SAVITRIBAI PHULE PUNE UNIVERSITY

A PROJECT REPORT ON

"Intelligent Traffic Light Control using Image Processing"

SUBMITTED TOWARDS THE
PARTIAL FULFILLMENT OF THE REQUIREMENTS OF

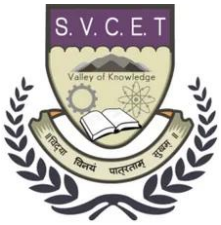
BACHELOR OF ENGINEERING (Computer Engineering)

BY

Maya Ravindra Gaikwad	B190994207
Komal Balasaheb Pawale	B190994218
Monali Harishchandra Shirole	B190994225
Sachin Shantaram Sabale	B190994223

Under the Guidance of

Prof. Nangare Vaibhav L.



DEPARTMENT OF COMPUTER ENGINEERING

Sahyadri Valley College of Engineering Technology
At-Rajuri, Tal-Junnar, Dist-Pune (412 411)

2022-2023

Sahyadri Valley College of Engineering Technology

DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

This is to certify that the project phase-II report entitled

"Intelligent Traffic Light Control using Image Processing"

Submitted by

Maya Ravindra Gaikwad	B190994207
Komal Balasaheb Pawale	B190994218
Monali Harishchandra Shirole	B190994225
Sachin Shantaram Sabale	B190994223

It is a bonfide work carried out by Students under the supervision of **Prof. Nangare Vaibhav L.** and it is submitted towards the partial fulfillment of the requirement of Bachelor of Engineering (Computer Engineering) Project.

Prof. Nangare Vaibhav L.	**Prof. P. Balaramudu**
Internal Guide & Examiner	Principal
Prof. Mundhe Bhalchandra B.	**Prof. ……..….**
Head of Department	External Examiner

Abstract

Increasing traffic congestion is a constant source of frustration, time loss, and expense to users and managers of transportation systems. Cities, countries, and state transportation agencies are persistently searching for ways to mitigate urban traffic congestion, while minimizing costs and maintenance requirements. India battleswith the dual challenge of pollution and congestion. Fifteen out of the top twenty most polluted cities in the world belong to India. In economic terms, the congestion losses combined for India's top four metros are over USD 22 billion annually. These high levels of congestion have a huge cost in the form of reduced productivity, fuel wastage, accidents, and traffic-related stress, simply due to time spent in traffic jams. Despite the increase in road length, newly constructed highways, and better connectivity, the problem of traffic congestion persists. With increasing vehicular traffic and limited road space, there is a dire need to adopt solution-centric and advancedtechnological measures to achieve free traffic flows in the capital city.

Technology can play a pivotal role in identifying these mobility gaps and transforming existingtransportation services. In urban areas, traffic signals are the limiting factors and common congestion points. Therefore, controlling traffic congestion relies on have an efficient and well-managed traffic signal control policy. There is no doubt that signals are one of the most powerful tools for urban traffic control available to city authorities and their correct installation can improve both traffic flow and the safety of all road users. A Smart Traffic Light System leverages technology to im- prove traffic outcomes by introducing a sensing network, which provides feedback to the existing network, so that it can adapt to the changing traffic density patterns and provide necessary signals to the controller in real-time.. The proposed model controls the clearance time of each lane in a sequential manner and is function of real time traffic density. The approach is rather hybrid – a blend of sensors networksand camera technology.

Keywords:-image processing, congestion control, smart traffic control system , Accident prevention system ,emergency prioritization.

Acknowledgments

It gives us great pleasure in presenting the preliminary project report on
"Intelligent Traffic Light Control using Image Processing"

I would like to take this opportunity to thank my internal guide **Prof. Nangare Vaibhav L.** for giving me all the help and guidance I needed. I am really grateful to them for their kind support. Their valuable suggestions were very helpful.

I am also grateful to **Prof. Munde B. B.** Head of Computer Engineering Department, Sahyadri Valley College of Engineering Technology for his indispensable support, suggestions.

In the end our special thanks to our principle **Prof. P. Balaramudu Sir** for providing various resources such as laboratory with all needed software platforms, continuousInternet connection, for Our Project.

Maya Ravindra Gaikwad	B190994207
Komal Balasaheb Pawale	B190994218
Monali Harishchandra Shirole	B190994225
Sachin Shantaram Sabale	B190994223

(B E. Computer Engineering)

INDEX

Acknowledgement **II**

Abstract **III**

1 INTRODUCTION **2**

1.1 Introduction 3

1.2 Objectives of project report 3

1.3 Organization of project report 4

2 LITERATURE SURVEY **5**

2.1 Study Of Research Paper 6

3 Project Requirement **9**

3.1 Problem Statement 10

3.2 Project Scope 10

3.2.1 Project Scope and Aim 10

4 Project Requirement **11**

4.1 Introduction 12

4.1.1 User Classes and Characteristics 12

4.1.2 Assumptions and Dependencies 12

4.2 Functional Requirement 13

4.2.1 System Feature 1(Functional Requirement) 13

4.3 External Interface Requirement 13
4.3.1 Software Requirements(Platform Choice) 13
4.3.2 Hardware Requirements 13
4.4 Non-Functional Requirement 13
4.4.1 Performance Requirements 13
4.4.2 Safety Requirements: 14
4.4.3 Security Requirements 14
4.4.4 Software Quality Attributes 14
4.5 Analysis Models: (SDLC Model To Be Applied) 15

5 System Analysis 19
5.1 System Architecture 20
5.1.1 Flowchart:- 21
5.2 Algorithm 21
5.3 Layers of CNN 25
5.3.1 Convolutional Layer 25
5.3.2 Pooling/Subsampling Layer 26
5.3.3 ReLU 27
5.3.4 Fully connected layers 27
5.4 Data Flow Diagram 28
5.4.1 Level 0 Data Flow Diagram 28
5.5 UML Diagram 29
5.5.1 Use-cases 29
5.5.2 Activity Diagram: 30
5.5.3 Class Diagram: 31
5.5.4 Sequence Diagram: 32
5.5.5 Component Diagram: 33
5.5.6 Deployment Diagram: 33

6 Software Information 34
6.1 Tools and Technology Used 35

7 Project Plan **37**

7.1 Project Estimates 38

7.1.1 Effort Estimate Table: 38

7.1.2 Project Description: 38

7.1.3 Estimation of KLOC: 39

7.2 Risk Management 39

7.2.1 Overview of Risk Mitigation, Monitoring, Management 39

7.3 Time Line Chart 42

8 Testing **43**

8.1 Introduction 44

8.2 Types Of Testing 44

8.3 Test Strategy And Approach 46

8.3.1 Test Objectives 46

8.3.2 Features To Be Tested 46

8.3.3 Integration Testing 46

8.3.4 Security Testing 46

8.3.5 Performance Testing 47

8.3.6 Regression Testing 47

8.3.7 Usability Testing 47

8.4 Test Result 48

9 RESULT **49**

9.1 Screenshots 50

10 CONCLUSION **53**

11 REFERENCES **55**

List of Figures

4.1 Waterfall Model 16

5.1 Block Diagram 20
5.2 Flowchart 21
5.3 CNN Architecture 21
5.4 RGB image layears 23
5.5 llustration of a biological neuron (left) and its mathematical model (right) 23
5.6 Convolution operation on image matrix 25
5.7 Pictorial representation of max pooling and average pooling 26
5.8 Pictorial representation of ReLU functionality 27
5.9 DFD Level 0 28
5.10 DFD Level 1 28
5.11 UseCase Diagram 29
5.12 Activity Diagram 30
5.13 Class Diagram 31
5.14 Sequence Diagram 32
5.15 Component Diagram 33
5.16 Deployment Diagram 33

7.1 Time Line Chart 42

List of Tables

7.1 Effort Estimate Table 38
7.2 Project Scheduling 38

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Traffic congestion has long been recognized as an economic and social impediment worldwide having a detrimental effect on human productivity, air quality, fuel us- age and overall quality of life. Increasing traffic congestion is a constant source of frustration, time loss, and expense to users and managers of transportation sys- tems. Cities, countries, and state transportation agencies are persistently searching for ways to mitigate urban traffic congestion, while minimizing costs and mainte- nance requirements. The congestion and delays that characterize much of the re- gion's transportation system have also intensified other social and environmental problems such as productivity losses, wasted energy, degraded air quality, and in- creased vehicular accidents. In future, for the overall development of the country itis important that infrastructure, of which road traffic is a very important part, should be made state-of-the-art. This makes the study very valid in the present conditions. The problem of traffic is a complex one requiring design, planning, engineering and institutional inputs for developing a proper solution. In urban areas, traffic signals are the limiting factors and common congestion points. Therefore, controlling traf- fic congestion relies on having an efficient and well-managed traffic signal control policy. There is no doubt that signals are one of the most powerful tools for urban traffic control available to city authorities and their correct installation can improveboth traffic flow and the safety of all road users.. The problem can be solved by us- ing present emergent technologies like IOT and image processing. A large amount of research is going on in these fields. In our proposed system we have made use ofthese technologies in order to develop a smart and intelligent traffic control system. Both of the methods individually have their own drawbacks individually.

1.2 OBJECTIVES OF PROJECT REPORT

- To review the current situation of traffic congestion in Pune city.
- To find & manage mental causes of traffic congestion in city.
- To analyze the impact of traffic congestion in Pune city.

- To recommend some solutions to reduce traffic congestion.

1.3 ORGANIZATION OF PROJECT REPORT

- Chapter 2 Deals with the Project Related Work i.e Literature Survey.
- Chapter 3 Giving an overall view of the techniques used in the system
- Chapter 4 Deals with System Design.
- Chapter 5 Project Plan
- Chapter 6 Implementation Part
- Conclusion and at last references

CHAPTER 2

LITERATURE SURVEY

2.1 STUDY OF RESEARCH PAPER

- Adil Hilmani , Abderrahim Maizate, and Larbi Hassouni, Automated Real-Time Intelligent Traffic Control System for Smart Cities Using Wireless Sensor Networks, Wireless Communications and Mobile Computing Volume 2020.

  In this paper, we propose an intelligent traffic control system based on the design of a wireless sensor network (WSN) in order to collect data on road traffic and also on available parking spaces in a smart city. In addition, the proposed system has innovative services that allow drivers to view the traffic rate and the number of available parking spaces to their destination remotely using an Android mobile application to avoid traffic jams and to take another alternative route to avoid getting stuck and also to make it easier for drivers when looking for a free parking space to avoid unnecessary trips. Our system integrates three smart subsystems connected to each other (crossroad management, parking space management, and a mobile application) in order to connect citizens to a smart city.

- Duy Nhat Nguyen, Adaptive Traffic Control System: Design And Simulation, Concordia University, July 2015.

  This paper offers such a solution based on an adaptive traffic control algorithm which takes the road network topology and dynamically varying traffic streams as input, and guarantees dependable and optimal mobility for vehicles. The algorithm calculates dependable passages for vehicles to cross road intersections, and enables point-to-point travel by minimizing travel time and maximizing fuel consumption. The adaptive algorithm is embedded in the Arbiter, managed by an Intersection Manager at every road intersection. A distributed traffic management architecture, consisting of a hierarchy of road managers, is proposed in the thesis. Extensions to the adaptive algorithm and the architecture are given.

- Peng Jing, Hao Huang and Long Chen, An Adaptive Traffic Signal Control in

a Connected Vehicle Environment: A Systematic Review, School of Automotive and Traffic Engineering, Jiangsu University, 22 August 2017

The purpose of this paper is to critically review the existing methods of adaptive traffic signal control in a connected vehicle environment and to compare the advantages or disadvantages of those methods. Further, a systematic framework on connected vehicle based adaptive traffic signal control is summarized to support the future research. Future research is needed to develop more efficient and generic adaptive traffic signal control methods in a connected vehicle environment.

- MADHUKAR, Adaptive Traffic Signal Control Using Fuzzy Logic, IJCRT — Volume 6, Issue 2 April 2020

  These systems were designed to handle regular pattern of traffic. But today traffic is fluctuating frequently that demands new computing power in the traffic management system. This paper presents the system which can handle the various sizes of vehicles on the road. This model first evaluates the concatenated vehicle size by considering all size vehicles. Calculated vehicle size is then fed into the fuzzy controller with the other input parameters, traffic flow, and traffic density to calculate cycle length. In this basically, three parameters are taken, named as Size of Vehicles, Traffic Density, and Traffic Flow. Fuzzy rules are built by considering the Size of Vehicles, Traffic Density and Traffic Flow, cycle length is calculated using various datasets. The cycle which is calculated using the model of this paper is then compared with the fixed time control cycle length

- Hong K. Lo H. E Chow, “Adaptive Resolution, and Accuracy, March 2002; Accepted: July 2002

  Adaptive traffic control system (ATCS) aims at controlling the imminent traffic, which is yet to arrive and hence not known perfectly. An ATCS can use historical data based on time of day or day of week, or real-time detected data,

to formulate control strategies, in the hope that the current or historical arrival profile will remain representative for the upcoming situation. More advanced ATCS may include a short-term traffic prediction module for improved prediction accuracy. In any case, the performance of ATCS depends on its ability to predict the upcoming traffic pattern.

CHAPTER 3

PROJECT REQUIREMENT

3.1 PROBLEM STATEMENT

Vehicle detection from UAV images has some difficulties due to the extremely high resolution of the images. These problems were also examined in the study. Especially, the performance of methods has been compared to find the car on video frames. Thus, precision and average Intersection over Union (IoU) of the car was investigated. In accordance with real-time applications, the methods were performed over real data.

3.2 PROJET SCOPE

3.2.1 Project Scope and Aim

- Scope : The scope of this study is to compare the methods of object detection methods, which are added every day, by using the aerial and terrestrial videos to find the appropriate methods according to the data group. In addition, the deficiencies and advantages of the methods are revealed and the results that will form the basis for new models are obtained.
- Aim: This study aims to compare object detection methods on aerial and terrestrial videos to find suitable methods according to the data group. In addition to the local video, the videos obtained from **unmanned aerial vehicle (UAV)**were also used in the study.

CHAPTER 4

PROJECT REQUIREMENT

4.1 INTRODUCTION

This software requirement specification (SRS) report expresses complete description about proposed System. This document includes all the functions and specifications with their explanations to solve related problems.

4.1.1 User Classes and Characteristics

Basic knowledge of using computers is adequate to use this application.
Knowledge of how to use a mouse or keyboard and internet browser is necessary.
The user interface will be friendly enough to guide the user.

4.1.2 Assumptions and Dependencies

- Assumptions:

  [1] The product must have an interface which is simple enough to understand.

  [2] All the software such as python, etc are installed and running on the computers

- Dependencies:

  [1] All necessary software's are available for implementing and use of the system. Proposed system would be designed, developed and implemented based on the software requirements specifications document. users should have basic knowledge of computer and we also assure that the users will be given software training documentation and reference material.

  [2] Well Trained dataset

4.2 FUNCTIONAL REQUIREMENT

4.2.1 System Feature 1(Functional Requirement)

Functional requirement describes features, functioning, and usage of a product/system or software from the perspective of the product and its user. Functional requirements are also called as functional specifications were synonym for specification is design.

4.3 EXTERNAL INTERFACE REQUIREMENT

4.3.1 Software Requirements(Platform Choice)

- Tools - Python IDE
- Programming Language - Python
- Software Version - Python 3.5

4.3.2 Hardware Requirements

- Processor - Pentium IV/Intel I3 core
- Speed - 1.1 GHz
- RAM - 512 MB (min)
- Hard Disk - 20GB
- Keyboard - Standard Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - LED Monitor

4.4 NON-FUNCTIONAL REQUIREMENT

4.4.1 Performance Requirements

- High Speed :- System should process requested task in parallel for various action to give quick response. Then system must wait for process completion.

- Accuracy:- System should correctly execute process, display the result accurately. System output should be in user required format.

4.4.2 Safety Requirements:

The data safety must be ensured by arranging for a secure and reliable transmission media. The source and destination information must be entered correctly to avoid any misuse or malfunctioning. Password generated by user is consisting of characters, special character number so that password is difficult to hack. So, that user account is safe.

4.4.3 Security Requirements

Secure access of confidential data (user's details). Information security means protecting information and information systems from unauthorized access, use, disclosure, disruption, modification or destruction.The terms information security, computer security and information assurance are frequently incorrectly used interchangeably. These fields are interrelated often and share the common goals of protecting the confidentiality, integrity and availability of information; however, there are some subtle differences between them.

4.4.4 Software Quality Attributes

[1] Runtime System Qualities: Runtime System Qualities can be measured as the system executes.

[2] Functionality: The ability of the system to do the work for which it was intended.

[3] Performance:The response time, utilization, and throughput behavior of the system. Not to be confused with human performance or system delivery time.

[4] Security:A measure of systems ability to resist unauthorized attempts at usage or behavior modification, while still providing service to legitimate users.

[5] Availability: (Reliability quality attributes falls under this category) the measure of time that the system is up and running correctly; the length of time between failures and the length of time needed to resume operation after a failure.

[6] Usability: The ease of use and of training the end users of the system. Sub qualities: learn ability, efficiency, affect, helpfulness, control.

[7] Interoperability: The ability of two or more systems to cooperate at runtime.

4.5 ANALYSIS MODELS: (SDLC MODEL TO BE APPLIED)

One of the basic notions of the software development process is SDLC models which stands for Software Development Life Cycle models. SDLC is a continuous process, which starts from the moment, when its made a decision to launch the project, and it ends at the moment of its full remove from the exploitation. There is no one single SDLC model. They are divided into main groups, each with its features and weaknesses. Evolving from the first and oldest waterfall SDLC model, their variety significantly expanded.

The SDLC models diversity is predetermined by the wide number of product types starting with a web application development to a complex medical software. And if you take one of the SDLC models mentioned below as the basis in any case, it should be adjusted to the features of the product, project, and company. The most used, popular and important SDLC models are given below:

[1] Waterfall Model

[2] Iterative Model

[3] Spiral Model

[4] V-shaped Model

[5] Agile Model

Waterfall Model

Waterfall is a cascade SDLC model, in which development process looks like the flow, moving step by step through the phases of analysis, projecting, realization, testing, implementation, and support. This SDLC model includes gradual execution of every stage completely. This process is strictly documented and predefined with features expected to every phase of this software development life cycle model.

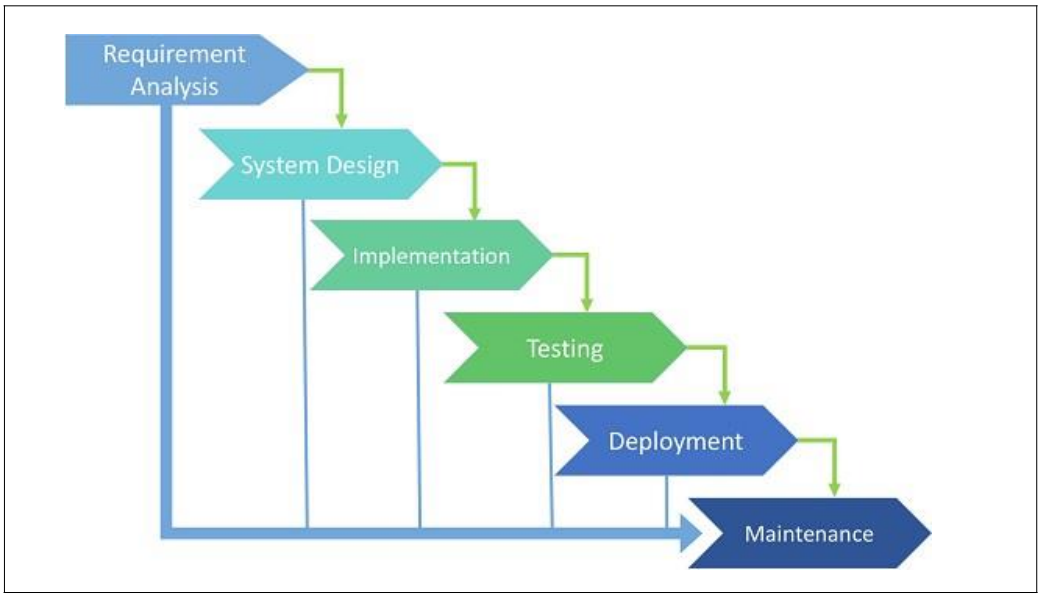


Figure 4.1: Waterfall Model

[1] Planning and requirement analysis

Each software development life cycle model starts with the analysis, in which the stakeholders of the process discuss the requirements for the final product. The goal of this stage is the detailed definition of the system requirements. Besides, it is needed to make sure that all the process participants have clearly understood the tasks and how every requirement is going to be implemented. Often, the discussion involves the QA specialists who can interfere the process with additions even during the development stage if it is necessary.

[2] Designing project architecture

At the second phase of the software development life cycle, the developers are actually designing the architecture. All the different technical questions that may appear on this stage are discussed by all the stakeholders, including the customer. Also, here are defined the technologies used in the project, team load, limitations, time frames, and budget. The most appropriate project decisions are made according to the defined requirements.

[3] Development and programming

After the requirements approved, the process goes to the next stage actual development. Programmers start here with the source code writing while keeping in mind previously defined requirements. The system administrators adjust the software environment, front-end programmers develop the user interface of the program and the logics for its interaction with the server. The programming by itself assumes four stages:-

- Algorithm development
- Source code writing
- Compilation
- Testing and debugging

[4] Testing

The testing phase includes the debugging process. All the code flaws missed during the development are detected here, documented, and passed back to the

developers to fix. The testing process repeats until all the critical issues are removed and software work ow is stable.

[5] Deployment

When the program is finalized and has no critical issues it is time to launch it for the end users. After the new program version release, the tech support team joins. This department provides user feedback; consult and support users during the time of exploitation. Moreover, the update of selected components is included in this phase, to make sure, that the software is up-to-date and is invulnerable to a security breach.

CHAPTER 5

SYSTEM ANALYSIS

5.1 SYSTEM ARCHITECTURE

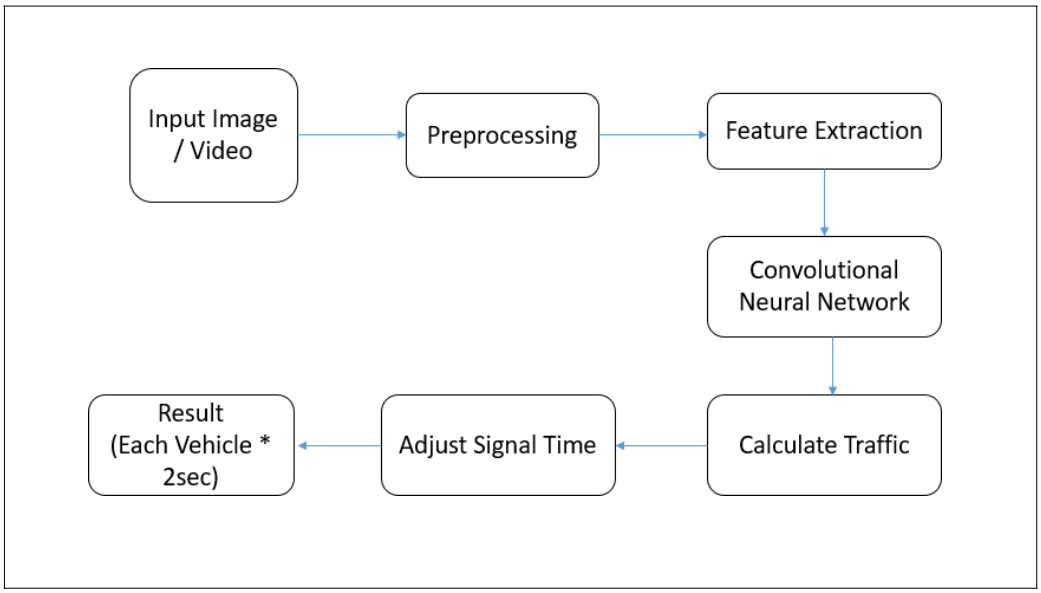


Figure 5.1: Block Diagram

we are traffic dataset for protype based traffic signal controlling based on vehicles is proposed in the system where we apply algorithm (CNN: - Convolutional Neural Networks).Convolutional Neural Networks is a popular deep learning technique for current visual recognition tasks. There are four layered concepts in Convolutional Neural Networks:

- Convolution,
- ReLu,
- Pooling and
- Full Connectedness (Fully Connected Layer).

5.1.1 Flowchart:-

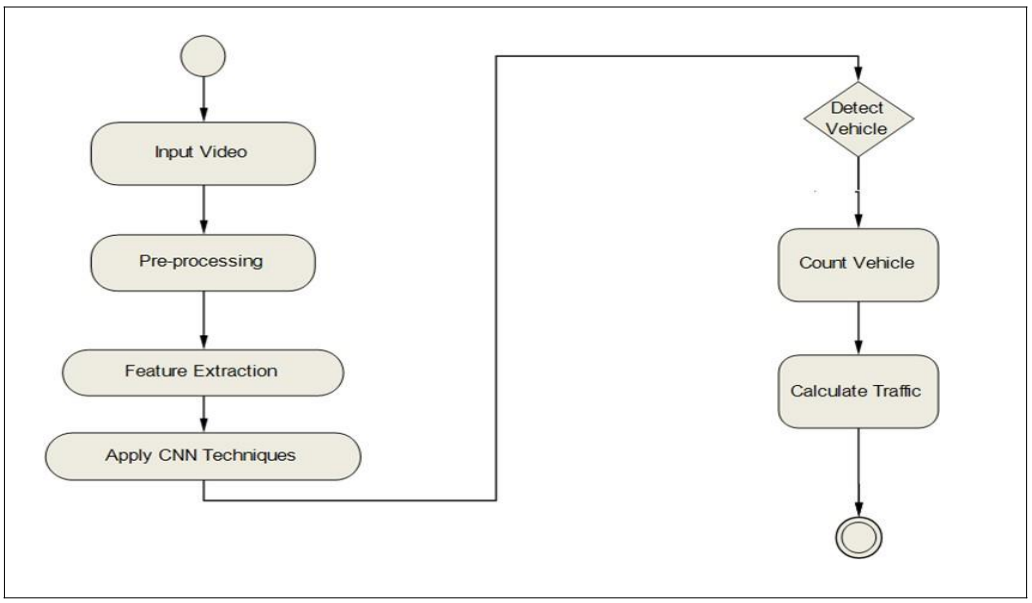


Figure 5.2: Flowchart

5.2 ALGORITHM

Convolutional Neural Networks

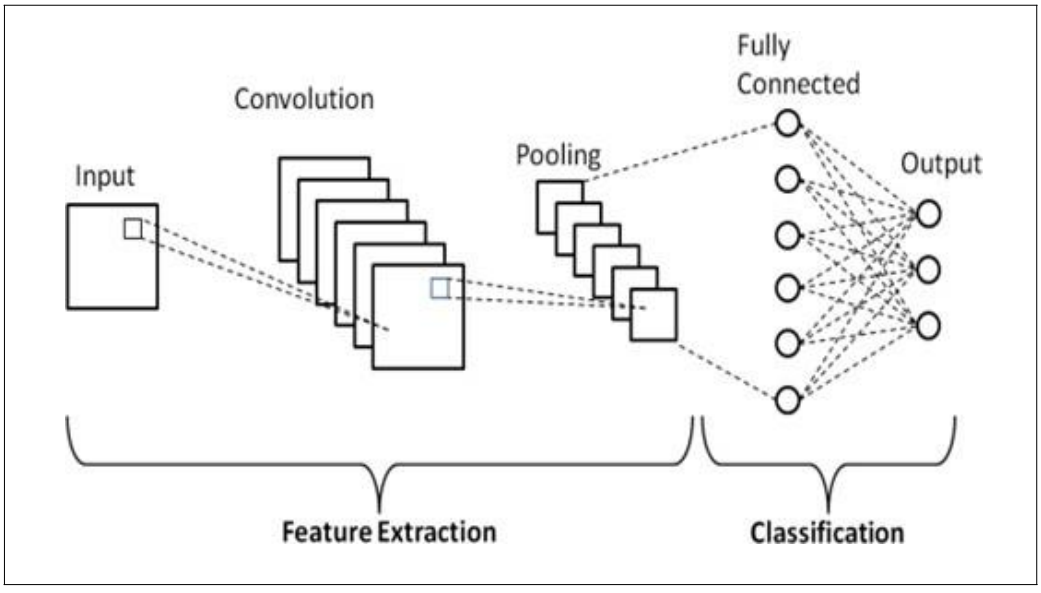


Figure 5.3: CNN Architecture

CNN or the convolutional neural network (CNN) is a class of deep learning neural networks. In short think of CNN as a machine learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and be able to differentiate one from the other.

CNN works by extracting features from the images. Any CNN consists of the following:

- The input layer which is a grayscale image
- The Output layer which is a binary or multi-class labels
- Hidden layers consisting of convolution layers, ReLU (rectified linear unit) layers, the pooling layers, and a fully connected Neural Network

It is very important to understand that ANN or Artificial Neural Networks, made up of multiple neurons is not capable of extracting features from the image. This is where a combination of convolution and pooling layers comes into the picture. Similarly, the convolution and pooling layers can't perform classification hence we need a fully connected Neural Network. Before we jump into the concepts further let's try and understand these individual segments separately.

The role of CNN is to reduce the images into a form that is easier to process, without losing features critical towards a good prediction. This is important when we need to make the algorithm scalable to massive datasets.

The challenge with images having multiple color channels is that we have huge volumes of data to work with which makes the process computationally intensive. In other worlds think of it like a complicated process where the Neural Network or any machine learning algorithm has to work with three different data (R-G-B values in this case) to extract features of the images and classify them into their appropriate categories

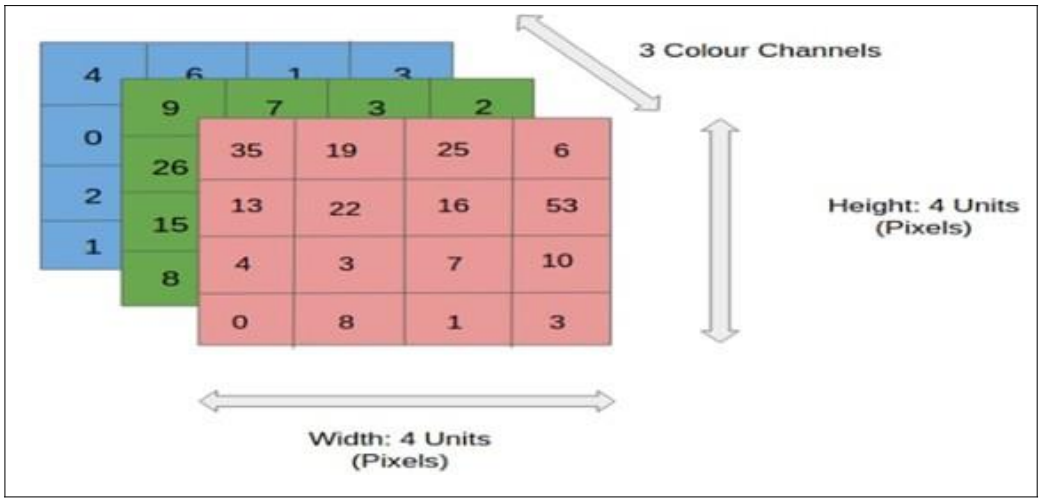


Figure 5.4: RGB image layears

The role of CNN is to reduce the images into a form that is easier to process, without losing features critical towards a good prediction. This is important when we need to make the algorithm scalable to massive datasets.

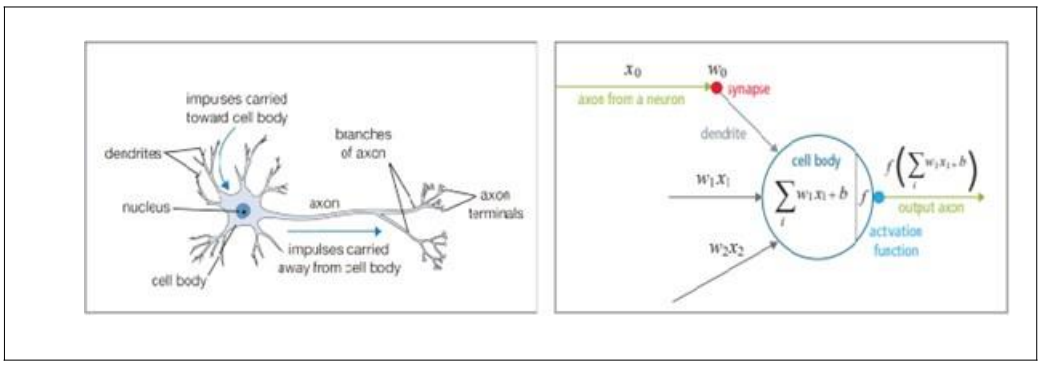


Figure 5.5: llustration of a biological neuron (left) and its mathematical model (right)

In a real animal neural system, a neuron is perceived to be receiving input signals from its dendrites and producing output signals along its axon. The axon branches out and connects via synapses to dendrites of other neurons. When the combination of input signals reaches some threshold condition among its input dendrites, the neuron is triggered and its activation is communicated to successor neurons.

In the neural network computational model, the signals that travel along the axons (e.g., x0) interact multiplicatively (e.g., w0x0) with the dendrites of the other

Neuron based on the synaptic strength at that synapse (e.g., w0). Synaptic weights are learnable and control the influence of one neuron or another. The dendrites carry the signal to the cell body, where they all are summed. If the final sum is above a specified threshold, the neuron fires, sending a spike along its axon. In the computational model, it is assumed that the precise timings of the firing do not matter and only the frequency of the firing communicates information. Based on the rate code interpretation, the firing rate of the neuron is modeled with an activation function f that represents the frequency of the spikes along the axon. A common choice of activation function is sigmoid. In summary, each neuron calculates the dot product of inputs and weights, adds the bias, and applies non-linearity as a trigger function (for example, following a sigmoid response function).

A CNN is a special case of the neural network described above. A CNN consists of one or more convolutional layers, often with a subsampling layer, which are followed by one or more fully connected layers as in a standard neural network.

The design of a CNN is motivated by the discovery of a visual mechanism, the visual cortex, in the brain. The visual cortex contains a lot of cells that are responsible for detecting light in small, overlapping sub-regions of the visual field, which are called receptive fields. These cells act as local filters over the input space, and the more complex cells have larger receptive fields. The convolution layer in a CNN performs the function that is performed by the cells in the visual cortex .

A typical CNN for recognizing traffic signs is shown in Figure 4. Each feature of a layer receives inputs from a set of features located in a small neighborhood in the previous layer called a local receptive field. With local receptive fields, features can extract elementary visual features, such as oriented edges, end-points, corners, etc., which are then combined by the higher layers.

In the traditional model of pattern/image recognition, a hand-designed feature extractor gathers relevant information from the input and eliminates irrelevant variabil-

ities. The extractor is followed by a trainable classifier, a standard neural network that classifies feature vectors into classes.

5.3 LAYERS OF CNN

By stacking multiple and different layers in a CNN, complex architectures are built for classification problems. Four types of layers are most common: convolution layers, pooling/subsampling layers, non-linear layers, and fully connected layers.

5.3.1 Convolutional Layer

If we observe Figure carefully we will see that the kernel shifts 9 times across image. This process is called Stride. When we use a stride value of 1 (Non-Strided) operation we need 9 iterations to cover the entire image. The CNN learns the weights of these Kernels on its own. The result of this operation is a feature map that basically detects features from the images rather than looking into every single pixel value.

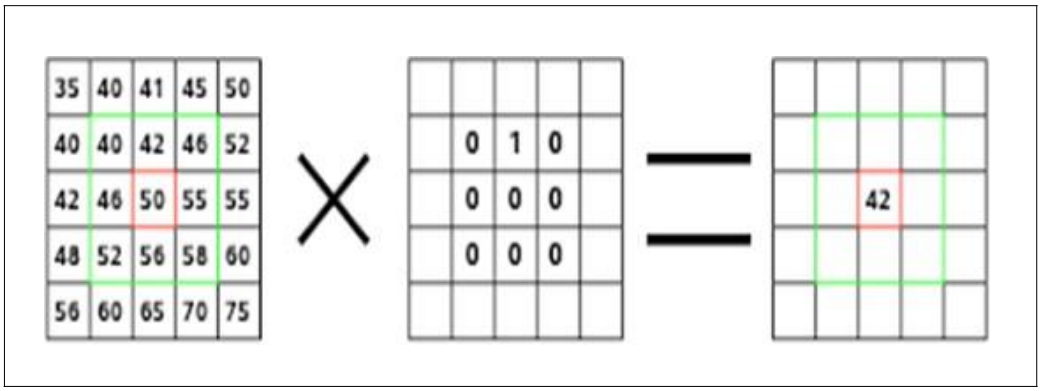


Figure 5.6: Convolution operation on image matrix

The general objective of the convolution operation is to extract high-level features from the image. We can always add more than one convolution layer when building the neural network, where the first Convolution Layer is responsible for capturing gradients whereas the second layer captures the edges. The addition of layers depends on the complexity of the image hence there are no magic numbers on how many layers to add. Note application of a 3 x 3 filter results in the original image

results in a 3 x 3 convolved feature, hence to maintain the original dimension often the image is padded with values on both ends.

5.3.2 Pooling/Subsampling Layer

The pooling layer applies a non-linear down-sampling on the convolved feature often referred to as the activation maps. This is mainly to reduce the computational complexity required to process the huge volume of data linked to an image. Pooling is not compulsory and is often avoided. Usually, there are two types of pooling, Max Pooling, that returns the maximum value from the portion of the image covered by the Pooling Kernel and the Average Pooling that averages the values covered by a Pooling Kernel. Figure below provides a working example of how different pooling techniques work.

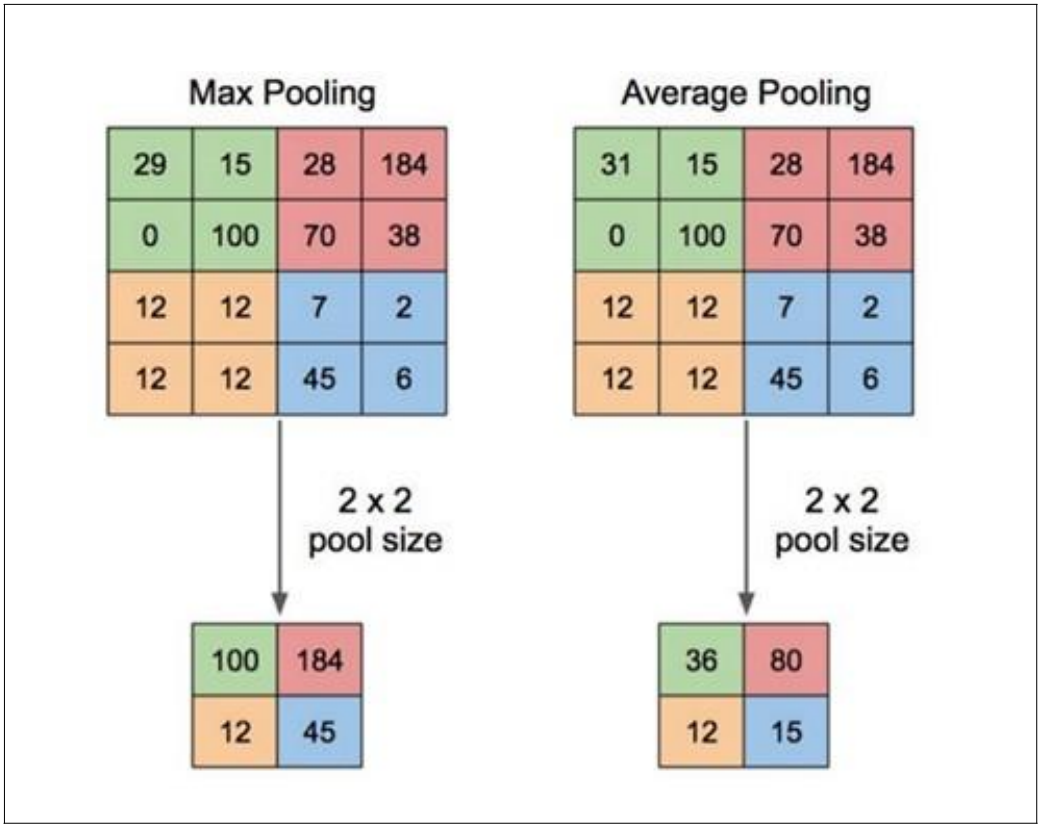


Figure 5.7: Pictorial representation of max pooling and average pooling

Non-linear layers Neural networks in general and CNNs in particular rely on a non-linear "trigger" function to signal distinct identification of likely features on each hidden layer. CNNs may use a variety of specific functions —such as rectified linear units (ReLUs) and continuous trigger (non-linear) functions—to efficiently implement this non-linear triggering

5.3.3 ReLU

A ReLU implements the function y = max(x,0), so the input and output sizes of this layer are the same. It increases the nonlinear properties of the decision function and of the overall network without affecting the receptive fields of the convolution layer. The advantage of a ReLU is that the network trains many times faster. ReLU functionality is illustrated in Figure 8, with its transfer function plotted above the arrow.

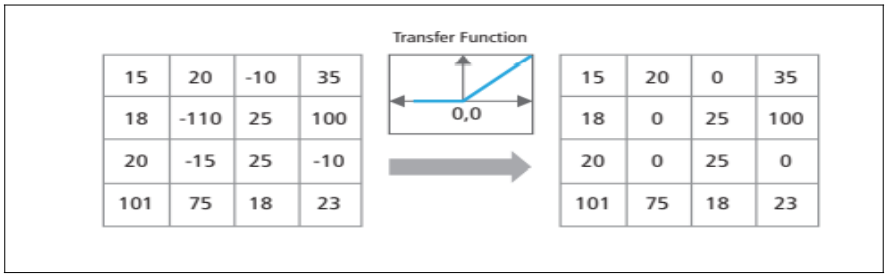


Figure 5.8: Pictorial representation of ReLU functionality

5.3.4 Fully connected layers

Fully connected layers are often used as the final layers of a CNN. These layers mathematically sum a weighting of the previous layer of features, indicating the precise mix of "ingredients" to determine a specific target output result. In case of a fully connected layer, all the elements of all the features of the previous layer get used in the calculation of each element of each output feature. Figure below explains the fully connected layer L. Layer L-1 has two features, each of which is 2x2, i.e., has four elements. Layer L has two features, each having a single element.

5.4 DATA FLOW DIAGRAM

5.4.1 Level 0 Data Flow Diagram

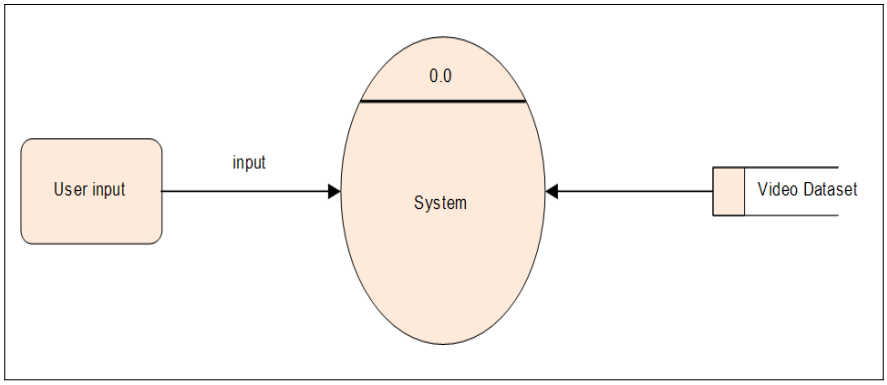


Figure 5.9: DFD Level 0

5.4.1.1 Level 1 Data Flow Diagram

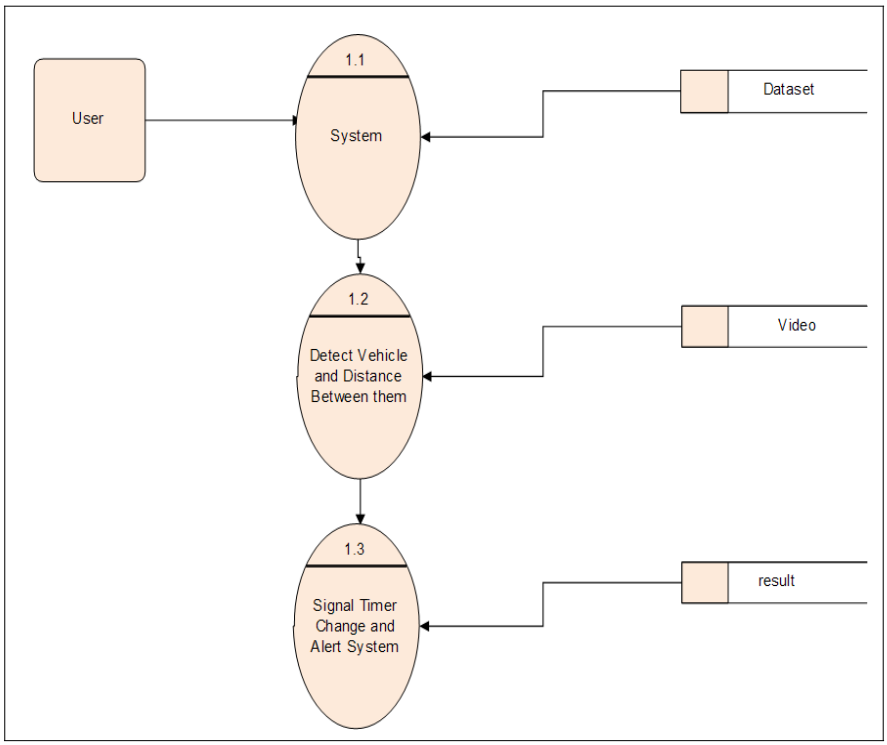


Figure 5.10: DFD Level 1

5.5 UML DIAGRAM

5.5.1 Use-cases

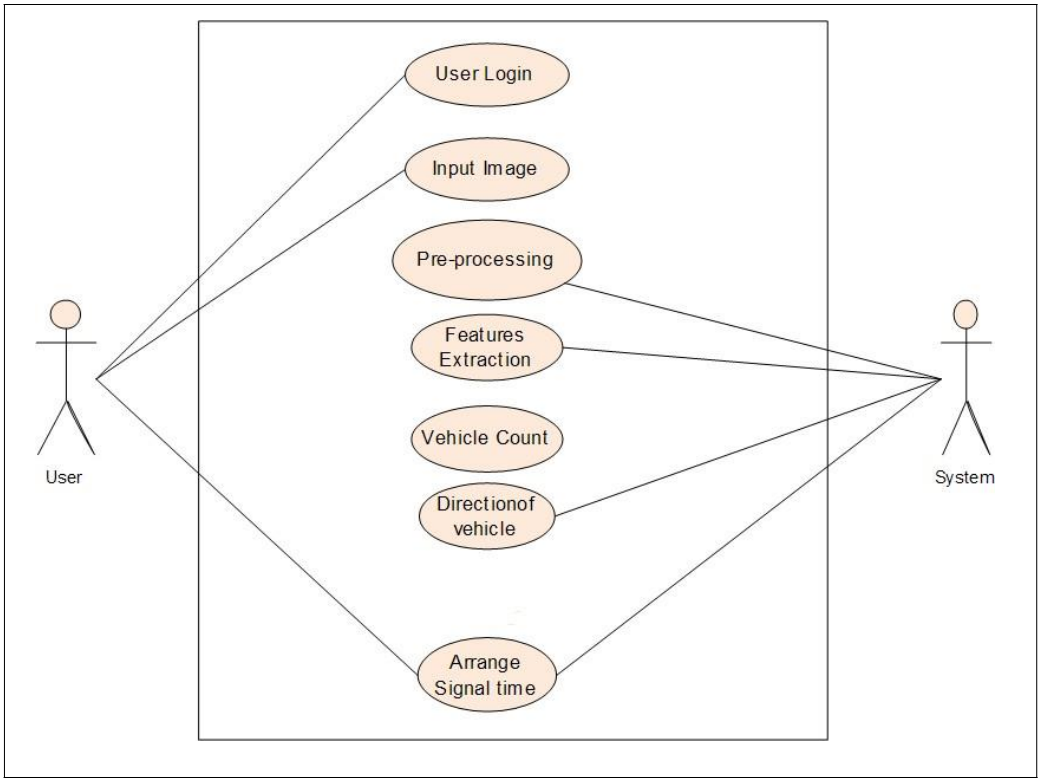


Figure 5.11: UseCase Diagram

5.5.2 Activity Diagram:

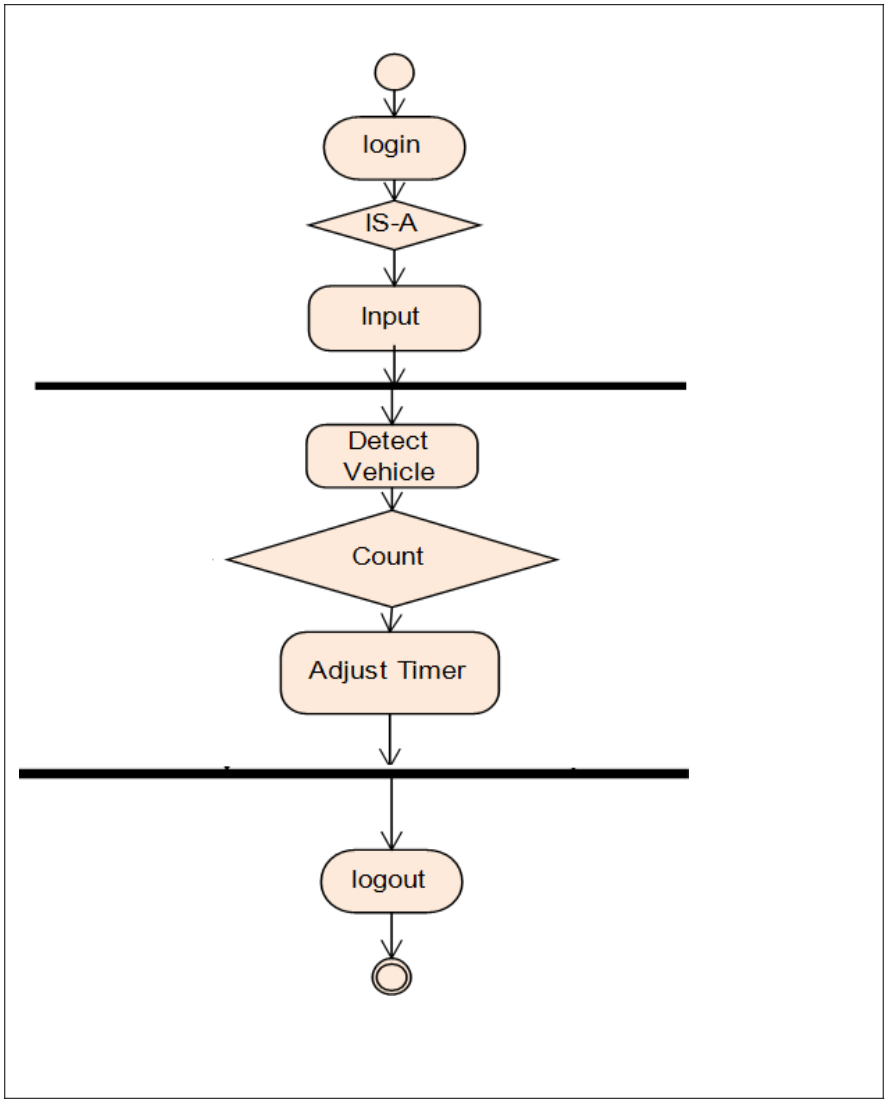


Figure 5.12: Activity Diagram

5.5.3 Class Diagram:

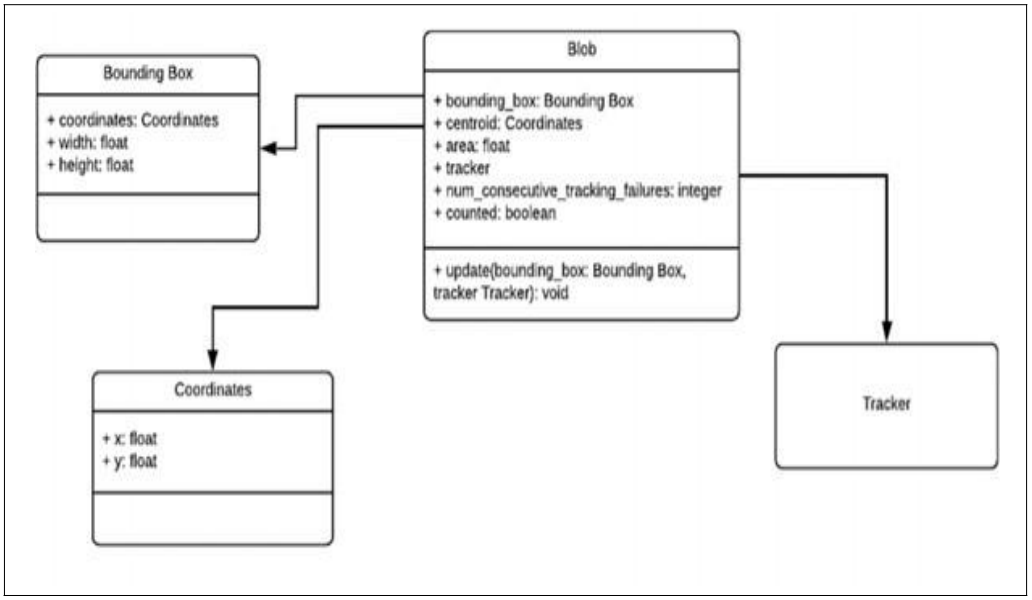


Figure 5.13: Class Diagram

5.5.4 Sequence Diagram:

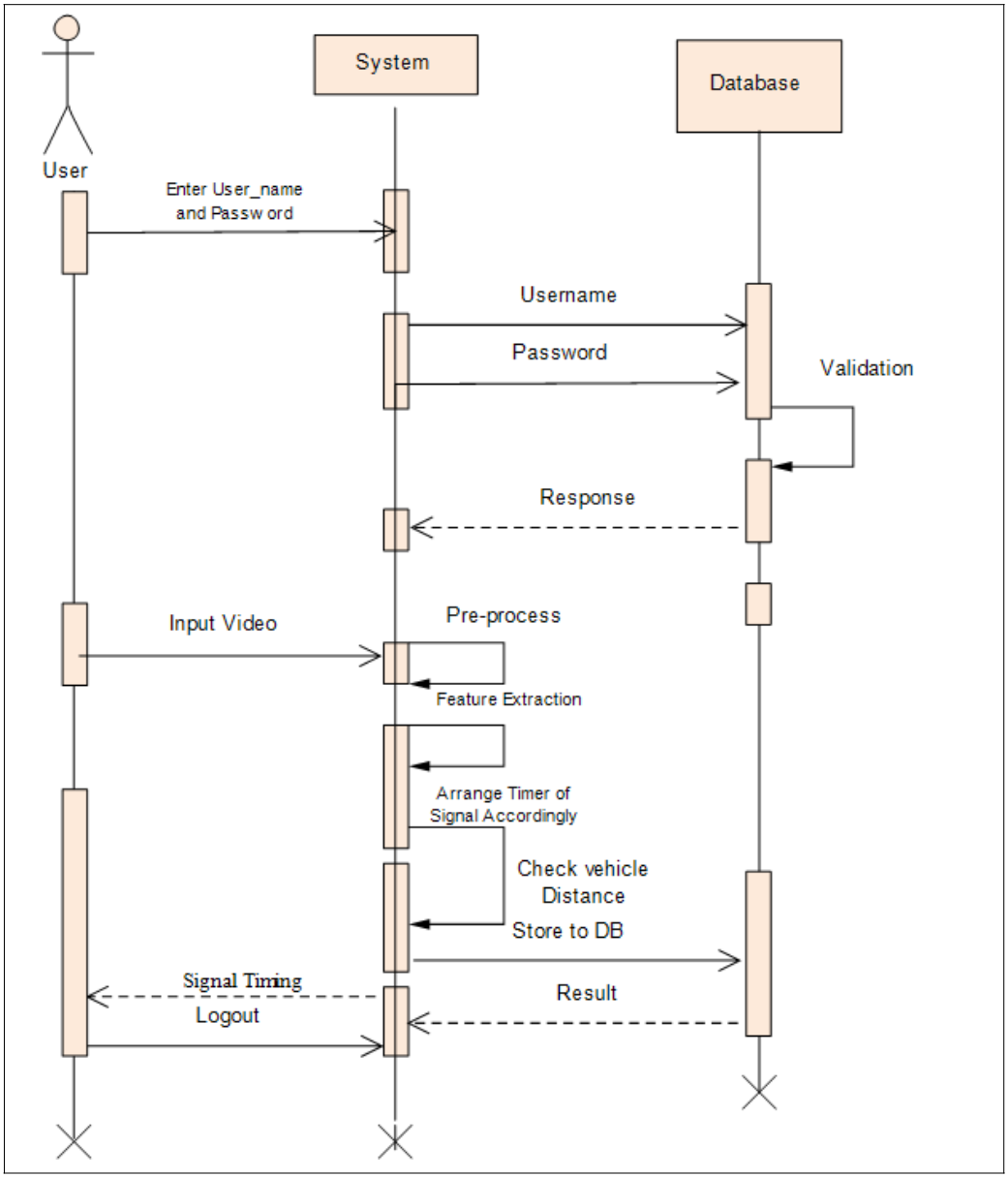


Figure 5.14: Sequence Diagram

5.5.5 Component Diagram:

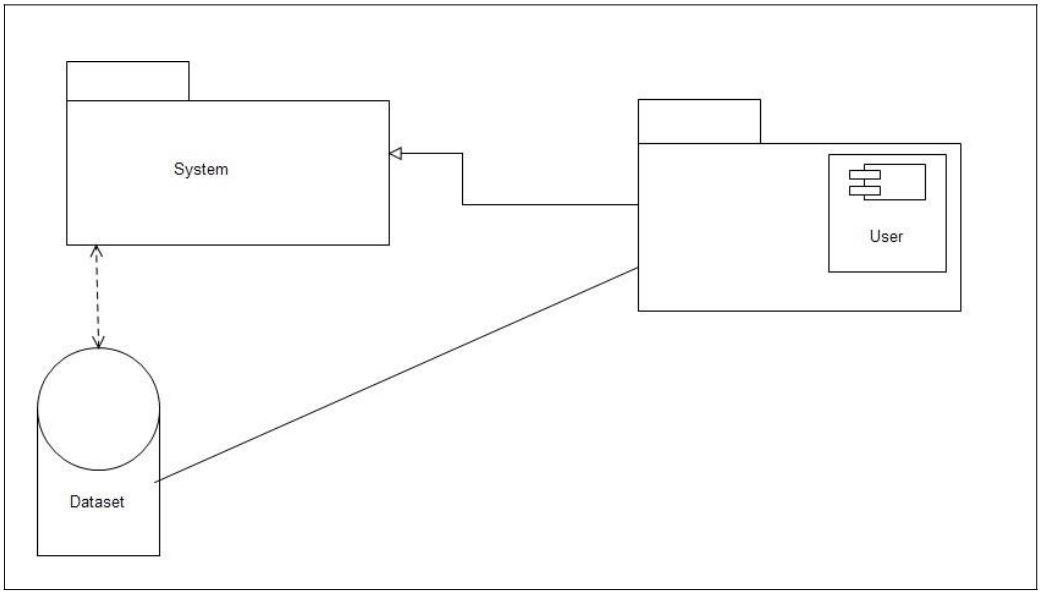


Figure 5.15: Component Diagram

5.5.6 Deployment Diagram:

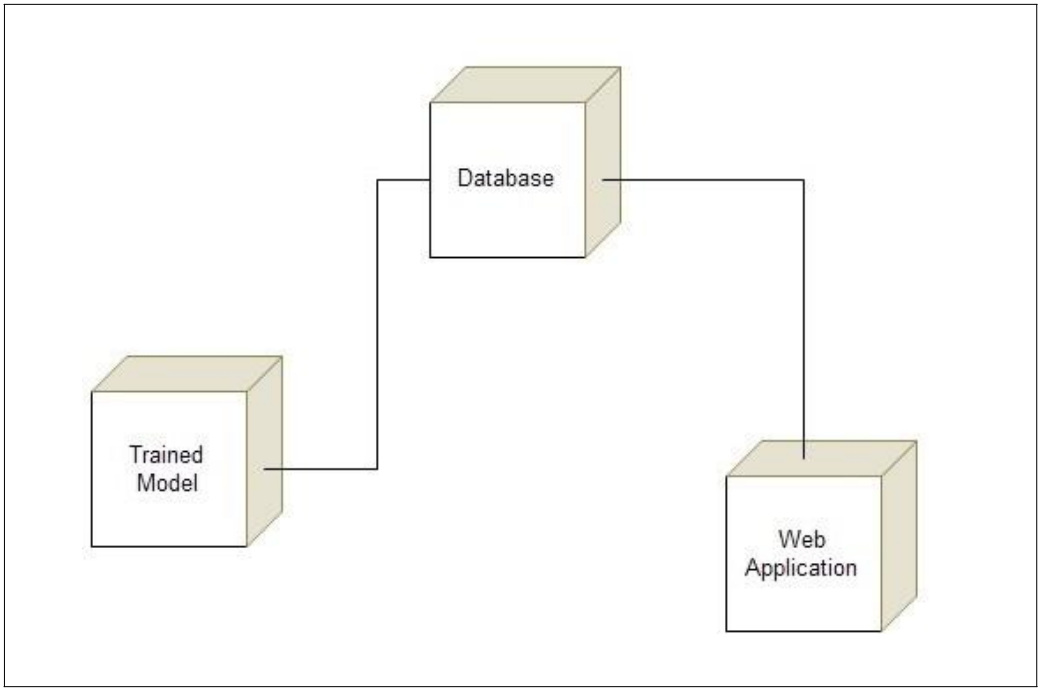


Figure 5.16: Deployment Diagram

CHAPTER 6

SOFTWARE INFORMATION

6.1 TOOLS AND TECHNOLOGY USED

Python

Python is an interpreted, high-level, general-purpose programming language. Created by **Guido van Rossum** and first released in 1991, Python has a design philoso- phy that emphasizes code readability, notably using signifficant whitespace. It pro- vides constructs that enable clear programming on both small and large scales.VanRossum led the language community until July 2018. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including pro-cedural, object-oriented, and functional programming.

Python features a comprehensive standard library, and is referred to as ”batteries included”. Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open-source software and has a community-based development model. Python and CPython are managed by the non-profit Python Software Foundation.

Python is a general-purpose object-oriented programming language with high-level programming capabilities. It has become famous because of its apparent and easily understandable syntax, portability and easy to learn. Python is a programming language that includes features of C and Java. It provides the style of writing an elegant code like C, and for object-oriented programming, it offers classes and objects like Java.

- Python was developed in the late eighties, i.e., late 1980’s by Guido van Rossum at the National Research Institute for Mathematics and Computer Science in the Netherlands as a successor of ABC language capable of exception handling and interfacing.
- Python is derived from programming languages such as ABC, Modula 3, small talk, Algol-68. Van Rossum picked the name Python for the new language from a TV show, Monty Python’s Flying Circus.
- Python page is a file with a .py extension that contains could be the combina-

tion of HTML Tags and Python scripts.

- In December 1989 the creator developed the 1st python interpreter as a hobby and then on 16 October 2000, Python 2.0 was released with many new features. On 3rd December 2008, Python 3.0 was released with more testing and includes new features.
- Python is an open source scripting language., which means that anyone can download it freely from www.python.org and use it to develop programs. Its source code can be accessed and modified as required in the project.Python is one of the official languages at Google.

Features of Python

[1] Easy to Learn and Use. Python is easy to learn and use.

[2] Expressive Language. Python language is more expressive means that it is more understandable and readable.

[3] Interpreted Language.

[4] Cross-platform Language.

[5] Free and Open Source.

[6] Object-Oriented Language.

[7] Extensible.

[8] Large Standard Library.

CHAPTER 7

PROJECT PLAN

7.1 PROJECT ESTIMATES

7.1.1 Effort Estimate Table:

Task	Effort weeks	Deliverables	Milestones
Analysis of existing systems & compare with proposed one	4 weeks		
Literature survey	1 weeks		
Designing & planning	2 weeks		
System flow	1 weeks		
Designing modules & its´ de-Liverables	2 week	Modules: design document	
Implementation	7 weeks	Primary system	
Testing	4 weeks	Test Reports	Formal
Documentation	2 weeks	Complete project report	Formal

Table 7.1: Effort Estimate Table

7.1.2 Project Description:

Phase	Task	Description
Phase 1	Analysis	Analyse the information given in the IEEE paper.
Phase 2	Literature survey	Collect raw data and elaborate on literature surveys.
Phase 3	Design	Assign the module and design the process flow control.
Phase 4	Implementation	Implement the code for all the modules and integrate all the modules.
Phase 5	Testing	Test the code and overall process weather the process works properly.
Phase 6	Documentation	Prepare the document for this project with conclusion and future enhancement.

Table 7.2: Project Scheduling

7.1.3 Estimation of KLOC:

The number of lines required for implementation of various modules can be estimated as follows:

Sr.No.	Modules	KLOC
1	Graphical User Interface	0.20
2	Back-end Algorithm Implementation	1.2
3	Front-Side Coding	1.2
4	Back-end Connectivity	0.6

Thus the total number of lines required is approximately 2.60 KLOC.

D = (Total KLOC / KLOC in a day)/30

= (3.6/0.025)/30

= 4.8

7.2 RISK MANAGEMENT

7.2.1 Overview of Risk Mitigation, Monitoring, Management

Risk management organizational role

Each member of the organization will undertake risk management. The development team will consistently be monitoring their progress and project status as to identify present and future risks as quickly and accurately as possible. With this said, the members who are not directly involved with the implementation of the product will also need to keep their eyes open for any possible risks that the development team did not spot. The responsibility of risk management falls on each member of the organization, while William Lord maintains this document.

Business Impact Risk

- Amount and quality of documentation that must be produced and delivered to customer the customer will be supplied with a complete online help file and users manual for Game Forge. Coincidentally, the customer will have access

to all development documents for Game Forge, as the customer will also be grading the project.

- Governmental constraints in the construction of the product none known.
- Costs associated with late delivery Late delivery will prevent the customer from issuing a letter of acceptance for the product, which will result in an incomplete grade for the course for all members of the organization.
- Costs associated with a defective product Unknown at this time.

Customer Related Risks

- Have you worked with the customer in the past? Yes, All team members have completed at least one project for the customer, though none of them have been to the magnitude of the current project.
- Does the customer have a solid idea of what is required? Yes, the customer has access to both the System Requirements Specification, and the Software Requirements Specification.
- Will the customer agree to spend time in formal requirements gathering meetings to identify project scope? Unknown. While the customer will likely participate if asked, the inquiry has not yet been made.

Process Risks

- Does senior management support a written policy statement that emphasizes the importance of a standard process for software development? N/A. PA Software does not have a senior management. It should be noted that the structured method has been adopted. At the completion of the project, it will be determined if the software method is acceptable as a standard process, or if changes need to be implemented.
- Has your organization developed a written description of the software process to be used on this project? Yes.

- Are staff members willing to use the software process? Yes. The software process was agreed upon before development work began.
- Is the software process used for other products? N/A. PA Software has no other projects currently.

Technical Issues

- Are facilitated application specification techniques used to aid in communication between the customer and the developer? The development team will hold frequent meetings directly with the customer. No formal meetings are held (all informal). During these meetings the software is discussed and notes are taken for future review.
- Are specific methods used for software analysis? Special methods will be used to analyze the software progress and quality. These are a series of tests and reviews to ensure the software is up to speed. For more information, see the Software Quality Assurance and Software Configuration Management documents.
- Do you use a specific method for data and architectural design? Data and architectural design will be mostly object oriented. This allows for a higher degree data encapsulation and modularity of code.

Technology Risk

- Is the technology to be built new to your organization? No
- Does the software interface with new or unproven hardware? No
- Is a specialized user interface demanded by the product requirements? Yes.

Development Environment Risks Is a software project management tool available? No. No software tools are to be used. Due to the existing deadline, the development team felt it would be more productive to begin implementing the project than trying to learn new software tools. After the completion of the project software tools may be implemented for future projects.

7.3 TIME LINE CHART

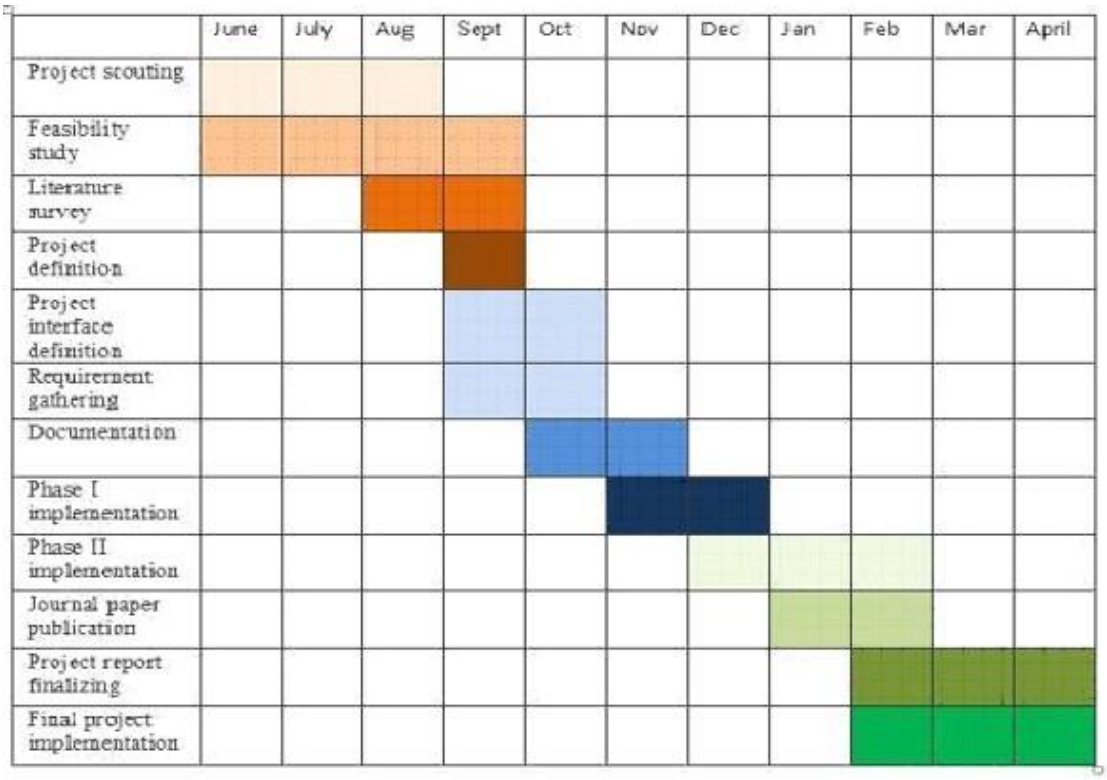


Figure 7.1: Time Line Chart

CHAPTER 8

TESTING

8.1 INTRODUCTION

Software testing is a method for examining the ability of a framework and confirms that it meets its results. It is produced by developers to keep up the quality of programming; software testing still remains a craftsmanship, because of less comprehension of the terms of testing. Fundamental issue in regards to the software testing is from the complication of programming: It can't test entire program with less complexity.. Testing is more than just debugging. The purpose of testing can be quality assurance, verification and validation, or reliability estimation. There are 2 major types of testing's

[1] Correctness testing

[2] Reliability testing.

Software testing has to deal between expenses, time and quality. Presently authors will see the test cases for the making website. Developers execute functional non-functional testing system. Software testing shall design test cases with higher probability of ending mistakes. To meet the correct objective and to give the efficient test cases the maxi- mum numbers of errors should be reported. Higher the number better is test objective achieved. Software testing shall guarantee that user accepts the software released for him to operate without errors.

8.2 TYPES OF TESTING

[1] Unit testing

Unit testing involves the design of test cases that validate that the internal-program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests performbasic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path

of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

[2] Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

[3] Functional Testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

[4] System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

[5] Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

8.3 TEST STRATEGY AND APPROACH

Field testing will be performed manually and functional tests will be written in detail.

8.3.1 Test Objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

8.3.2 Features To Be Tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

8.3.3 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

8.3.4 Security Testing

Security Testing is defined as a type of Software Testing that ensures software systems and applications are free from any vulnerabilities, threats, risks that may cause a big loss. Security testing of any system is about finding all possible loopholes and weaknesses of the system which might result into a loss of information, revenue, repute at the hands of the employees or outsiders of the Organization. The goal of security testing is to identify the threats in the system and measure its potential vulnerabilities, so the system does not stop functioning or is exploited. The client needs

to enter his login id and password. If the user name and password is valid then only the client is allowed to log in.

8.3.5 Performance Testing

In developing the system, we are going to use Java which will reduce the response time. In Performance Testing, We are going to test Response time for each Screen. It is a type of non-functional testing. Performance testing is testing that is performed; to determine how fast some aspect of a system performs under a particular work-load. It can serve different purposes like it can demonstrate that the system meets performance criteria. It can compare two systems to find which performs better. Or it can measure what part of the system or workload causes the system to perform badly. This process can involve quantitative tests done in a lab, such as measuring the response time or the number of MIPS (millions of instructions per second) at which a system functions.

8.3.6 Regression Testing

Regression Testing is defined as a type of software testing to confirm that a recent program or code change has not adversely affected existing features. Regression Testing is nothing but a full or partial selection of already executed test cases which are re-executed to ensure existing functionalities work fine. This testing is done to make sure that new code changes should not have side effects on the existing functionalities. It ensures that the old code still works once the new code changes are done. Testing done to ensure that, the changes to the application have not adversely affected previously tested functionality. Here testing will take care of the test cases passed during the first module testing will not be affected in the subsequent rounds of module testing.

8.3.7 Usability Testing

Usability testing is an informal process, reviewing your design with a few users, or a formal process in an established usability lab. Either way, the purpose is the same i.e. learning first hand from the users where your design works and where it needs

improvement. The following points would be considered:-

- **Effectiveness:** Accuracy and completeness with which specified users can achieve specified goals in a particular environment.
- **Efficiency:** The resources expended in relation to the accuracy and completeness of the goals achieved.
- **Satisfaction:** The comfort and acceptability of the work system to its users and other people affected by its use.
- **Learn ability:** How easy it is for users to accomplish basic tasks the first time they encounter the design.
- **Memory ability:** When the user returns to the design after a period of not using it, how easily can they re-establish proficiency.

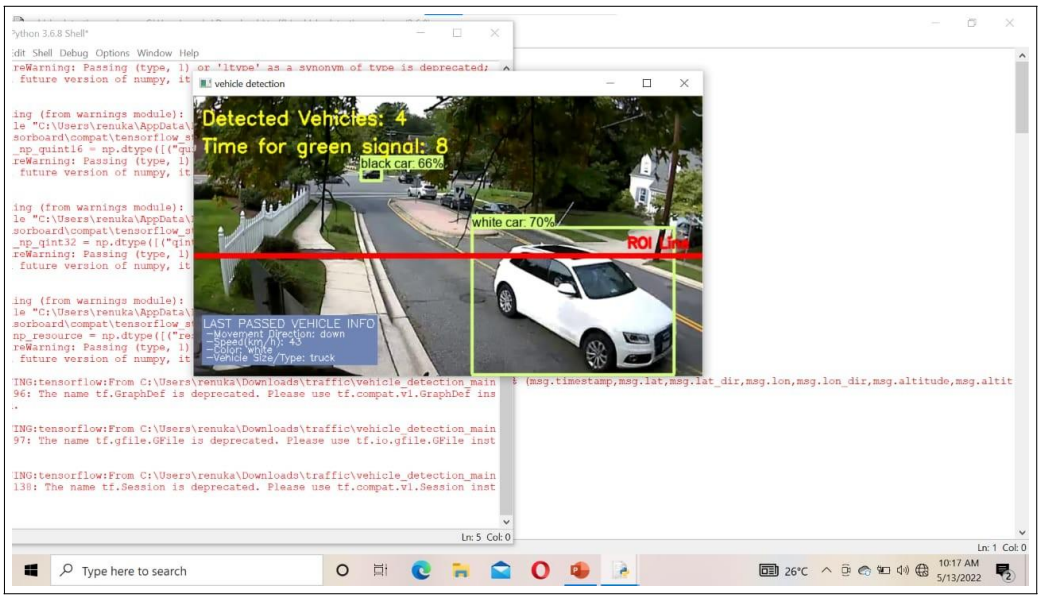
8.4 TEST RESULT

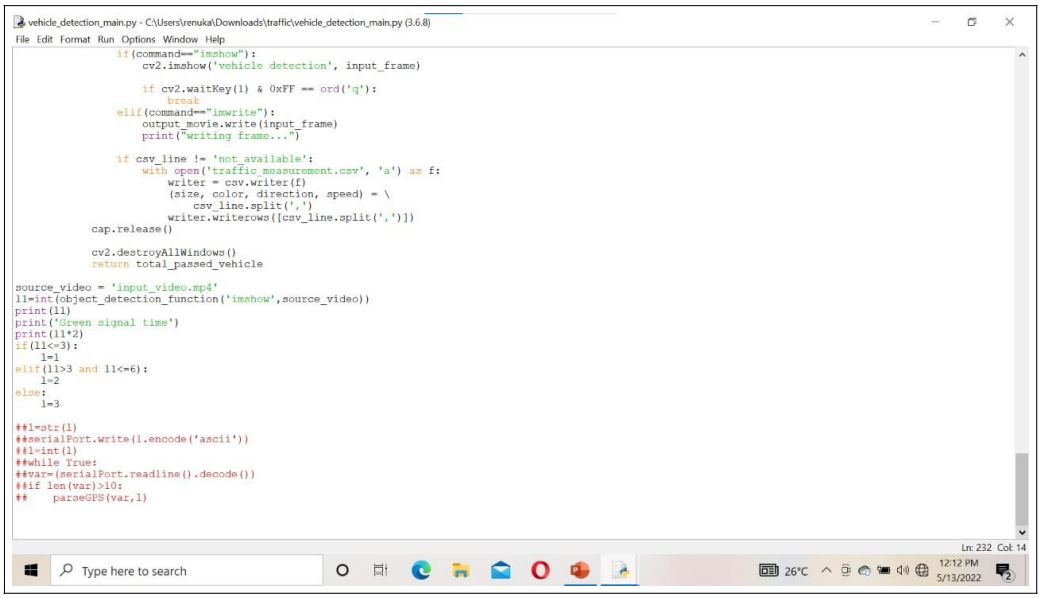
The listed tests were conducted in the software at the various developments stages. Unit testing was conducted. The errors were debugged and regression testing was performed. The integration testing will be performed once the system is integrated with other related systems like Inventory, Budget etc. Once the design stage was over the Black Box and White Box Testing was performed on the entire application. The results were analyzed and the appropriate alterations were made. The test results proved to be positive and henceforth the application is feasible and test approved.

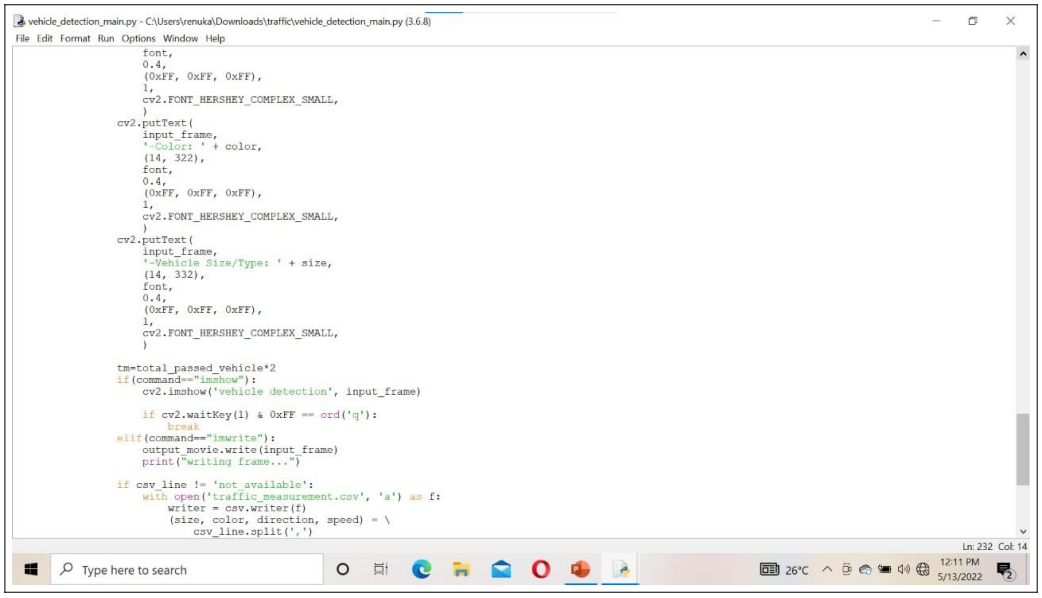
CHAPTER 9

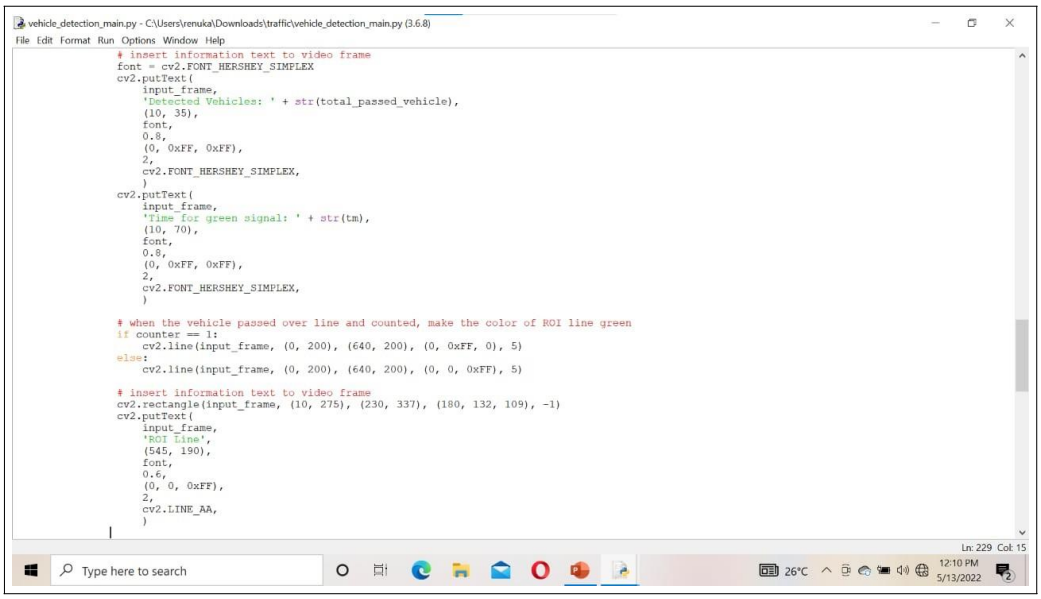
RESULT

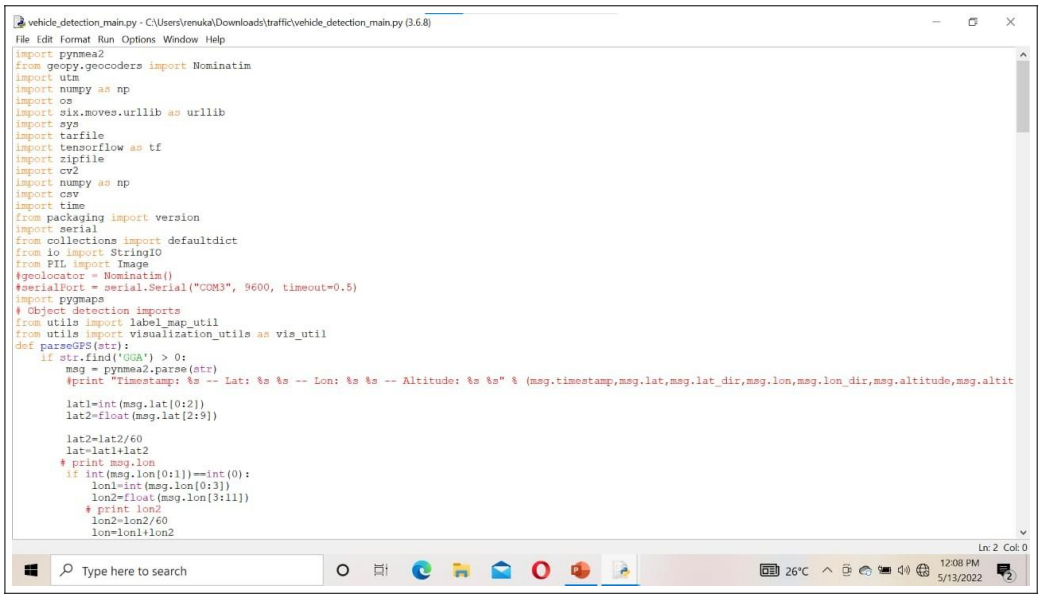
9.1 SCREENSHOTS






vehicle_detection_main.py - C:\Users\renuka\Downloads\traffic\vehicle_detection_main.py (3.6.8)
File Edit Format Run Options Window Help
font,
0.4,
(0xFF, 0xFF, 0xFF),
1,
cv2.FONT_HERSHEY_COMPLEX_SMALL,
)
cv2.putText(
input_frame,
'-Color: ' + color,
(14, 322),
font,
0.4,
(0xFF, 0xFF, 0xFF),
1,
cv2.FONT_HERSHEY_COMPLEX_SMALL,
)
cv2.putText(
input_frame,
'-Vehicle Size/Type: ' + size,
(14, 332),
font,
0.4,
(0xFF, 0xFF, 0xFF),
1,
cv2.FONT_HERSHEY_COMPLEX_SMALL,
)
tm=total_passed_vehicle*2
if(command=="imshow"):
cv2.imshow('vehicle detection', input_frame)
if cv2.waitKey(1) & 0xFF == ord('q'):
break
elif(command=="imwrite"):
output_movie.write(input_frame)
print("writing frame...")
if csv_line != 'not_available':
with open('traffic_measurement.csv', 'a') as f:
writer = csv.writer(f)
(size, color, direction, speed) = \
csv_line.split(',')
Ln: 232 Col: 14
Type here to search
26°C
12:11 PM
5/13/2022


vehicle_detection_main.py - C:\Users\renuka\Downloads\traffic\vehicle_detection_main.py (3.6.8)
File Edit Format Run Options Window Help
# insert information text to video frame
font = cv2.FONT_HERSHEY_SIMPLEX
cv2.putText(
input_frame,
'Detected Vehicles: ' + str(total_passed_vehicle),
(10, 35),
font,
0.8,
(0, 0xFF, 0xFF),
2,
cv2.FONT_HERSHEY_SIMPLEX,
)
cv2.putText(
input_frame,
'Time for green signal: ' + str(tm),
(10, 70),
font,
0.8,
(0, 0xFF, 0xFF),
2,
cv2.FONT_HERSHEY_SIMPLEX,
)
# when the vehicle passed over line and counted, make the color of ROI line green
if counter == 1:
cv2.line(input_frame, (0, 200), (640, 200), (0, 0xFF, 0), 5)
else:
cv2.line(input_frame, (0, 200), (640, 200), (0, 0, 0xFF), 5)
# insert information text to video frame
cv2.rectangle(input_frame, (10, 275), (230, 337), (180, 132, 109), -1)
cv2.putText(
input_frame,
'ROI Line',
(545, 190),
font,
0.6,
(0, 0, 0xFF),
2,
cv2.LINE_AA,
)
Ln: 229 Col: 15
Type here to search
26°C
12:10 PM
5/13/2022


vehicle_detection_main.py - C:\Users\renuka\Downloads\traffic\vehicle_detection_main.py (3.6.8)
File Edit Format Run Options Window Help
import pynmea2
from geopy.geocoders import Nominatim
import utm
import numpy as np
import os
import six.moves.urllib as urllib
import sys
import tarfile
import tensorflow as tf
import zipfile
import cv2
import numpy as np
import csv
import time
from packaging import version
import serial
from collections import defaultdict
from io import StringIO
from PIL import Image
#geolocator = Nominatim()
#serialPort = serial.Serial("COM3", 9600, timeout=0.5)
import pygmaps
# Object detection imports
from utils import label_map_util
from utils import visualization_utils as vis_util
def parseGPS(str):
if str.find('GGA') > 0:
msg = pynmea2.parse(str)
#print "Timestamp: %s -- Lat: %s %s -- Lon: %s %s -- Altitude: %s %s" % (msg.timestamp,msg.lat,msg.lat_dir,msg.lon,msg.lon_dir,msg.altitude,msg.altit
lat1=int(msg.lat[0:2])
lat2=float(msg.lat[2:9])
lat2=lat2/60
lat=lat1+lat2
# print msg.lon
if int(msg.lon[0:1])==int(0):
lon1=int(msg.lon[0:3])
lon2=float(msg.lon[3:11])
# print lon2
lon2=lon2/60
lon=lon1+lon2
Ln: 2 Col: 0
Type here to search
26°C
12:08 PM
5/13/2022

CHAPTER 10

CONCLUSION

Conclusion

The proposed model provides solution to the growing traffic congestion problem and can effectively replace the existing traditional methodologies or traffic control system. Depending on traffic the signal will be adjusted and the accident detection is also proposed where alert is generated if vehicles are going to dash each other . The system can turned out to be very promising while implementing and testing independent components or modules based on the concept and approach we proposed.

CHAPTER 11

REFERENCES

[1] Adil Hilmani , Abderrahim Maizate, And Larbi Hassouni, “Automated Real-time Intelligent Traffic Control System For Smart Cities Using Wireless Sensor Networks”, Wireless Communications And Mobile Computing Volume 2020.

[2] Madhukar, “Adaptive Traffic Signal Control Using Fuzzy Logic”, Ijret — Volume 6, Issue 2 April 2020

[3] Peng Jing, Hao Huang And Long Chen, An Adaptive Traffic Signal Control In A Connected Vehicle Environment: A Systematic Review, School Of Automotive And Traffic Engineering, Jiangsu University, 22 August 2017

[4] Hong K. Lo H. E Chow, “Adaptive Resolution, And Accuracy, March 2002; Accepted: July 2002

[5] Duy Nhat Nguyen, Adaptive Traffic Control System: Design And Simulation, Concordia University, July 2015.