Visvesvaraya Technological University
Belgaum, Karnataka- 590014

A Project Report On

"SMART TRAFFIC SURVEILLANCE SYSTEM USING RCNN AND KALMAN FILTER"

Submitted in the partial fulfilment of the requirements for the award of the Degree of

BACHELOR OF ENGINEERING

In

INFORMATION SCIENCE AND ENGINEERING

Presented By:-
Mohit Chordia (1DS14IS056)
Onam C Bhartia (1DS14IS066)
Paritosh Anand (1DS14IS067)
Ybhav M (1DS14IS122)

Under The Guidance of:-
Mrs. Latha A P
Assistant Professor,
Department of ISE,DSCE

2016-2017

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

DAYANANDA SAGAR COLLEGE OF ENGINEERING

SHAVIGE MALLESHWARA HILLS, KUMARASWAMY LAYOUT, BANGALORE-78

DAYANANDA SAGAR COLLEGE OF ENGINEERING

Shavige Malleshwara Hills, Kumaraswamy Layout

Bangalore-560078

Department of Information Science and Engineering

2017-2018

Certificate

This is to certify that the Project Work entitled **—"SMART TRAFFIC SURVEILLANCE SYSTEM USING RCNN AND KALMAN FILTER"** is a bonafide work carried out by **Mohit Chordia** (1DS14IS056), **Onam C Bhartia** (1DS14IS066), **Paritosh Anand** (1DS14IS067), **Ybhav M** (1DS14IS122) in partial fulfilment for the 8th semester of Bachelor of Engineering in Information Science & Engineering of the Visvesvaraya Technological University, Belgaum during the year 2017-2018. The Project Report has been approved as it satisfies the academics prescribed for the Bachelor of Engineering degree.

_____	_____	_____
Signature of Guide	Signature of HOD	Signature of Principal
[Mrs. Latha A P]	[**Dr. K.N. Rama Mohan Babu**]	[Dr C.P.S Prakash]

Name of the Examiners	Signature with Date
1._____	_____
2._____	_____

ACKNOWLEDGEMENT

It is great pleasure for us to acknowledge the assistance and support of a large number of individuals who have been responsible for the successful completion of this project.

We take this opportunity to express our sincere gratitude to **Dayananda Sagar College of Engineering** for having provided us with a great opportunity to pursue our Bachelor Degree in this institution.

In particular we would like to thank **Dr. C. P. S Prakash,** Principal, Dayananda Sagar College of Engineering for his constant encouragement and advice.

Special thanks to **Dr. K.N. Rama Mohan Babu,** HOD, Department of Information Science & Engineering, Dayananda Sagar College of Engineering for his motivation and invaluable support well through the development of this project.

We are highly indebted to our internal guide **Mrs. Latha A P,** Asst. Professor,
Department of Information Science & Engineering, Dayananda Sagar College of Engineering for her constant support and guidance. He has been a great source of inspiration throughout the course of this project.

Finally, we gratefully acknowledge the support of our families during the completion of the project.

Mohit Chordia (1DS14IS056)
Onam C Bhartia (1DS14IS066)
Paritosh Anand (1DS14IS067)
Ybhav M (1DS14IS122)

ABSTRACT

Detecting the objects in the video and tracking their motion to identify their characteristics has been emerging as a demanding research area with implementation like vehicle tracking.

An important large number of applications in diverse disciplines are employed for change detection in its work, such as video surveillance, medical diagnosis and treatment, remote sensing, underwater sensing and civil infrastructure. One of the video surveillance branches is the traffic image analysis which included the moving/motion vehicle detection and segmentation approaches.

Due to increase in the number of day to day vehicle there is not enough manpower to support manual traffic surveillance. To overcome this, significant application of video-based supervision system using artificial intelligence is brought in the traffic surveillance. The proposed system will track vehicles and flag any possible traffic rule breakers.

In this project we will be using artificial neural network for vehicle tracking. This neural network is a derivative of Convolutional Neural Network. The rule conforming part of the system uses application of Kalman Filter to decide whether the vehicle is following traffic rules or not.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

To take advantage of the video in real-time, human must monitor the system continuously in order to alert security officers if there is an emergency. The need for intelligent video surveillance systems which can monitor and respond to situation in real time have increased due to the high-cost and low efficiency of the existing surveillance system.

1.1 Overview:

For the past two decades, because of the development of computer technologies and computer vision, target tracking has evolved in many ways and has spread out to a large number of areas from traffic control to radar missile targeting. We will focus here on moving vehicle tracking and track the drivers who are not abiding to the traffic rules. Due to increase in the number of day to day vehicle there is not enough manpower to support it. To prevent the people from this problem one solution is to increase the significant application of video-based supervision system that is traffic surveillance. Traffic flow is the study of interactions between vehicles, drivers, and background obstacles such as road signals, trees and weather conditions. Vehicle classification in a traffic flow is considered a difficult task due to similarity in appearances among different vehicles.

The detection of moving object's regions of change in the same image sequence which captured at different intervals is one of interested fields in computer vision. An important large number of applications in diverse disciplines are employed the change detection in its work, such as video surveillance, medical diagnosis and treatment, remote sensing, underwater sensing and civil infrastructure. One of the video surveillance branches is the traffic image analysis which included the moving/motion vehicle detection and segmentation approaches. Even though various research papers have been showed for moving vehicle detection (background subtraction, frame differencing and motion based methods) but still a tough task to detect and segment the vehicles in the dynamic scenes.

The success or failure of any tracking algorithm depends a lot on the degree that the tracked object can be distinguished from its surroundings. In particular, the set of features used by the tracking algorithm to represent the object(s) being tracked plays a major role in tracking performance.

In many public places such as airport, parking lots, train stations, and banks there is need of surveillance to prevent the accident or harmful incident so that surveillance cameras are installed in such places. To take advantage of the video in real-time, human must monitor the system continuously in order to alert security offers if there is an emergency. The need for intelligent video surveillance systems which can monitor and respond to situation in real time have increased due to the high-cost and low efficiency of the existing surveillance system. Object tracking having aim to obtain a record of the moving object one or more targets over time and space. By locating and

tracking moving objects in a video sequence in real time, we can develop a real time alert system to enhance current Surveillance system.

To be an effective traffic surveillance tool, whether by mimicking loop detectors or actually tracking vehicles, a video image processing system (VIPS) should meet several stringent requirements like Real-time vehicle detection and feature extraction, detection of anomalies with high accuracy and cost efficiency. Even though a number of commercial VIPS for monitoring traffic have been introduced to the market, many of these criteria still cannot be met.

The task of reliably detecting and tracking moving objects in surveillance video, which forms a basis for higher level intelligence applications, has many open questions. Our work focuses on developing a framework to detect moving objects and generate reliable tracks from real-world surveillance video. After setting up a basic system that can serve as a platform for further automatic tracking research, we tackle the question of variation in distances between the camera and the objects in different parts of the scene (object depth) in surveillance videos. A feedback-based solution to automatically learn the distance variation in static camera video scenes is implemented based on object motion in different parts of the scene.

1.2 Problem Statement:

Even though a number of commercial video image processing systems for monitoring traffic have been introduced to the market, many important criteria still cannot be met. Following is a list of such important criteria which this project has incorporated:

1. Automatic segmentation of each vehicle from the background and from other vehicles so that all cars are detected.

2. Correctly detect all types of cars.

3. Function under a wide range of traffic conditions such as light traffic, congestion, varying speeds in different lanes.

4. Operate in real-time.

1.3 Objectives:

1. Correctly detect all types of cars on the road.

2. Function under a wide variety of lighting conditions such as sunny, overcast, twilight etc. and for dynamic backgrounds and different viewing angles.

3. Detect inconsistent drivers using anomaly detection to flag possible traffic rule breaking.

CHAPTER 2

LITERATURE SURVEY

2.1 Material referred

Various Books and information materials from the web regarding Object tracking and detection have been studied through in order to achieve the required information concern to this project. Among them, following are the key points extracted through:

1. S.Srilekha, G.N.Swamy and A.Anudeep Krishna, "A Novel approach for Detection and Tracking of Vehicles using Kalman filter" in CICN 2015. DOI:10.1109/CICN.2015.53.
   This paper deals with: Prediction of vehicle location for a small time span
   Advantage: Highly accurate prediction for normal conditions
   Disadvantages: Susceptible to background noise, Cannot manage dynamic background pixels.

2. Hui Li, Peirui Bai and Huajun Song, "Car tracking algorithm based on Kalman filter and compressive tracking" in CISP 2014. DOI: 10.1109/CISP.2014.7003744.
   This paper deals with: Detailed survey of tracking using Compressive Tracking and overcoming partial occlusion using Kalman Filter(prediction).
   Advantages: Higher level of accuracy, Works for different viewing angles, Robust in dynamic background conditions.
   Disadvantages: Not accurate enough for practical applications, Doesn't use Kalman filter for vehicle classification, Cannot overcome complete occlusion.

3. Hiroshi Unno, Kouki Ojima and Kelkichi Hayashibe, "Vehicle Motion Tracking Using Symmetry of Vehicle and Background Subtraction" in Intelligent Vehicles Symposium 2007. DOI: 10.1109/IVS.2007.4290269.
   This paper deals with: Tracking vehicle motion
   Advantage: Proposed algorithm enables stable vehicle motion tracking in video frames
   Disadvantage: The experiments yielded results under the limited conditions.

4. Mao Shan, Stewart Worrall and Eduardo Nebot, "Long term vehicle motion prediction and tracking in large environments" in ITSC 2011. DOI: 10.1109/ITSC.2011.6082922.
   This paper deals with: Predicting and tracking vehicle position.
   Advantage: long term vehicle prediction and tracking in a large unstructured area.
   Disadvantage: Delay in observing vehicle position.

5. D.M. Gavrila and V. Philomin, "Real-time object detection for smart vehicles", in ICCV.1999. DOI: 10.1109/ICCV.1999.791202.
This paper deals with: An efficient, shape-based object detection method based on Distance Transforms and describes its use for real-time vision on-board vehicles.
Advantage: A method for shape-based object detection using Distance Transforms, which takes a combined coarse-to-fine approach in shape and parameter space, incorporating a multi-stage segmentation technique as well.
Disadvantage: Although we dealt with a sizeable amount of shape variation when considering pedestrian shapes, this method might be not the most appropriate to detect pedestrians very close to the camera when shape variations become even larger.

6. Wang Zhiqiang and Liu Jun, “A review of object detection based on convolutional neural network” in CCC 2017. DOI: 10.23919/ChiCC.2017.8029130.
This paper deals with: Problem in the current CNN based Object detection and probable means to improve the performance.
Advantages:
1) Region proposal greatly decreases time-complexity of the next actions and quality of region proposals are higher than sliding windows.
2) Fast R-CNN uses multi-task loss on each labeled Region of Interest(RoI) to jointly train for classification and bounding-box.
Disadvantages:
1) The mean average precision using the hand-crafted technique is 22.58% as compare to 43.93% using R-CNN.
2) SPP-net cannot update the convolutional layers while training.

7. Ross Girshick, "Fast R-CNN", in ICCV 2015. DOI: 10.1109/ICCV.2015.169.
This paper deals with: Fast Region-based Convolutional Network method (Fast R-CNN) for object detection.
Advantage: Sparse object proposals appear to improve detector quality.
Disadvantage: The time it take for the regional proposal method is too long, and the task of getting regional proposal is a bottleneck.

8. Lu Ming, “Image segmentation algorithm research and improvement” in ICACTE 2010. DOI: 10.1109/ICACTE.2010.5579114.
This paper deals with: Image Segmentation techniques and algorithm.
Advantage: OTSU method in the process of selecting threshold is essentially a process of finding the optimal solution to achieve greater efficiency.
Disadvantage: In the genetic algorithm, the selection of crossover probability and mutation probability is the key to affect the algorithm behavior and performance, which directly affects the convergence of the algorithm.

9. Xinyi Zhou, Wei Gong, WenLong Fu and Fengtong Du, “Application of deep learning in object detection” in ICIS 2017. DOI:10.1109/ICIS.2017.7960069.0
   This paper deals with: Data Sets and types of Neural network.
   Advantages: The use of the Faster R-CNN on the new datasets has given great success in the technology of deep learning in image classification, object detection, face identification and many other vision tasks.
   Disadvantage: Uneven quantity and uneven size during the process of image lowers the accuracy of recognition.

10. Madalina Cosmina Popescu and Lucian Mircea Sasu, "Feature extraction ,feature selection and machine learning for image classification: A case study", in OPTIM 2014. DOI: 10.1109/OPTIM.2014.6850925.
    This paper deals with: The background extraction of the Object to be tracked.
    Advantages: The two classifiers (evolutionary and genetic algorithms) with the largest number of good results rely on local models, building predictions based on data that cluster together.
    Disadvantages: Greedy search, best first search with backward search and principal component analysis should be avoided for this problem, as their induced performance is constantly poor across all classifiers.

2.2 Video surveillance in the context of Computer vision

Detection and tracking of moving objects are the important tasks of the computer vision.
The video surveillance systems not only need to track the moving objects but also interpret their patterns of behaviours. This means solving the information and integration the pattern.

- **Advantages**
- Minimizes the user interaction.
- Less amount of prohibitive bandwidth.
- Minimizes the cost and time.

2.3 Motivation

- Due to increase in the number of day to day vehicle there is not enough manpower to support it. To prevent people from this problem one solution is to increase the significant application of video-based supervision system that is traffic surveillance.
- In many public places such as airport, parking lots, train stations, and banks there is need of surveillance to prevent accident or harmful incident so that surveillance cameras are installed in such places.
- Solving the problems like Real-time vehicle detection and feature extraction, detection of anomalies with high accuracy and cost efficiency present in existing surveillance systems.

CHAPTER 3

REQUIREMENT

3.1 Functional Requirements

- The User should be able to view the detected cars in the video feed in real time.
- The User should be able to view speed of the vehicles detected in the video feed.
- The User should get to know if any traffic violation is done by any vehicle in the Video feed.

3.2 Non Functional Requirements

- User Interface should be easily accessible.
- The vehicle detection should occur in near real time.
- The Application should not fail in the middle of operation.

3.3 Software Requirements:

- OS version: Windows 10 (64-bit)
- Coding Language: Python, C++
- Python version: 2.7
- IDE: MATLAB (R2018a)

3.4 Hardware Requirement

- Processor: Intel Core™ i5
- Hard Disk: 500GB
- RAM: 8GB
- GPU: Compute Power 3.0 or more

CHAPTER 4

SYSTEM ANALYSIS & DESIGN

4.1 Analysis

Analysis is the process of breaking a complex topic or substance into smaller parts to gain a better understanding of the problem. Analysts in the field of engineering look at requirements, structures, mechanisms, and systems dimensions. Analysis is an exploratory activity. The Analysis Phase is where the project lifecycle begins.

The Analysis Phase is where we break down the deliverables in the high-level Project Charter into the more detailed requirements. The Analysis Phase is also the part of the project where we identify the overall direction that the project will take through the creation of the project strategy.

4.2 Definitions, Acronyms, and Abbreviations

- **ANN**

  Artificial neural networks (ANNs) or connectionist systems are computing systems vaguely inspired by the biological neural networks that constitute animal brains. Such Systems learn (i.e. progressively improve performance on) tasks by considering examples, generally without task-specific programming.

- **CNN:**

  In machine learning, a convolutional neural network (CNN or ConvNet) is a class of deep, feed-forward artificial neural networks that has successfully been applied to analyzing visual imagery.

  CNNs use a variation of multilayer perceptrons designed to require minimal pre-processing. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics.

- **R-CNN:**

  The goal of R-CNN is to take in an image, and correctly identify where the main objects (via a bounding box) in the image. But how do we find out where these bounding boxes are? R-CNN does what we might intuitively do as well - propose a bunch of boxes in the image and see if any of them actually correspond to an object.

- **Faster R-CNN:**

  Faster R-CNN has two networks: region proposal network (RPN) for generating region proposals and a network using these proposals to detect objects. The main different here with Fast R-CNN is that the later uses selective search to generate region proposals. The time cost of generating region proposals is much smaller in RPN than selective search, when RPN shares the most computation with the object detection network. Briefly, RPN ranks region boxes (called anchors) and proposes the ones most likely containing objects.

- **Kalman Filter:**

  Kalman filtering, also known as linear quadratic estimation (LQE), is an algorithm that uses a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more accurate than those based on a single measurement alone, by estimating a joint probability distribution over the variables for each timeframe. The filter is named after Rudolf E. Kalman, one of the primary developers of its theory

4.3 System Design

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could see it as the application of systems theory to product development.

The objective of the design phase is to produce overall design of the software. It aims to figure out the modules that should be in the system to fulfil all the system requirements in an efficient manner. The design will contain the specification of all these modules, their interaction with other modules and the desired output from each module. The output of the design process is a description of the software architecture. The design phase is followed by two sub phases

- High Level Design
- Detailed Level Design

4.4 System Architecture Diagram

System architecture is a conceptual model that defines the structure, behaviour, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviours of the system. A system architecture can comprise system components that will work together to implement the overall system. The below figure shows a general block diagram describing the activities performed by this project.

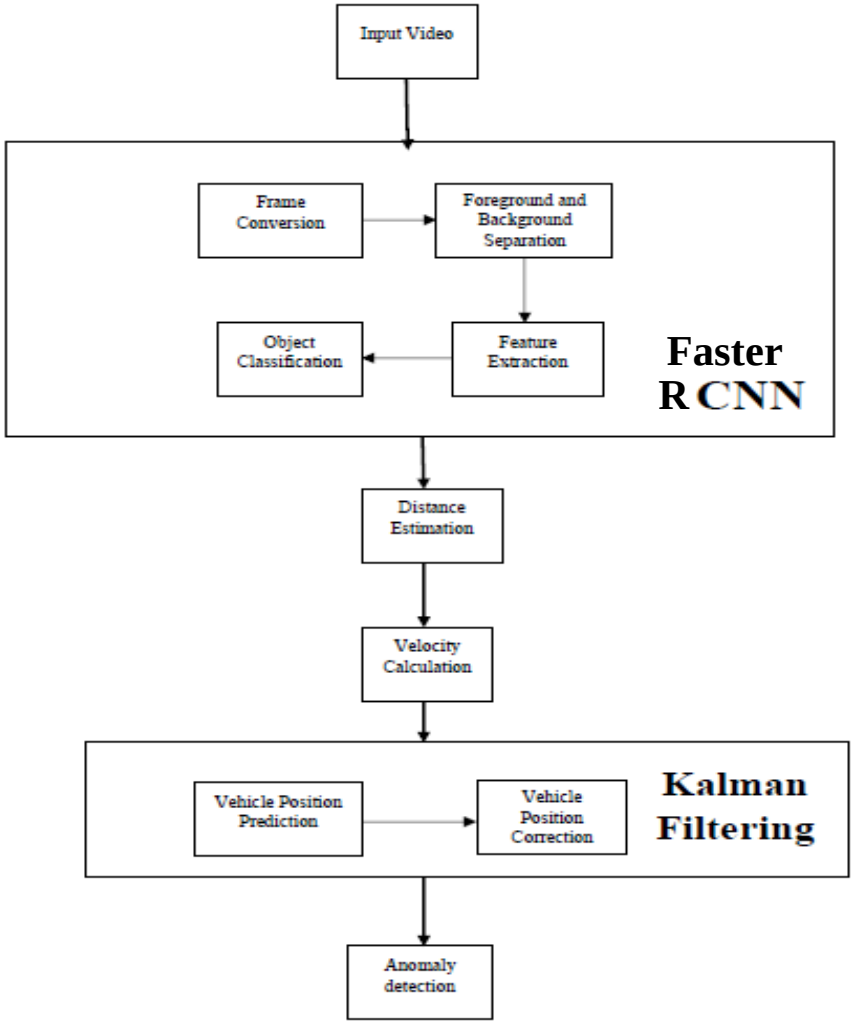

Fig 4.1: System architecture of the project

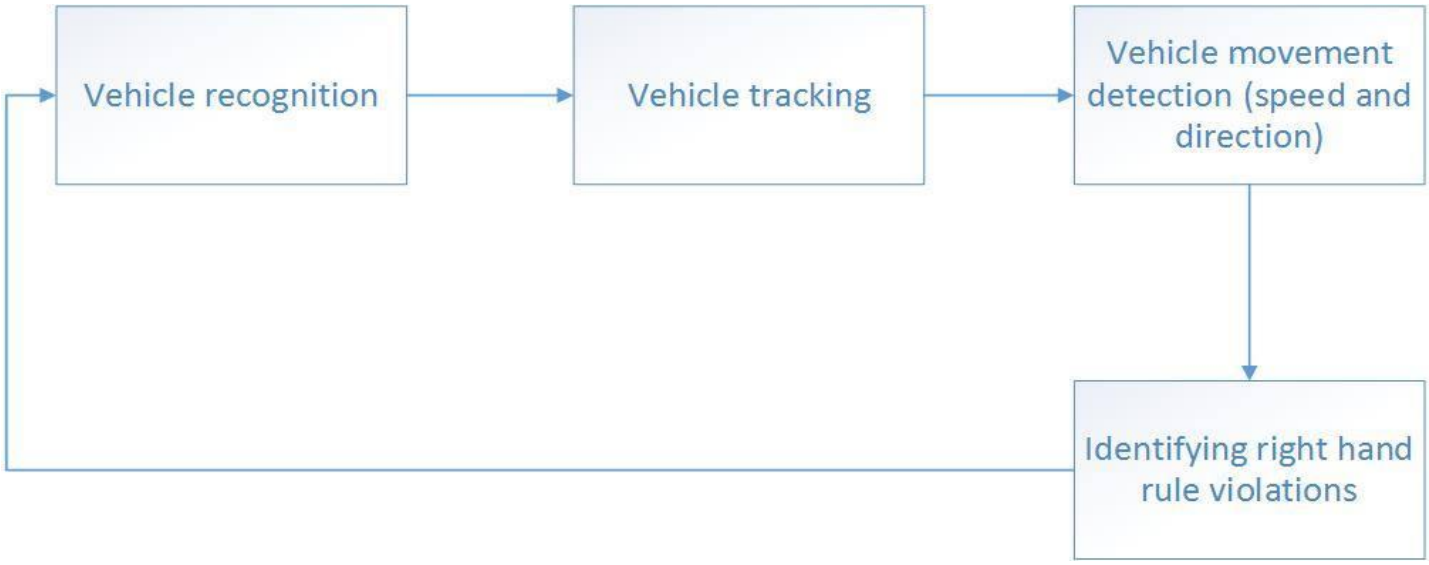

Fig 4.2: Architecture of Intersection Module

Key Components of this Project are:

i Faster RCNN.

ii Kalman Filter.

iii Image Segmentation.

4.4.1: Faster RCNN

- In the below diagram, the input is fed to the network of stacked Conv, Pool and Dense layers.
- The convolutional layer can be thought of as the eyes of the CNN. The neurons in this layer look for specific features. If they find the features they are looking for, they produce a high activation.

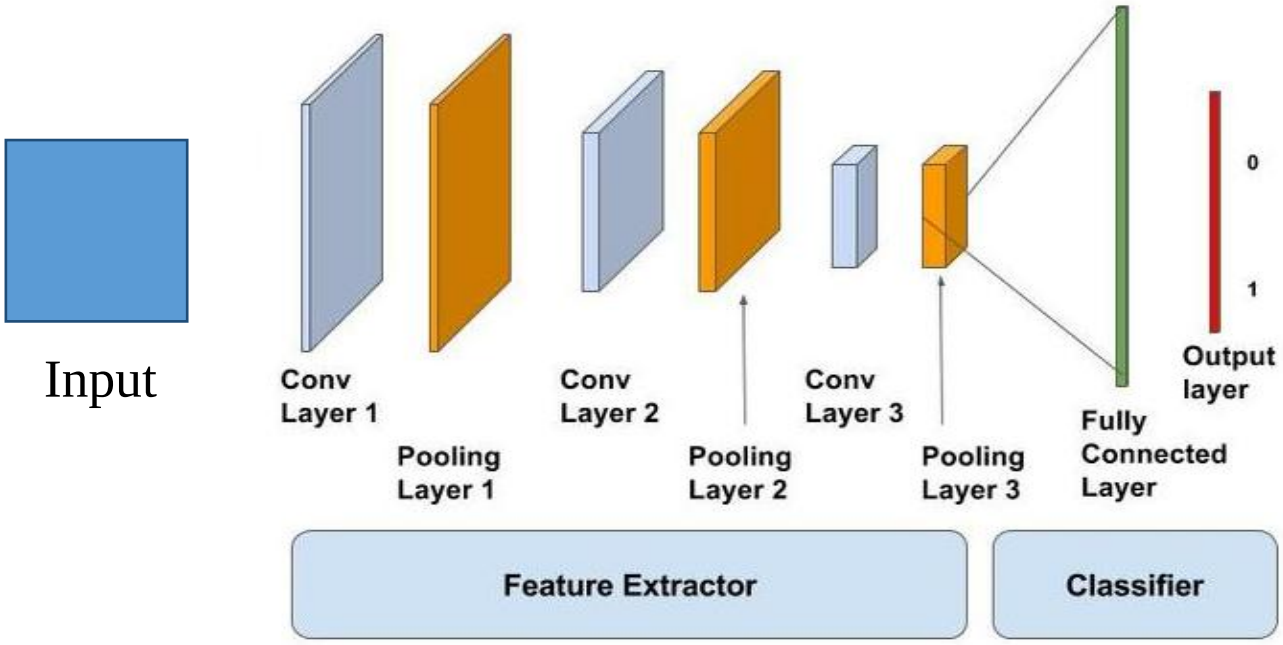

Fig 4.3: Architecture of Faster R-CNN

- In image processing, to calculate convolution at a particular location, we extract x sized chunk from the image centred at location. We then multiply the values in this chunk element-by-element with the convolution filter (also sized x) and then add them all to obtain a single output as shown in the below figure.

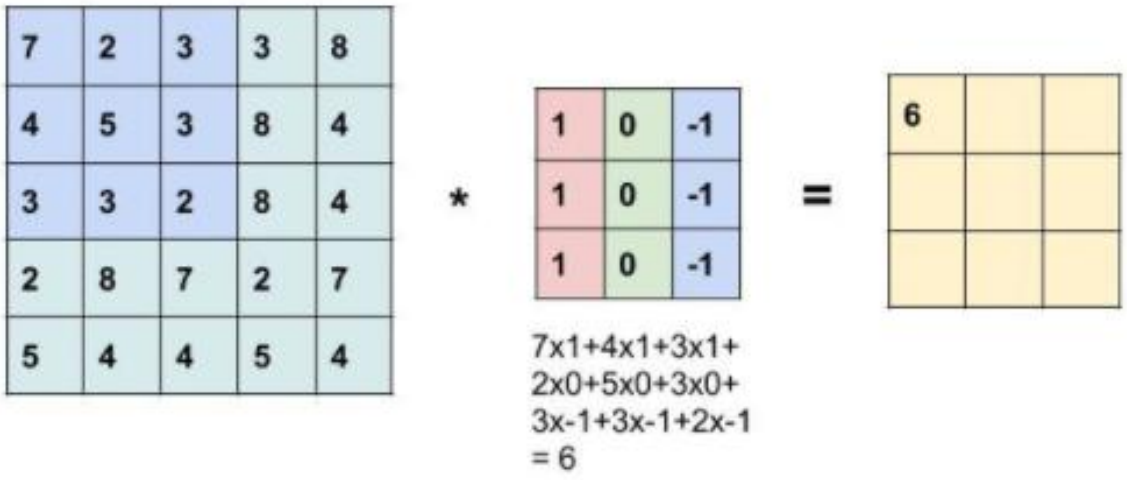

Fig 4.4: Chunk extraction from image

4.4.2 Kalman Filter:

- Kalman filter is used to predict the trajectory of the car.
- Kalman filtering, also known as linear quadratic estimation, is an algorithm that uses a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more accurate than those based on a single measurement alone, by estimating a joint probability distribution over the variables for each timeframe.
- Kalman filter is an optimal recursive data processing algorithm and is a linear minimum variance error estimation algorithm for the state sequence of a dynamic system.
- The algorithm works in a two-step process.
- The system is assumed as a linear and discrete system, and the noise in the system is Gauss white noise.

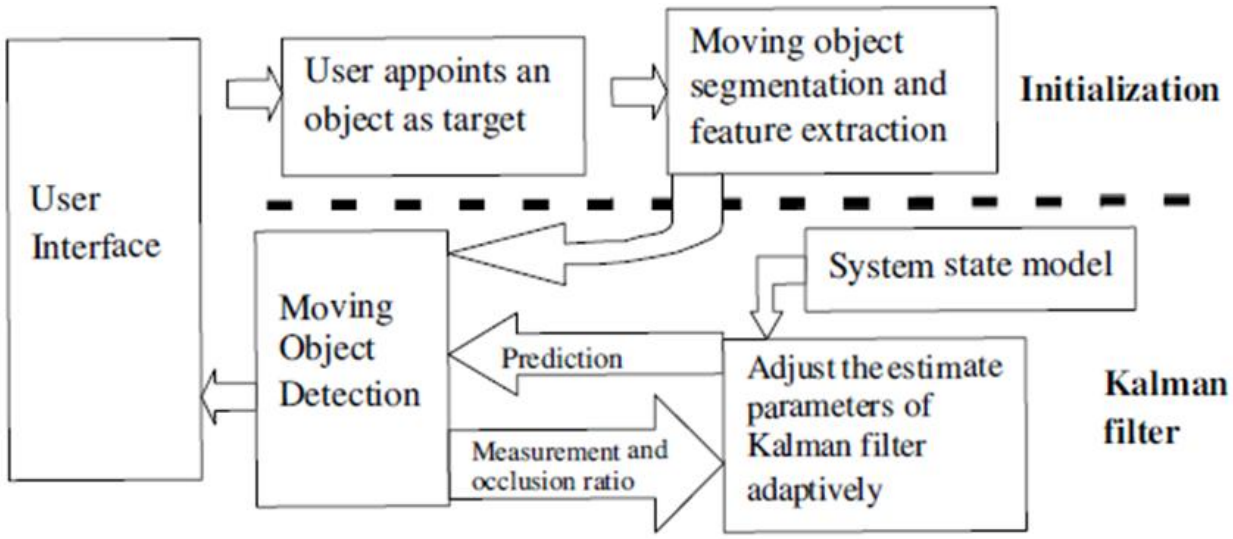

Fig 4.5: Architecture of Kalman Filter

4.4.3 Image Segmentation:

- First, the image is stretched, to reduce autocorrelation. The image is split into R, G, and B channels which are stretched separately, and then the inverse of splitting operation is applied to put the channels back together.
- The resulted image is transformed from RGB into HSV colour space, in order to use the saturation channel.
- This step helps us to separate foreground from background, since in the image the saturation of foreground is higher than the saturation of background.
- Next, an adaptive threshold is applied, to obtain a binary image. The threshold is computed for each image using data from image histogram.
- To eliminate black holes, we apply the following set of morphological operations: dilation, hole filling, erosion, and extraction of central connected component.

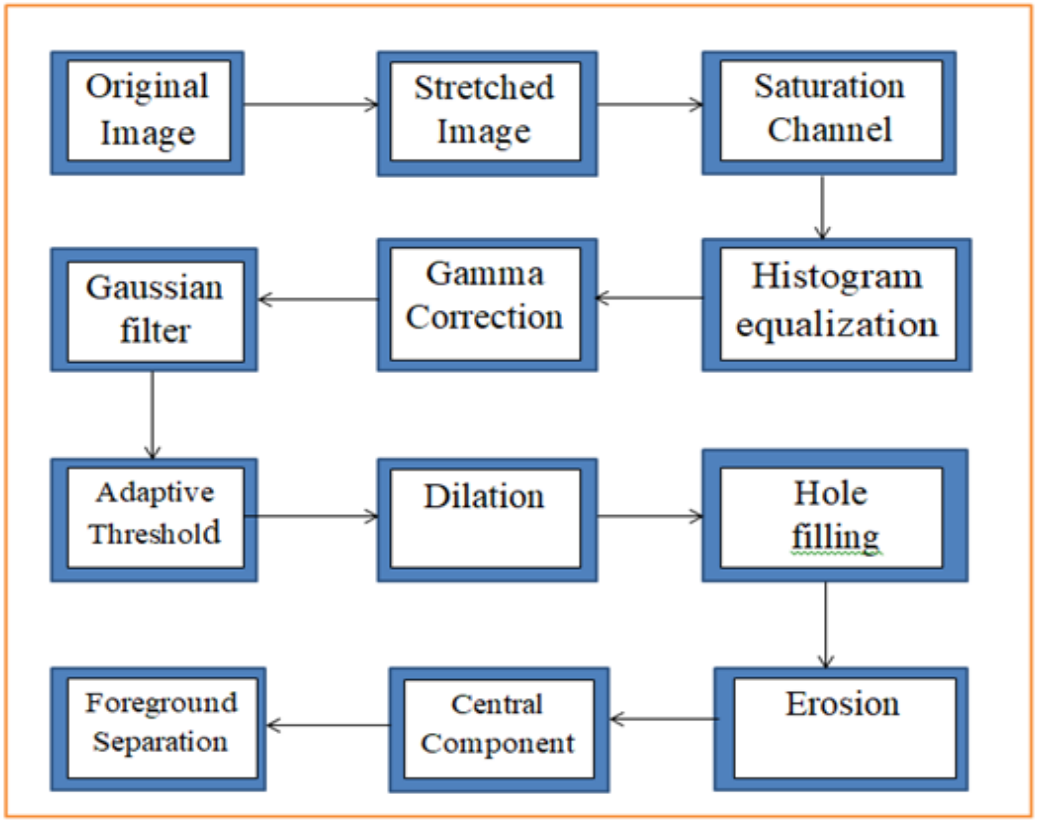

Fig 4.6: Methods of Image Segmentation in R-CNN

4.5 HIGH LEVEL DESIGN

4.5.1 Introduction

In the high level design, the proposed functional and non-functional requirements of the software are depicted. Overall solution to the architecture is developed which can handle those needs. This chapter involves the following consideration.

- Design consideration
- Data flow diagram

4.5.2 Design consideration

There are several design consideration issues that need to be addressed or resolved before getting down designing a complete solution for the system. These design considerations include having different modes for different types of roads. This is so because rules change for different roads, as in, highways have different rules as compared to intersections. Also occlusion must be factored in. Not only this, various other factors play in which can't be controlled. These include natural features like the weather conditions, the angle and the intensity of sunlight, the movement of the camera, and the colour of the cars.

4.5.3 Data Flow Diagram

A data flow diagram is the graphical representation of the flow of data through an information system. DFD is very useful in understanding a system and can be efficiently used during analysis.
A DFD shows the flow of data through a system. It views a system as a function that transforms the inputs into desired outputs. Any complex systems will not perform this transformation in a single step and a data will typically undergo a series of transformations before it becomes the output.
With a data flow diagram, users are able to visualize how the system will operate that the system will accomplish and how the system will be implemented, old system data flow diagrams can be drawn up and compared with a new systems data flow diagram to draw comparisons to implement a more efficient system.

Data flow diagrams can be used to provide the end user with a physical idea of where they input, ultimately as an effect upon the structure of the whole system.

In the perspective of the project, Data Flow Diagram (DFD) is a special chart type which lets graphically illustrate the "flow" of data through various application components. So the Data Flow Diagrams can be successfully used for visualization of data processing or structured design, for creation an overview of the project, in order to exploring the high-level design in terms of data flows and documenting the major data flows.

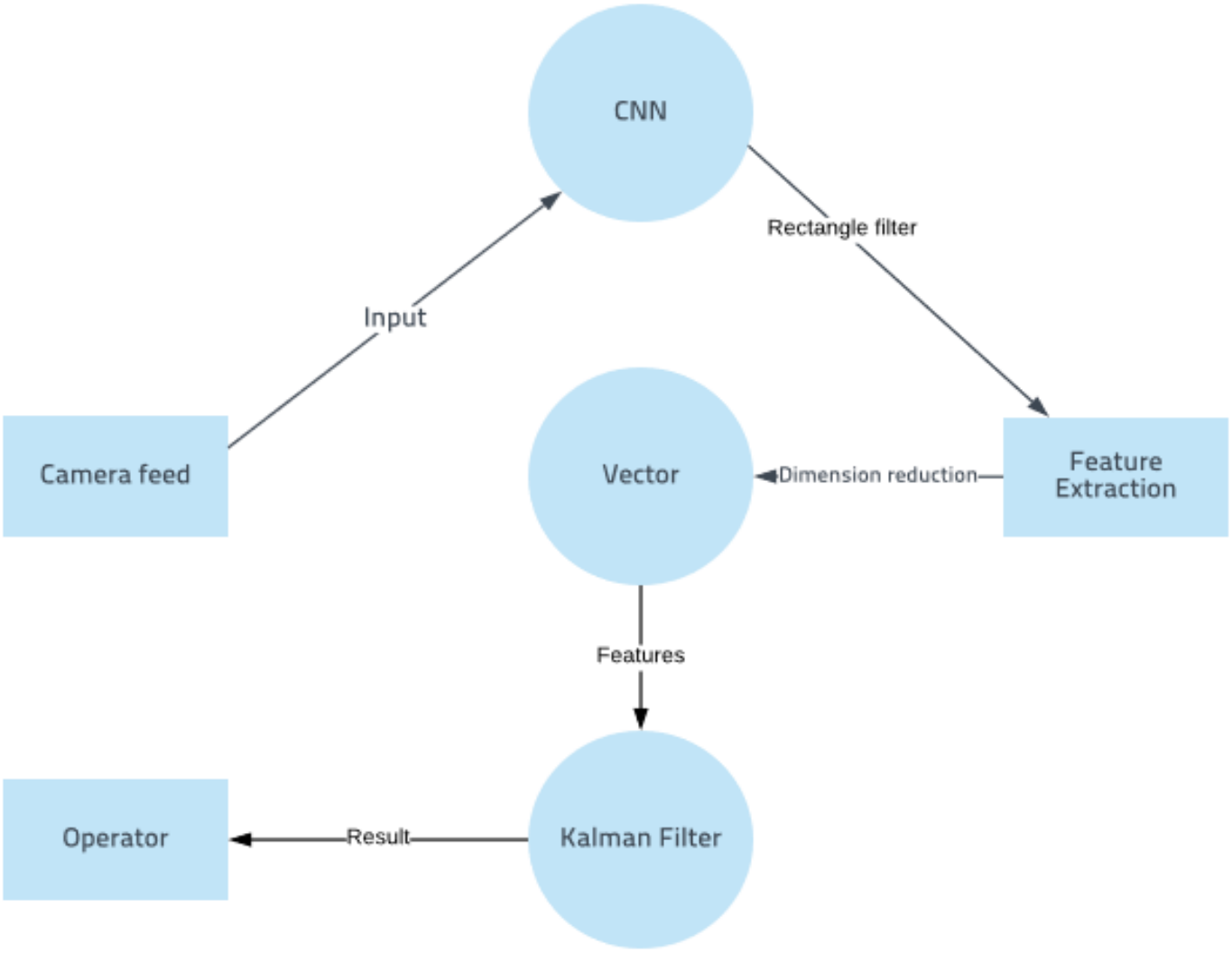

Figure 4.7:

Dataflow diagram of "Smart traffic Surveillance system using RCNN and Kalman Filter

4.5.4 Flowchart

Flowcharts are used in designing and documenting simple processes or programs. Like other types of diagrams, they help visualize what is going on and thereby help understand a process, and perhaps also find flaws, bottlenecks, and other less-obvious features within it. There are many different types of flowcharts, and each type has its own repertoire of boxes and notational conventions. The two most common types of boxes in a flowchart are:

- A processing step, usually called activity, and denoted as a rectangular box.
- A decision usually denoted as a diamond.

A flowchart is described as "cross-functional" when the page is divided into different swim lanes describing the control of different organizational units. A symbol appearing in a particular "lane" is within the control of that organizational unit. This technique allows the author to locate the responsibility for performing an action or making a decision correctly, showing the responsibility of each organizational unit for different parts of a single process.

Below is the flowchart for process of Message Sending Process.

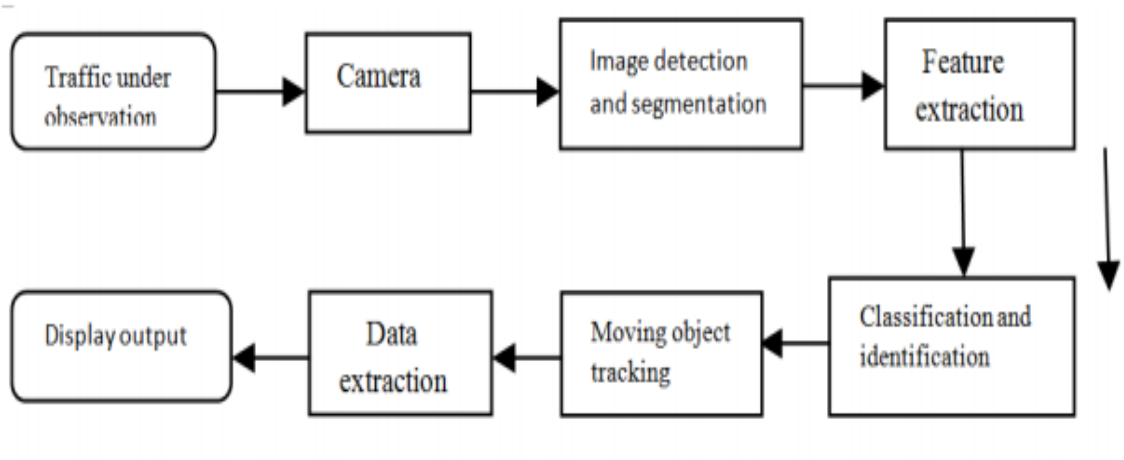

Fig 4.8: Flowchart of the Project

4.6 LOW LEVEL DESIGN

4.6.1 Introduction

During the detailed phase, the view of the application developed during the high level design is broken down into modules and programs. Logic design is done for every program and then documented as program specifications. For every program, a unit test plan is created.

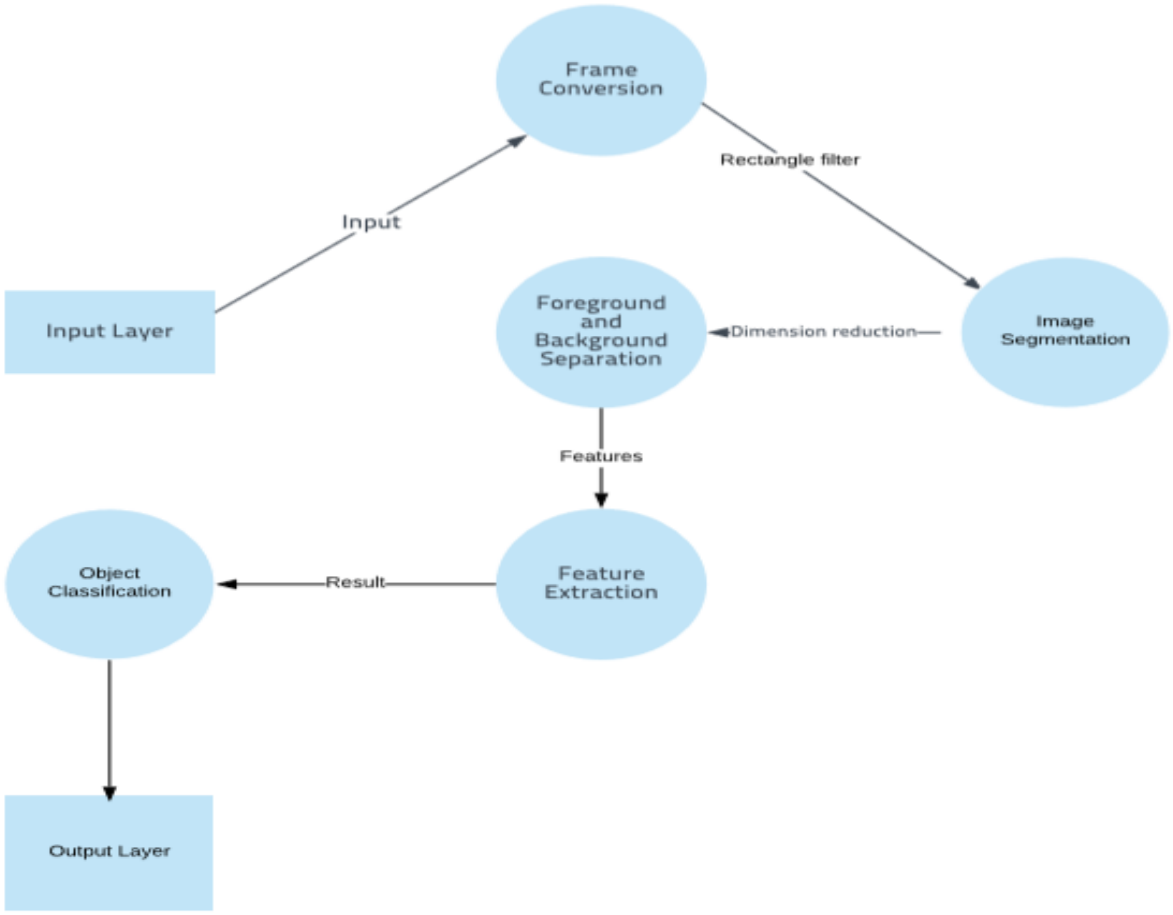

Figure 4.9: Low Level DFD of Faster R-CNN

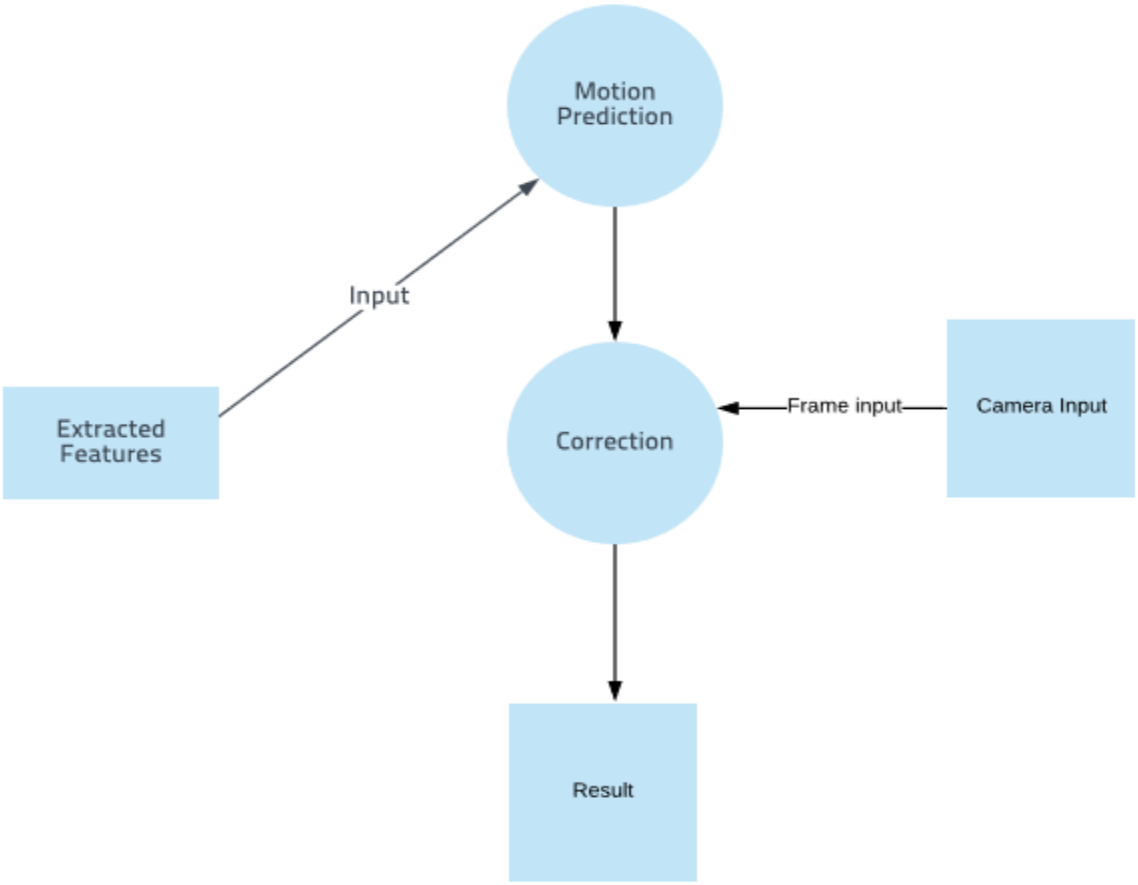

Figure 4.10: Low Level DFD of Kalman Filter.

4.6.2 Use Case Diagram

The external objects that interact directly with the system are called ***actors.*** Actors include humans, external devices and other software systems. The important thing about actors is that they are not under control of the application. In this project, user of the system is the actor. To find use cases, for each actor, list the fundamentally different ways in which the actor uses the system. Each of these ways is a use case.

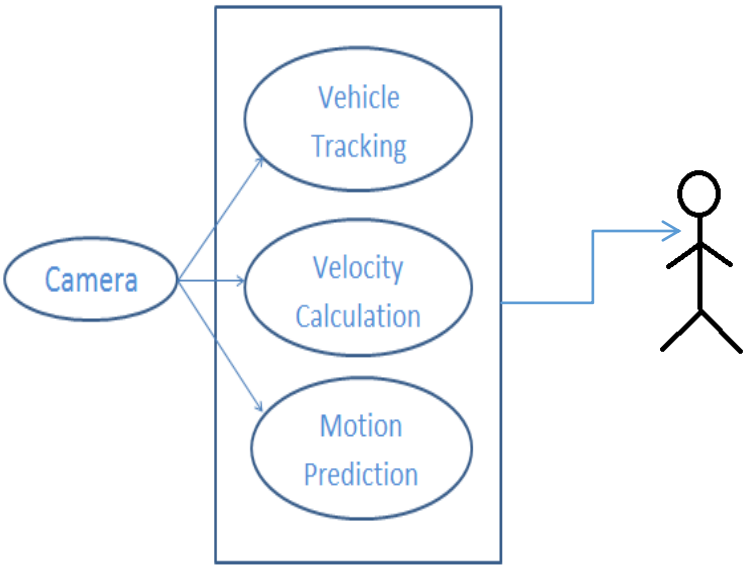

Fig 4.11: Use Case Diagram for Video feed

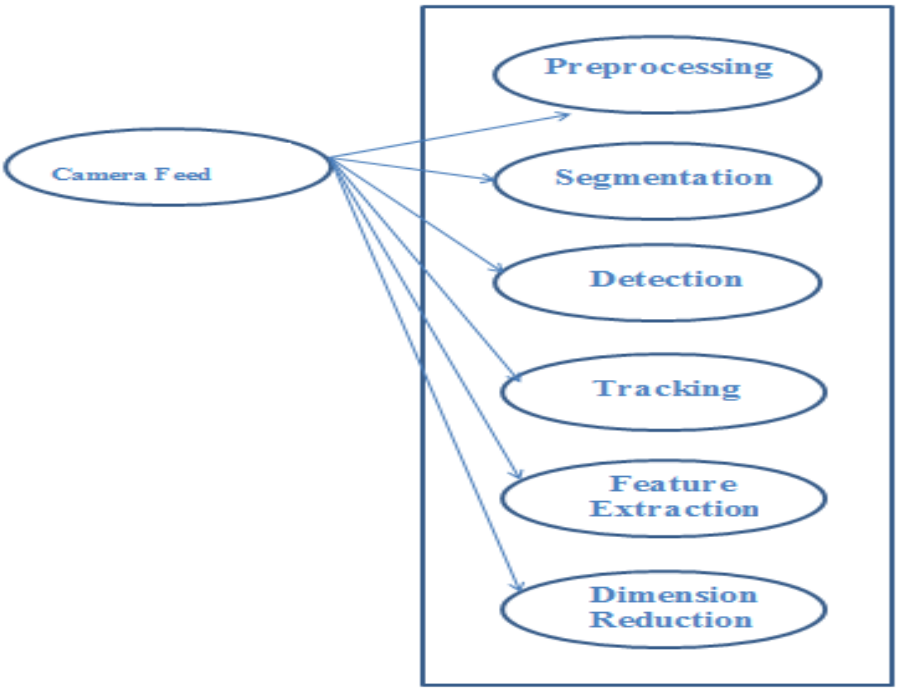

Fig 4.12: Use Case Diagram for Faster R-CNN

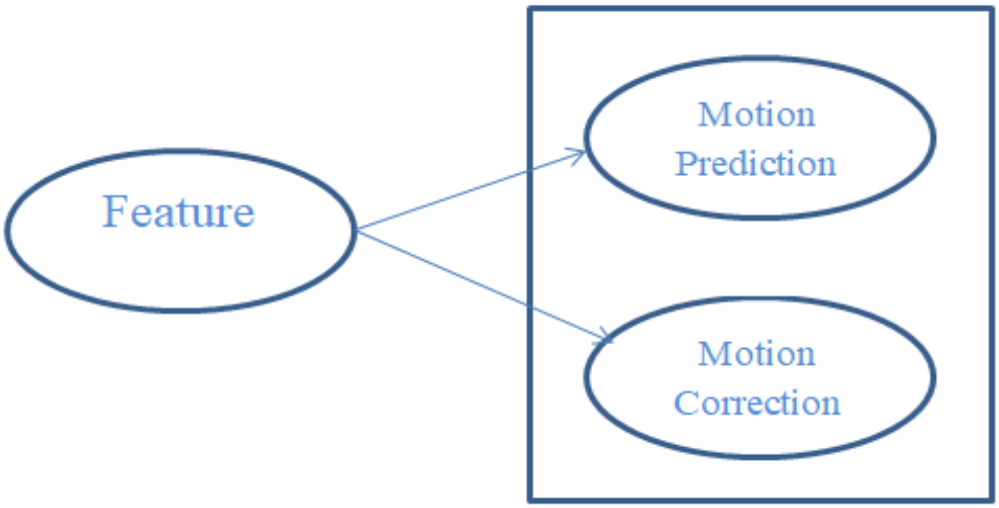

Fig 4.13: Use Case Diagram for Kalman Filter.

4.6.3 Sequence Diagram

A sequence diagram in a Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It shows the participants in an interaction and the sequence of messages among them; each participant is assigned a column in a table.

Below section shows the sequence diagrams in this application

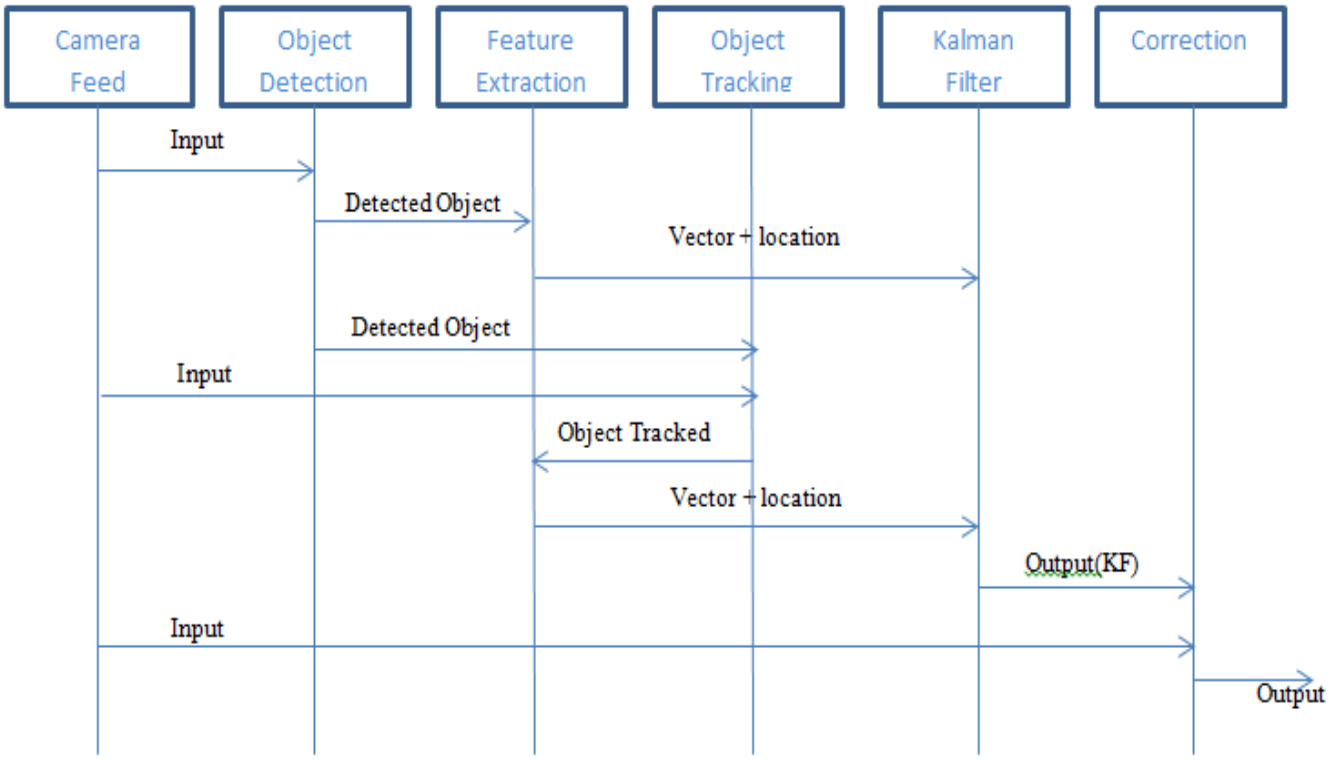

Figure 4.14: Sequence Diagram for Smart Traffic Surveillance system using Faster RCNN and Kalman Filter

CHAPTER 5

IMPLEMENTATION

5.1 Introduction

Implementation is the realization of an application, or execution of a plan, idea, model, design, specification, standard, algorithm, or policy. In other words, an implementation is a realization of a technical specification or algorithm as a program, software component, or other computer system through programming and deployment. Many implementations may exist for a given specification or standard.

Implementation is one of the most important phases of the Software Development Life Cycle (SDLC). It encompasses all the processes involved in getting new software or hardware operating properly in its environment, including installation, configuration, running, testing, and making necessary changes. Specifically, it involves coding the system using a particular programming language and transferring the design into an actual working system.

5.2 Overview of System Implementation

This project is implemented considering the following aspects:

1. Usability Aspect.
2. Technical Aspect.

5.3 Usability Aspect

The usability aspect of implementation of the project is realized using two principles:

5.3.1 The project is implemented using MATLAB

The environment used for this project is MATLAB and the programming language used is Python.

MATLAB (matrix laboratory) is a multi-paradigm numerical computing environment. A proprietary programming language developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, C#, Java, Fortran and Python.

MATLAB is not only a programming language, but a programming environment as well. You can perform operations from the command line, as a sophisticated calculator. Or you can create programs and functions that perform repetitive tasks, just as any other computer language. One of the most important features of the MATLAB interface is the help. It is very thorough and you can learn almost anything you need from it. Also MATLAB, being vectorised, makes working with matrices quite easy.

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Its readability and huge library makes it one of the most preferred programming languages.

5.4 Technical Aspect

The technical aspect of implementation of the project is realized using following principle:

5.4.1 The project is implemented by using MATLAB

Image Tracking can be done efficiently in several methods/platforms. Of these methods/platforms, 2 are most widely used. This is so because best results are obtained through these platforms. They are:

a) Tensor Flow and OpenCV in Microsoft Visual Studio
b) MATLAB

We considered MATLAB for development of this project because of several reason as follows:

- **Matrix Element**

  Its basic data element is the matrix. A simple integer is considered an matrix of one row and one column. Several mathematical operations that work on arrays or matrices are built-in to the Matlab environment. For example, cross-products, dot-products, determinants, inverse matrices.

- **Recording of the processing used**

  MATLAB is a general purpose programming language. When it is used to process images one generally writes function files, or script files to perform the operations. These files form a formal record of the processing used and ensures that the final results can be tested and replicated by others should the need arise.

- **Access to implementation details**

  MATLAB provides many functions for image processing and other tasks. Most of these functions are written in the MATLAB language and are publicly readable as plain text files. Thus the implementation details of these functions are accessible and open to scrutiny. The defense can examine the processing used in complete detail, and any challenges raised can be responded to in an informed way by the prosecution.

- **Numerical accuracy**

  Another advantage of MATLAB is that it allows one to ensure maximal numerical precision in the final result.

  In general, image files store data to 8 bit precision. This corresponds to a range of integer values from 0-255. A pixel in a colour image may be represented by three 8 bit numbers, each representing the red, green and blue components as an integer value between 0 and 255. However as soon as one reads this image data into memory and starts to process it it is very easy to generate values that lie outside the range 0-255.

  Being a general programming language it is possible to have complete control of the precision with which one represents data in MATLAB. An image can be read into memory and the data cast into double precision floating point values. All image processing steps can then be performed in double precision floating point arithmetic, and at no intermediate stage does one need to rescale the results to integers in the range 0-255. Only at the final point when the image is to be displayed and/or written to file does it need to be rescaled. Here one can use histogram truncation to eliminate extreme pixel values so that the bulk of the image data is properly represented.

- **Advanced algorithms**

  MATLAB is a scientific programming language and provides strong mathematical and numerical support for the implementation of advanced algorithms. It is for this reason that MATLAB is widely used by the image processing and computer vision community. New algorithms are very likely to be implemented first in MATLAB; indeed they may only be available in MATLAB.

- **Vector operations and Graphical output**

  Vectorised operations have several advantages like adding two arrays together needs only one command, instead of a for or while loop. The graphical output is optimized for interaction. You can plot your data very easily, and then change colors, sizes, scales, etc, by using the graphical interactive tools.

5.5 Implementation Support

5.5.1 Installation of MATLAB

Following are the requirements for installation of Android Studio on Windows Operating System:

- Microsoft Windows 7/8/10 (64-bit)
- GB RAM minimum, 8 GB RAM recommended
- GB of available disk space minimum, 4 GB Recommended
- 1280x800 minimum screen resolution
- PYTHON 2.7
- GPU of compute power 3.0 or above

We should have already downloaded MATLAB.

To install MATLAB on Windows, we should proceed as follows:

(i) Launch the .exe file you downloaded.

(ii) Follow the setup wizard to install MATLAB.

CHAPTER 6

PSEUDO CODE

Pseudo Code uses the structural conventions of a normal programming language, but is intended for human reading rather than machine reading. Pseudo code typically omits details that are essential for machine understanding of the algorithm, such as variable declarations, system-specific code and some subroutines. The programming language is augmented with natural language description details, where convenient, or with compact mathematical notation. The purpose of using pseudocode is that it is easier for people to understand than conventional programming language code, and that it is an efficient and environment-independent description of the key principles of an algorithm. It is commonly used in textbooks and scientific publications that are documenting various algorithms, and also in planning of computer program development, for sketching out the structure of the program before the actual coding takes place.

This project basically consists of 2 parts. These 2 parts are distinctive as one looks for violations in highways and the other in intersections. As such the logic is different and hence the project is divided into 2 modules.

The project consists of following 2 python Modules:

i) HighwayMain.m
ii) IntersectionMain.m

The pseudo code is as follows:

HIGHWAYMAIN.m

clc;

clear all;

close all;

```
inputvideo=vision.VideoFileReader('C:\Users\Onam\Desktop\final project\dataset.mp4');
inputvideo1=VideoReader("C:\Users\Onam\Desktop\final project\dataset.mp4");
nframes = inputvideo1.NumberOfFrames;
vid1=vision.VideoPlayer;
while~isDone(inputvideo)
 frame1=step(inputvideo);
 step(vid1,frame1);
 pause(0.005);
end
position =  [1 50];
ii = 1;
imwrite(frame1,'C:\Users\Onam\Desktop\final project\referenceimage.jpg','jpg');
release(inputvideo);
release(vid1);
referenceimage=imread('C:\Users\Onam\Desktop\final project\referenceimage.jpg');
vid2=vision.VideoFileReader('C:\Users\Onam\Desktop\final project\dataset.mp4');
for i=2:nframes
clc
 frame=step(vid2);
 filename = [sprintf('%03d',ii) '.jpg'];
 fullname = fullfile('C:\Users\Onam\Desktop\final project\','imgs',filename);
```

```
 frame2=((im2double(frame))-(im2double(referenceimage)));

  imwrite(frame,fullname);

 frame1=im2bw(frame2,0.2);

 [labelimage]=bwlabel(frame1);

 stats=regionprops(labelimage,'basic');

 BB=stats.BoundingBox;

 X(i)=BB(1);

 Y(i)=BB(2);

 Dist=((X(i)-X(i-1))^2+(Y(i)-Y(i-1))^2)^(1/2)

 Z(i)=Dist;

  M=median(Z);

Speed=(Dist)*(120/18);

 if(Dist>10&&Dist<20)

 RGB=insertText(frame,position,'Medium
Speed','AnchorPoint','LeftTop','BoxOpacity',0.0,'TextColor','yellow');

 fullname1 = fullfile('C:\Users\Onam\Desktop\final project\','spd',filename);

imwrite(RGB,fullname1);

 display('MEDIUM SPEED');

  elseif(Dist<10)

 RGB=insertText(frame,position,'Slow
Speed','AnchorPoint','LeftTop','BoxOpacity',0.0,'TextColor','green');

 fullname1 = fullfile('C:\Users\Onam\Desktop\final project\','spd',filename);
```

```
imwrite(RGB,fullname1);

 display('SLOW SPEED');

  else

 RGB = insertText(frame,position,'Fast
Speed','AnchorPoint','LeftTop','BoxOpacity',0.0,'TextColor','red');

 fullname1 = fullfile('C:\Users\Onam\Desktop\final project\','spd',filename);

 fullname2 = fullfile('C:\Users\Onam\Desktop\final project\','ovrspd',filename);

imwrite(RGB,fullname1);

imwrite(RGB,fullname2);

 display('FAST SPEED');

 end

 S=strel('disk',6);

 frame3=imclose(frame1,S);

 step(vid1,frame1);

 pause(0.005);

  ii = ii+1;

end

 M=median(Z);

Speed=(M)*(120/8);

 release(vid1)
```

```
imageNames = dir(fullfile('C:\Users\Onam\Desktop\final project\','spd','*.jpg'));
imageNames = {imageNames.name}';
outputVideo = VideoWriter(fullfile('C:\Users\Onam\Desktop\final project\','output.avi'));
outputVideo.FrameRate = inputvideo1.FrameRate;
open(outputVideo)
for ii = 1:length(imageNames)
   img = imread(fullfile('C:\Users\Onam\Desktop\final project\','spd',imageNames{ii}));
   writeVideo(outputVideo,img)
end
close(outputVideo)
shuttleAvi = VideoReader(fullfile('C:\Users\Onam\Desktop\final project\','output.avi'));
ii = 1;
while hasFrame(shuttleAvi)
   mov(ii) = im2frame(readFrame(shuttleAvi));
   ii = ii+1;
end
videoFReader = vision.VideoFileReader('C:\Users\Onam\Desktop\final project\output.avi');
videoPlayer = vision.VideoPlayer;
while ~isDone(videoFReader)
  videoFrame = videoFReader();
  videoPlayer(videoFrame);
```

```
  pause(0.1)
end
release(videoPlayer);
release(videoFReader);
trafficVid = VideoReader('C:\Users\Onam\Desktop\final project\output.avi')
get(trafficVid)
%implay('traffic.mj2');
darkCarValue = 75;
darkCar = rgb2gray(read(trafficVid,71));
noDarkCar = imextendedmax(darkCar, darkCarValue);
imshow(darkCar)
figure, imshow(noDarkCar)
sedisk = strel('disk',2);
noSmallStructures = imopen(noDarkCar, sedisk);
imshow(noSmallStructures)
nframes = trafficVid.NumberOfFrames;
I = read(trafficVid, 1);
taggedCars = zeros([size(I,1) size(I,2) 3 nframes], class(I));
for k = 1 : nframes
    singleFrame = read(trafficVid, k);
     % Convert to grayscale to do morphological processing.
    I = rgb2gray(singleFrame);
```

```
 noDarkCars = imextendedmax(I, darkCarValue);
    % Remove lane markings and other non-disk shaped structures.
noSmallStructures = imopen(noDarkCars, sedisk);
% Remove small structures.
noSmallStructures = bwareaopen(noSmallStructures, 150);
% Get the area and centroid of each remaining object in the frame. The
% object with the largest area is the light-colored car.  Create a copy
% of the original frame and tag the car by changing the centroid pixel
% value to red.
taggedCars(:,:,:,k) = singleFrame;
   stats = regionprops(noSmallStructures, {'Centroid','Area'});
if ~isempty([stats.Area])
    areaArray = [stats.Area];
    [junk,idx] = max(areaArray);
    c = stats(idx).Centroid;
    c = floor(fliplr(c));
    width = 2;
    row = c(1)-width:c(1)+width;
    col = c(2)-width:c(2)+width;
    taggedCars(row,col,1,k) = 255;
    taggedCars(row,col,2,k) = 0;
    taggedCars(row,col,3,k) = 0;
```

```
    end
end
frameRate = trafficVid.FrameRate;
implay(taggedCars,frameRate);
extract2('Demo_d.jpg',4);
extract('Demo_d.jpg');
businessCard   = imread('C:\Users\Onam\Desktop\final project\lpimg.jpg');
     ocrResults     = ocr(businessCard)
     recognizedText = ocrResults.Text;
     figure;
     imshow(businessCard);
     imwrite(businessCard,'C:\Users\Onam\Desktop\final project\lpimg1.jpg','jpg')
     text(600, 150, recognizedText, 'BackgroundColor', [1 1 1]);
     fid = fopen('C:\Users\Onam\Desktop\final project\noPlate.txt', 'wt'); % This portion of code writes
the number plate
    fprintf(fid,'%s\n',recognizedText);      % to the text file, if executed a notepad file with the
    fclose(fid);                    % name noPlate.txt will be open with the number plate written.
    winopen('C:\Users\Onam\Desktop\final project\noPlate.txt')
```

FASTER R-CNN:

```
data = load('fasterRCNNVehicleTrainingData.mat');

vehicleDataset = data.vehicleTrainingData;

data = load('fasterRCNNVehicleTrainingData.mat');

vehicleDataset = data.vehicleTrainingData;data = load('fasterRCNNVehicleTrainingData.mat');

vehicleDataset = data.vehicleTrainingData;

dataDir = fullfile(toolboxdir('vision'),'visiondata');

vehicleDataset.imageFilename = fullfile(dataDir, vehicleDataset.imageFilename);

I = imread(vehicleDataset.imageFilename{10});

I = insertShape(I, 'Rectangle', vehicleDataset.vehicle{10});

I = imresize(I,3);

figure

imshow(I)

idx = floor(0.6 * height(vehicleDataset));

trainingData = vehicleDataset(1:idx,:);

testData = vehicleDataset(idx:end,:);

inputLayer = imageInputLayer([32 32 3]);

filterSize = [3 3];

numFilters = 32;

middleLayers = [

. convolution2dLayer(filterSize, numFilters, 'Padding', 1)

  reluLayer()
```

```
    convolution2dLayer(filterSize, numFilters, 'Padding', 1)
    reluLayer()
    maxPooling2dLayer(3, 'Stride',2)
    ];
finalLayers = [
    fullyConnectedLayer(64)
    reluLayer()
    fullyConnectedLayer(width(vehicleDataset))
    softmaxLayer()
    classificationLayer()
];
layers = [
    inputLayer
    middleLayers
    finalLayers
    ]
optionsStage1 = trainingOptions('sgdm', ...
    'MaxEpochs', 10, ...
    'MiniBatchSize', 256, ...
    'InitialLearnRate', 1e-3, ...
    'CheckpointPath', tempdir);
optionsStage2 = trainingOptions('sgdm', ...
```

```
    'MaxEpochs', 10, ...
    'MiniBatchSize', 128, ...
    'InitialLearnRate', 1e-3, ...
    'CheckpointPath', tempdir);
optionsStage3 = trainingOptions('sgdm', ...
    'MaxEpochs', 10, ...
    'MiniBatchSize', 256, ...
    'InitialLearnRate', 1e-3, ...
    'CheckpointPath', tempdir);
optionsStage4 = trainingOptions('sgdm', ...
    'MaxEpochs', 10, ...
    'MiniBatchSize', 128, ...
    'InitialLearnRate', 1e-3, ...
    'CheckpointPath', tempdir);
options = [
    optionsStage1
    optionsStage2
    optionsStage3
    optionsStage4
    ];
doTrainingAndEval = false;
if doTrainingAndEval
```

```
    rng(0);
    detector = trainFasterRCNNObjectDetector(trainingData, layers, options, ...
        'NegativeOverlapRange', [0 0.3], ...
        'PositiveOverlapRange', [0.6 1], ...
        'BoxPyramidScale', 1.2);
else
        detector = data.detector;
end
I = imread(testData.imageFilename{1});
 [bboxes,scores] = detect(detector,I);
I = insertObjectAnnotation(I,'rectangle',bboxes,scores);
figure
imshow(I)
if doTrainingAndEval
    resultsStruct = struct([]);
    for i = 1:height(testData)
        I = imread(testData.imageFilename{i});
        [bboxes, scores, labels] = detect(detector, I);
        resultsStruct(i).Boxes = bboxes;
        resultsStruct(i).Scores = scores;
        resultsStruct(i).Labels = labels;
    end
```

```
    results = struct2table(resultsStruct);
else
    results = data.results;
end
expectedResults = testData(:, 2:end);
 [ap, recall, precision] = evaluateDetectionPrecision(results, expectedResults);
figure
plot(recall,precision)
xlabel('Recall')
ylabel('Precision')
grid on
title(sprintf('Average Precision = %.2f', ap))
```

ACF DETECTOR CLASS:

```
classdef AcfDetector
    %ACFDETECTOR Detect Cars using acf detector
    %   Aggregated Channel Features
    properties
        detector
    end
    methods
        function obj = AcfDetector(modelAddress)
            model = load(modelAddress);
            obj.detector = model.detector;
        end
            function bboxes = detect(obj, image)
            bboxesWithScores = acfDetect(image, obj.detector);
            bboxes = bboxesWithScores(:,1:4);
        end
    end
end
```

BGS DETECTOR CLASS:

```
classdef BgsDetector
    %BGSDETECTOR A object detector based on background subtraction
    %   This class detect add moving object in a movie
    properties
        bgs
        blobAnalyser
    end
    methods
        % Constructor
        function obj = BgsDetector()
            obj.bgs = vision.ForegroundDetector(...
                'NumTrainingFrames', 100, ... % 5 because of short video
                'InitialVariance', 30*30, ...
                'LearningRate', .005); % initial standard deviation of 30
            obj.blobAnalyser = vision.BlobAnalysis(...
                'CentroidOutputPort', false, 'AreaOutputPort', false, ...
                'BoundingBoxOutputPort', true, ...
                'MinimumBlobAreaSource', 'Property', 'MinimumBlobArea', 250);
        end
        function bboxes = detect(obj, image)
            foreground = step(obj.bgs, image );
```

```
            bboxes   = step(obj.blobAnalyser, foreground);
            bboxes = double(bboxes);
        end
    end
end
```

KALMAN FILTER CLASS:

```
classdef MultiobjectKalmanTracker < handle
    properties
        tracks
        nextId
    end
    methods
        function obj = MultiobjectKalmanTracker()
            obj.nextId = 0;
            obj.initializeTracks();
        end
        function tracks = track(obj, detections)
            obj.predictNewLocationsOfTracks();
            [assignments, unassignedTracks, unassignedDetections] = ...
                obj.detectionToTrackAssignment(detections);
            obj.updateAssignedTracks(detections, assignments);
```

```
            obj.updateUnassignedTracks(unassignedTracks);
            obj.deleteLostTracks();
            obj.createNewTracks(detections(unassignedDetections,:));
            obj.deleteInvalidTracks();
            tracks = obj.tracks;
        end
    end
    methods (Access = private)
        function initializeTracks(obj)
            % create an empty array of tracks
            obj.tracks = struct(...
                'id', {}, ...
                'bbox', {}, ...
                'kalmanFilter', {}, ...
                'age', {}, ...
                'totalVisibleCount', {}, ...
                'consecutiveInvisibleCount', {});
        end
        function tracks = predictNewLocationsOfTracks(obj)
            for i = 1:length(obj.tracks)
                %bbox = tracks(i).bbox;
                % Predict the current location of the track.
```

```
        predictedBbbox = predict(obj.tracks(i).kalmanFilter);
        % Shift the bounding box so that its center is at
        % the predicted location.
        %predictedCentroid = predictedCentroid - bbox(3:4) / 2;
        obj.tracks(i).bbox = predictedBbbox;
    end
    tracks = obj.tracks;
end
function [assignments, unassignedTracks, unassignedDetections] = ...
        detectionToTrackAssignment(obj, bboxes)
    nTracks = length(obj.tracks);
    nDetections = size(bboxes, 1);
    % Compute the cost of assigning each detection to each track.
    cost = zeros(nTracks, nDetections);
    for i = 1:nTracks
        for j=1:nDetections
            cost(i,j)=-rectint(obj.tracks(i).bbox,bboxes(j,:))/ ...
            sqrt(obj.tracks(i).bbox(3)*obj.tracks(i).bbox(4)* ...
            bboxes(j,4)*bboxes(j,3));
        end
        %cost(i, :) = distance(tracks(i).kalmanFilter, centroids);
    end
```

```
    % Solve the assignment problem.
    costOfNonAssignment = -0.01;
    [assignments, unassignedTracks, unassignedDetections] = ...
        assignDetectionsToTracks(cost, costOfNonAssignment);
end
%% Update Assigned Tracks
% The |updateAssignedTracks| function updates each assigned track with the
% corresponding detection. It calls the |correct| method of
% |vision.KalmanFilter| to correct the location estimate. Next, it stores
% the new bounding box, and increases the age of the track and the total
% visible count by 1. Finally, the function sets the invisible count to 0.
function updateAssignedTracks(obj, bboxes, assignments)
    numAssignedTracks = size(assignments, 1);
    for i = 1:numAssignedTracks
        trackIdx = assignments(i, 1);
        detectionIdx = assignments(i, 2);
        %       centroid = centroids(detectionIdx, :);
        bbox = bboxes(detectionIdx, :);
        % Correct the estimate of the object's location
        % using the new detection.
        correct(obj.tracks(trackIdx).kalmanFilter, bbox);
        % Replace predicted bounding box with detected
```

```
        % bounding box.
        %tracks(trackIdx).bbox = bbox;
        % Update track's age.
        obj.tracks(trackIdx).age = obj.tracks(trackIdx).age + 1;
        % Update visibility.
        obj.tracks(trackIdx).totalVisibleCount = ...
            obj.tracks(trackIdx).totalVisibleCount + 1;
        obj.tracks(trackIdx).consecutiveInvisibleCount = 0;
    end
end
%% Update Unassigned Tracks
% Mark each unassigned track as invisible, and increase its age by 1.
 function updateUnassignedTracks(obj,unassignedTracks)
    for i = 1:length(unassignedTracks)
        ind = unassignedTracks(i);
        obj.tracks(ind).age = obj.tracks(ind).age + 1;
        obj.tracks(ind).consecutiveInvisibleCount = ...
            obj.tracks(ind).consecutiveInvisibleCount + 1;
    end
end
%% Delete Lost Tracks
% The |deleteLostTracks| function deletes tracks that have been invisible
```

```
% for too many consecutive frames. It also deletes recently created tracks
% that have been invisible for too many frames overall.
function deleteLostTracks(obj)
    if isempty(obj.tracks)
        return;
    end
    invisibleForTooLong = 50;
    ageThreshold = 8;
    % Compute the fraction of the track's age for which it was visible.
    ages = [obj.tracks(:).age];
    totalVisibleCounts = [obj.tracks(:).totalVisibleCount];
    visibility = totalVisibleCounts ./ ages;
    % Find the indices of 'lost' tracks.
    lostInds = (ages < ageThreshold & visibility < 0.6) | ...
        [obj.tracks(:).consecutiveInvisibleCount] >= invisibleForTooLong;
    % Delete lost tracks.
    obj.tracks = obj.tracks(~lostInds);
end
function deleteInvalidTracks(obj)
    if isempty(obj.tracks)
        return;
    end
```

```
    bboxes = reshape([obj.tracks(:).bbox]',4,length(obj.tracks))';
    heights = bboxes(:,4);
    widths = bboxes(:,3);
    validTracks = heights>0 & widths>0;
    obj.tracks = obj.tracks(validTracks);
end
%% Create New Tracks
% Create new tracks from unassigned detections. Assume that any unassigned
% detection is a start of a new track. In practice, you can use other cues
% to eliminate noisy detections, such as size, location, or appearance.
        function createNewTracks(obj, bboxes)
    %    centroids = centroids(unassignedDetections, :);
    %bboxes = bboxes(unassignedDetections, :);
 for i = 1:size(bboxes, 1)
        %centroid = centroids(i,:);
        bbox = bboxes(i, :);
        % Create a Kalman filter object.
        %kalmanFilter = configureKalmanFilter('ConstantVelocity', ...
        %    bbox, [100, 100], [100 , 10], 1e5);
        StateTransitionModel=[1 0 0 0 1 0 0 0; ...
            0 1 0 0 0 1 0 0; ...
            0 0 1 0 0 0 1 0; ...
```

```
    0 0 0 1 0 0 0 1; ...
    0 0 0 0 1 0 0 0; ...
    0 0 0 0 0 1 0 0; ...
    0 0 0 0 0 0 1 0; ...
    0 0 0 0 0 0 0 1];
MeasurementModel= [1 0 0 0 0 0 0 0; ...
    0 1 0 0 0 0 0 0; ...
    0 0 1 0 0 0 0 0; ...
    0 0 0 1 0 0 0 0];
kalmanFilter=vision.KalmanFilter(StateTransitionModel,MeasurementModel);
kalmanFilter.MeasurementNoise = [1e3 0 0 0; ...
    0 1e3 0 0; ...
    0 0 1e4 0; ...
    0 0 0 1e4];
kalmanFilter.State = [bbox 0 0 0 0];
%  kalmanFilter = configureKalmanFilter('ConstantVelocity', ...
%     centroid, [5, 5], [5, 5], 100);
% Create a new track.
newTrack = struct(...
    'id', obj.nextId, ...
    'bbox', bbox, ...
    'kalmanFilter', kalmanFilter, ...
```

```
                'age', 1, ...
                'totalVisibleCount', 1, ...
                'consecutiveInvisibleCount', 0);
            % Add it to the array of tracks.
            obj.tracks(end + 1) = newTrack;
            % Increment the next id.
            obj.nextId = obj.nextId + 1;
        end
    end
  end
end
```

LICENSE PLATE EXTRACTION FUNCTION:

```
function [] = extract(inp)

grey_image = rgb2gray(imread(inp));

strel_square = strel('square', 3);    % Creates structuring element 3-by-3 square

q = grey_image;

r = grey_image;

for p = 1:100;

   q = imerode(q,strel_square);

   r = imdilate(r,strel_square);

end

for p = 1:100;

   q = imdilate(q,strel_square);

   r = imerode(r,strel_square);

end

imModified = imsubtract(imadd(grey_image,imsubtract(grey_image,q)), imsubtract(r,grey_image));

high_pass_filter = [0 1 0;1 -4 1;0 1 0]; % High Pass Filter

imModified_doub = imsubtract(imModified,imfilter(imModified, high_pass_filter));

[x,y,z] = size(grey_image);

for ipo = 1:x;

   for jpo = 1:y-1;

      if imModified_doub(ipo,jpo)>110

         imModified_doub(ipo,jpo) = 255;
```

```
        end
    end
end
imComp = imcomplement(imModified);
ieopen = imModified;
ieclostrel_square = imComp;
for kk = 1:10;
    ieopen = imerode(ieopen,strel_square);
    ieclostrel_square = imerode(ieclostrel_square,strel_square);
end
iclostrel_squarenotfinal = imreconstruct(ieclostrel_square,imComp);
iclostrel_squarefinal = imcomplement(iclostrel_squarenotfinal);
ibothat = imsubtract(iclostrel_squarefinal,imModified);
strel_squarehr = strel('rectangle',[1,3]);
xclostrel_square = ibothat;
for k = 1:20;
    xclostrel_square = imdilate(xclostrel_square,strel_squarehr);
end
for k = 1:20;
    xclostrel_square = imerode(xclostrel_square,strel_squarehr);
end
for k = 1:25;
```

```
    xclostrel_square = imerode(xclostrel_square,strel_squarehr);

end

for k = 1:25;

    xclostrel_square = imdilate(xclostrel_square,strel_squarehr);

end

strel_squarerect = strel('rectangle',[9,15]);

erodeDilate = imdilate(im2bw(xclostrel_square,0.7),strel_squarerect);   % Increase the threshold if the number plate is more whitish

numberPlate = zeros(x, y);

imMultiplied = immultiply(erodeDilate,grey_image);

imDouble = im2double(imMultiplied);

grey_image = im2double(rgb2gray(imread(inp)));

for ipo = 1:x;

    for jpo = 1:y;

        if imDouble(ipo,jpo)==0.0

            numberPlate(ipo,jpo) = 1.0;

        else

            numberPlate(ipo,jpo) = grey_image(ipo,jpo);

        end

    end

end

figure(1), imshow(numberPlate);
```

```
multipliedBinary = im2bw(imMultiplied,0.5);
figure(2), imshow(multipliedBinary);
plate = imadd(imcomplement(erodeDilate),multipliedBinary);
figure(3), imshow(plate);
imwrite(plate,'E:\8th sem\sa\number-plate-extraction-master\lpimg.jpg','jpg');
```

LICENSE PLATE NUMBER ENHANCEMENT FUNCTION:

```
function [] = extract2(inp,area)
img = imread(inp);
[x,y] = size(rgb2gray(img));
imGraySobel = edge(rgb2gray(img),'sobel');
row=1;
while(row<x)
   coloumn=1;
   while(coloumn<y)
      t=coloumn;
      temp=0;
      while(t<y&&imGraySobel(row,t)==0)
         temp=temp+1;
            t=t+1;
      end
      if(temp>48)
```

```
            coloumn=t+1;
        else
            while(coloumn<=t-1)
                imGraySobel(row,coloumn)=1;
                coloumn=coloumn+1;
            end
            coloumn=t+1;
        end
    end
    row=row+1;
end
greyCopy=imGraySobel;
row=0;
coloumn=0;
while(row<x)
    coloumn=0;
    while(coloumn<y)
        c=coloumn;
        r=row;
        temp=0;
        for i=1:5
            for j=1:5
```

```
            if(r+i<x && c+j<y && greyCopy(r+i,c+j)==1)
                temp=temp+1;
            end
        end
    end
    if(temp>15)
        for i=1:5
            for j=1:5
                greyCopy(r+i,c+j)=1;
            end
        end
    else
        for i=1:5
            for j=1:5
                greyCopy(r+i,c+j)=0;
            end
        end
    end
    coloumn=coloumn+5;
  end
  row=row+5;
end
```

```
plate = bwareaopen(imsubtract(greyCopy, bwareaopen(greyCopy, 4800)), 900);
[L, num] = bwlabel(plate, 4);
i=1;
j=1;
while (i<x)
   j=1;
   while (j<y)
      if((L(i,j)==5) || (L(i,j)==6) || (L(i,j)==9))
         plate(i,j)=1;
      else
         plate(i,j)=0;
      end
      j=j+1;
   end
   i=i+1;
end
%figure,imshow(plate)
plate1=plate;
if (area==1)
   plate = bwareaopen(plate1, 1100);
   plate2=imsubtract(plate1,plate);
   %figure, imshow(plate2);
```

```
else
    CC = bwconncomp(plate);
    numPixels = cellfun(@numel,CC.PixelIdxList);
    [biggest,idx] = max(numPixels);
    plate(CC.PixelIdxList{idx}) = 0;
    plate2=imsubtract(plate1,plate);
    %figure, imshow(plate2);
end
i=1;
while (i<x)
    j=1;
    while (j<y)
        if(plate2(i,j)==0)
            img(i,j,1) = 0;
            img(i,j,2) = 0;
            img(i,j,3) = 0;
        end
        j=j+1;
    end;
    i=i+1;
end;
gray=rgb2gray(img);
```

```
if (area==1)
   gray = imcomplement(gray);
end;
gray = im2double(gray);
[x,y]=size(gray);
for i1 = 1:x;
   for j1 = 1:y;
      if gray(i1,j1) == 0.0
         gray(i1,j1) = 1.0;
      end
   end
end
for i1 = 1:x
   gray(i1, y) = 1.0;
end;
for j1 = 1:y
   gray(x, j1) = 1.0;
end;
if (area == 1)
   level = 0.6;
end;
if (area == 2)
```

```
    level = 0.2;
end;
if (area == 3)
    level = 0.3;
end;
if (area == 4)
    level = 0.6;
end;
imBinary =im2bw(gray,level);
imBinary = imerode(imBinary, strel('rectangle', [2, 1]));
imBinary = imerode(imBinary, strel('rectangle', [1, 3]));
imBinaryComp = imcomplement(imBinary);
figure,imshow(imBinary);
figure,imshow(imBinaryComp);
```

INTERSECTION MODULE MAIN:

```
clc

clear

close all

%%

% tested for

% 'video_data_sunday/IMG_5108.mov'

% 'video_data_sunday/IMG_6914.mov' P1&P3

% 'video_data_sunday/IMG_6919.mov'

% 'video_data_sunday/IMG_6915_01.mov'

% 'video_data_sunday/IMG_6917_01.mov' the turn: p1 bbox "jumps" on the p2 box (which
disappears)

%%

videoPath = 'C:\Users\Onam\Desktop\final project\source_code\video_data_sunday\IMG_6919.mov';

videoId = getVideoId(videoPath); % this is just the IMG_6919 part of the videoPath

default_params =  loadParameters('default_params');

custom_params = loadParameters(videoId);

params = setstructfields(default_params, custom_params);

car_tracking(videoPath, params);
```

LOAD PARAMETERS METHOD:

```
function params = loadParameters(paramId)

paramsCustom = strcat('C:\Users\Onam\Desktop\traffic-violation-detection-master\traffic-violation-detection-master\source_code\setup_params\', paramId,'.mat');

paramsDefault = 'C:\Users\Onam\Desktop\traffic-violation-detection-master\traffic-violation-detection-master\source_code\setup_params\default_params.mat';

try

   params = load(paramsCustom);

   fprintf('Loaded initial params from: %s \n', paramsCustom);

catch

   params = load(paramsDefault);

   fprintf('Could not load %s \nUsing %s instead\n', paramsCustom, paramsDefault);

end

end
```

CAR TRACKS INITIALIZATION METHOD

```
function car_tracks = initializeCarTracks()

%Function for initializing car tracks

%   Detailed explanation goes here

car_tracks = struct(...

        'id', {}, ...

        'age', {}, ...

        'bbox', {}, ...
```

```
        'bbox_initial', {}, ...
        'velocity', {}, ...
        'priority', {}, ...
        'kalmanFilter', {}, ...
        'totalVisibleCount', {}, ...
        'consecutiveInvisibleCount', {});
end
```

DETECT CARS METHOD:

```
function [ centroids, bboxes, mask] = detectCars(system_object,frame,freehandMask)
%The detectObjects function returns the centroids and the bounding boxes of
%the detected objects. It also returns the binary mask,
%which has the same size as the input frame. Pixels with a value of 1 correspond
%to the foreground, and pixels with a value of 0 correspond to the background.
   % Detect the foreground in the current video frame
   mask = system_object.detector.step(frame);
   mask(freehandMask==0) = 0;
   % Use morphological opening to remove noise in the foreground
   se = strel('square', 3);

   mask = imopen(mask, se);
   mask = imclose(mask, se);
```

```
    mask = imfill(mask, 'holes');
    % Detect the connected components with the specified minimum area, and
    % compute their bounding boxes
    % Perform blob analysis to find connected components.
    [area, ~, identified_boxes] = system_object.blobAnalyser.step(mask);
    area;
    bboxes=[];
    centroids=[];
    %identified_boxes
    for k = 1:size(identified_boxes,1)
        bb = identified_boxes(k,:);
        xwidth = double(bb(3));
        ywidth = double(bb(4));
        frac = xwidth/ywidth;
        if frac >= 0.9
            centroids(k,:) = [(bb(1)+bb(3)/2) (bb(2)+bb(4)/2)];
            bboxes(k,:) = bb;
        end
    end
end
```

CAR TRACKING:

```
function car_tracking(video, params)

% detects cars, detects traffic violations

%   Detailed explanation goes here

% Create System objects used for reading video, detecting moving objects,

% and displaying the results.

system_object = setupSystemObject(video, params);

reportedViolations = []; % list of car ids with priority 1 or 3 that were reported to violate the right hand rule

%% Create an empty array of tracks

car_tracks = initializeCarTracks();

%% ID tracker

nextID = 1;

%% Detect moving objects and track them across video frames

frame = system_object.reader.step(); % read frames

if params.useFreehandMask

   %%freehandMask = freehand_mask(frame, getVideoId(video));

   freehandMask = params.freehandMask;

else

   freehandMask = ones(size(frame));

end

while ~isDone(system_object.reader)
```

```
    frame = system_object.reader.step(); % read frames

    [centroids, bounding_boxes, mask] = detectCars(system_object,frame, freehandMask);

    car_tracks = predictNewLocationOfTracks(car_tracks);

    [assignments, unassignedTracks, unassignedDetections] = ...

        detectionToTrackAssignment(car_tracks,centroids);

    car_tracks = updateAssignedTracks(assignments, centroids, bounding_boxes, car_tracks);

    car_tracks = updateUnassignedTracks(car_tracks, unassignedTracks);

    car_tracks = deleteLostTracks(car_tracks);

    [car_tracks, nextID ] = createNewTracks(params, centroids, bounding_boxes,
unassignedDetections, car_tracks, nextID);

    car_tracks = updateVelocity(car_tracks);

    displayTrackingResults(frame, car_tracks, mask, system_object);

    reportedViolations = checkForViolations(car_tracks, reportedViolations);

end

end
```

DETERMINE PRIORITY:

```
function priority = determinePriority(bbox, priorityPolygons)
    % bbox(1,2) are the upper left corner coords
    % use the lower left corner
    car = bbox(1:2);
    carWidth = bbox(3);
    carHeight = bbox(4);
    polyP1 = priorityPolygons.polyP1;
    polyP2 = priorityPolygons.polyP2;
    polyP3 = priorityPolygons.polyP3;
   if inpolygon(car(1)+carWidth, car(2)+carHeight, polyP3(:,1), polyP3(:,2))
        priority = 3;
    elseif  inpolygon(car(1)+carWidth, car(2)+carHeight, polyP2(:,1), polyP2(:,2))
        priority = 2;
    elseif inpolygon(car(1), car(2)+carHeight, polyP1(:,1), polyP1(:,2))
        priority = 1;
    else
        priority = -1;
    end
end
```

DETECT VOILATIONS:

```
function reportedViolations=checkForViolations(car_tracks, reportedViolations)
carsP1 = car_tracks([car_tracks(:).priority] == 1);
carsP2 = car_tracks([car_tracks(:).priority] == 2);
carsP3 = car_tracks([car_tracks(:).priority] == 3);
if length(carsP1) + length(carsP3) == 0
   return
end
for i = 1:length(carsP2)
   % width or length cannot be 0
   if any(carsP2(i).bbox(3:4) <=0)
      continue
   end
   % case priority 1 vs 2
   for j = 1:length(carsP1)
      if ~(any(carsP1(j).bbox(3:4) <= 0) || any(reportedViolations == carsP1(j).id))
         reportedViolations = determineIfViolaitonP12(carsP1(j), carsP2(i), reportedViolations);
      end
   end
   % case priority 2 vs 3
   for j = 1:length(carsP3)
      if  ~(any(carsP3(j).bbox(3:4) <= 0) || any(reportedViolations == carsP3(j).id))
```

```
            reportedViolations = determineIfViolaitonP23(carsP2(i), carsP3(j), reportedViolations);
        end
    end
end
end
```

CHAPTER 7

TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every possible conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. The system has been verified and validated by running the test data and live data.

Under the following tables the testing of different components of the project is described:

7.1: Test Cases for Highway Module

Table 7.1: Test Cases for Highway Module

Test Case Id	Description	Input State	Expected o/p	Actual o/p	Result
1	To detect the speed of a car and determine the anomaly	Traffic Video in which cars are passing.	All the cars has to be detected	All the cars are being detected	Pass
2	To detect the speed of a car and determine the anomaly	Traffic video in which a car performs an anomaly.	The car performing an anomaly must be detected and its screenshot must be captured.	The car performing the anomaly is detected and its screenshot is captured	Pass
3	To detect the speed of a car and determine the anomaly	Traffic video in which a car performs an anomaly.	The number plate of the car must be detected and the license plate number must be stored in the text document.	The number plate of the car is detected and the license plate number is stored in the text document.	Pass

7.2: Test Cases for Intersection Module

Table 7.2: Test Cases for Intersection Module

Test Case Id	Description	Input State	Expected o/p	Actual o/p	Result
1	Detection of the violation of a Right-hand rule.	A video with cars passing in different lanes.	The passing of first car should not show any violation	The first car didn't show violation	Pass
2	Detection of the violation of a Right-hand rule.	A video with cars passing in different lanes.	The low priority car if passed before the high priority, violation should be detected	Since the low priority car passed before the high priority, violation is detected.	Pass

CHAPTER 8

RESULTS

8.1 Results for Anomaly Detection

Fig 8.1: Screenshot of car on Highway

Fig 8.2: Background subtraction

Fig 8.3: Detecting car with anomaly

8.1.1 Results of Number plate detection

Fig 8.4: Car Demo

Fig 8.5: Background subtraction

Fig 8.6: Number plate captured

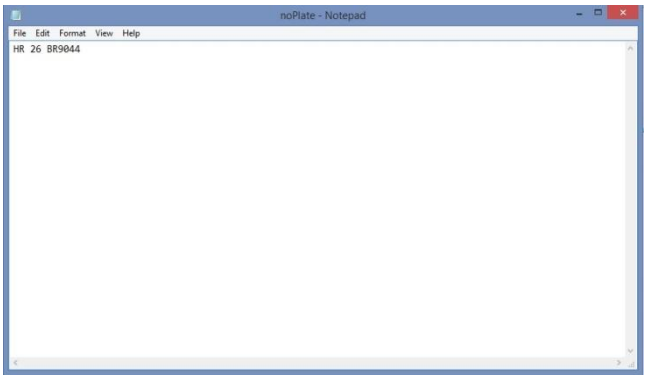

Fig 8.7: Storing in a text document

8.2 Results for Intersection/Right-hand rule violation

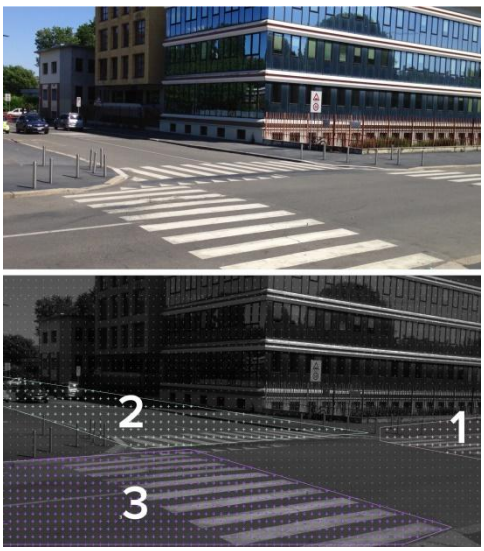

Fig 8.8: Showing the priorities of the road

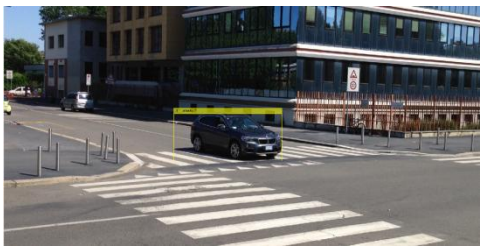

Fig 8.9: car travelling in one of the road

Fig 8.10: Background subtraction.

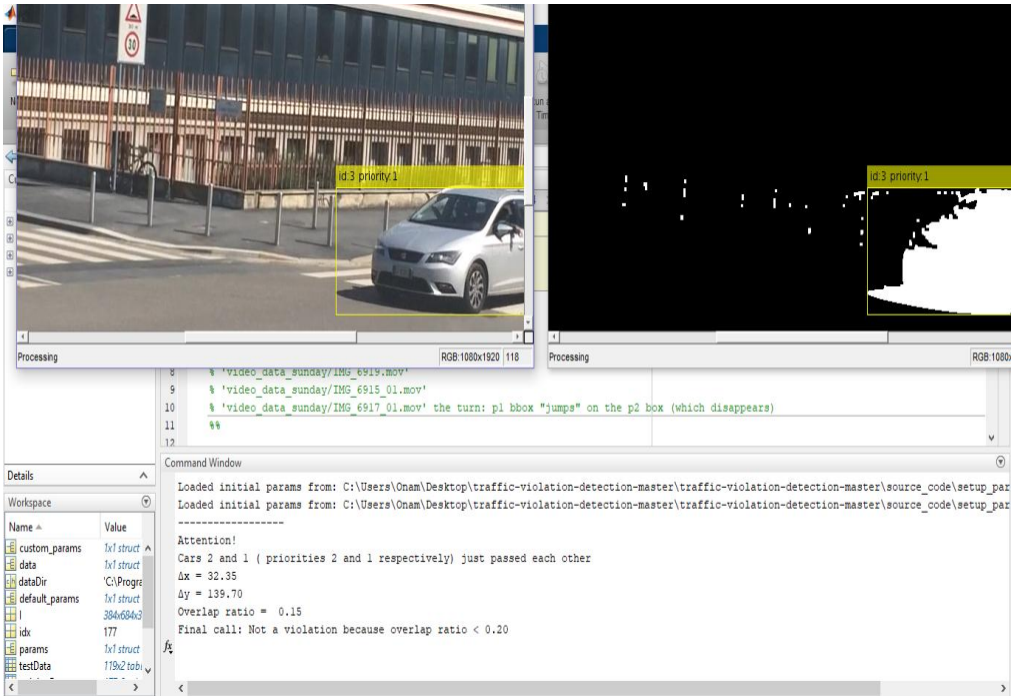

Fig 8.11: Car not performing any Violation.

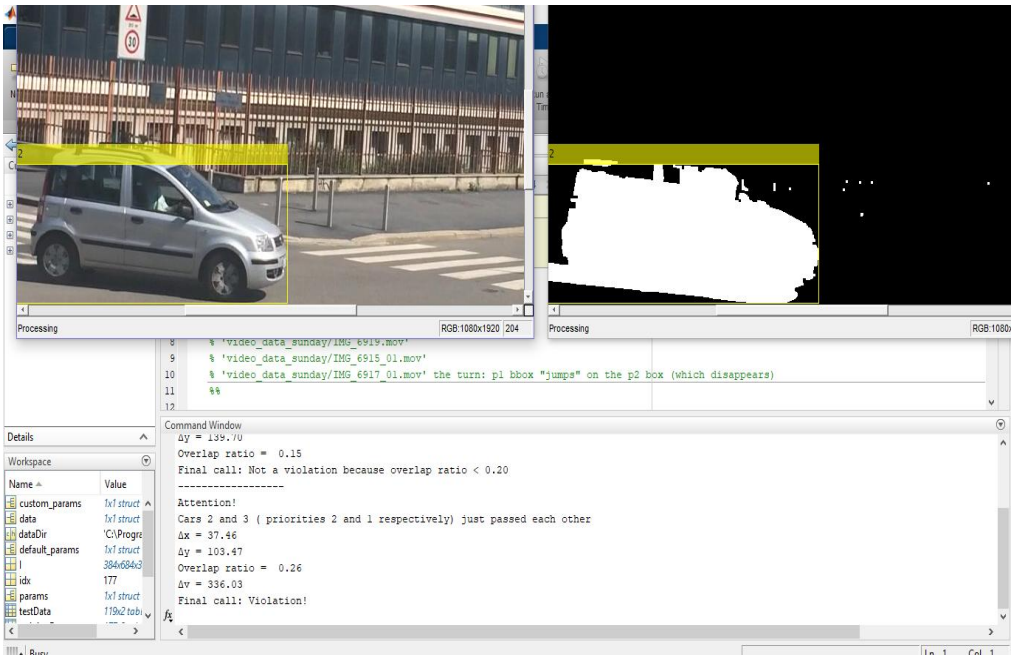

Fig 8.12: Car performing a Violation.

CHAPTER 9

CONCLUSION & FUTURE SCOPE

- The major limitation of this project is that it classifies the cars only for the car travelling in single direction. Thus this will be the future work so as to detect the intersection of cars in all the directions.
- The other future scope will be to improve the accuracy of classifying the cars for violating the right-hand rule.
- To calculate accurate speed when the camera is not stable.

CHAPTER 10

REFERENCES

1. S.Srilekha, G.N.Swamy and A.Anudeep Krishna, “A Novel approach for Detection and Tracking of Vehicles using Kalman filter” in CICN 2015. DOI:10.1109/CICN.2015.53.
2. Shaoqing Ren, Kaiming He, Ross Girshick and Jian Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks" in TPAMI 2016. DOI: 10.1109/TPAMI.2016.2577031.
3. Hui Li, Peirui Bai and Huajun Song, “Car tracking algorithm based on Kalman filter and compressive tracking” in CISP 2014. DOI: 10.1109/CISP.2014.7003744.
4. Madalina Cosmina Popescu and Lucian Mircea Sasu, "Feature extraction ,feature selection and machine learning for image classification: A case study", in OPTIM 2014. DOI: 10.1109/OPTIM.2014.6850925.
5. Hiroshi Unno, Kouki Ojima and Kelkichi Hayashibe, “Vehicle Motion Tracking Using Symmetry of Vehicle and Background Subtraction” in Intelligent Vehicles Symposium 2007. DOI: 10.1109/IVS.2007.4290269.
6. Ross Girshick, "Fast R-CNN", in ICCV 2015. DOI: 10.1109/ICCV.2015.169.
7. Lu Ming, “Image segmentation algorithm research and improvement” in ICACTE 2010. DOI: 10.1109/ICACTE.2010.5579114.
8. Xinyi Zhou, Wei Gong, WenLong Fu and Fengtong Du, “Application of deep learning in object detection” in ICIS 2017. DOI:10.1109/ICIS.2017.7960069.
9. Mao Shan, Stewart Worrall and Eduardo Nebot, “Long term vehicle motion prediction and tracking in large environments” in ITSC 2011. DOI: 10.1109/ITSC.2011.6082922.
10. D.M. Gavrila and V. Philomin, "Real-time object detection for smart vehicles", in ICCV.1999. DOI: 10.1109/ICCV.1999.791202.