A

Mini Project Report on

REAL TIME VIDEO BASED VEHICLE DETECTION, COUNTING AND CLASSIFICATION SYSTEM

Submitted for partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

by

B. RAJESH	**20K81A0508**
G. SAI KUMAR	**20K81A0520**
R. DEEKSHITH GOUD	**20K81A0549**
A. SOUMITH REDDY	**20K81A0503**

Under the Guidance of

Mr. P. AKHIL

ASSISTANT PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

St. MARTIN'S ENGINEERING COLLEGE

UGC Autonomous
Affiliated to JNTUH, Approved by AICTE,
Accredited by NBA & NAAC A+, ISO 9001:2008 Certified
Dhulapally, Secunderabad - 500 100

NOVEMBER - 2023

St. MARTIN'S ENGINEERING COLLEGE

UGC Autonomous
NBA & NAAC A+ Accredited
Dhulapally, Secunderabad - 500 100
www.smec.ac.in

Certificate

This is to certify that the project entitled **"Real time video based vehicle detection, counting, classification system"** is being submitted By **B.RAJESH(20K81A0508), G.SAI KUMAR (20K81A0520), R.DEEKSHITH GOUD(20K81A0549), A.SOUMITH REDDY(20K81A0503)** in fulfilment of the requirement for the award of degree of **BACHELOR OF TECHNOLOGY in DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING** is recorded of bonafide work carried out by them. The result embodied in this report have been verified and found satisfactory.

Guide
Mr. P.
Associate Professor
Department of CSE

Head of the Department
Dr. R.SANTHOSHKUMAR
Professor & Head
Department of CSE

Internal Examiner **External Examiner**

Date:

Place:

St. MARTIN'S ENGINEERING COLLEGE

UGC Autonomous
NBA & NAAC A+ Accredited
Dhulapally, Secunderabad - 500 100
www.smec.ac.m

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We, the students of **'Bachelor of Technology in Department of Computer Science and Engineering',** session: 2020 - 2024, **St. Martin's Engineering College, Dhulapally, Kompally, Secunderabad,** hereby declare that the work presented in this Project Work entitled **"Real Time Video based Vehicle Detection, Counting and Classification System"** is the outcome of our own bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics. This result embodied in this project report has not been submitted in any university for award of any degree.

GADUSANDULA SAI KUMAR	**20K81A0520**
RANGU DEEKSHITH GOUD	**20K81A0549**
ALMAWAR SOUMITH REDDY	**20K81A0503**
BANDAR RAJESH	**20K81A0508**

CONTENTS

CERTIFICATE ii
DECLARATION iii
ACKNOWLEDGEMENT iv
ABSTRACT vii
LIST OF FIGURES viii

CHAPTER 1 INTRODUCTION 1
1.1 Introduction
1.2 Problem Statement
1.3 Objectives

CHAPTER 2 LITERATURE SURVEY 3

CHAPTER 3 SYSTEM ANALYSIS 5
3.1.1 Existing System
3.1.1.1 Disadvantages System
3.1.2 Proposed System
3.1.2.1 Advantages System
3.2 Project Requirements

CHAPTER 4 SYSTEM DESIGN 14
4.1 Project Architecture
4.2 Detailed Design
4.3 Algorithm
4.4 UML Diagrams
4.4.1 Class Diagram
4.4.2 Use Case Diagram
4.4.3 Sequence Diagram
4.4.4 Activity Diagram
4.4.5 Component Diagram
4.4.6 Deployment Diagram
4.4.7 Package Diagram
4.4.8 Profile Diagram

CHAPTER 5 MODULES 26
5.1 Background Learning Module
5.2 Foreground Extraction Module
5.3 Vehicle Classification Module

CHAPTER 6 SOFTWARE ENVIRONMENT 27
6.1 Software Description

ST MARTIN'S ENGINEERING COLLEGE TECHNOLOGY FOR PROSPERITY UGC AUTONOMOUS

CHAPTER 7 SOURCE CODE 39

CHAPTER 8 EXPERIMENTAL RESULTS 41
7.1 Dataset Description

CHAPTER 9 SYSTEM TESTING 45

CHAPTER 10 CONCLUSION AND FEATURE ENHANCEMENT 49

REFERENCES 50

ST.MARTIN'S ENGINEERING COLLEGE
SMEC
TECHNOLOGY FOR PROSPERITY
UGC AUTONOMOUS

ABSTRACT

Traffic Analysis has been a problem that city planners have dealt with for years. Smarter ways are being developed to analyze traffic and streamline the process. Analysis of traffic may account for the number of vehicles in an area per some arbitrary time period and the class of vehicles. People have designed such mechanism for decades now but most of them involve use of sensors to detect the vehicles i.e. a couple of proximity sensors to calculate the direction of the moving vehicle and to keep the vehicle count. Even though over the time these systems have matured and are highly effective, they are not very budget friendly. The problem is such systems require maintenance and periodic calibration. Therefore, this study has purposed a vision based vehicle counting and classification system. The system involves capturing of frames from the video to perform background subtraction in order detect and count the vehicles using Gaussian Mixture Model (GMM) background subtraction then it classifies the vehicles by comparing the contour areas to the assumed values. The substantial contribution of the work is the comparison of two classification methods. Classification has been implemented using Contour Comparison (CC) as well as Bag of Features (BoF) method.

UGC AUTONOMOUS

LIST OF FIGURES

Figure No.	Figure Title	Page No.
4.1	Project Architecture	14
4.2	Detail Diagram	15
4.3	Linear SVM Algorithm and Non Linear SVM Algorithm	17
4.4.1	Class Diagram	20
4.4.2	Use case Diagram	21
4.4.3	Sequence Diagram	21
4.4.4	Activity Diagram	22
4.4.5	Component Diagram	23
4.4.6	Deployment Diagram	24
4.4.7	Package Diagram	25
4.4.8	Profile Diagram	25
6.1	Python Website	31
6.2	Python version	31
6.3	Python release version	32
6.4	Windows versions	32
6.5	Python Application	33
6.6	Python Installation	34
6.7	Python Setup	34
6.8	Command Prompt	35
6.9	Python Test	35
7.1	Command prompt	42
7.2	Project home page	42
7.3	Project home page	43

ST. MARTIN'S ENGINEERING COLLEGE
SMEC
TECHNOLOGY FOR PROSPERITY
UGC AUTONOMOUS

7.4	Project home page	43
7.5	Accuracy of each vehicle	44

ST.MARTIN'S ENGINEERING COLLEGE
SMEC
TECHNOLOGY FOR PROSPERITY
UGC AUTONOMOUS

CHAPTER - 1

INTRODUCTION

1.1 Introduction

The need of efficient management and monitoring of road traffic has increased in last few decades because of the increase in the road networks, the number and most importantly the size of vehicles. Intelligent traffic surveillance systems are very important part of modern day traffic management but the regular traffic management techniques such as wireless sensor networks[1], Inductive loops[2] and EM microwave detectors[3] are expensive, bulky and are difficult to install without interrupting the traffic. A good alternative to these techniques can be video based surveillance systems. The videos stored by these surveillance systems are generally analyzed by humans, which is a time consuming Job. To overcome this constraint, the need of more robust, automatic video based surveillance systems has increased interest in field of computer vision. The objectives of a traffic surveillance system is to detect, track and classify the vehicles but they can be used to do complex tasks such as driver activity recognition, lane recognition etc. The traffic surveillance systems can have applications in a range of fields such as, public security, detection of anomalous behavior, accident detection, vehicle theft detection, parking areas, and person identification. A Traffic surveillance system usually contains two parts, hardware and software. Hardware is a static camera installed on the roadside that captures the video feed and the software part of the system is concerned with processing and analyses.These systems could be portable with a microcontroller attached to the camera for the real-time processing and analyses or just the cameras that transmit the video feed to a centralized computer for further processing. Various approaches were made to develop such systems that can detect, count and classify the vehicles and can be used for traffic surveillance in intelligent transportation systems. This section covers the discussion about such systems and the knowledge about the methods used to develop such systems. Computer vision technology is using for traffic monitoring in many countries [10], [11]. The development of computer vision technology over video based traffic monitoring for detecting moving vehicles in video streams become an essential part in ITS, [13]. A good number of work has been done on vehicle tracking and detection using computer vision technology. In 2005, Hasegawa and Kanade [14] introduced a system for detecting and classifying the moving objects by its type and colour. In

this process, a series of images of a specific location were supplied and vehicles from these images were identified. In 2013, Nilesh et al. designed and developed a system using python with OpenCV for detecting and counting moving vehicles. It can automatically identify and count moving objects as vehicle in real-time or from recorded videos, which basically used background subtraction, image filtering, image binary and segmentation method. In 2014, Da Li et al. developed real-time moving vehicle detection, tracking, and counting system also using python with OpenCV including adaptive subtracted background method in combination with virtual detector and blob tracking technology. Virtual detector constructs a set of rectangular regions in each input image frame and blobtracking method generates input image frames, the absolute difference between the background image and foreground blobs corresponding to the vehicles on the road. The above systems have some limitations like tackling shadows, occlusion of multiple vehicles that appear in a single region. Peek Traffic Corporation commercially developed several video traffic detection systems at the present time.

1.2 Problem statement

vehicle detection and counting is a challenging task due to many reasons such as: small size of the vehicles, different types and orientations, similarity in visual appearance of vehicles and some other objects (e.g., air conditioning units on the buildings, trash bins, and road marks), and attributes have been extracted by the proposed coupled regional convolution network method which merges an AVPN and a vehicle attribute learning network. Fast and Faster R-CNN have been explored . In order to overcome the limitations in Fast and Faster R-CNN, a new architecture has been proposed. They have improved the detection accuracy of the small-sized objects by using the resolution of the output of the last convolution layer and adapting anchor boxes of RPN as feature map.

1.3 Objectives

The system could be used for detection, recognition and tracking of the vehicles in the video frames and then classify the detected vehicles according to their size in three different classes. The proposed system is based on three modules which are background learning, foreground extraction and vehicle classification as shown in Background subtraction is a classical approach to obtain the foreground image or in other words to detect the moving objects.

CHAPTER – 2

LITERATURE SURVEY

Detection of vehicles in a video based traffic surveillance system is first and very important phase as it greatly impacts the other algorithms such as tracking and classification of the vehicles hence an accurate detection and segmentation of the foreground moving object is very important. Many of the techniques are used for foreground detection like frame differencing [12]. Frame differencing can be considered as the simplest foreground detection and segmentation method as it is based on the close relationship among the sequence of motion images. An improved frame differencing method was presented by Collins [7] which uses difference between multiple frames to compute the foreground instead of just using the initial frame. Another method named as Optical Flow Field method was brought out by Gibson[13]. Wu, K, et al. proposed that the optical flow represents the velocity of mode within an image [14].Optical Flow method takes image within the detecting area as a vector field of velocity; each vector represents the transient variation of the position for a pixel in the scenery. Another method used to detect foreground is average model [8]. In average model the average grey value of a pixel in a sequence of frames is considered as the background value of same pixel. A GMM was proposed by Friedman, N. and S. Russell.[15] and was refined for real time tracking by Stauffer, C. and W.EL Grimson[16, 17]. Gaussian Mixture Model relies on assumptions that the background is visible more frequently than any foreground regions. Elgammal proposed Kernel density estimation based nonparametric background model [18]. Kernel density estimation method evaluates the samples of video data using kernel functions and it samples data which has maximal probability density as background is selected. A bag of features model is the one which represents images as order less collections of local features [19]. The name bag of features came from the bag of words representation used in text based information retrieval. Scale Invariant Feature Transform algorithm (SIFT) was introduced David Lowe in 1999 [20] and refined in 2004 [21]. SIFT is used to extract the features from dataset. In SIFT features are invariant to image scaling, rotation and partially invariant to change in illumination and 3D camera viewpoint [21]. SIFT with SVM requires less number of dataset. With any supervised learning model, first SVM istrained to cross validate the classifier. Then trained machine is used to make predict classify

Bhushan et al. [23], proposed a vehicle detection method based on morphological operations including binarization, edge detection, and top-hat processing, masking operation. Proposed system fails to give good results n cloudy environment. Raul et al. [24], proposed a classification technique for moving objects in which information processing is employed via clustering and classification algorithms. Proposed methodology provides sufficient accuracy for it to be employed for real-time applications but still the system can be further improved for vehicle classification in complex scenarios such as weather and illumination conditions. A. Suryatali and V.B. Dharmadhikari in [25], proposed a Computer Vision based vehicle detection that counts and also classifies the vehicles in to heavy and light weight vehicles; object detection is accomplished by making use of kalman filter for background subtraction and then openCV library is used finally to detect the object in processed frame. Nilakorn et al. [26], proposed vehicle detection and counting prototype which uses different steps for background subtraction and object detection then uses CV techniques such as thresholding, hole-filling and adaptive morphological dilation to remove noise and enhance the foreground objects in particular frame from video. Proposed system provides limited functionality for the objects appearing in detection zone if they are occluded or small.

SIR M. VISVESVARAYA INSTITUTE OF TECHNOLOGY
TECHNOLOGY FOR PROSPERITY
UGC AUTONOMOUS

CHAPTER – 3

SYSTEM ANALYSIS

3.1.1 Existing System

A vehicle detection and classification system using time spatial image and multiple virtual detection line[6]. A two-step K nearest neighborhood (KNN) algorithm is adopted to classify vehicles via shape invariant and texture based features. Experiments confirm the better accuracy and low error rate of proposed method over existing methods since it also considers the various illumination conditions. People have designed such mechanism for decades now but most of them involve use of sensors to detect the vehicles i.e. a couple of proximity sensors to calculate the direction of the moving vehicle and to keep the vehicle count. Even though over the time these systems have matured and are highly effective, they are not very budget friendly. The problem is such systems require maintenance and periodic calibration.

3.1.1.1 Disadvantages System

Detecting vehicle with low accuracy

Existing using Mixture Model relies

3.1.2 Proposed System

The system could be used for detection, recognition and tracking of the vehicles in the video frames and then classify the detected vehicles according to their size in three different classes. The proposed system is based on three modules which are background learning, foreground extraction and vehicle classification as shown in fig. 1. Background subtraction is a classical approach to obtain the foreground image or in other words to detect the moving objects. We have proposed an adaptive video based vehicle detection, classification, counting for real-time traffic data collection. The proposed system was built using python programming language and OpenCV. The main objective for developing this system is to collect vehicle count and classification data. So that we can build intelligent transportation network based on historical traffic data. The proposed system can engender traffic data by detecting, classifying, counting It's a plug & play system and applied YOLO algorithm as a background subtraction technique.

The proposed system was tested at different six locations in Hyderabad under different traffic and environmental conditions.

Proposed System:

Detecting vehicle with High accuracy

Counting and classification vehicles

Existing using YOLO model

3.1.2.1 Advantages System

Detection of multiple moving vehicles in a video sequence
Tracking of the detected vehicles.
Identification of Vehicle types.
Counting the total number of vehicles passing in videos

3.2 Project Requirements

The project involved analyzing the design of few applications so as to make the application more users friendly. To do so, it was really important to keep the navigations from one screen to the other well ordered and at the same time reducing the amount of typing the user needs to do. In order to make the application more accessible, the browser version had to be chosen so that it is compatible with most of the Browsers.

Functional Requirements

- Graphical User interface with the User.

Non Functional Requirements

• **Maintainability:** Maintainability is used to make future maintenance easier, meet new requirements. Our project can support expansion.

• **Robustness:** Robustness is the quality of being able to withstand stress, pressures or changes in procedure or circumstance. Our project also provides it.

• **Reliability:** Reliability is an ability of a person or system to perform and maintain its functions in circumstances. Our project also provides it.

• **Size:** The size of a particular application plays a major role, if the size is less then efficiency will be high. The size of database we have developed is 5.05 MB.

• **Speed:** If the speed is high then it is good. Since the no of lines in our code is less, hence the speed is high.

• **Power Consumption:** In battery-powered systems, power consumption is very important. In the requirement stage, power can be specified in terms of battery life.

However the allowable wattage can't be defined by the customer. Since the no of lines of code is less CPU uses less time to execute hence power usage will be less.

In given database author has given five different types of dataset which describe below

Dataset 1: This dataset can be used to train ML algorithms and this trained model can be used to predict harvest time

Dataset 2: This dataset cab used to train ML algorithms which can be used to predict growth

Dataset 3: This can be used to predict phenology stage.

Dataset 4 and 5 can be used to predict maturity.

Numpy

Python has a strong set of data types and data structures. Yet it wasn't designed for MachineLearning per say. Enter numpy (pronounced as num-pee). Numpy is a data

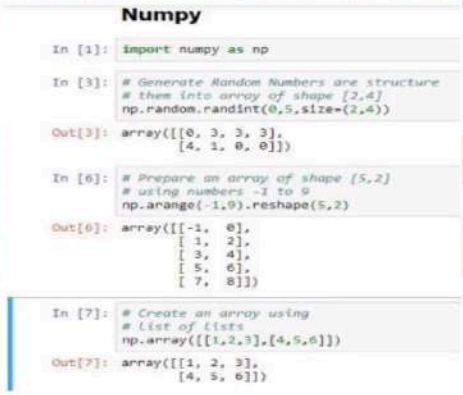


Figure 3.1. Numpy

handling library,particularly one which allows us to handle large multi-dimensional arrays along with a hugecollectionofmathematical operations.Thefollowingis aquick

snippetofnumpyin action.

Numpy isn't just a data handling library known for its capability to handle multidimensional data. It is also known for its speed of execution and vectorization capabilities.It provides MATLAB style functionality and hence requires some learning before you can get comfortable. It is also a core dependency for other majorly used libraries like pandas, matplot lib and so on.It's documentation itself is a good starting point.

Pandas

Think of relational data, think pandas. Yes, pandas is a python library that provides flexibleand expressive data structures (like dataframes and series) for data manipulation. Built on topofnumpy, pandas is as fast andyeteasierto use.

Figure 3.2. Pandas

Pandas provides capabilities to read and write data from different sources like CSVs, Excel,SQL Databases, HDFS and many more. It provides functionality to add, update and deletecolumns, combine or split dataframes/series, handle datetime objects, impute null/missingvalues, handle time series data, conversion to and from numpy objects and so

on. If you areworking on a real-world Machine Learning use case, chances are, you would need pandassooner than later. Similar to numpy, pandas is also an important component of the SciPy orScientificPython Stack .

Scipy

Pronounced as Sigh-Pie, this is one of the most important python libraries of all time. Scipy isa scientific computing library for python. It is also built on top of numpy and is a part of theScipyStack.

Figure 3.3. Scipy

It provides modules/algorithms for linear algebra, integration, image processing, optimizations, clustering, sparse matrix manipulation and many more..

Matplotlib

Another component of the Sci Py stack, matplotlib is essentially a visualization library. It works seamlessly with numpy objects(and its high-level derivatives like pandas).

Matplotlib provides a MATLAB like plotting environment to prepare high-quality figures/charts for publications,notebooks,web applications and so on.

Figure 3.4. Lines, Bars, Markers

Matplolib is a high customizable low-level library that provides a whole lot of controls and knob stop repare any type of visualization/figure. Given its low-level nature, it requires a bit of getting used to along with plenty of code to get stuff done. Its well documented and extensible design has allowed a whole list of high-level visualization libraries to be built on top. Some of which, we will discuss in the coming sections.:

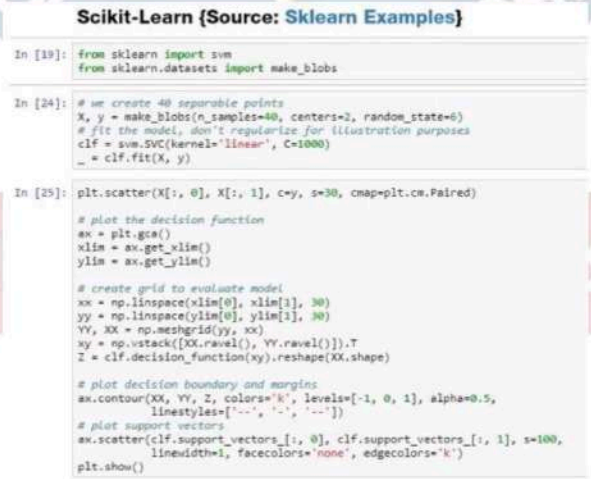


Figure 3.5. Scikit Learn

Scikit-Learn

Designed as an extension to the SciPy library, scikit-learn has become the de-facto standard for many of the machine learning tasks. Develop as part of Google Summer of Code project, it has now become a widely contributed open source project with over

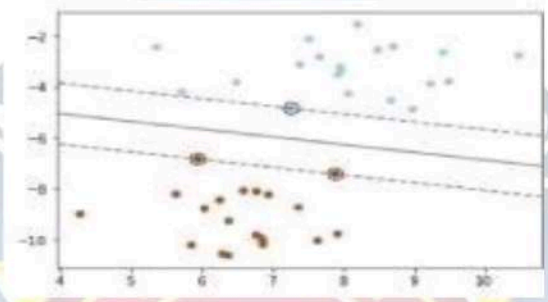


Figure 3.6. Graph

Scikit-learn provides a simple yet powerful fit-transform and predict paradigm to learn from data, transform the data and finally predict. Using this interface, it provides capabilities to prepare classification, regression, clustering and ensemble models. It also provides a multitude of utilities for preprocessing, metrics, model evaluation techniques, etc.

Important features of scikit-learn:

Simple and efficient tools for data mining and data analysis. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means, etc.

Accessible to everybody and reusable in various contexts.

Built on the top of NumPy, SciPy, and matplotlib.

Open source, commercially usable–BSD license.

Scikit-learn provides a simple yet powerful fit-transform and predict paradigm to learn from data, transform the data and finally predict. Using this interface, it provides capabilities to prepare classification, regression, clustering and ensemble models. It also provides a multitude of utilities for preprocessing, metrics, model evaluation techniques, etc.

Visualization

Seaborn

Built on top of matplotlib, seaborn is a high level visualization library. It provides so phisticated styles straight out of the box (which would take some good amount of effort if done using matplotlib).

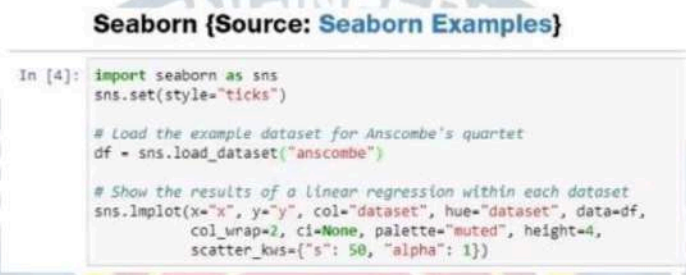


Figure 3.7. Seaborn

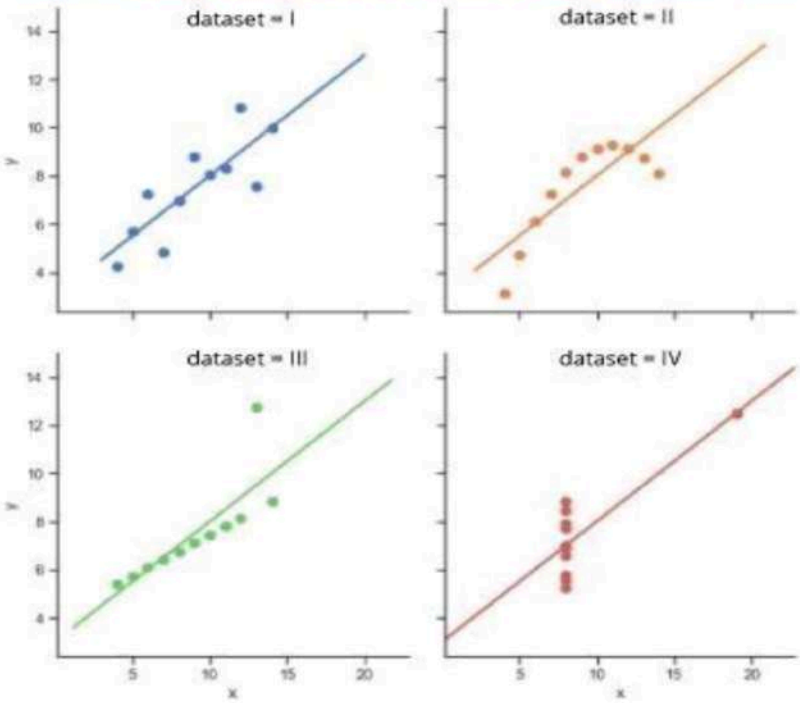


Figure 3.8. Datasets

Hardware Requirements

- **System** : Pentium IV 2.4 GHz.
- **Hard Disk** : 40 GB.
- **Floppy Drive** : 1.44 Mb.
- **Monitor** : 14’ Colour Monitor.
- **Mouse** : Optical Mouse.
- **Ram** : 512 Mb.

Software Requirements

- **Operating system** : Windows 7 Ultimate.
- **Coding Language** : Python.
- **Front-End** : Python.

ST.MARTIN'S ENGINEERING COLLEGE
TECHNOLOGY FOR PROSPERITY
SMEC
UGC AUTONOMOUS

CHAPTER – 4

SYSTEM DESIGN

4.1 Project Architecture

The system could be used for detection, recognition and tracking of the vehicles in the video frames and then classify the detected vehicles according to their size in three different classes. The proposed system is based on three modules which are background learning, foreground extraction and vehicle classification as shown in fig. 1. Background subtraction is a classical approach to obtain the foreground image or in other words to detect the moving objects.

Background Learning Module This is the first module in the system whose main purpose is to learn about the background in a sense that how it is different from the foreground. Furthermore as proposed system works on a video feed, this module extracts the frames from it and learns about the background. In a traffic scene captured with a static camera installed on the road side, the moving objects can be considered as the foreground and static objects as the background. Image processing algorithms are used to learn about the background using the above mentioned.

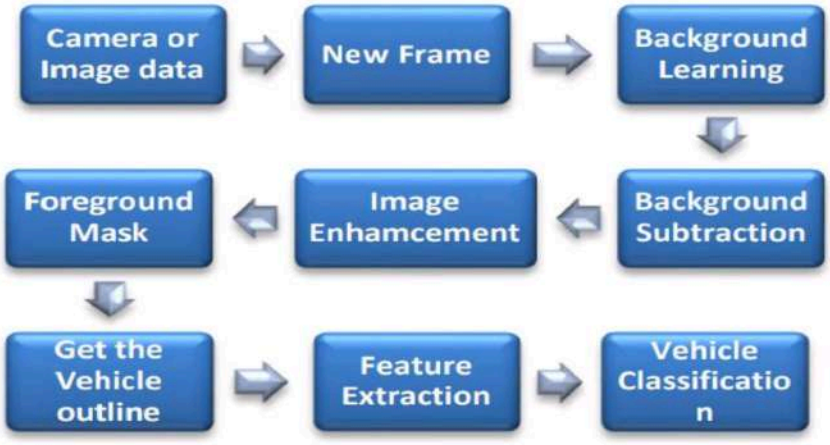


Figure 4.1 Project Architecture

Foreground Extraction Module This module consists of three steps, background subtraction, image enhancement and foreground extraction. Background is subtracted so that foreground objects are visible. This is done usually by static pixels of static objects to binary 0. After background subtraction image enhancement techniques such as noise filtering, dilation and

erosion are used to get proper contours of the foreground objects. The final result obtained from this module is the foreground.

Vehicle Classification Module The third and the last module in the proposed system is classification. After applying foreground extraction module, proper contours are acquired. Features of these contours such as centroid, aspect ratio, area, size and solidity are extracted and are used for the classification of the vehicles.

4.2 Detail Diagram

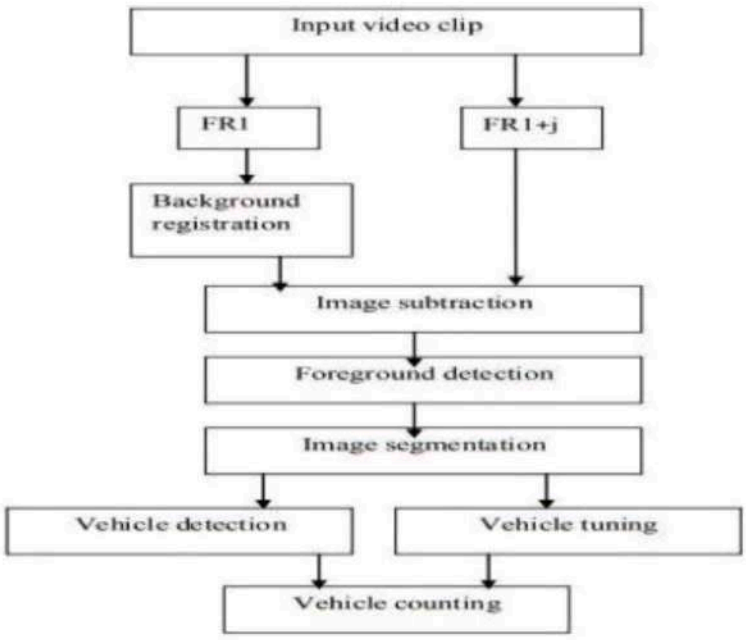


Figure 4.2 Detail Diagram

4.3 Algorithm

SVM Algorithm

Support Vector Machine Algorithm

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that

we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

A simple linear SVM classifier works by making a straight line between two classes. That means all of the data points on one side of the line will represent a category and the data points on the other side of the line will be put into a different category. This means there can be an infinite number of lines to choose from.

What makes the linear SVM algorithm better than some of the other algorithms, like k-nearest neighbors, is that it chooses the best line to classify your data points. It chooses the line that separates the data and is the furthest away from the closet data points as possible. A 2-D example helps to make sense of all the machine learning jargon. Basically you have some data points on a grid. You're trying to separate these data points by the category they should fit in, but you don't want to have any data in the wrong category. That means you're trying to find the line between the two closest points that keeps the other data points separated. So the two closest data points give you the support vectors you'll use to find that line. That line is called the decision boundary.

Example: SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat. Consider the below diagram:

Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm.

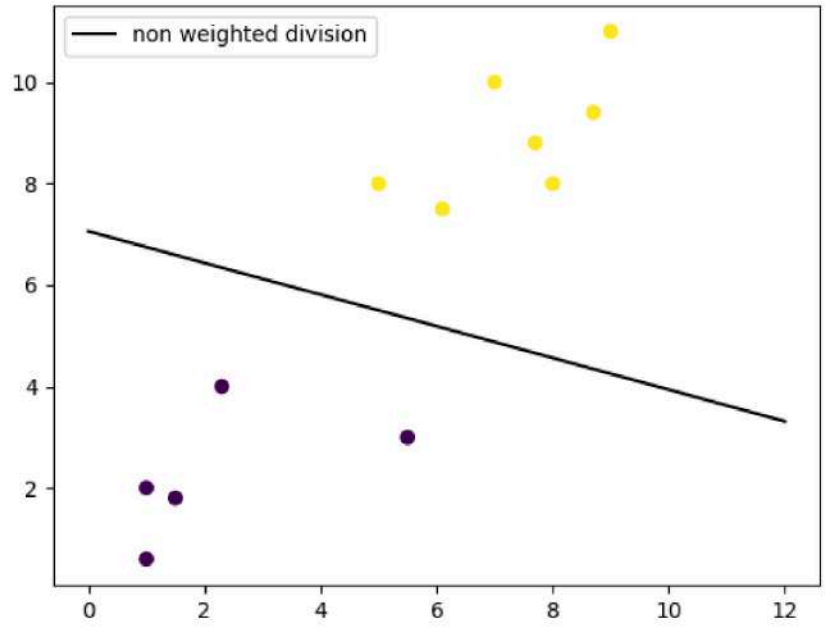


Figure 4.3 Linear SVM

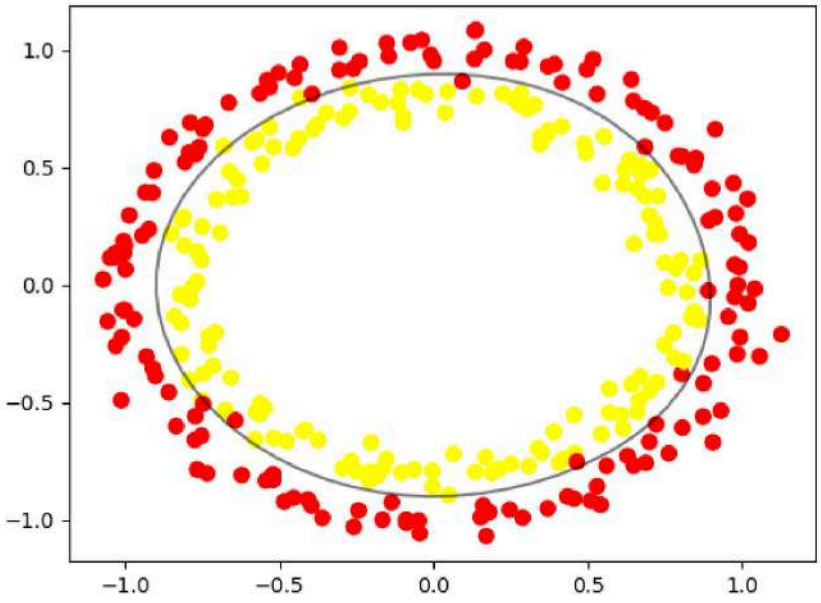


Figure 4.3 Non Linear SVM

Implementation:

In the very first frame of the video, I define a ROI by drawing a close line on the image. The goal is to recognize that ROI in a later frame, but that ROI is not a salient vehicle. It is just a part of an vehicle, and it can deform, rotate, translate and even not be fully in the frame.

Active strategy to choose a search window for vehicle detection using an image context was proposed a deep CNN framework (AttentionNet) to capture the vehicle by sequential actions with top-down attention. AttentionNet has achieved satisfactory performance on vehicle detection benchmark, by sequentially refining the bounding boxes. Proposed a sequential search strategy to detect visual vehicles in images, where the detection model was trained by proposed a deep RL framework to select a proper action to capture an vehicle in an image.

One of the compelling features of our network is its simplicity: the classifier is simply replaced by a mask generation layer without any smoothness prior or convolution structure. However, it needs to be trained with a huge amount of training data: vehicles of different sizes need to occur at almost every location.

Visual tracking solves the problem of finding the position of the target in a new frame from the current position. The proposed tracker dynamically pursues the target by sequential actions controlled by the ADNets. The ADNet predicts the action to chase the target moving from the position in the previous frame. The bounding box is moved by the predicted action from the previous position, and then, the next action is sequentially predicted from the moved position. By repeating this process over the test sequence, we solve the vehicle tracking problem. The ADNet is pre-trained by SL as well as RL. During actual tracking, online adaptation is conducted.

In this module detected vehicles will be counted and these counted results will be updated frequents based on vehicle detection, results will be printed streaming video using opencv.

Proper background subtraction is a vital pre-processing step in creation of any visual surveillance system as the accuracy of whole process of classification of the objects depends on it. The systems like visitor counter [27] in which a static camera captures the video of people entering a building and the system could count the number or a system where a camera captures the video of the vehicles on the road for the similar purpose. The background subtraction could be an easy job if we already have an image of the background like the image of the building or the road. In cases defined above, background image could be removed and foreground objects

could be obtained but most of the time the situation is varying. The backgrounds can be dynamic or initial information of the scene might not be available. Furthermore, the background subtraction becomes more difficult if the objects in the video also have shadows since they also move with the people or vehicles, then the normal background subtraction will detect the shadows as foreground objects too.

4.4 UML Diagrams

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.

4.4.1 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

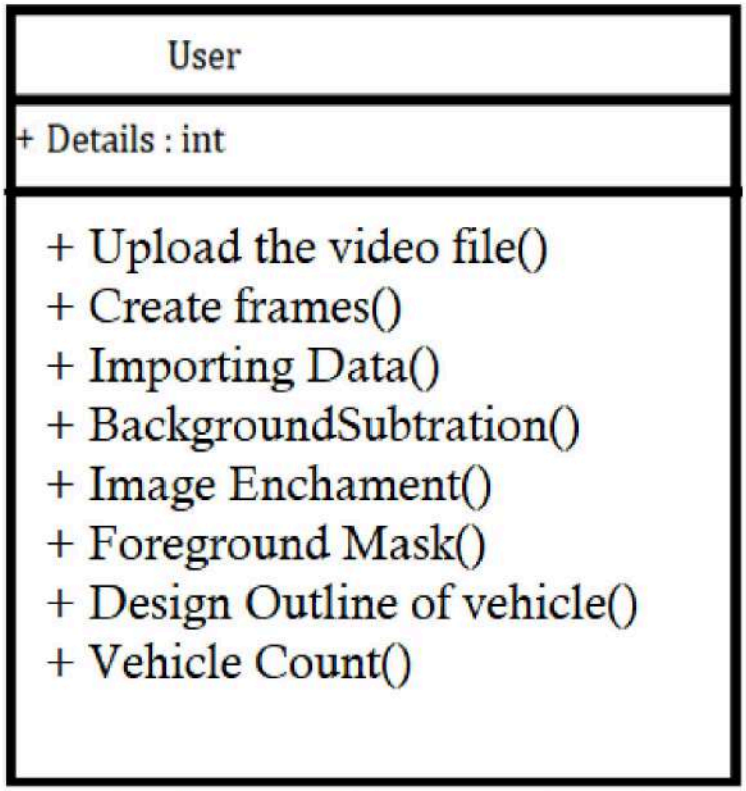


Figure 4.4 Class Diagram

UGC AUTONOMOUS

4.4.2 USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

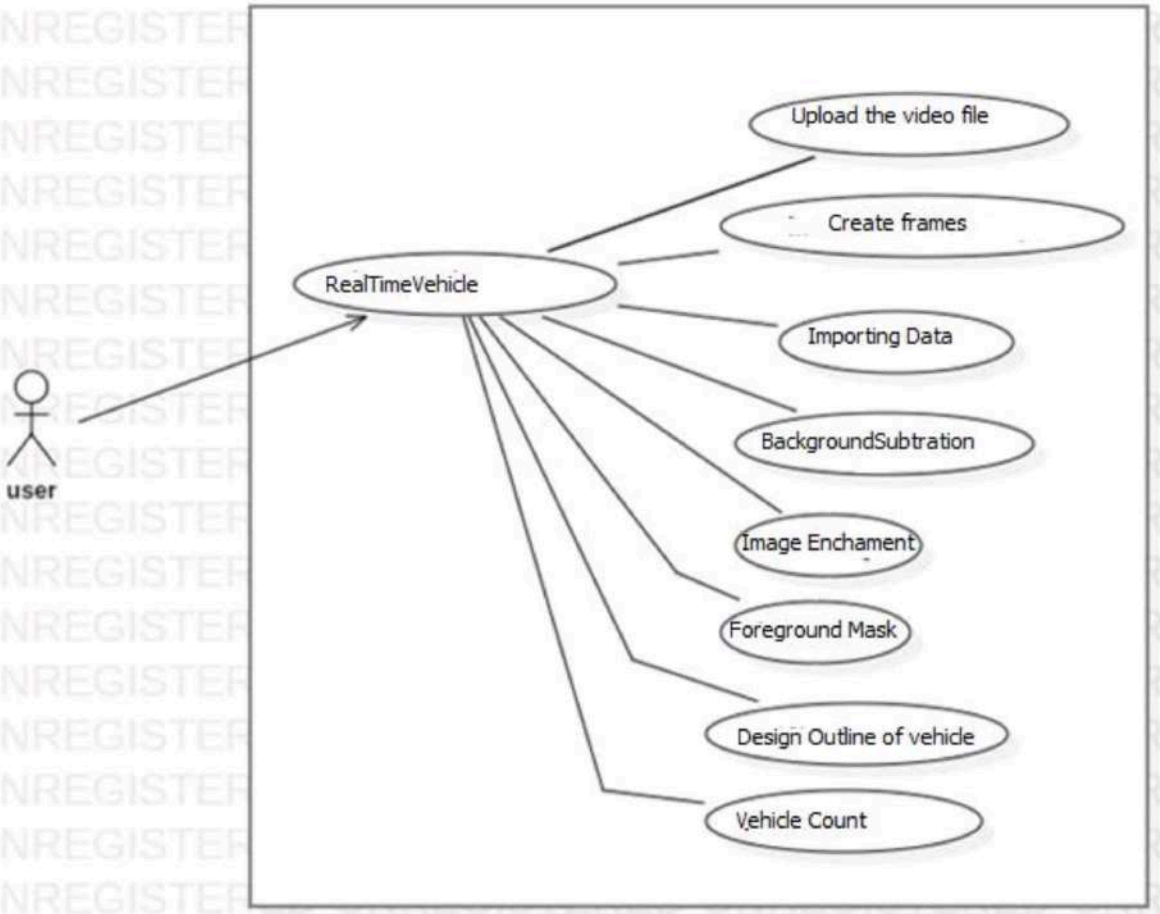


Figure 4.5 Use Case Diagram

4.4.3 SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

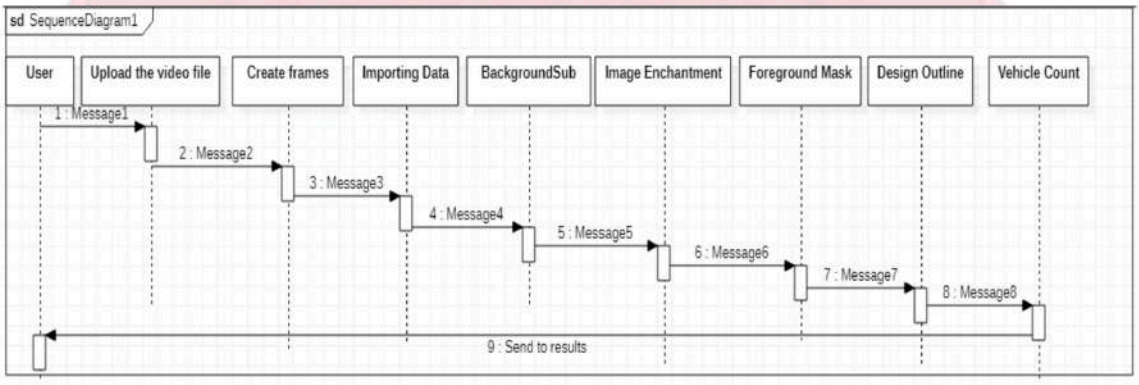


Figure 4.6 Sequence Diagram

4.4.4 Activity diagrams

Activity diagrams are graphical representations of workflows of stepwise activities and actions[1] with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities.[2][3] Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.[citation needed]Activity diagrams are graphical representations of workflows of stepwise activities and actions[1] with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities.[2][3] Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.[citation needed]

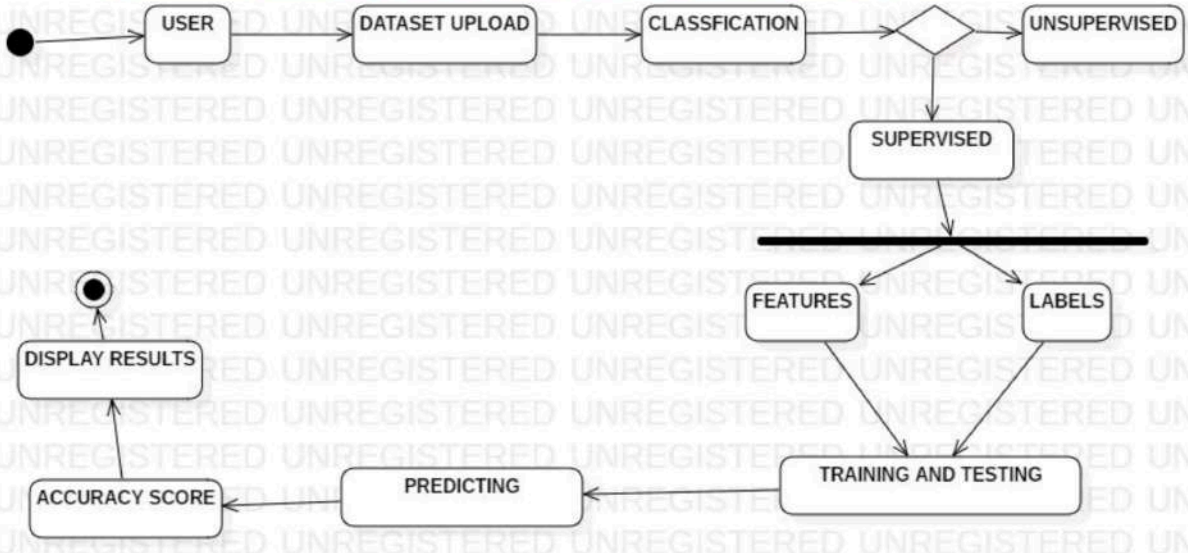


Figure 4.7 Activity Diagram

4.4.5 Component Diagram

The component diagram extends the information given in a component notation element. One way of illustrating the provided and required interfaces by the specified component is

in the form of a rectangular compartment attached to the component element.[2] Another accepted way of presenting the interfaces is to use the ball-and-socket graphic convention. A provided dependency from a component to an interface is illustrated with a solid line to the component using the interface from a "lollipop", or ball, labelled with the name of the interface. A required usage dependency from a component to an interface is illustrated by a half-circle, or socket, labelled with the name of the interface, attached by a solid line to the component that requires this interface. Inherited interfaces may be shown with a lollipop, preceding the name label with a caret symbol. To illustrate dependencies between the two, use a solid line with a plain arrowhead joining the socket to the lollipop.[3]

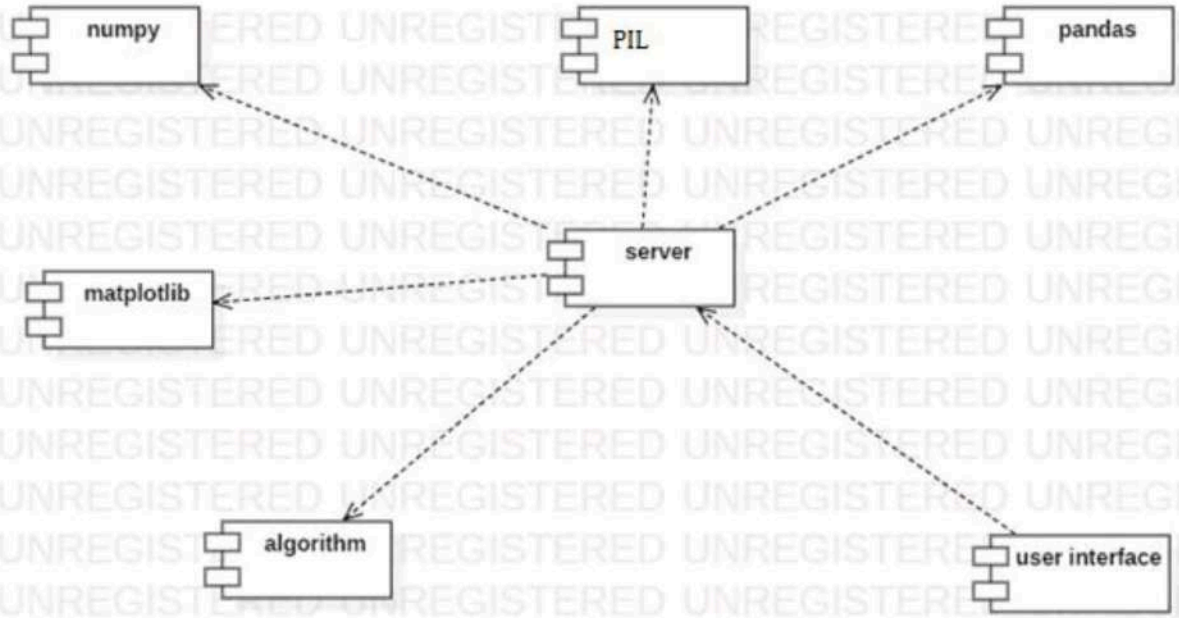


Figure 4.8 Component Diagram

UGC AUTONOMOUS

4.4.6 Deployment diagram

A deployment diagram in the Unified Modeling Language models the physical deployment of artifacts on nodes.[1] To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have subnodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

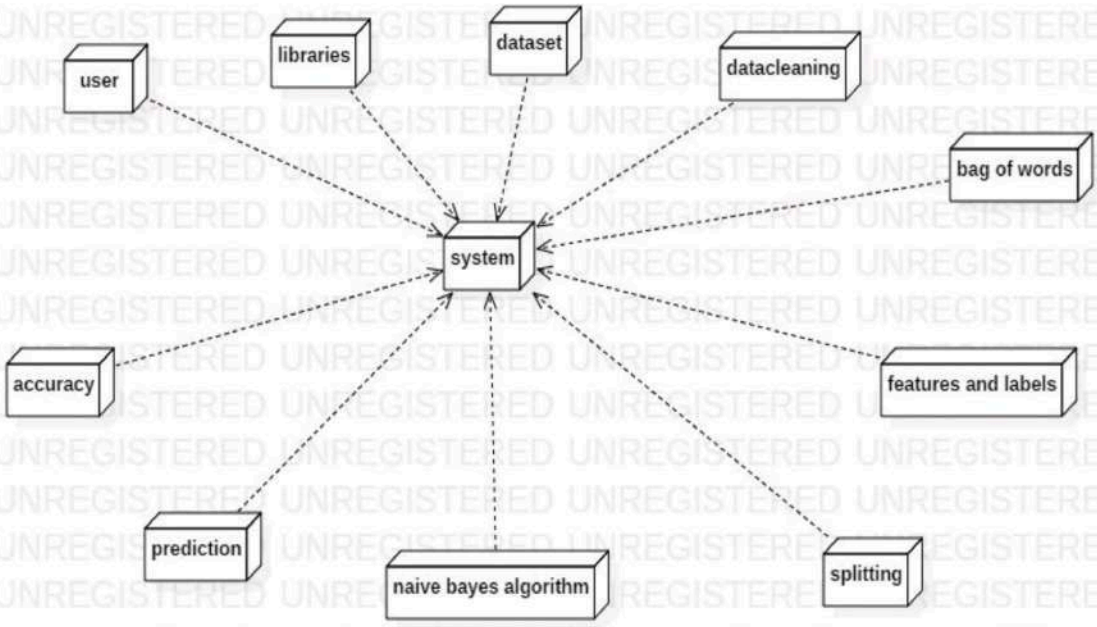


Figure 4.9 Deployment Diagram

4.4.7 Package Diagram

Package diagram is UML structure diagram which shows structure of the designed system at the level of packages. The following elements are typically drawn in a package diagram: package, packageable element, dependency, element import, package import, package merge.

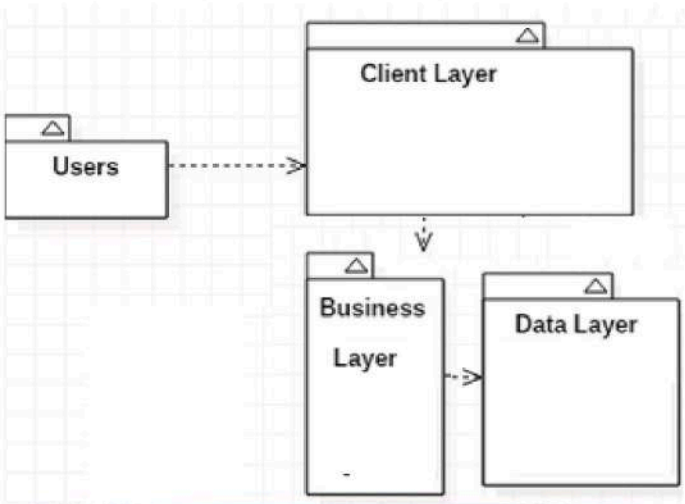


Figure 4.10 Package Diagram

4.4.8 Profile Diagram

A Profile diagram is any diagram created in a «profile» Package. Profiles provide a means of extending the UML. They are based on additional stereotypes and Tagged Values that are applied to UML elements, connectors and their components.

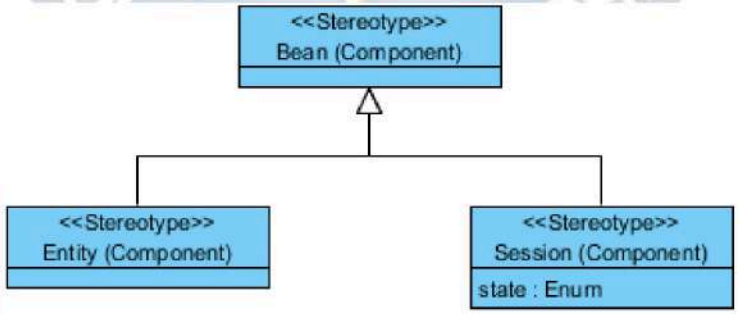


Figure 4.11 Profile Diagram

CHAPTER – 5

MODULES

Modules Description:

5.1 Background Learning Module

This is the first module in the system whose main purpose is to learn about the background in a sense that how it is different from the foreground. Furthermore as proposed system works on a video feed, this module extracts the frames from it and learns about the background. In a traffic scene captured with a static camera installed on the road side, the moving objects can be considered as the foreground and static objects as the background. Image processing algorithms are used to learn about the background using the above mentioned technique.

5.2 Foreground Extraction Module

This module consists of three steps, background subtraction, image enhancement and foreground extraction. Background is subtracted so that foreground objects are visible. This is done usually by static pixels of static objects to binary 0. After background subtraction image enhancement techniques such as noise filtering, dilation and erosion are used to get proper contours of the foreground objects. The final result obtained from this module is the foreground

5.3 Vehicle Classification Module

The third and the last module in the proposed system is classification. After applying foreground extraction module, proper contours are acquired. Features of these contours such as centroid, aspect ratio, area, size and solidity are extracted and are used for the classification of the vehicles.

UGC AUTONOMOUS

CHAPTER – 6

SOFTWARE ENVIRONMENT

6.1 Software Description

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++or Java. It provides constructs that enable clear programming on both small and large scales. 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, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation. 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.What is Python ?

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991. It is used for web development (server-side), software development, mathematics, system scripting.

What can Python do

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

Why Python

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).

- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-orientated way or a functional way.

The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.

In this tutorial Python will be written in a text editor. It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.Python Syntax compared to other programming languages

Python was designed for readability, and has some similarities to the English language with influence from mathematics.

Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.

Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

Introduction

Python applications will often use packages and modules that don't come as part of the standard library. Applications will sometimes need a specific version of a library, because the application may require that a particular bug has been fixed or the application may be written using an obsolete version of the library's interface.

This means it may not be possible for one Python installation to meet the requirements of every application. If application A needs version 1.0 of a particular module but application B needs version 2.0, then the requirements are in conflict and installing either version 1.0 or 2.0 will leave one application unable to run.

The solution for this problem is to create a virtual environment, a self-contained directory tree that contains a Python installation for a particular version of Python, plus a number of additional packages.

Different applications can then use different virtual environments. To resolve the earlier example of conflicting requirements, application A can have its own virtual environment with version 1.0 installed while application B has another virtual environment with version 2.0. If application B requires a library be upgraded to version 3.0, this will not affect application A's environment.Python is the most widely used multi-purpose, high-level programming language at the moment. Python supports both Object-Oriented and Procedural programming paradigms. Python programmes are typically smaller than those written in other programming languages such as Java. Programmers must type relatively little, and the language's indentation requirement ensures that their code is always readable.Python is used by almost all tech giants, including Google, Amazon, Facebook, Instagram, Dropbox, Uber, and others.Python's greatest strength is its vast collection of standard libraries, which can be used for the following.

- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like OpenCV, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks

History of Python : -

What are the similarities between Python and the alphabet? Yes, both begin with ABC. It is abundantly clear that the programming language ABC is being referred to when we talk about ABC in the context of Python. ABC is a broadly useful programming language and programming climate, which had been created in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde and Informatica). Influencing the development of Python was ABC's greatest accomplishment. Python was first thought of at the end of the 1980s. During that time, Guido van Rossum worked on a distributed operating system called Amoeba at the CWI. Guido van Rossum stated in an interview with Bill Venners 1: At Centrum voor Wiskunde en Informatica (CWI), I worked as an implementer on a team developing a language called ABC in the early 1980s. I don't have any idea how well individuals know ABC's effect on Python. I

attempt to make reference to ABC's impact since I'm obliged to all that I mastered during that undertaking and to individuals who chipped away at it." Guido van Rossum went on later in the same interview: " I recalled all my experience and a portion of my dissatisfaction with ABC.I decided to try to create a straightforward scripting language with some of ABC's best features without its drawbacks. So I began composing. I made a straightforward virtual machine, a basic parser, and a basic runtime. I adapted the various ABC parts I liked into my own creation. I developed a basic syntax, substituted indentation for curly braces or begin-end blocks for statement grouping, and a small number of powerful data types: a hash table (or word reference, as we call it), a rundown, strings, and numbers."

How to Install Python on Windows and Mac :

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your System Requirements. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a Windows 64-bit operating system. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheatsheet here.The steps on how to install Python on Windows 10, 8 and 7 are divided into 4 parts to help understand better.

Download the Correct version into the system

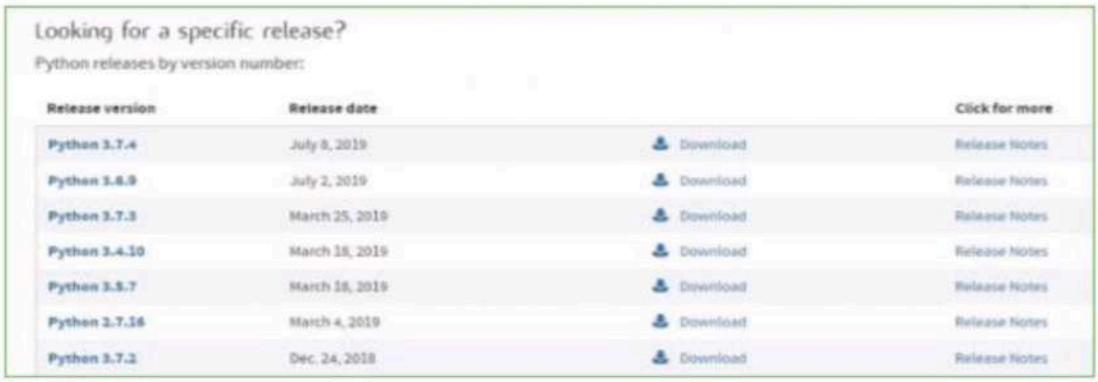
Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: https://www.python.org

Now, check for the latest and the correct version for your operating system.

Step 2: Click on the Download Tab.

Figure 6.1 Official Site

Figure 6.2 Python Version



Looking for a specific release?

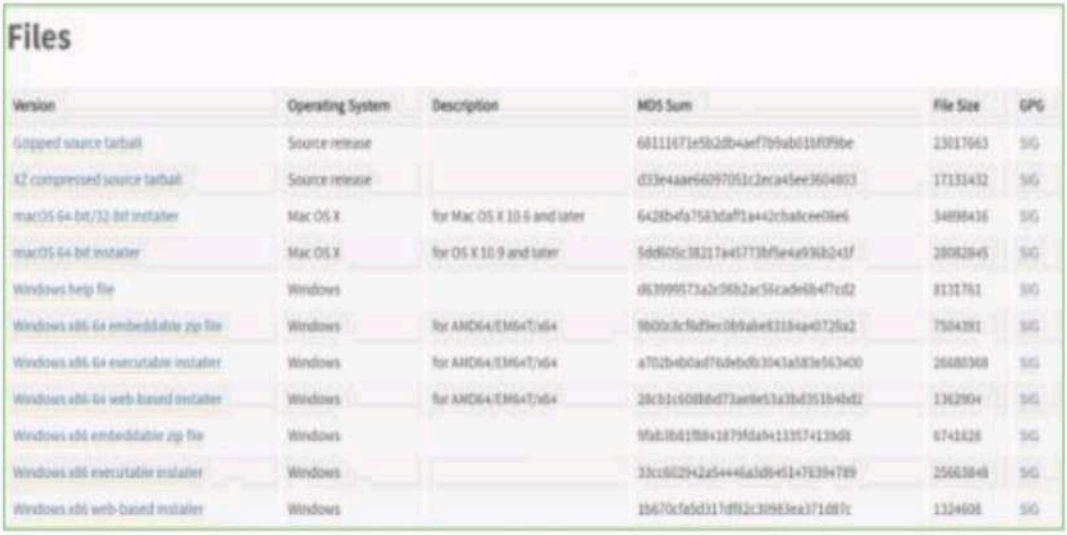
Python releases by version number:

Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	Download	Release Notes
Python 3.6.9	July 2, 2019	Download	Release Notes
Python 3.7.3	March 25, 2019	Download	Release Notes
Python 3.4.10	March 18, 2019	Download	Release Notes
Python 3.5.7	March 18, 2019	Download	Release Notes
Python 2.7.16	March 4, 2019	Download	Release Notes
Python 3.7.2	Dec. 24, 2018	Download	Release Notes

Figure 6.3. Release Versions

Step 3: Scroll down the page until you find the Files option.

Step 4: Here you see a different version of python along with the operating system.



Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		68111671e5b2db4aef7b9ab018f0f9be	23017663	SIG
XZ compressed source tarball	Source release		d33e4aae66097051c2eca45ee3604803	17131432	SIG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	6428b4fa7583daff1a442cba8cee08e6	34898416	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5dd605c38217a457738f5e4a936b243f	28082845	SIG
Windows help file	Windows		d639995732a2c98b2ac56cade68b4f7cd2	8131761	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9b00c8cf6d9ec0bf4abe83184a072fa2	7584391	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b40dad76deb0b3041a583e563400	26680368	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28c31c6088b6d73ae8e53a3bd351b4bd2	1362904	SIG
Windows x86 embeddable zip file	Windows		9fab3b819884879f4a941335741390f8	6741628	SIG
Windows x86 executable installer	Windows		33cc6029426a5444a1a0b451e7639e789	25663848	SIG
Windows x86 web-based installer	Windows		1b670cfa5d317df82c30983ea371d87c	1324608	SIG

Figure 6.4 windows versions

• To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.

To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.

Step 2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.

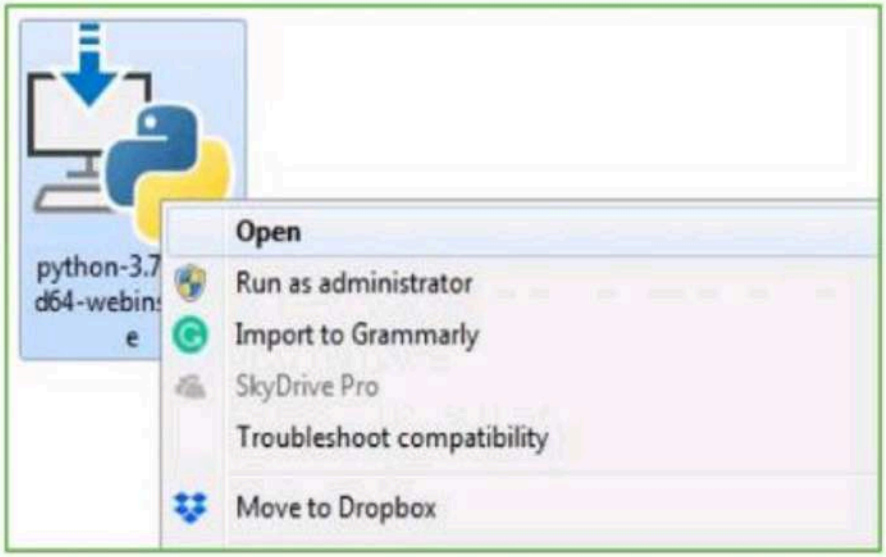


Figure 6.5 Python Application

Figure 6.6 Python Installation

Step 3: Click on Install NOW After the installation is successful. Click on Close.

Figure 6.7 Python setup

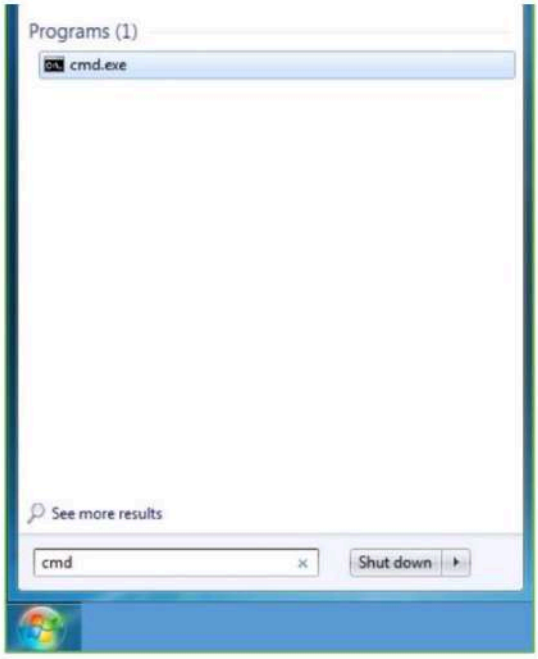


Figure 6.8 Command prompt

Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type python –V and press Enter.

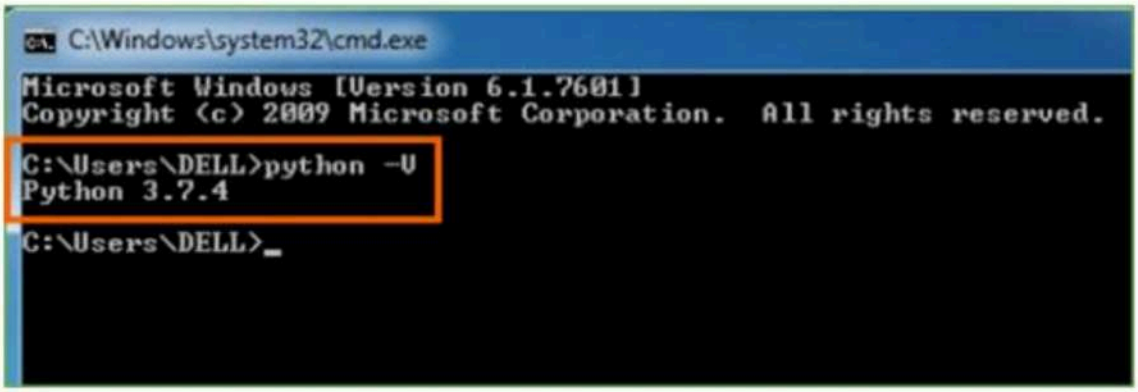


Figure 6.9 Python test

Step 5: You will get the answer as 3.7.4

You will see that the command given is launched. With this, we end our tutorial on how to install Python. You have learned how to download python for windows into your respective operating system.

Note: Unlike Java, Python doesn't need semicolons at the end of the statements otherwise it won't work.

MACHINE LEARNING : -

Before we investigate the subtleties of different AI techniques, how about we start by seeing what AI is, and what it isn't. Although machine learning is frequently referred to as a subfield of artificial intelligence, I believe that categorization can frequently be initially misleading. The investigation of AI positively emerged from research in this specific circumstance, however in the information science use of AI techniques, it's more useful to consider AI for the purpose of building models of information.

In its most basic form, machine learning is the creation of mathematical models that aid in data comprehension. Learning" enters the conflict when we give these models tunable boundaries that can be adjusted to noticed information; The program can thus be regarded as "learning" from the data. These models can be used to predict and comprehend aspects of newly observed data once they have been fitted to data that has already been observed. The more philosophical discussion regarding the degree to which this kind of mathematical, model-based "learning" is comparable to the "learning" exhibited by the human brain will be left up to the reader. Understanding the issue setting in AI is crucial for utilizing these devices successfully, thus we will begin for certain general classifications of the kinds of approaches we'll examine here.

Classes Of Machine Inclining :-

Machine learning can be broken down into two main categories at the most fundamental level: learning under supervision and unsupervised learning.

Modeling the relationship between the data's measured features and some label is one aspect of supervised learning. New, unidentified data can be labeled with the help of this model once it has been established. Regression and classification tasks are two more subcategories of this: The labels in classification are discrete categories, whereas the labels in regression are continuous quantities. In the following section, we will see examples of both kinds of supervised learning.

Unsupervised learning is often referred to as "letting the dataset speak for itself," and it involves modeling the features of a dataset without using any labels. These models incorporate undertakings like bunching and dimensionality decrease. Bunching calculations distinguish

unmistakable gatherings of information, while dimensionality decrease calculations look for additional brief portrayals of the information. In the following section, we will see examples of both kinds of unsupervised learning.

Need for AI

Individuals, as of now, are the most keen and high level species on earth since they can think, assess and tackle complex issues. On the opposite side, computer based intelligence is still in its underlying stage and haven't outperformed human knowledge in numerous viewpoints. Then the inquiry that is the need to make machine learn? The most appropriate justification behind doing this is, "to decide, in view of information, with proficiency and scale".6.2 Packages and Versions

astunparse ==1.6.3

certifi ==2022.12.7

charset-normalizer ==3.0.1

contourpy ==1.0.7

cycler ==0.11.0

Django ==3.0.4

et-xmlfile ==1.1.0

fonttools ==4.38.0

gast ==0.3.3

google-pasta ==0.2.0

grpcio ==1.51.1

h5py ==2.10.0

idna ==3.4

importlib-metadata ==6.0.0

joblib ==1.2.0

ST.MARTIN'S ENGINEERING COLLEGE SMEC TECHNOLOGY FOR PROSPERITY UGC AUTONOMOUS

Keras ==2.3.1

Keras-Applications ==1.0.8

Keras-Preprocessing ==1.1.2

kiwisolver ==1.4.4

Markdown ==3.4.1

matplotlib ==3.6.3

mysql-connector-python ==8.0.32

mysqlclient ==2.1.1

numpy ==1.21.4

oauthlib ==3.2.2

openpyxl ==3.1.0

opt-einsum ==3.3.0

packaging ==23.0

pandas ==1.3.5

Pillow ==9.4.0

pip ==20.1.1

protobuf ==3.20.3

pyparsing ==3.0.9

ST.MARTIN'S ENGINEERING COLLEGE
SMEC
TECHNOLOGY FOR PROSPERITY
UGC AUTONOMOUS

CHAPTER - 7

SOURCE CODE

```
import tkinter as tk
from tkinter import filedialog
from tkinter import *
import os
import subprocess
import numpy
 #initialise GUI
top=tk.Tk()
top.geometry('1200x720')
top.title('RealTime Vechicle Project')
bg = PhotoImage(file = "a.png")
canvas1 = Canvas( top, width = 800, height = 800)
canvas1.pack(fill = "both", expand = True)
# Display image
canvas1.create_image( 0, 0, image = bg, anchor = "nw")
# top.configure(background= bg)
label=Label(top,background='#CDCDCD', font=('arial',15,'bold'))
sign_image = Label(top)
def classify(file_path):
    # print(file_path)
    Str="Main1.py --input "+file_path+" --output outputVideos/bridgeOut.avi --yolo yolo-"
    # harActivity = "Main1.py --input inputVideos/bridge.mp4 --output outputVideos/bridgeOut.avi --
yolo yolo-"
    subprocess.call("python "+Str)
def show_classify_button(file_path):
```

```
    classify_b=Button(top,text="Get RealTime Vechicle Reading",command=lambda:
classify(file_path),padx=10,pady=5)

    classify_b.configure(background='#364156', foreground='white',font=('arial',10,'bold'))

    classify_b.place(relx=0.79,rely=0.46)

    button2_canvas = canvas1.create_window( 800, 300, anchor = "nw", window = classify_b)

def upload_video():

    try:

        file_path=filedialog.askopenfilename()

        label.configure(text='')

        show_classify_button(file_path)

    except:

        pass

upload=Button(top,text="Upload A Video",command=upload_video,padx=10,pady=5)

upload.configure(background='#364156', foreground='white',font=('arial',10,'bold'))

upload.pack(side=BOTTOM,pady=50)

button1_canvas = canvas1.create_window( 600, 500, anchor = "nw", window = upload)

sign_image.pack(side=BOTTOM,expand=True)

label.pack(side=BOTTOM,expand=True)

heading = Label(top, text="RealTime Vechicle Project",pady=20, font=('arial',20,'bold'))

heading.configure(background='#CDCDCD',foreground='#364156')

heading.pack()

button2_canvas = canvas1.create_window( 500, 100, anchor = "nw", window = heading)

top.mainloop()
```

CHAPTER – 8

EXPERIMENTAL RESULTS

7.1 Dataset Description

In this project author is divided the project into two parts and in first part he gave brief literature on technologies which can be used to improve vineyard growth and in second part he describe 'real time traffic Database' which can be used to train various machine learning algorithms such as SVM. Once we trained model on ML algorithms then that trained model can be used to predict grape growth, harvest time and phenology (development cycle) type on new test images.

In given database author has given five different types of dataset which describe below

Dataset 1: This dataset can be used to train ML algorithms and this trained model can be used to predict harvest time

Dataset 2: This dataset cab used to train ML algorithms which can be used to predict growth

Dataset 3: This can be used to predict phenology stage.

Dataset 4 and 5 can be used to predict maturity.

Here in this project we are using first dataset to real time traffic. we are skipping as it's taking too time for execution due to huge images and for same reason we have implemented only SVM algorithm.

You can see all images inside 'real time traffic Database' folder and this folder contains 3 different datasets for harvesting images, growth rate and phenology type. Below screen shots showing dataset images

Screen Shots

Dataset Description and Screen shots with Matter

You can see all images inside 'real time traffic Database' folder and this folder contains 3 different datasets for harvesting images, growth rate and phenology type. Below screen shots showing dataset images

ST.MARTIN'S ENGINEERING COLLEGE TECHNOLOGY FOR PROSPERITY UGC AUTONOMOUS

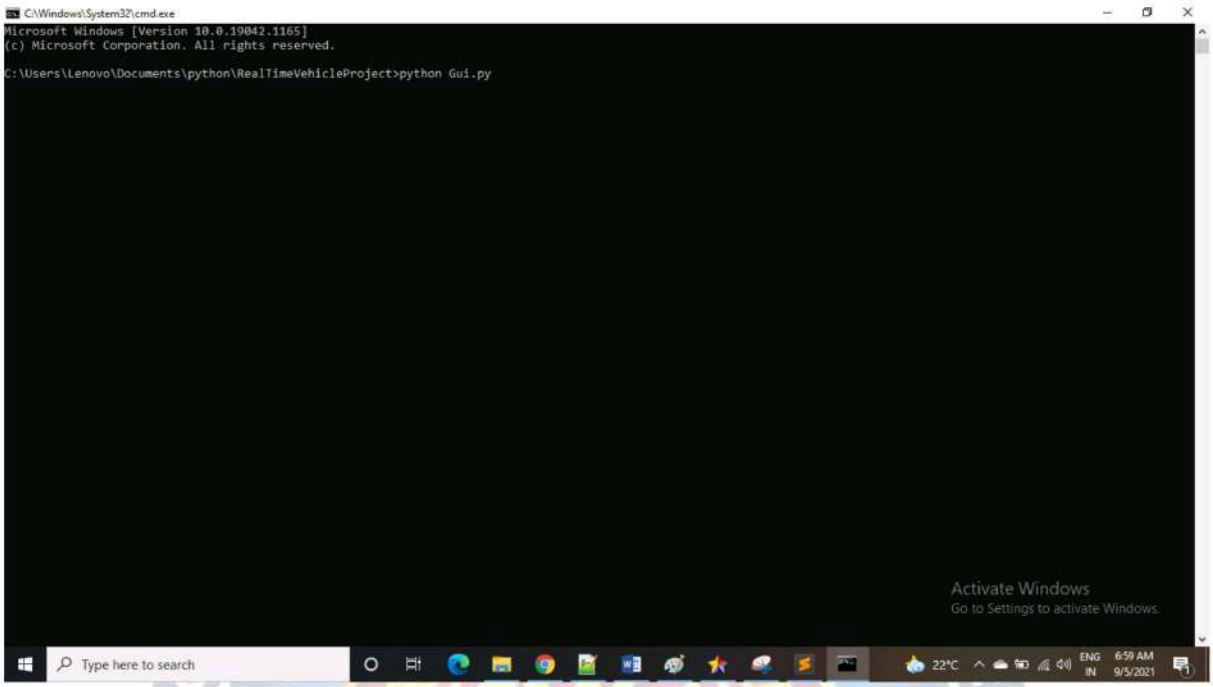


Figure 1.command prompt

Open the project folder

Open command prompt from the project folder address path

Inside address bar enter cmd and run python GUI.py file

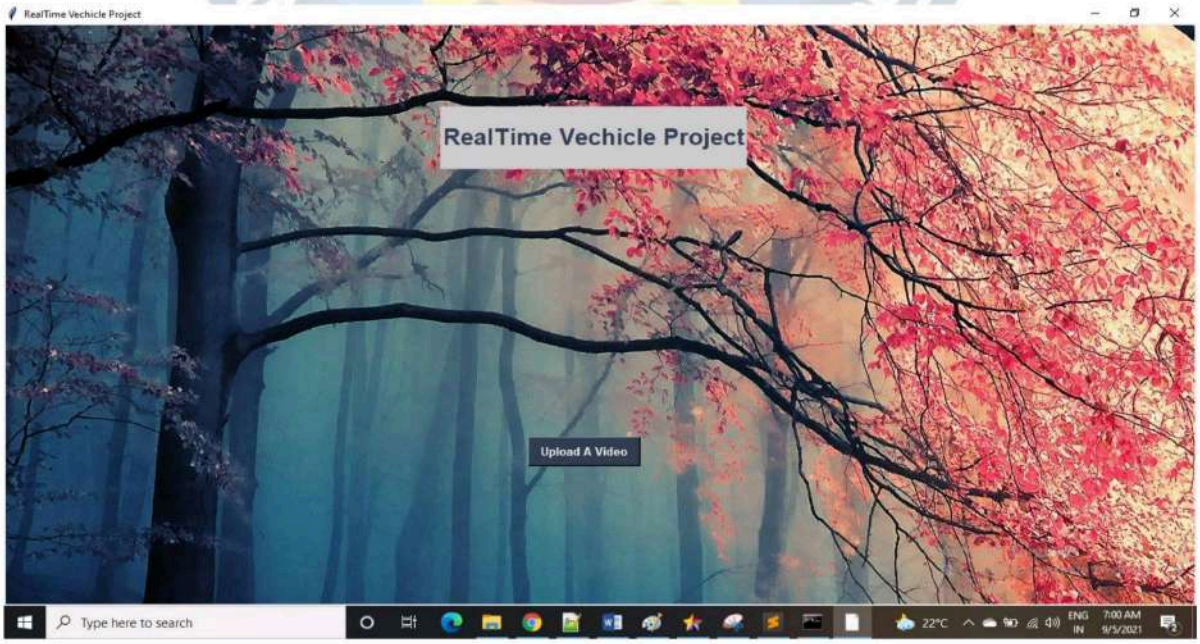


Figure 2. project home page

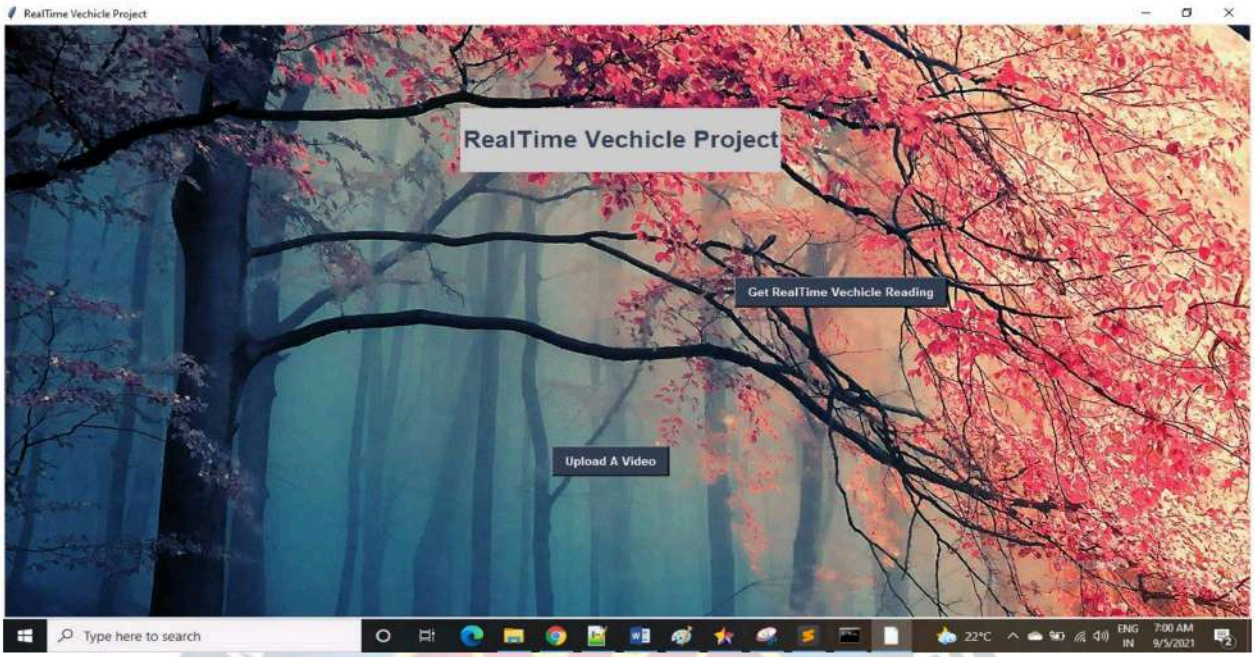


Figure 3. project home page

It loads video from local path

Click on real time Vehicle Reading Button

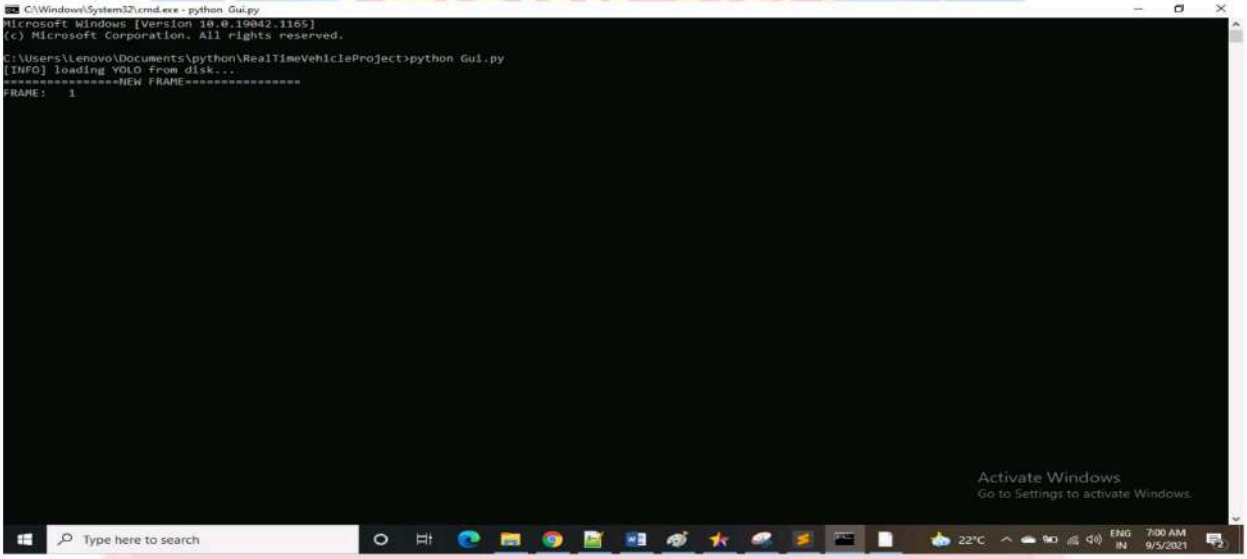


Figure 4. project home page

It identifies Real time vehicles and displays accuracy for each vehicle

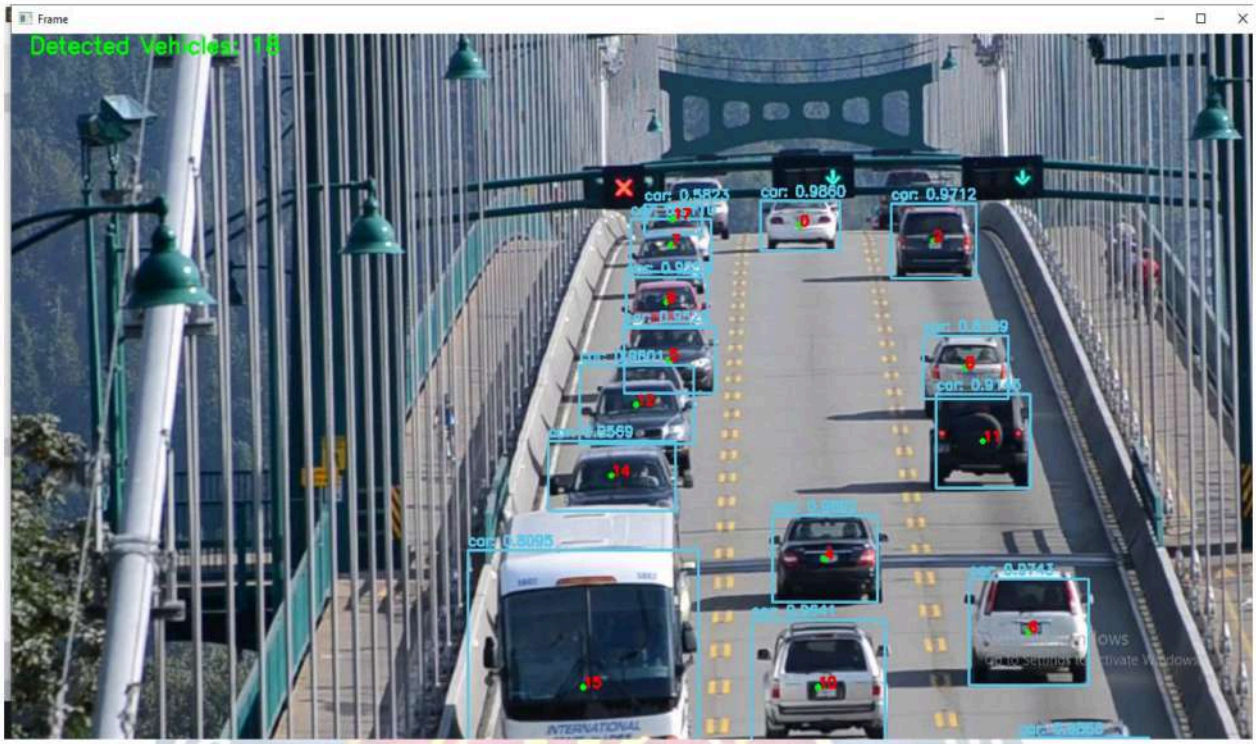


Figure 5. Accuracy of each vehicle

TECHNOLOGY FOR PROSPERITY
UGC AUTONOMOUS

CHAPTER – 9

SYSTEM TESTING

9.1 Test Cases

The purpose of testing is to discover errors. Testing is the process of trying to discover every 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 mKNNer. There are various types of test. Each test type addresses a specific testing requirement.

9.2 TYPES OF TESTS

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 perform basic 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.

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.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked

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.

System Test

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.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cKNNot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cKNNot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

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.

Test strategy and approach

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

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.

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.

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.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

9.3 Test Results: All the test cases mentioned above passed successfully. No defects encountered.

TEST CASES:

S.no	Test Case	Excepted Result	Result	Remarks(IF Fails)
1.	Loading Dataset	Loading Dataset Success	Pass	Image must be resize resolution will get better results
2.	Preprocessing	Loading the preprocessing	Pass	
3.	Feature extractment			
4.	Training and testing slipt	Detected images draw square and writing stress emotions	Pass	Images must be clearly to detect facial expression
5.	Model execution	PyImage libaray will load the process and start the live	Pass	If library not available then failed
6.	Algorithm	If tensorflow not installed then it will fail	Pass	Depends on system configuration and tensorflow library
7.	Results	Load the dataset and process the Algorithm	Pass	The dataset must be media folder
8.	Predict Train and Test data	Predicted and original salary will be displayed	Pass	Trains and test size must be specify otherwise failed

UGC AUTONOMOUS

CHAPTER – 10

CONCLUSION AND FEATURE ENHANCEMENT

The proposed solution is implemented on python, using the OpenCV bindings. The traffic camera footages from variety of sources are in implementation. A simple interface is developed for the user to select the region of interest to be analyzed and then image processing techniques are applied to calculate vehicle count and classified the vehicles. We have developed video based vehicle detection, classification, counting for real-time traffic data collection. We have used Background Subtraction Yolo algorithm, OpenCV, and python for developing the system. In the proposed system, we have considered all day and night shadowing, and different lighting situations. Also, we have considered the moving shadow of moving vehicles.

ST.MARTIN'S ENGINEERING COLLEGE
SMEC
TECHNOLOGY FOR PROSPERITY
UGC AUTONOMOUS

REFERENCES

[1] I. Alam, M. F. Ahmed, M. Alam, J. Ulisses, D. M. Farid, S. Shatabda, and R. J. F. Rossetti, "Pattern mining from historical traffic big data," in IEEE Technologies for Smart Cities (TENSYMP 2017). IEEE, July 2017, pp. 1–5.

[2] I. Alam, D. M. Farid, and R. J. F. Rossetti, "The prediction of traffic flow with regression analysis," in Emerging Technologies in Data Mining and Information Security, ser. Advances in Intelligent Systems and Computing, A. A., D. P., M. J., B. A., and D. S., Eds. Springer, Singapore, 2019, vol. 813, pp. 661–671.

[3] A. Talebpour, H. S. Mahmassani, and S. H. Hamdar, "Effect of information availability on stability of traffic flow: Percolation theory approach," Transportation Research Procedia, vol. 23, pp. 81–100, 2017.

[4] M. Dell'Orco and M. Marinelli, "Modeling the dynamic effect of information on drivers' choice behavior in the context of an advanced traveler information system," Transportation Research Part C: Emerging Technologies, vol. 85, pp. 168–183, 2017.

[5] A. Csikos, T. Charalambous, H. Farhadi, B. Kulcs ´ ar, and H. Wymeersch, ´ "Network traffic flow optimization under performance constraints," Transportation Research Part C: Emerging Technologies, vol. 83, pp. 120–133, 2017.

[6] M. Zhou, X. Qu, and X. Li, "A recurrent neural network based microscopic car following model to predict traffic oscillation," Transportation Research Part C: Emerging Technologies, vol. 84, pp. 245–264, 2017.

[7] F. B. Ghorghi and H. Zhou, "Traffic control devices for deterring wrongway driving: historical evolution and current practice," Journal of Traffic and Transportation Engineering, vol. 4, pp. 280–289, 2017.

[8] A. Abadi, T. Rajabioun, and P. A. Ioannou, "Traffic flow prediction for road transportation networks with limited traffic data," IEEE Transactions on Intelligent Transportation Systems, vol. 16, no. 2, pp. 653–662, 2015.

[9] S.-Y. Cheung, and P.P. Varaiya, "Traffic surveillance by wireless sensor networks: Final report", PhD diss., University of California at Berkeley, 2006.

[10] S. Oh, S. Ritchie, and C. Oh, "Real-time traffic measurement from single loop inductive signatures", Transportation Research Record: Journal of the Transportation Research Board, (1804), pp. 98-106, 200