

#### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

#### SCHOOL OF ENGINEERING AND TECHNOLOGY, SHARDA UNIVERSITY, GREATER NOIDA

Real-time Vehicle Detection and Tracking System using Cascade Classifier and Background Subtractor

#### A project submitted

#### In partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science and Engineering

**By**

**Priyanshi Verma (180101237)**

**Aprajita Singh (180101070)**

**Kajal Verma (180101137)**

**Arya Kumar (180101076)**

**Supervised by:**

**Dr. Vijendra Singh, Assistant Professor (CSE)**

**Session, 2022**

# CERTIFICATE

This is to be certify that the report entitled **“Real-time Vehicle Detection and Tracking System using Cascade Classifier and Background Subtractor”** submitted by **“**Priyanshi Verma (180101237) Aprajita Singh(180101070) Kajal Verma(180101137) Arya Kumar (180101076) to the Sharda University, towards the fulfillment of the requirements of the degree of **“Bachelor of Technology”** is record of the Bonafide final year Project work carried out by all of us in the **“**Department of Computer Science and Engineering, School of Engineering and Technology, Sharda University**”**.

The results/outcomes in this Project has not been submitted in part or full to any other University/Organization for any award of any others Degree/Diploma.

Signature of the Guide

Name: Dr. Vijendra Singh

Designation: Assistant. Professor (CSE)

Signature of Head of Department

Name: Prof. (Dr.) Nitin Rakesh

Place:

Date:

**Signature of External Examiner Date:**

# ACKNOWLEDGEMENT

The major project is the golden opportunity for learning and developing the skills in terms of technical skills, time management, team management etc. We consider ourself blessed and honored to collaborate with so many wonderful people who lead in completion of this major project.

First and foremost, we would like to express our profound gratitude to Dr. Nitin Rakesh (HOD), and Dr. Parma Nand (Dean) of Computer Science and Engineering Department for their contributions to the completion of our project **Vehicle Detection and Tracking**.

We would like to express special grace to our mentor **Dr Vijendra Singh** whoprovide usthis golden opportunity to do perform such wonderful project on the topic of **Vehicle Detection and Tracking**, which also helped us doing a lot of Research and we explored a lot of new things, his valuable guidance and suggestions helped us in every phase in this project. We are really grateful to him.

Secondly, we would also like to greet our colleagues who helped us a lot in finalizing the project within short span of time and without their support and coordination, we would not have been able to complete this project. I hope we will achieve more in our future endeavors.

Also, CSE department of our college monitored our progress and arranged all the facilities to make this process for the project easier. We choose this time to admit their benefaction gratefully.

Name and signature of Students

Priyanshi Verma (2018013701)

Aprajita Singh (2018012404)

Kajal Verma (2018013687)

Arya Kumar (2018014534)

# Abstract

The density of the particular area in this world increases as population of that area grows. This is frequently go along with an increase in the number of vehicles on the road, resulting in an increase in traffic congestion. The vehiclee such as a car, bus, or truck which is used to carry entities from place to place. Though Vehicle detection and tracking platforms plays a very necessary role and are so useful in highway traffic management, traffic surveillance, and traffic planning. Vehicle detection consists of vehicle tracking, vehicle counts, traffic analysis, average speed of vehicle, and vehicle categorization etc. There are diverse advantages of Vehicle Detection and Tracking. For instance, vehicle detection and counting system could be useful for traffic police because it allows them to monitor each and every vehicle from a single location such as how many vehicles have crossed this toll which vehicle at what time and doesn't require much manual effort. It is cost-effective also. From the past years this system Intelligent Transportation received a lot of attention which makes a major concern. This is a prevalent issue that can be resolved multiple use cases.

Also, as number of vehicles increasing everyday causes heavy traffic accidents considered as a major health problem result in death and long-term suffering for the victim's family, as well as disabilities, loss of productivity and hospitalization. In 2016, there were 4,80,652 traffic accidents in the United States, with 4,94,624 people injured and 1,50,785 people killed. Several issues can be resolved by implementing this approach like production will be high, less pollution, falling off the count of accidents, less manual efforts, and cost-effective etc. The proposed approach is put to the test using a dataset that contains video streams with common highway issues such changing lighting, traffic, camera vibration, and image blurring. We will use OpenCV library for doing all image processing operations and for classification. The haar cascade classifier is used for detecting and counting the cars.

Vehicle counting systems, intelligent parking systems along with Autonomous Driving Assistant Systems (ADAS), there is a high demand for Real-time Vehicle Detection System for detecting any mobile architecture. The importance of processing a real-time stream of frames of a video stands out to be the major outlook for creating a lightweight and efficient model for giving potent results. For this purpose, we need an algorithm which can be easily configured according to the hardware and perform well, even in low processing power thereby reducing inventory costs. In this paper, we have scrutinized the Cascade Classifier and the Background Subtractor MOG2 modus operandi’s for assuaging the competency of each model on different devices based on Frames Per Second (FPS) value and the amount of processing required to run in real-time. The comparison of these models has also been carried out with MobileNet and Xailient architectures for validation. The results gave us an idea of the need of including a complexity perspective into an algorithm for its actuation for proving its need in real-time.

***Index Terms*** Cascade Classifier, Background Subtractor MOG2, Computer Vision (CV), Mobile Net, Single Shot Object Detection (SSD), Xailient.

**Contents**

Title Page… i

[CERTIFICATE ii](#_bookmark0)

[ACKNOWLEDGEMENT iii](#_bookmark1)

[Abstract iv](#_bookmark2)

[Chapter1: INTRODUCTION 6](#_bookmark3)

* 1. Problem Definition……………………………………………………………………………………………………………….9
  2. [Project Overview/ Requirement Specifications……………………………………………………………..……………………………………….10](#_bookmark5)
  3. [Hardware Specifications 14](#_bookmark6)
  4. [Software Specifications 15](#_bookmark7)

[Chapter2: Literature Survey 16](#_bookmark8)

* 1. [Existing System: 16](#_bookmark9)
  2. [Proposed System 18](#_bookmark10)
  3. [Feasibility Study 19](#_bookmark11)

[Chapter 3: System Analysis and Design 21](#_bookmark13)

* 1. [Software Requirement Specification 21](#_bookmark14)
  2. [Flowcharts 26](#_bookmark15)
  3. [Test Steps and Testing 29](#_bookmark16)

[Chapter 4: RESULTS / OUTPUTS 37](#_bookmark18)

[Chapter 5: Conclusion 40](#_bookmark19)

[Chapter 6: References 41](#_bookmark21)

# Chapter1: INTRODUCTION

With the increase number of humans existing over the crust, the increment in the ownership of vehicles has been escalated to an extreme extent. Humans are capable of quickly recognizing cars in videos or photographs, as well as distinguishing between various types of vehicles. Computer algorithms and programmers are greatly influenced by the types of data. The operation might be made easier or more difficult depending on factors such as the weather or the quantity of light available. For vehicle detection and tracking, there are a variety of techniques and methodologies introduced. These approaches come in a wide range of forms are Recurrent Neural network and Support Vector Machine etc. Simultaneously, we have a wide range of vehicle types and shapes. Object detection is the act of identifying objects in a video sequence's frames, whereas object tracking is the process of locating a recognized object over a period of time. So many strategies introduced. Many solutions, such as Background removal, Mean-shift, and Optical flow algorithms, have been devised from time to time to increase the efficiency of object detection and tracking Most other algorithms, such as Frame Differencing, are built on the foundation of background subtraction. The basic idea is to extract the moving target from the sequence of image or video. A detector, a core data model, a hypothesis, and a hypothesis verifier are required to perform the task of detecting the item in an image or video system. Object tracking is crucial in the image processing applications. Object observation and tracking is a powerful search area, with the applications ranging from computerised video surveillance to the robotic vision, vehicle navigation, traffic detection, facility identification, and many more. A video is made up of a succession of images, each of which is called a frame. There are both moving and motionless objects are in the photograph sequence. Anything that moves, whether it's a person, a bird, a vehicle, or anything else.

Background subtractor is a mutual way to recognize the objects which are in motion in a series of vertical camera frames. Background subtraction detects changes in the frame by subtracting the incoming frames from set reference frames. As per the Statista Data, the number of registered vehicles in India was reported to be 295 million until 2019. It elucidated a compound growth of 10% from 2007 to 2019 in India itself. Due to this, we can often see that the roads are jammed due to extensive traffic, increase in accidents, increase in security breaches, etc. Thus, it becomes extremely important for ensuring the safety of human beings and aid the traffic issues, to develop a surveillance system for real time analysis of these vehicular prospect. Due to the increase in number of mobile structures on road, heavy obstruction is faced by common people due to traffic.

They proposed recursive equations for updating the parameters of a Gaussian mixture model and selecting the right number of components for each pixel in real time. For superior kernel density estimation, KNN used. MOG improve this adaptive background mixture model using a strategy. They used various equations at different phases after reexamining the update equations. This enables their system to learn faster and more precisely, as well as adapt to changing circumstances more efficiently. It was created using their background model and a computational color space. As a result, color video sequences yielded the best results. Gaussian mixture probability density is used in their adaptive technique. Recursive equations were utilized to update the parameters on a regular basis as well as to pick the right number of components for each pixel. Also, the demand for real time system is increasing per day. It plays a major role in robotics, healthcare, industrial manufacturing and high-precision industries, which majorly depends upon real-time data for nonstop improvement in safety, efficacy, and dependability, it needs real-time processing more and real-time system require more processing powers with real time response.

In this articulation, we collated Haar Cascade Classifier and Background SubtractorMOG2 approach for detecting vehicles in a video stream in real-time and also tracking those vehicles through counting by initiating vehicle counter. Both these detectors have been fine tuned for obtaining higher efficacy. We’ve also integrated real-time Frames Per Second (FPS) value for understanding the performance of model on varied hardware systems developing the low-specs and high-specs ones for clear explanation. For each backdrop pixel, the Combination of Gaussians, or Moog, is the mixture of the k Gaussians distribution model with k values between 3 and 5. Distinct distributions, according to inventor, indicates different background and foreground colors. Each of the employed distributions on the model has a weight that is proportional to how long each color stays on that pixel. As a result, when the pixel distribution's weight is less, that pixel is classed as foreground pixel.

But the major constraint revolves around the hardware requirements of these models. The increment in computational complexity hinders the overall performance of any workflow and when we consider doing it real-time, the approach revolving around the resources needs to be aggregated for concluding any modus operandi. Furthermore, for validating the efficacy of these models to be integrated into real-time, we also colligate them with different models such as Mobile Net Single Shot Detection (SSD) and Xailient for deep explication. This analysis gave us an insight of the efficacy of each model and their potency in Vehicle Detection prospect and also an ideation about the impact of each model over varied hardware systems and their stability in terms of efficiency on those system.

**Overview:**

To describe the learning model used, the specification and implementation of algorithms, we needed to investigate the most stable system using some machine learning techniques to the subject each model over varied hardware systems and their stability in terms of efficiency on those systems. We define the project and compare the strategies proposed in this chapter.

**Project Description:**

As already many works have been done in past times for vehicle detection and tracking. Detectors such as You Only Look Once (YOLO), Retina Net, etc. has been incorporated for Vehicle Detection widely. But the major constraint is all about how much hardware used in these models while using it in real time. The increase in computational complexity will decrease the overall performance of any workflow the approach revolving around the resources needs to be aggregated So the motive is to find the system that can be easily implemented in real time and also taking the look how many cores of CPU used and how much processing power is used?

We combined Cascade Classifier and Background Subtractor approach for detecting vehicles in a video in real-time and also tracking those vehicles through counting by initiating vehicle counter in real-time. Both these detectors have been fine tuned for obtaining higher efficacy. We’ve also integrated real-time Frames Per Second (FPS) value for understanding the performance of model on varied hardware systems developing the low-specs and high-specs ones for clear explanation.

Furthermore, for validating the efficacy of these models to be integrated into real-time, we also compared them with different models such as MobileNet Single Shot Detection (SSD) and Xailient for deep explication. This analysis gave us an insight of the efficacy of each model and their potency in Vehicle Detection prospect and also an ideation about the impact of each model over varied hardware systems and their stability in terms of efficiency on those systems.

## Problem Definition

The project's goal is to process Vehicle Detection and Tracking Model to identify the performance of each model on different devices based on Frames Per Second (FPS) value which defines amount of processing required to run in real-time.

Real-time processing is a way for processing data very instantly. This method does not require any pauses or waiting. These systems process data as soon as it is received and output the processed information. Real-time processing, by its very nature, necessitates a constant flow of data. Real-time processing can be found in bank ATMs, and traffic control systems, to name a few because processing takes place as data is entered, it requires continuous stream of input data to produce continuous output.

The importance of processing a real-time stream of frames of a video stands out to be the major outlook for creating a lightweight and efficient model for giving potent results. For this purpose, we need an algorithm which can be easily configured according to the hardware and perform well, even in low processing power thereby reducing inventory costs.

Object detection has attracted the attention of the research industry recently. Researchers are trying to determine the subject in order to achieve a level of accuracy. Machine learning is used to find objects. There are many strategies for doing this work but in order to identify the best model among the proposed models, this thesis will explore the topic of visualization at the same time comparing the two models and suggesting the best one that offers high accuracy.

## Project Overview/ Requirement Specifications

* + 1. **Functional Requirements**
       1. **Introduction**

Due to increase in number of vehicles on road every day, The US publisher Ward's states the estimation of year 2019, there were **1.4 billion** motor vehicles is in the use. Due to this, we can often see that the jammed roads due to extensive traffic, increase in accidents etc. For aiding these all the problems, there are so many researches are in progress and done in past times for vehicle detection, tracking and counting. Thus, it becomes extremely important for ensuring the safety of human beings and aid the traffic issues, to develop a surveillance system in real ­time. For scanning a vehicle on a road, we need to detect it and further we can track it. Thus, detectors such as You Only Look Once (YOLO), Retina Net, etc. has been incorporated for Vehicle Detection widely. But the major constraint revolves around the hardware requirements of these models. The increase in computational complexity decreases overall performance of any work.

In this, we collated Cascade Classifier and Background Subtractor approach for detecting vehicles in a video stream in real-time and also tracking those vehicles through counting by initiating vehicle counter. Both these detectors have been fine tuned for obtaining higher efficacy. We’ve also integrated real-time Frames Per Second (FPS) value for understanding the performance of model on varied hardware systems developing the low-specs and high-specs ones for clear explanation Furthermore, for validating the efficacy of these models to be integrated into real-time, we also colligate them with different models such as Mobile Net Single Shot Detection (SSD) and Xailient for deep explication.

* + - 1. **Input**

We have searched different videos where enormous number of cars moving on road. As we’re dealing with Frame Rate, we have tried to keep such video as input having higher speed moving vehicles when compared to other. We also wanted a video which could challenge the model for predicting accurate results. As a result, after filtering through our requirements we selected a video downloaded from pexels.com which was freely available to utilize for research purpose.

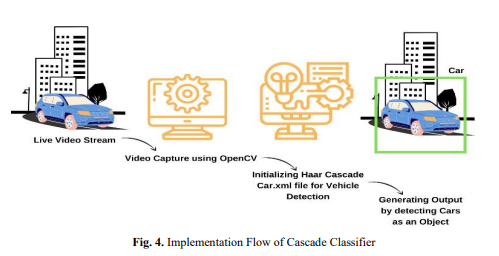


**Fig. 1 Input**

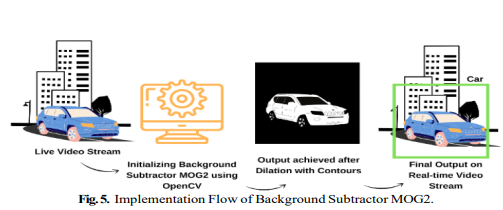
* + - 1. **Processing**

Using Open CV, for all image processing operations of the vehicle and for classification haar cascade classifier for identifying and counting the vehicles on the highway. We have used some common libraries open CV, matplotlib, and NumPy. To eliminate the noise from the image, we have also used Gaussian Blur i.e. image processing technique using Gaussian Blur function. It is also extensively used in graphics design to reduce the noise and smooth the image so that further preprocessing produces the best results. The Gaussian blur technique diminishes the features of image while simultaneously reducing the noise of image. And then applied some dilation which is morphological technique. In this technique we try to fill pixels with elements, which is also known as kernels, to fill in the vacant parts of the images as per need. As a result, the detection of traits is done in different stages. If the window is in the wrong place, we see that the first two-rectangular section is active, and the window is discarded before the second phase begins. Only one window, containing the object, runs in both directions and sees the car.

The goal of employing morphological techniques is to eliminate noise in the image. The majority of the operations are a mix of two processes: dilatation and erosion. We have implemented these techniques dilation and erosion. The dilation procedure increases the size of an item this technique is performed to fill in the gaps or to fill the missing pixels. While, the erosion operation is a complement to the dilation operation. The erosion operation reduces the size of the thing. As a result, it is utilized to erase the noise between two objects.



**Fig.2.** Implementation Flow of Cascade Classifier



**Fig. 3.** Implementation Flow of Background Subtractor MOG2

* + - 1. **Error handling**

During the application of cascade classifier, we have observed that detection of other objects such as tree is detected by the model, so we have added the background subtractor and added the concept of FPS value which led to handle the mistakes of model and help to make the accurate result even in fast moving vehicles.

* + 1. **Functional Requirements**

These are the things we have observed and tried to give the solution for using the system in real time

**N1:** Feedback should identify the different sizes of vehicles moving with different speed.

**N2:** Complexity as compared to detection algorithm such as YOLO is less.

**N3:** Less complexity led to use of less processing power and more accurate results.

**N4:** Cost effective because we are using CPU only, GPU is not added in the picture.

* + 1. **Nonfunctional Requirements**

Non-functional requirements are which specifies how the system perform a certain function of its own. It describes system behavior from technical point of view.

* + - 1. **Performance Requirements**

We aggregated two systems with both lower processing power and high processing power.In comparison to the time and assets used, execution is defined by the measure of useful work done by a PC framework or PC system.

* + - 1. **Reliability**

The system necessarily give stability and possibility to reuse the results in the future. The system should be working in a real time which allows the users to view 24/7.

* + - 1. **Availability**

Accessibility is a general concept used in PC systems and system management to describe the measure of time over a one-year span that the framework assets are available in the wake of partial system disappointments. A structure with all its properties that is continually available is seen as fruitful.

* + - 1. **Security**

Security (or PC security) in registration is the technique to ensure that information placed on a PC can not be accessed or negotiated without approval by any person. Data encryption and passwords are the majority of PC efforts to develop security. Encryption of information is the interpretation of data into a structure that is indiscernible without a method of disentanglement. A watchword is a mystery word or phrase that gives a client access to a particular project or structure.

* + - 1. **Maintainability**

It is defined as the probability within a given time of conducting a successful repair operation. As such, practicality tests the straightforwardness and speed at which, after a disappointment occurs, a system can be returned to operating status. Convenience is a trademark that is credited to a PC application in the event that it may be used rather than the one in which it was developed as part of operating systems without the need for major reconstruction. Porting is the job of performing whatever work that is necessary to maintain the PC program going in the new environment.

* + - 1. **Ability of Learning**

It is simple to operate and operate with less complexity.

## Hardware Specifications

## High-end Hardware

|  |  |  |
| --- | --- | --- |
| Hardware Utilized | Cores Used | CPU Utilization(%) |
| Honor Magicbook 15 | 4 | 80 |
| Honor Magicbook 15 | 1 | 60 |
| Honor Magicbook 15 | 4 | 85 |
| Honor Magicbook 15 | 4 | 72 |

|  |  |  |
| --- | --- | --- |
| **Hardware Utilized** | **Cores Used** | **CPU Utilization(%)** |
| Iball CompBook | 4 | 95 |
| Iball CompBook | 1 | 75 |
| Iball CompBook | 4 | 94 |
| Iball CompBook | 4 | 85 |

## Low-end Hardware

## Software Specifications

|  |
| --- |
| **Tensorflow** |
| **Numpy** |
| **OpenCV** |

**Language**

* Python

**Chapter2: Literature Survey**

## Existing System:

The current video-based systems are very sensitive in environmental conditions like weather or illuminations which will result in lesser accuracy and relatability. In this we will discuss some of them:

* Camera motion: when there is motion in front of camera view or the videos captured by unstable or vibrating cameras it will result in motion blur in videos which makes it harder to detect and track vehicles. Motion blur can be avoided by de-blurring or estimating a single kernel for the entire image.
* Congested traffic conditions: In urban traffic condition, traffic jams are very common so vehicle occlusion can occur anytime a vehicle passes behind other vehicles with respect to a camera. If the vehicle detection system is based on motion information it will affect the process of computing background frame.
* Low light conditions: In night or in dark tunnel vehicle detection becomes hard so cars can’t be detected by their visual features and only part which can be recognized is headlight/taillight lamps. The other challenge is it have to pair detected lamps to be considered as an individual car. This lack of features can affect detection and tracking process. Image binary conversions using an adequate threshold value can solve these situations.
* Vehicle shadows: In sunny weather condition vehicle shadows are often formed and accompany the moving vehicle. Theses shadow sometimes regarded as part of the vehicle and may affect the detection task. In these circumstances casting shadow elimination or edge detection to separate shadows from vehicles can result in better accuracy in detection.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Author** | | **Approach** | **Description** | **Advantage** | **Disadvantage** |
| **1** | Yanfeng Chen  Qingxiang Wu | Optical Flow  (Lucas-kanade method) | Even in the presence of camera motion, optical-flow is used to detect moving objects. | |  |  |  | | --- | --- | --- | | It Works for static Backgrounds and adaptable to dynamic backgrounds and it is easy to implement. |  | Effective ondynamicbackgrounds | | Noise sensitivity can indicate more complexity because it is very vulnerable to noise. |
| **2** | Lucas-kanade | Image registration  Technique | With the Newton-Raphson iteration, the gradient feature of special intensity is used to match the objects in the photographs. | It can also detect objects even if the object is rounded, scaled or trimmed. | The quality of image must be greater for matching of objects. |
| **3** | Hostettler, Wolfgang Birk, Denis Kleyko, Roland, and Evgeny Osipov | Vehicle Classification using Road Side Sensors | The method was tested on a large database, comprising 3074 vehicles in total using different algorithms and a variety of differentiation methods used as sensory networks and logical retrieval. | Best performance was given by the logistic regression with the classification rate 93.4% comparison with other of machine learning methods. | In this way the use of databases, as it focuses mainly on one class. |
| **4** | Seda Kul, Süleyman Eken, Ahmet Sayar | Vehicle detection and classification in the real time video streams | Vehicle categorization, feature extraction, foreground detection, and background subtraction are some of the techniques used. | During the day the results are good with an accuracy of 89.4%. | During the dark or bad weather condition they didn’t perform well. |

**Table:** Literature Survey on Vehicle Detection and Classification

## Proposed System:

Initiating for the Vehicle Detection prospect we scrutinized over diverse approaches, to ultimately formulate the potency along-with the shortcomings in these approaches. Vehicle Speed Calculation system utilizing Sparse Random Projection over Raspberry Pi was instantiated gaining an effective frame rate of 7.08 fps which was improved from 3.29 fps without optimization. The process needed more optimization for even better outcomes. Cheap Embedded System for Counting of Vehicle and Classifying it for Management of traffic was elucidated expelling the efficacy of 95.35% with the need of more training samples for better results. For aiding the safety of Self-Driving vehicles, a vision-based approach has been conferred with the integration of OpenCV python library for Object Detection on the go. The efficacy turned out to be decent giving an edge for integrating the given method in real-time.

The incorporation of Artificial Neural Network (ANN) for counting the number people inside public transport was exhibited for complying Transit Law in Ecuador using OpenCV gaining the effectiveness of 90% with the need of improvement for tackling obstacles for approximation. Moreover, perpetual driving aiding system was introduced using OpenCV for detecting lanes, detection of blind spot, etc. The model was implemented over Raspberry Pi for gauging accuracy. A vehicle counting system was also effectuated utilizing background subtraction method for specifically complying to The Sidoarjo Toll Road gaining an efficacy of 92.3% in the morning and 77.3% in the evening, giving an insight of the importance of camera position and the environmental outlook for accurate results. The same method of Background Subtraction for Vehicle Detection but in real-time was also elucidated, formulating an Intelligent Transportation Systems (ITS) for traffic monitoring which gained an average accuracy rate of 71.38% with the shortcoming of proper tuning. Furthermore, a real-time traffic data was collected for vehicle detection, counting, classification and speed measurement through adaptive video-based approach gaining an accuracy of 82% for real-time video and 81% for recorded video having the limitation of video processing in real-time. Integration of Haar Algorithm for Object Detection was scrutinized for analysing its potency through Arduino Board in real-time. You Only Look Once (YOLOv3) algorithm has also been consummated for performance analysis of Object Detection and Tracking outlook gaining an average accuracy of 92% over KITTI dataset. YOLOv3 performs well over images but when induced for real-time analysis requires a good system for its application. The CAMSHIFT algorithm has also been effectuated for vehicle detection in real-time where it performed well but with the issues of including shadows in its detections.

Thus, commemorating all the minutes of these approaches we found the major limitation in the resource’s perspective as well as the hindrance in the accuracy due to it. As a result, we came to a conclusion of actuating the modus operandi and thereby analyzing those results in real-time to provide an insight about the reason of shortcomings in the previous approaches.

* 1. **Feasibility Study:**

Any comprehension of the major specifications for the scheme is necessary for feasibility study. Whenever we talk about driving road safety has been an issue as long as cars have been in existence. Every year thousands of people died in India due to non-uniform driving which can be curable. Feasibility Dimensions for Computers would be as shown in:

* **Technology**

Is the project technically possible?

Is it a component of the state of the art?

Will failure be limited to the need for an implementation meeting the level?

* **Finance**

Is it financially practicable?

Is it realistic for the software company and its customer or company to achieve production at a reasonable pace?

* **Time**

Can the time for the idea to be sold, beat the competition?

* **Resources**

Will the corporation have the capital necessary for success?

Two major variables used in the study of viability are:

1. Technological Feasibility
2. Cost Feasibility
3. **Technical Feasibility** The purpose of this analysis is to check the technological viability, that is to say, the system's technical requirements. Any built system does not have a strong need for the technological resources required. This will add to intense strains on the intellectual resources available. It would bring to the customer's already firm hopes. Since this system can only be applied with minor to no modifications, a bare minimum must be met.
4. **Cost Feasibility** This study evaluates the economic impact of the scheme on the Autonomous Driving Assistant Systems (ADAS). It restricts the amount of money that can spend on the research and development of its strategy. It is necessary to justify the expenses. Thus, within the budget, the developed system was also developed and this was done because much of the technology used is readily accessible.

# Chapter 3: System Analysis and Design

## Software Requirement Specification

* + 1. **Open CV**

OpenCV (Open-Source Computer Vision Library) is a free software for computer visualization and machine learning. OpenCV was created to provide a common computer vision infrastructure to improve the use of machine vision in the field of commercial computer vision Intel created OpenCV in 1999, and is widely used now.

As a result, it is available for both educational and commercial use. C ++, C, Python, and Java are all included. Windows, Linux, Android, iOS, and Mac OS are all supported by the main interface of Android. It has more than 2500 algorithms developed by OpenCV for over 9 million users worldwide.

* + 1. **Image**

Before we get into image processing let's talk about the image itself. Most of us think of a picture as a

picture that we see in a wall or magazine etc.

A(x,y) = H(x,y) + B(x,y) Where, A(x,y)= function of noisy image, H(x,y)= function of image

noise, B(x,y)= function of original image. In theoretical terms, a picture that we look at is a function

of image intensity at a particular position in the image. i.e. *I(x,y) is an image function*where I = Intensity

at position (x,y) in an image.

**Types of digital images:**

There are typically three types of digital images.

1. Binary Images
2. Gray Scale Images
3. Color Images

**1.Binary images**

f: [a,b] \* [c,d] -> 0 or 255 (For binary images, the output of the function is either the brightest pixel 255

or the darkest pixel 0)

**2.Gray Scale images**

f: [a,b] \* [c,d] -> [min,max] (For gray-scale images, the output of the function is a range of possible

values from the brightest pixel 255 to the darkest pixel 0)

**3. Color Images**

For color images they are three functions stacked together as a “vector valued function. Those function represent red, blue and green pixel values.

* + 1. **Computer Vision**

Computer vision (CV) is a small artificial intelligence (AI) program that focuses on developing and deploying digital systems that process, analyze, and interpret visual data. The goal of computer vision is to allow computers to see an object or person in a digital image and to take appropriate action.

Computer Vision has a lot of overlap with the following fields:

• Image Processing is concerned with the alteration of images.

• Pattern Recognition discusses how to classify patterns using numerous strategies.

* + 1. **Image Processing**

Various scanning algorithms are used to process the images, such as Image inversion, Conversion of Grey Scale and thinning of images. Other techniques, such as noise reduction, image segmentation, cutting, and scaling, are also used. These techniques were used primarily in our project for 4-image recognition, but some were also used in touch mode, such as cutting the written character and scaling it to our input.

There are various operations you can perform using OpenCV

* Read and write images
* Capture and store videos
* Process the images
* Perform features detection
* Detect specific objects using the OpenCV library and capture the video, removing the background, and tracking objects many more.
  + 1. **Mobilenet SSD**

Mobilenet SSD is an acquisition model that combines a binding box and an item category from an embedded image. This Single Shot Detector (SSD) detection model practices Mobilenet as the support and could accomplish instant entity recognition for mobile based appliances.

Mobilenet SSD captures (3,300,300) images as input and output (1,3000,4) boxes and (1,3000,21) scores. Boxes containing offset values ​​(cx, cy, w, h) are from the default box. Scores contain the confidence levels for each of the 20 categories, with a value of 0 set for the background.

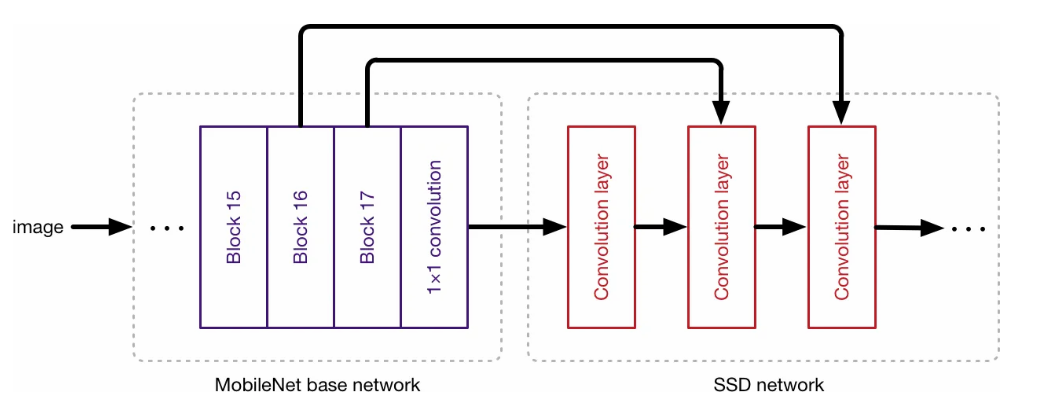
Single Shot or SSD object detection takes one shot to get multiple objects within an image. The SSD method is totally depend on a feed-forward convolutional network that produce a fixed-size set of binding boxes and notches for the occurrence of phase-level objects in the boxes.

It is made up of two parts:

* Remove map features
* Use the convolution filter to get things done

SSD is basically designed to be self-regulating of basic network, so it can work on any of the basic networks like VGG, YOLO model, MobileNet.

In order to additional address the applied restrictions of using high-end neural networks and power on low-end devices in real-time applications, MobileNet was integrated with the SSD framework. So, when MobileNet was used as the primary network on the SSD, it became the MobileNet SSD.



**Fig.4** Mobilenet SSD Overview

* + 1. **Xailient**

The Xailient model uses the selected attention method to detect it. Inspired by the workings of the human eye.

Xailient models have been developed to work on low-power devices with memory and slow service.

* + 1. **Haar Cascade classifier**

Haar Cascade Classifier is entrenched over ‘Haar Wavelets’. It can basically be coined as a series of rescaled ‘square-shaped’ functions which is collated to formulate a family of wavelet or create its basis. Haar Algorithm can be elucidated in four phases:

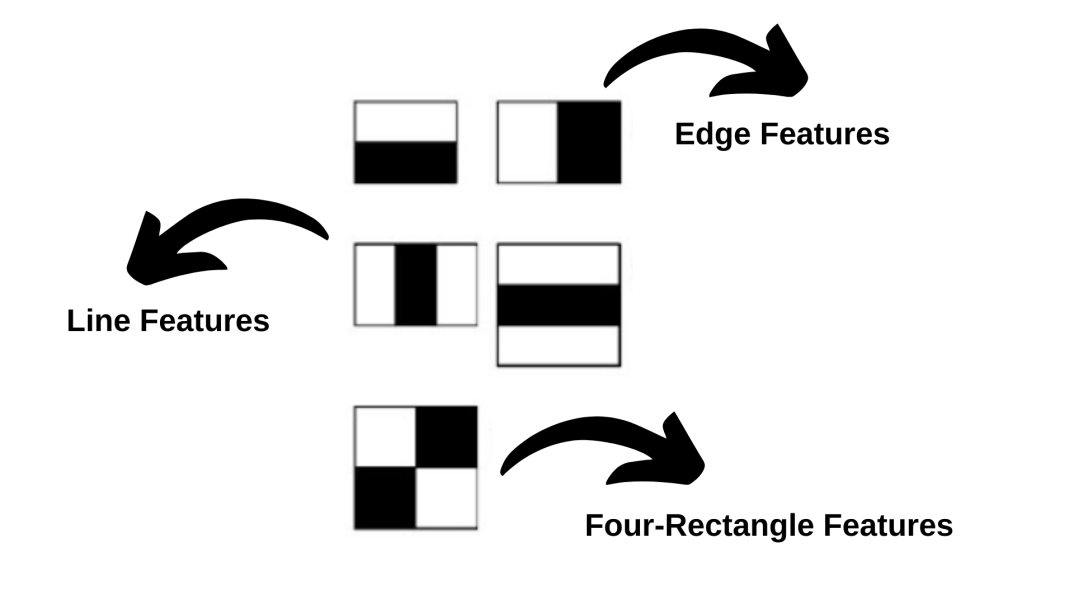
• The calculation of Haar Features

• The creation of integral images

• The utilization of Adaboost

• Finally, the Implementation of Cascading Classifiers

Basically, it can be subsumed as a Haar Wavelet operation for analysing pixels present in an image in the form of squares through a function. Training Data is utilized (if the **Segmentation** dataset is provided) for obtaining a higher order of efficacy with the collation of machine learning technique. The features present in the dataset is detected and computed through a concept known as an Integral Image. The Adaboost algorithm which is utilized for enhancing the learning perspective of a model through the detection of minute features and thereby making the output more efficient is used in Haar Cascades. The Cascade Classifiers are stored as an .xml files for any object detection. Figure.1 demonstrates the diverse and unique Haar Features for the Feature Extraction. In Integral creation, In an Integral Image, the sum of all pixels above and left of each point indicates that point's value. An Integral Image can be calculated in one pass across the image. For a window of 24x24 pixels, there are around 162,336 possible features, which would be prohibitively expensive to evaluate. As a result, the AdaBoost method is used to train the classifier using only the best characteristics. A cascade classifier consists of several classifiers connected in a logical order. It makes a number of small decisions about whether or not something is an item. The decision tree structure of the cascade classifier is degenerate.



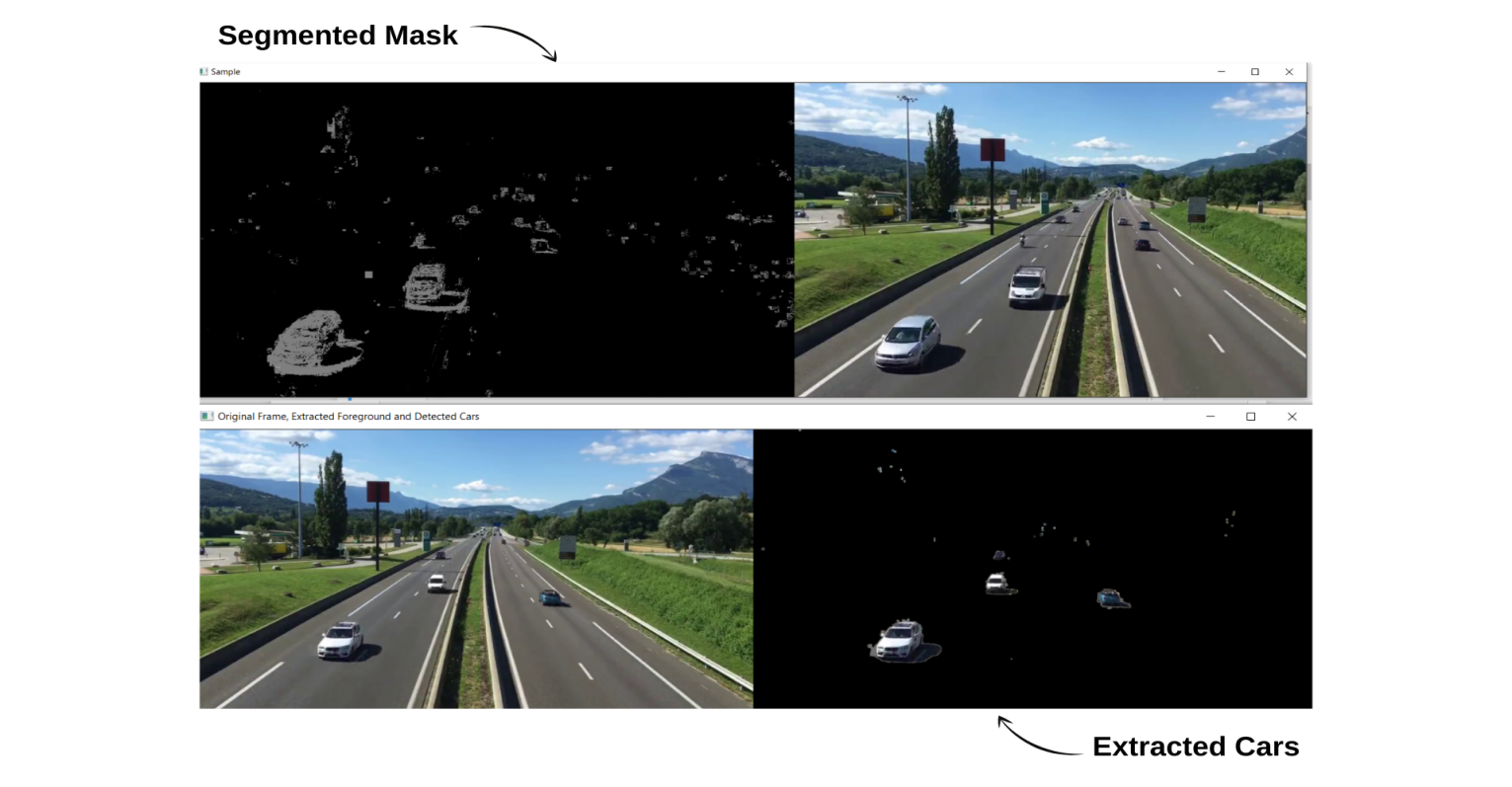
**Fig.5.** Casscade Classifier

* + 1. **Background Subtraction**

In many computers vision-based applications, background detection is a crucial stage in the preprocessing process. Consider the following scenarios: a visitor counter for persons entering or leaving a room, a traffic gadget that pulls vehicle data, and so on. In these circumstances, we must first separate persons or cars from the noise that contains irrelevant data. We need to separate the moving background from the static background from a technological standpoint. This challenge would have been much easier if we only had the background image, such as an empty room or an empty container. However, in most circumstances, such an image is not available, thus the backdrop must be identified and extracted from the images that are available. ItaOne of the methods for detecting objects in motion is Background Subtraction. Sometimes this method is also known as Foreground Detection. The process of detecting foreground from a background is one of famous techniques in field of Image Processing and Computer Vision. The modus operandi of Background Subtraction is incorporated for detection of moving entities over videos captured through static cameras. This approach is implemented on the basis of the difference of background reference and the frame.

Furthermore, with the background subtractor, morphology is also induced, which is also an image processing outlook that works on image segmentation for escalating segmentation accuracy. The morphological modus operandi is basically used for binary images, grey images and images with varied intensity values. The most widely used morphological techniques are Dilation and Erosion. Dilation majorly focuses on accentuating the objects through addition of layers, whereas Erosion turns out to be opposite of Dilation, i.e., it aids the erosion of edges of any entity.

Moreover, an edge detector exhibits an edge formulation in binary image elucidation, where the pixels at the edges are represented in white and the remaining area is converted to black. Thus, for better separation between these edges it’s necessary to have a bind between two edges. Contours is coined as the sequence of pixel information forming the encapsulating boundary of a region. The utilization of contours for vehicle detection stands out to be one of the most prominent modules. Thus, the collation of all these aspect aids in consummating the modus operandi for Vehicle Detection using Background Subtractor. These techniques have been induced in real-time for testing their applicability and proving their potency. Thus, implementation over these prospects was initiated. Figure.2 represents the Background Subtractor overview.



**Fig.6.** Demonstration of Background Subtractor Output

**Flowcharts**

**Process Flow for Cascade Classifier and Background Subtractor**

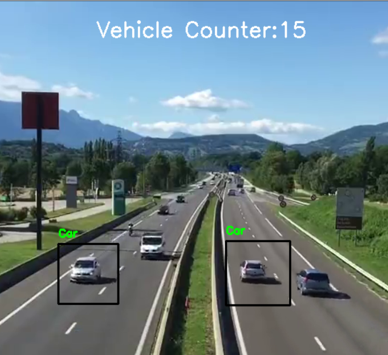
## Test Steps and Testing

* + 1. **Process Model**

## Initiating with the process, we have done the massive amount of literature survey and have read different papers related to vehicle detection and tracking and also visit different sites which shows different comparison for different algorithms about the accuracy. The idea about Mobile Net and Xailient’s comparison which has been compared on the basis of number of cores used, hardware used, percentage of CPU utilization has gathered from a website. After that we have move towards the Cascade Classifier and background subtractor MOG2.The idea about FPS came into picture because we are have to use this system on roads where there are large amount of vehicles moving with high speed, so better FPS value lead to accurate results. The modus operandi for the proposed outlook has been mentioned below.

* + 1. **Vehicle Detection Phase**

Cameras are used to capturing the real time video of the road and processing it Using Haar Cascade introduced by Viola and Jones provided edge or line detection features, which are used in the implemented approach. They are able to track the vehicle with focal area for the vehicle in motion in real time. The models are saved in XML files in the repository and can be accessed using OpenCV techniques. Vehicle detection, pedestrian detection, license plate detection, and other models are among them. These elements on the image make it simple to locate the image's boundaries or lines, as well as locations where the brightness of the pixels abruptly shift.

Pixels with a value of 1 are blacker on the haar, while pixels with a value of 0 are light. Each of these features a unique feature in the image. Any structure in an image with a rapid change of intensity, such as edge, line, or any other structure. The haar feature, for example, can find a straight line with black pixels on the right and bright pixels on the left in the image above. These are divided into three groups based on the quality that each person wants. The first pair of rectangular features is responsible for determining whether the edges are horizontal or vertical (as shown above). The second set of three rectangular elements is responsible for determining whether a simple surface is surrounded by black or vice versa. The final set of four rectangular symbols is responsible for determining how the pixel intensity varies across diagonals. The traversal of haar features on a picture now necessitates a great deal of arithmetic. As we can see, a single rectangle on either side requires 18-pixel value increases (for a rectangle enclosing 18 pixels). Consider doing this for the full image, with all of the different-sized haar features. This would be a frenetic procedure even with a high-performance system. Rectangles are returned as a list of observed items.

**Fig.7.** Vehicle Detection

* + 1. **Vehicle Counting Phase**

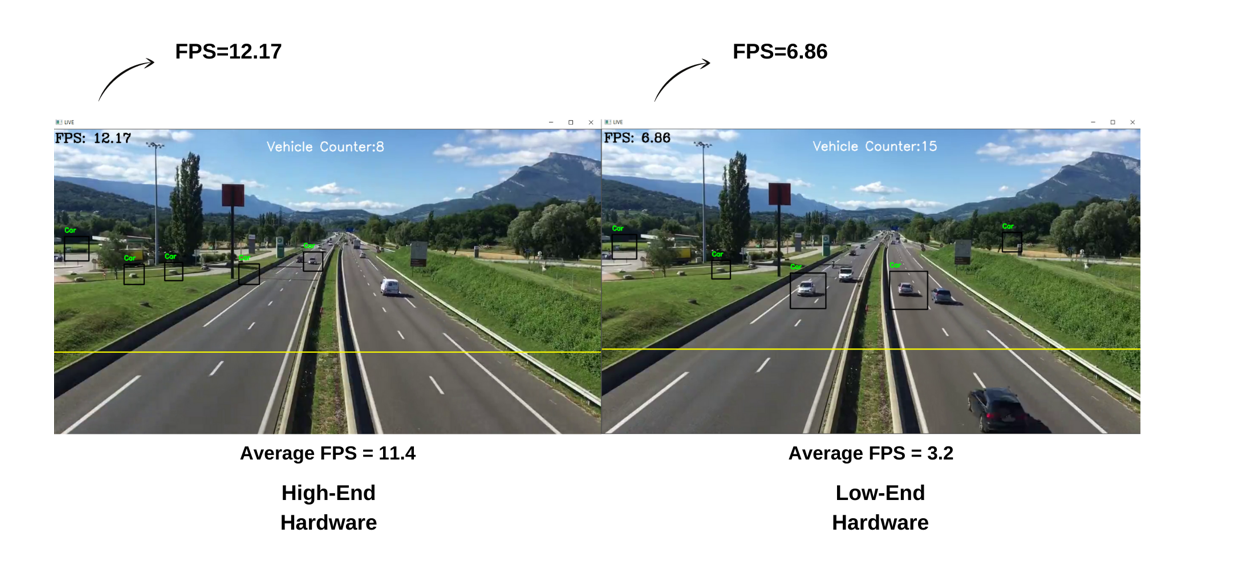
Now we'll do a video of the car detection and counting. The cv2.VideoWriter() method is used to create the output video from frames or images. It takes a path with extension as the first parameter, a codec for output format as the second parameter, and frames per second, height, and width as additional parameters. We'll now receive frames from the input video one by one, convert them to grayscale, and use car cascade to detect all cars in that frame.



**Fig.8.**Vehicle Counting

* + 1. **FPS Value**

We have also implemented FPS count and compared the value at different system. The word FPS (frames per second) refers to the number of recorded images per second displayed on a video device. More the FPS value the more accurate results we will find but there are always some limitations for everything we have tried to do fine tuning for better results. The human eye records images at a rate of 24 frames per second. When we display a picture at 50 frames per second, for example, the resulting image appears smoother and crisper to the human eye since it is receiving twice as much information. For video recording, there are numerous commonly used values.

If you're filming fast action in front of your camera, a higher frame rate will make the image more appealing to the viewer. The image will appear smoother and more natural as a result. High frames per second is commonly utilized in sports feeds, where there is a lot of fast movement in front of the camera. The smoothness and sharpness of the image are the key concerns of the viewer in these conditions. Nobody enjoys watching a ball fly through the air, leaving a fuzzy trail behind it. Variation of the value is shown below.

**Fig.9.** FPS Value Comparison

# 

* + 1. **Dataset Utilized**

We collated certain videos where we could see suitable number of cars that would be on a normal road. As we’re dealing with Frame Rate, we tried to keep videos having higher speed when compared to other. We also wanted a video which could challenge the model for predicting accurate results. As a result, after filtering through our requirements we selected a video downloaded from pexels.com which was freely available to utilize. Figure.3 demonstrates a static image of the chosen video.

# Fig.10. Demonstration of Dataset Utilized

* + 1. **Libraries Utilized**

As we’re dealing with vision-based modus operandi, we incorporated Computer Vision OpenCV python library for implementation of both the algorithms.

**NumPy** is basically a Python component that permits us to relate with arrays.

* It offers various functions working such as matrices, and linear algebra.
* Travis Oliphant introduced NumPy in year2005. It is an open-source library that any user is free to use.
* Numerical Python is mentioned as NumPy.

# NumPy arrays, distinct list, are stored in a single continuous memory location, giving permission to applications to access and manage in different ways easily. In computer science, this is called the reference area. This is the main reason why NumPy is so successful on the list. It has been redesigned to work with the latest CPU architecture.

* + 1. **Algorithms Implemented**

After deciding the Dataset and the computation framework, we stumbled upon the algorithmic outlook. Graphically represents the implementation overview of Cascade Classifier for the proposed system. Similarly, demonstrates the actuation of Background Subtractor MOG2 for the exhibited formulation.

For cascade classifier we utilized Cars.xml Haar Cascade File available over GitHub repository for analysis through OpenCV. Furthermore, we integrated Background Subtractor MOG2 which is an improved version of Gaussian Mixture Model i.e., Background Subtractor MOG. The major difference between both the prospects is, MOG2 has an option of choosing whether to include shadows for detection or not, whereas MOG doesn’t have it.

Also, MOG2 provides efficient adaptability for diverse scenes due to alteration in illumination. Due to all these improvements, we utilized MOG2 over MOG for our explication. We also inculcated Frames Per Second (FPS) value in real-time for both the models to gauge the efficacy of the model, through encapsulating processing anticipation.

* + 1. **Resources Utilized**

As we’re colligating the computational complexity outlook for inferring an elucidation, we aggregated two systems with both lower processing power and high processing power, but both explicated over CPU processing itself and no GPU was utilized.

1. The Low-End System

iball CompBook Pentium Quad Core Processor with Windows 10 Operating System.

1. The High-End System

Honor Magic book 15 Ryzen-5 Quad Core 3500U with Windows 10 operating system.

The High-End here doesn’t mean an extremely expensive system. We considered only the power of processors which can be consummated through their generations for inferring conclusion. The core remains the same i.e., both the systems are Quad Core Processors.

* + 1. **Metric Used for Analysis**

Frames per Second (FPS) Value has been inferred for gauging the potency of model over varied systems. The FPS value is calculated as: Demonstrated in Equation.1 and Equation.2.

Time Difference = Fps\_End\_Time – Fps\_Start\_Time …… (1)

FPS = 1 / Time Difference

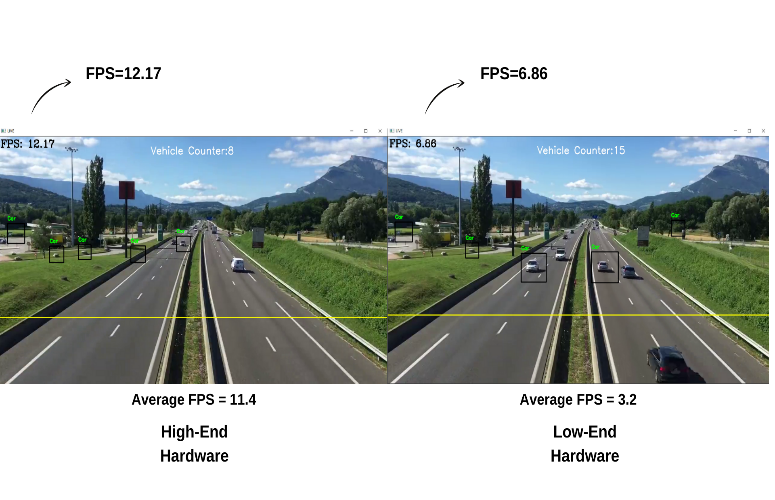
Also, the accuracy of both the algorithms has also been calculated, taking Vehicle Count (VC) as the reference point.

Equation.3 denotes the accuracy calculation.

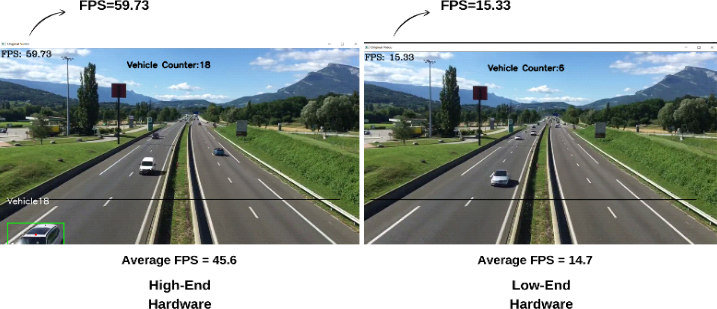
A = (Predicted VC/ Actual VC) \* 100 …………………… (3)

We ran the same code over both the systems and the result we got was worth the scrutiny as we got wonderful insights which was never juxtaposed in previous researches and the modus operandi proved to be an explication that was much needed. The experimental outcomes are elucidated in further section with real-time representation of Output with Frames Per Second (FPS) value and also the Vehicle Count.

# Chapter 4: RESULTS / OUTPUTS

As we implemented our proposed system in real-time, we got phenomenal insights. Figure.6 displays the output results for the Cascade Classifier with Frame Rate and the Vehicle Count in real-time. Similarly, Figure.7 represents the output results for the Background Subtractor MOG2 with Frame Rate and the Vehicle Count. The output is illustrated from the High-End device and the accuracy of these algorithms stood out to be best till now.

**Fig .11.** Real-time Output for Cascade Classifier with FPS value and Vehicle Counter.



**Fig .12.** Real-time Output for Background Subtractor MOG2 with FPS and Vehicle Counter.

|  |  |  |
| --- | --- | --- |
| **SN** | **Models** | **Accuracy( in %)** |
| **1.** | **Casscade Classifier** | **92** |
| **2.** | **Background Subtractor MOG2** | **97** |

**Table 1. Accuracy Illustration**

**Table 2. FPS Value Comparison on High-end Hardware for different models.**

|  |  |  |
| --- | --- | --- |
| **SN** | **Models** | **Accuracy( in %)** |
| **1.** | **Casscade Classifier** | **92** |
| **2.** | **Background Subtractor MOG2** | **97** |

**Table 3. FPS Value Comparison on Low-End Hardware for different models.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Models** | **Hardware Utilized** | **Cores Used** | **FPS Value** | **CPU Utilization(%)** |
| **MobileNet SSD [22]** | **Iball CompBook** | **4** | **0.8** | **95** |
| **Xailient [22]** | **Iball CompBook** | **1** | **8.2** | **75** |
| **Cascade Classifier** | **Iball CompBook** | **4** | **3.2** | **94** |
| **Background Subtractor MOG2** | **Iball CompBook** | **4** | **14.7** | **85** |

# Chapter 5: Conclusion

The above explanation proves that the efficiency of Background Subtractor MOG2 over Cascade Classifier in terms of accuracy for Vehicle Detection and Counting. Because the haar cascade is employed for object detection, this project has a very broad reach. It can be used to detect any form of object. For the specified object, we can even design our own unique haar cascade. As the feature extraction is escalated through Adaboost addition, the results for Background Subtractor MOG2 allows it to be a prominent solution for real-time Vehicle Detection for Traffic Monitoring Systems. The Cascade Classifier lags in terms of Vehicle Detecting and Tracking and also detects false entities. Furthermore, when the comparison is done upon utilization of resources Background Subtractor MOG2 and Xailient turns out to be the most efficient one. Through this explication, we can infer that a model possessing the capability of producing potent results along with the ability to consume less power can be brought to existence with fine tuning i.e., Background Subtractor MOG2. Moreover, this analysis can be done for a Traffic Surveillance System in real-time by building a prototype. Also, more tuning can be done for optimizing these algorithms for producing higher efficiency.

# Chapter 6: References

1. S. Choudhury, S. P. Chattopadhyay and T. K. Hazra, "Vehicle detection and counting using haar feature-based classifier," 2017 8th Annual Industrial Automation and Electromechanical Engineering Conference (IEMECON), 2017, pp. 106-109.
2. R. M. Waregaonkar, S. M. Deokar, S. S. Patankar and J. V. Kulkarni, "Development of prototype for vehicle speed measurement," 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2017, pp. 945-949.
3. S. K. Mankani, N. S. Kumar, P. R. Dongrekar, S. Sajjanar, Mohana and H. V. R. Aradhya, "Real-time implementation of object detection and tracking on DSP for video surveillance applications," 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2016, pp. 1965-1969.
4. A. P. Kulkarni and V. P. Baligar, "Real Time Vehicle Detection, Tracking and Counting Using Raspberry-Pi," 2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), 2020, pp. 603- 607.
5. C. R. Rashmi and C. P. Shantala, "Vehicle Density Analysis and Classification using YOLOv3 for Smart Cities," 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA), 2020, pp. 980-986.
6. V. S. Sindhu, "Vehicle Identification from Traffic Video Surveillance Using YOLOv4," 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), 2021, pp. 1768-1775.
7. F. H. Shubho, F. Iftekhar, E. Hossain and S. Siddique, "Real-time traffic monitoring and traffic offense detection using YOLOv4 and OpenCV DNN," TENCON 2021 - 2021 IEEE Region 10 Conference (TENCON), 2021, pp. 46- 51.
8. Chato, D. J. M. Chipantasi, N. Velasco, S. Rea, V. Hallo and P. Constante, "Image processing and artificial neural network for counting people inside public transport," 2018 IEEE Third Ecuador Technical Chapters Meeting (ETCM), 2018, pp. 1-5.
9. https://xailient.com/real-time-vehicle-detection-with-mobilenet-ssd-andxailient/
10. C. C. Paglinawan, A. N. Yumang, L. C. M. Andrada, E. C. Garcia and J. M. F. Hernandez, "Optimization of Vehicle Speed Calculation on Raspberry Pi Using Sparse Random Projection," 2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), 2018, pp. 1-6.
11. L. C. Pico and D. S. Benítez, "A Low-Cost Real-Time Embedded Vehicle Counting and Classification System for Traffic Management Applications," 2018 IEEE Colombian Conference on Communications and Computing (COLCOM), 2018, pp. 1-6.
12. D. Dong, X. Li and X. Sun, "A Vision-Based Method for Improving the Safety of Self-Driving," 2018 12th International Conference on Reliability, Maintainability, and Safety (ICRMS), 2018, pp. 167-171.
13. S. Raviteja and R. Shanmugha sundaram, "Advanced Driver Assitance System (ADAS)," 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), 2018, pp. 737-740.
14. H. Yuliandoko, M. D. Ayatullah and F. E. Purwita, "Automatic Vehicle Counting Using Raspberry pi and Background Subtractions Method in The Sidoarjo Toll Road," 2019 2nd International Conference of Computer and Informatics Engineering (IC2IE), 2019, pp. 83-86.
15. D. R. Ignatius Moses Setiadi, R. R. Fratama, N. D. Ayu Partiningsih, E. H. Rachmawanto, C. A. Sari and P. N. Andono, "Real-Time Multiple Vehicle Counter using Background Subtraction for Traffic Monitoring System," 2019 International Seminar on Application for Technology of Information and Communication (iSemantic), 2019, pp. 1-5.
16. A. Ghosh, M. S. Sabuj, H. H. Sonet, S. Shatabda and D. M. Farid, "An Adaptive Video-based Vehicle Detection, Classification, Counting, and Speedmeasurement System for Real-time Traffic Data Collection," 2019 IEEE Region 10 Symposium (TENSYMP), 2019, pp. 541-546.
17. L. Koraqi and F. Idrizi, "“Detection, identification and tracking of objects during the motion”," 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), 2019, pp. 1-6.
18. N. Jain, S. Yerragolla, T. Guha and Mohana, "Performance Analysis of Object Detection and Tracking Algorithms for Traffic Surveillance Applications using Neural Networks," 2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2019, pp. 690-696.
19. W. Hou, D. Xia and H. Jung, "Video Road vehicle detection and tracking based on OpenCV," 2020 International Conference on Information Science and Education (ICISE-IE), 2020, pp. 315-318.
20. A. C. Cormoş, R. Andrei Gheorghiu, V. A. STAN and I. Spirea Dănăilă, "Use of TensorFlow and OpenCV to detect vehicles," 2020 12th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), 2020, pp. 1-4.

|  |  |
| --- | --- |
| S/N | Activities |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| 1 | Project Title |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 | Title confirmation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | Literature Review |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 | Requirement Gathering |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5 | Preparation of progressive report |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6 | Progressive report submission |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 7 | Mid semester presentation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 8 | System design |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 9 | Writing of end semester report |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 10 | Submission of end semester report |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 11 | Final presentation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |