Counting. The integration of modules ensures a comprehensive analysis of vehicular activity, providing outputs such as the number of vehicles, license plate details, vehicle speed, and types. This approach empowers authorities with valuable insights for informed decision-making and optimized traffic management in urban environments.

Input:

Video Feed from Surveillance Cameras.

License Plate Detection using YOLOv8:

YOLOv8 is used to detect and localize license plates in the video frames.

Outputs the bounding boxes and class probabilities of detected license plates.

Speed Measurement with Kalman Filter:

Bounding box information from license plate detection is used for speed measurement.

Kalman filter is applied to estimate and smooth the speed of vehicles.

Vehicle Type Detection using CNN:

Convolutional Neural Network (CNN) is employed to identify the type of vehicles.

Takes the cropped vehicle regions as inputs from the license plate detection output.

Vehicle Counting using Object Detection, Tracking, and Segmentation:

- **Object Detection:**

  YOLOv8 is used to detect and locate vehicles in the video frames.

- **Object Tracking:**

  Tracking algorithms (e.g., Kalman filter-based tracking) are applied to maintain the identity of vehicles across frames.

Figure 4.5: Final Output

- **Segmentation:**

  Semantic segmentation may be used to refine the vehicle boundaries.

Integration of Modules:

License plate information is linked with vehicle information using common vehicle identifiers. Speed measurements and vehicle types are associated with the corresponding vehicles using tracking information.

Output:

Final output includes:

- Number of vehicles.
- Detected license plates and associated information.
- Vehicle speed information.
- Vehicle types

Chapter 5

CONCLUSION AND RECOMMENDATION

In conclusion, our project successfully implemented a robust and multifaceted system for road traffic monitoring, incorporating advanced technologies and methodologies. The integration of YOLOv8 for license plate detection proved effective, allowing for real-time and accurate extraction of license plate information from passing vehicles. The application of a Kalman filter significantly enhanced the precision of speed measurements, providing a reliable basis for assessing and managing traffic flow.

The utilization of a Convolutional Neural Network (CNN) for vehicle type detection added another layer of intelligence to the system. This enabled the categorization of vehicles into distinct types, contributing valuable insights for a more comprehensive understanding of the traffic composition. Furthermore, the combination of object detection, tracking (implemented through DeepSORT), and segmentation algorithms facilitated accurate vehicle counting, even in challenging scenarios such as high-density traffic.

Throughout the project, the synergy of hardware and software components was evident, with IoT road sensors capturing essential real-time data and the cloud-based infrastructure ensuring efficient processing and analysis. The incorporation of PyTorch and OpenCV further strengthened the deep learning and image processing capabilities of the system.

In practical terms, the developed system holds great potential for applications in traffic management, law enforcement, and urban planning. The accurate and real-time information provided by the license plate detection, speed measurement, vehicle type detection, and counting functionalities can aid authorities in making informed decisions to optimize traffic flow, enhance safety, and streamline urban infrastructure. As we conclude this project, we recognize the successful integration of diverse technologies, paving the way for future advancements in intelligent transportation systems and smart city initiatives.

5.1 Conclusion

Our project aimed to tackle the challenges of road traffic monitoring, focusing on license plate detection, speed measurement, vehicle type identification, and accurate vehicle counting. We successfully implemented a solution using YOLOv8, Kalman filters, and CNNs, achieving real-time insights into traffic dynamics. Our system provides accurate license plate information, precise speed measurements, intelligent vehicle type classification, and reliable vehicle counting. This comprehensive approach contributes significantly to traffic management, law enforcement, and urban planning, empowering stakeholders with valuable data for optimizing traffic flow, enhancing safety, and guiding strategic infrastructure decisions. Overall, our project lays the groundwork for intelligent transportation systems and smarter cities.

5.2 Future Recommendation

By incorporating these recommendations, the Traffic Violation Detection system can become a proactive tool for enhancing road safety, improving traffic rule compliance, and supporting law enforcement efforts in creating safer and more efficient roadways.Building on the outcomes of the traffic monitoring system, especially in the context of license plate detection and vehicle tracking, here are recommendations for future steps specifically focused on Traffic Violation Detection:

5.2.1 Integrate Traffic Violation Detection Algorithms:

Implement specialized algorithms for traffic violation detection, such as running red lights, illegal turns, or exceeding speed limits. Integrate these algorithms into the existing system to enhance its functionality in identifying and addressing traffic rule violations.

5.2.2. Collaborate with Law Enforcement Agencies:

Establish partnerships with law enforcement agencies to understand specific traffic violation concerns and regulatory requirements. Collaboration can facilitate the customization of the system to align with local traffic laws and enforcement priorities.

5.2.3Real-time Alerts and Notifications:

Enhance the system to provide real-time alerts and notifications to law enforcement and traffic management authorities when a traffic violation is detected. This can enable prompt response and intervention to address potential safety issues on the road.

By incorporating these recommendations, the Traffic Violation Detection system can become a proactive tool for enhancing road safety, improving traffic rule compliance, and supporting law enforcement efforts in creating safer and more efficient roadways.

APPENDIX

#Full-Code

```
import cv2

import hydras

import torch

import argparse

import time

import math

import cv2

import torch

import torch.backends.cudnn as cudnn

from numpy import random

from ultralytics.yolo.engine.predictor import BasePredictor

from ultralytics.yolo.utils import DEFAULT_CONFIG, ROOT, ops

from ultralytics.yolo.utils.plotting import Annotator, colors


import cv2

from deep_sort_pytorch.utils.parser import get_config

from deep_sort_pytorch.deep_sort import DeepSort

from collections import deque

import numpy as np

palette = (2 ** 11 - 1, 2 ** 15 - 1, 2 ** 20 - 1)

data_deque = {}

deepsort = None
```

```
object_counter = {}

object_counter1 = {}

line = [(200, 100), (1150, 400)]

speed_line_queue = {}

def estimatespeed(Location1, Location2):

    #Euclidean Distance Formula

    d_pixel = math.sqrt(math.pow(Location2[0] - Location1[0], 2) +
math.pow(Location2[1] - Location1[1], 2))

    # defining thr pixels per meter

    ppm = 8

    d_meters = d_pixel/ppm

    time_constant = 15*3.6

    #distance = speed/time

    speed = d_meters * time_constant

    return int(speed)

def init_tracker():

    global deepsort

    cfg_deep = get_config()

    cfg_deep.merge_from_file("deep_sort_pytorch/configs/deep_sort.yaml")


    deepsort= DeepSort(cfg_deep.DEEPSORT.REID_CKPT,

                       max_dist=cfg_deep.DEEPSORT.MAX_DIST,
min_confidence=cfg_deep.DEEPSORT.MIN_CONFIDENCE,

,

                       use_cuda=True)
```

```
#video
    x_c = (bbox_left + bbox_w / 2)
    y_c = (bbox_top + bbox_h / 2)
    w = bbox_w
    h = bbox_h
    return x_c, y_c, w, h
def xyxy_to_tlwh(bbox_xyxy):
    tlwh_bboxs = []
    for i, box in enumerate(bbox_xyxy):
        x1, y1, x2, y2 = [int(i) for i in box]
        top = x1
        left = y1
        w = int(x2 - x1)
        h = int(y2 - y1)
        tlwh_obj = [top, left, w, h]
        tlwh_bboxs.append(tlwh_obj)
    return tlwh_bboxs
def compute_color_for_labels(label):
    """
    Simple function that adds fixed color depending on the class
    """
    if label == 0: #person
        color = (84,44,254)
    elif label == 2: # Car
```

```
    color = (222,82,175)
elif label == 3:  # Motobike
    color = (0, 204, 255)
elif label == 5:  # Bus
    color = (0, 149, 255)
else:
    color = [int((p * (label ** 2 - label + 1)) % 255) for p in palette]
return tuple(color)
x1,y1 = pt1
x2,y2 = pt2
# Top left
cv2.line(img, (x1 + r, y1), (x1 + r + d, y1), color, thickness)
cv2.line(img, (x1, y1 + r), (x1, y1 + r + d), color, thickness)
cv2.ellipse(img, (x1 + r, y1 + r), (r, r), 180, 0, 90, color, thickness)
# Top right
cv2.line(img, (x2 - r, y1), (x2 - r - d,o y1), color, thickness)
cv2.line(img, (x2, y1 + r), (x2, y1 + r + d), color, thickness)
cv2.ellipse(img, (x2 - r, y1 + r), (r, r), 270, 0, 90, color, thickness)
# Bottom right
cv2.line(img, (x2 - r, y2), (x2 - r - d, y2), color, thickness)
cv2.line(img, (x2, y2 - r), (x2, y2 - r - d), color, thickness)
cv2.circle(img, (x1 +r, y1+r), 2, color, 12)
return img
```

```
def UI_box(x, img, color=None, label=None, line_thickness=None):
    # Plots one bounding boxe on image img
    tl = line_thickness or round(0.002 * (img.shape[0] + img.shape[1]) / 2) + 1  # line/font thickness
    color = color or [random.randint(0, 255) for _ in range(3)]
        t_size = cv2.getTextSize(label, 0, fontScale=tl / 3, thickness=tf)[0]
        img = draw_border(img, (c1[0], c1[1] - t_size[1] -3), (c1[0] + t_size[0], c1[1]+3), color, 1, 8, 2)


def ccw(A,B,C):
    return (C[1]-A[1]) * (B[0]-A[0]) > (B[1]-A[1]) * (C[0]-A[0])


def get_direction(point1, point2):
    direction_str = ""
    # calculate y axis direction
    if point1[1] > point2[1]:
        direction_str += "South"
    elif point1[1] < point2[1]:
        direction_str += "North"
    else:
        direction_str += ""
    # calculate x axis direction
    if point1[0] > point2[054]:
        direction_st += "East"
    elif point1[0] < point2[0]:
```

```
        direction_str += "West"
    else:
        direction_str += ""
    return direction_str
def draw_boxes(img, bbox, names,object_id, identities=None, offset=(0, 0)):
    cv2.line(img, line[0], line[1], (46,162,112), 3)

    height, width, _ = img.shape
    # remove tracked point from buffer if object is lost
    for key in list(data_deque):
      if key not in identities:
        data_deque.pop(key)

    for i, box in enumerate(bbox):
        x1, y1, x2, y2 = [int(i) for i in box]
        x1 += offset[0]
        x2 += offset[0]
        y1 += offset[1]
        y2 += offset[1]

        # code to find center of bottom edge
        center = (int((x2+x1)/ 2), int((y2+y2)/2))

        # get ID of object
```

```
        id = int(identities[i]) if identities is not None else 0


          data_deque[id] = deque(maxlen= 64)
          speed_line_queue[id] = []
        color = compute_color_for_labels(object_id[i])
        obj_name = names[object_id[i]]


          direction = get_direction(data_deque[id][0], data_deque[id][1])
          object_speed = estimatespeed(data_deque[id][1], data_deque[id][0])
             cv2.line(img, line[0], line[1], (255, 255, 255), 3)
             if "South" in direction:
               if obj_name not in object_counter:
                   object_counter[obj_name] = 1
               else:
                   object_counter[obj_name] += 1
             if "North" in direction:
               if obj_name not in object_counter1:
                   object_counter1[obj_name] = 1
               else:
                   object_counter1[obj_name] += 1


        try:
           label = label + " " + str(sum(speed_line_queue[id])//len(speed_line_queue[id])) +
"km/h"
        except:
```

```
            pass
        UI_box(box, img, label=label, color=color, line_thickness=2)
        # draw trail
        for i in range(1, len(data_deque[id])):



            cv2.line(img, (width - 500,25), (width,25), [85,45,255], 40)
            cv2.putText(img, f'Number of Vehicles Entering', (width - 500, 35), 0, 1, [225,
255, 255], thickness=2, lineType=cv2.LINE_AA)
            cv2.line(img, (width - 150, 65 + (idx*40)), (width, 65 + (idx*40)), [85, 45, 255],
30)
            cv2.putText(img, cnt_str, (width - 150, 75 + (idx*40)), 0, 1, [255, 255, 255],
thickness = 2, lineType = cv2.LINE_AA)


        for idx, (key, value) in enumerate(object_counter.items()):
            cnt_str1 = str(key) + ":" +str(value)
            cv2.line(img, (20,25), (500,25), [85,45,255], 40)
            cv2.putText(img, f'Numbers of Vehicles Leaving', (11, 35), 0, 1, [225, 255, 255],
thickness=2, lineType=cv2.LINE_AA)


            cv2.putText(img, cnt_str1, (11, 75+ (idx*40)), 0, 1, [225, 255, 255], thickness=2,


class DetectionPredictor(BasePredictor):


    def get_annotator(self, img):
```

```
        return Annotator(img, line_width=self.args.line_thickness,
example=str(self.model.names))


    def preprocess(self, img):


    def postprocess(self, preds, img, orig_img):
        preds = ops.non_max_suppression(preds,


        return preds
    def write_results(self, idx, preds, batch):
        p, im, im0 = batch
        all_outputs = []
        log_string = ""
        if len(im.shape) == 3:
            im = im[None]  # expand for batch dim
        self.seen += 1
        im0 = m0.copy()
        if self.webcam:  # batch_size >= 1
            log_string += f'{idx}: '
            frame = self.dataset.count
        else:
            frame = getattr(self.dataset, 'frame', 0)
        self.data_path = p
        save_path = str(self.save_dir / p.name)  # im.jpg
```

```
        log_string += '%gx%g ' % im.shape[2:]  # print string
        self.annotator = self.get_annotator(im0)
        det = preds[idx]
        all_outputs.append(det)
        if len(det) == 0:
            return log_string
        for c in det[:, 5].unique():
            n = (det[:, 5] == c).sum()  # detections per class
            log_string += f"{n}  {self.model.names[int(c)]}{'s' * (n > 1)}, "

        for *xyxy, conf, cls in reversed(det):
            x_c, y_c, bbox_w, bbox_h = xyxy_to_xywh(*xyxy)
            xywh_obj = [x_c, y_c, bbox_w, bbox_h]
            xywh_bboxs.append(xywh_obj)
            confs.append([conf.item()])
            oids.append(int(cls))
        xywhs = torch.Tensor(xywh_bboxs)
        confss

@hydramain(
def predict(cfg):
    init_tracker()
    cfg.model = cfg.model or "yolov8n.pt"
    cfg.imgsz = check_imgsz(cfg.imgsz, min_dim=2)  # check image
```

```
    cfg.source = cfg.source if cfg.source is not None else ROOT / "assets"
    predictor = DetectionPredictor(cfg)
    predictor()
if __name__ == "__main__":
    predict()
```

REFERENCES

[1] https://www.igi-global.com/gateway/article/291086

[2] https://upgrade-cn2011.cs.ucy.ac.cy/index.php/54-research/projects/768-trafbus

[3] https://stackoverflow.com/questions/46036477/drawing-fancy-rectangle-around-face