Semi-Supervised Vehicle Detection, Classification and Counting System Using a Background Subtraction-based Algorithm

An Undergraduate Thesis

Presented to the Faculty of the

Electrical and Electronics Engineering Department

College of Engineering

Mindanao State University - General Santos City

In Partial Fulfillment of the

Requirements for the Degree of

Bachelor of Science in Electronics Engineering

Myles Antonette R. Solatorio

Pauline Joy B. Acebedo

Researchers

Dr. Cristina P. Dadula, PECE

Adviser

January 2022

## **Abstract**

The paper proposes a vehicle detection, classification and counting system that can be used for traffic flow surveillance and is capable of running in real-time. The system uses a background subtraction-based detection algorithm. The system will use footage collected from CCTV cameras installed along roads to monitor traffic. The proposed algorithm also accounts for the segmentation of the shadows casted by vehicles. Homography-based perspective manipulation will be used to calibrate the perspective effect on the footage caused by the cameras’ different positions. The vehicles will be classified according to their dimensions and will be counted according to their class. The data will then be shown real-time in the video-feed and in a dashboard.

*Index Terms-*vehicle detection, vehicle classification, background subtraction, traffic statistics, perspective correction

**Table of Contents**

[**Chapter 1**](#_jbospu3b89as) **5**

[1.1 Background of the Study](#_72qxsvem5xqy) 5

[1.2 Statement of the Problem](#_xhz016dlqax3) 7

[1.3 Objectives](#_8zukohuhm6ld) 8

[1.3.1 General Objectives](#_8k56cuueko42) 8

[1.3.2 Specific Objectives](#_8skjrgwrw8bs) 8

[1.4 Significance of the Study](#_a8zlgdy7r117) 9

[1.5 Assumptions, Scope and Delimitations](#_8dz853j5n3qm) 9

[1.5.1 Assumptions](#_kz5elapetoqe) 9

[1.5.2 Scope](#_1zfhm0dqhyvc) 9

[1.5.3 Delimitations](#_lcafqq265puk) 10

[1.6 Theoretical Background](#_f0xzxyg0hct6) 10

[Background Subtraction](#_8ybyvfj663hc) 10

[Contours](#_t3qq4bfhr5yw) 11

[Perspective Manipulation](#_navhacslf41t) 12

[**Chapter 2**](#_km4dtqthcvm1) **14**

[2.1 Traffic Management in the Philippines](#_b2qaib7p00au) 14

[2.2 OpenCV](#_jhgcr48hsmnj) 16

[2.3 Background Subtraction](#_f3ir4zcfcj6x) 16

[2.3.1 Background Modeling Methods](#_r80twpqixcip) 17

[2.3.1.1 GMG](#_zfq04cftl6tz) 17

[2.3.1.2 Mixture of Gaussians (MOG)](#_vy4bg3rj7iar) 17

[2.4.1.3 Mixture of Gaussians 2 (MOG2)](#_g6o5vws0g87m) 17

[2.4 Intelligent Traffic Systems with Computer Vision](#_p9leln71nt7e) 18

[2.5 Perspective Manipulation](#_lnz0d8mr8tlo) 21

[2.5.1 Inverse Perspective](#_z4yttnw3ucd1) 21

[2.5.2 Homography](#_546pnemaimjs) 21

[**Chapter 3**](#_2amdg2kvbkdr) **24**

[3.1 System Specifications](#_5y6cyoo73wsh) 24

[3.2 Supervised Initialization](#_68xkqhacv0fn) 26

[3.2.1 Selection of Region of Interest (ROI)](#_1dmsswkpf24t) 26

[3.2.2 Perspective Transformation](#_hmpbwawy8ju6) 26

[3.3 Detection](#_uq6pzpnojlvk) 27

[3.3.1 Background Modeling](#_krot2kwh40uc) 27

[3.3.2 Foreground Extraction](#_j1e8glxzrx8) 27

[3.3.3 Contour Detection](#_erl4e45hy7rn) 28

[3.4 Virtual Line](#_l70dp4m9cgff) 28

[3.4.1 Vehicle Counting](#_yoy1ttpzftep) 28

[3.4.2 Contour Analysis](#_s2x3iqwnnr41) 28

[3.4.3 Vehicle Classification](#_7w2b9cmho0z1) 29

[3.5 Real-time Data Dashboard](#_z2xn273zzbgm) 29

[**References**](#_o03t6trzuc3w) **30**

[**Appendices**](#_58uosi2dnpt6) **33**

[Tables](#_hugq734kx4cm) 33

[Table 1](#_r5l2gjk7itkx) 33

[Table 2](#_9882ptu1kcp7) 35

[Table 3](#_qlidpw4j8uel) 36

## Chapter 1

**INTRODUCTION**

**Contents**

[1.1 Background of the Study](#_72qxsvem5xqy) [5](#_72qxsvem5xqy)

[1.2 Statement of the Problem](#_xhz016dlqax3) [7](#_xhz016dlqax3)

[1.3 Objectives](#_8zukohuhm6ld) [8](#_8zukohuhm6ld)

[1.3.1 General Objectives](#_8k56cuueko42) [8](#_8k56cuueko42)

[1.3.2 Specific Objectives](#_8skjrgwrw8bs) [8](#_8skjrgwrw8bs)

[1.4 Significance of the Study](#_a8zlgdy7r117) [9](#_a8zlgdy7r117)

[1.5 Assumptions, Scope and Delimitations](#_8dz853j5n3qm) [9](#_8dz853j5n3qm)

[1.5.1 Assumptions](#_kz5elapetoqe) [9](#_kz5elapetoqe)

[1.5.2 Scope](#_1zfhm0dqhyvc) [9](#_1zfhm0dqhyvc)

[1.5.3 Delimitations](#_lcafqq265puk) [10](#_lcafqq265puk)

[1.6 Theoretical Background](#_f0xzxyg0hct6) [10](#_f0xzxyg0hct6)

[Background Subtraction](#_8ybyvfj663hc) [10](#_8ybyvfj663hc)

[Contours](#_t3qq4bfhr5yw) [11](#_t3qq4bfhr5yw)

[Perspective Manipulation](#_navhacslf41t) [12](#_navhacslf41t)

### 1.1 Background of the Study

According to the Transport Sector Assessment, Strategy, and Road Map report of ADB(2012), transport services in urban areas consist mainly of public utility vehicles, taxis, tricycles, and pedicabs that are privately owned and operated. Additionally, 6.6 million vehicles in the country were motorcycles which can be credited to the vehicle’s portability. However, motorcycle users are vulnerable to road crashes and contribute significantly to traffic congestion. The most recent report of Numbeo published last January 2021 shows that the country scored 192.88 for the traffic index and 243.20 for the inefficiency index. Road congestion costs our economy a significant portion of the gross domestic product (ADB, 2012). High levels of traffic congestion and density cause huge delays in transport of goods and delivery of services, and increased fuel wastage. The long time spent by vehicles on the road also contributes to the increase in emission of pollutants. Thus, minimizing the time spent by vehicles on the road can help lessen their carbon footprint. Plenty of Filipinos spend a few hours each day stuck in traffic. The Philippines is always among the top countries with longest traffic wait times. Commuters lose valuable time that can instead be spent on productive activities. Traffic costs people income opportunities. Filipinos can do away with delayed transactions and the mental and physical fatigue caused by traffic.

In the last decade or so, authorities in the Philippines have increasingly depended on CCTV cameras to monitor real-time traffic especially in metropolitan areas. However, the utilization of these systems leaves so much to be desired. These camera systems are manually monitored by operators and are usually only used for monitoring accidents and congestion. Quantifying the information provided by these systems would be helpful in optimizing traffic mobility and improving traffic flow. The apparent lack of comprehensive traffic flow and network analysis in the Philippines, which is crucial for urban planning and road infrastructure management is rather concerning. Cities in general in the Philippines do not have adequate knowledge on use of information and technology in traffic management (Lidasan et al, 2009).

By incorporating intelligent transport technologies in Philippine roads, we can monitor these figures and assess the conditions of traffic in the country. However, one factor that hinders the development of intelligent transportation systems in the Philippines is that these often require enormous resources. Other systems that use sensors, for instance, are often expensive and require complicated installation methods. Thus, it is also important to develop affordable, and easily adaptable and deployable systems. The use of computer vision in traffic monitoring has been steadily gaining traction over the years. These advancements have shown remarkable results and achievements. Various researches have shown that detection and classification algorithms using background subtraction techniques perform well. Thus, it has become one of the most widely used detection methods.

However, while background subtraction is efficient for vehicle detection, Xia et al. (2016) discussed that the performance of the model can suffer in a complex environment. The variation in lighting, presence of leaves or wind, and even a slight movement of the camera can affect the effectiveness. Moreover, identifying shadows casted by vehicles is a huge challenge (Yang, 2017). Since the shadow also moves with the vehicle, the algorithm can detect it as a part of the vehicle. As a result, the shapes might be distorted. It can also cause more issues, especially when long shadows are cast, due to the overlap with other shadows or other objects. There are many more factors that could limit the capabilities of these vehicle detection systems such as glare, light from vehicle headlights, weather conditions and varying shooting angles of different cameras among others (Alpatov, 2018).

Thus, the researchers have proposed an improved background subtraction-based vehicle detection, counting and classification system. The project aims to build upon existing works and methods, and implement several techniques to resolve the issues presented. This will provide information on the make-up of vehicles traversing the roads such as what types of vehicles, how many are there per size class, and what are the trends in these numbers throughout the day.

### 1.2 Statement of the Problem

Computer vision-based systems in general are limited by spatial constraints, imperfect image processing techniques and insufficient computational resources.

Maqbool et al (2018) and Alpatov et al. (2018) pointed out several limitations in implementing a background subtraction-based system for vehicle detection. One of which is the shadows from vehicles caused by the position of the sun. Since they move with the vehicle, they can be detected as part of the foreground. The system can also have difficulty detecting fast moving vehicles (Maqbool et al, 2018). In addition, the camera angle can also affect the system (Memon et al., 2018). Thus, it’s been recommended to use calibration techniques to improve the efficiency. Memon et al. (2018) also pointed out in their study that the accuracy of their system depends on the judgment of the human supervisor that defines the imaginary line. In their system, the user had to manually define the position of the imaginary line where the centroid of the vehicle crosses to be counted. Aside from the issues discussed above, other factors include changes in scene illumination caused by background interferences, and different configuration of observed road intersections and road sections (Alpatov, 2018).

### 1.3 Objectives

#### 1.3.1 General Objectives

The main objective of this project is to create a computer vision-based vehicle detection and classification system using a background subtraction-based algorithm.

#### 1.3.2 Specific Objectives

Specifically, the researchers aim to:

* Use OpenCV to employ image processing techniques.
* Apply homography to calibrate the perspective across different cameras.
* Estimate the vehicles’ dimensions.
* Classify vehicles according to sizes.
* Count vehicles according to class.
* Deploy the system in roads with varying traffic volumes.
* Test the system in several types of environments, lighting conditions, and weather conditions.
* Publish a dashboard/web app that will show the real time data (timelines, etc.)

### 1.4 Significance of the Study

If the project is deployed successfully, especially in highly urbanized areas, it can help local governments to employ efficient routes, implement better policies, and plan effective road infrastructure projects. Efficient traffic networks also translate to safer commute and roads. With better road networks, we can significantly reduce the risk of accidents and hazards in roads. This could also provide road users with vital information that would help them understand traffic flow and interactions better, and consequently, navigate the roads better and more safely. By acquiring these statistics, we can also keep a historical database of these figures to analyze the trends in traffic conditions over time. Using the insights that we can gain from these analyses, we can create better road infrastructure projects and plan more efficient road networks.

### 1.5 Assumptions, Scope and Delimitations

#### 1.5.1 Assumptions

1. Once set up, the cameras will be stationary.
2. Initialization will only be run once.
3. The cameras that will be used should not produce distortion.
4. Ideally, the cameras should be positioned and configured as uniformly as possible to minimize perspective effect.

#### 1.5.2 Scope

1. The project will be conducted in General Santos City.
2. The proposed timeline of the conduct of the study will be from February to June.
3. The project will only use computer vision and image processing techniques offered by OpenCV natively.
4. Machine learning methods will not be used.

#### 1.5.3 Delimitations

1. Due to the inconsistencies in results that might be introduced by different camera specifications and positions, the project will only consider overhead cameras.
2. Since the project will use a dimension-based classification algorithm, separate cameras will be used for each lane as the system might have trouble detecting small objects.
3. The system will be deployed only on four-lane roads.

### 1.6 Theoretical Background

#### 1.6.1 Background Subtraction

Background subtraction is one of the most popular methods used for object detection. This method separates the moving objects, called the foreground, from the static information, called the background (Bouwmans, 2012). This is done by modeling the background through isolating parts of the image that do not include any moving object. Aside from being easy to set up, it is also resource-efficient. The general algorithm of most background subtraction techniques can be broken down into three phases (Yang, 2017).

Background initialization: In this phase, a background model is built by referencing a certain number of frames - usually extracted from the beginning of the footage (Bouwmans, 2012). There are various approaches to choose from such as fuzzy, statistical, etc.

Background maintenance: Video compositions, especially of complex scenes, are dynamic. Therefore, the background composition can also change. When this happens, it is important for the algorithm to be able to adapt and update the background model accordingly. This is a crucial part of the algorithm as the foreground extraction will fail if the background model is compromised.

The first two phases are part of the sub-process called background modeling (OpenCV Documentation, n.d.).

Foreground detection: The succeeding frames are then compared with the background model to compute the foreground. Each pixel of the frame is classified and divided into the background and foreground. Then, a mask is made and the foreground is subtracted from the whole image.

#### 1.6.2 Contours

According to the official documentation of OpenCV, a contour is “a curve joining all the continuous points (along the boundary), having the same color or intensity.” The contours are a useful tool for shape analysis and object detection and recognition. Contours can be used to analyze shapes, and to detect and recognize objects. Ideally, images are binarized before determining the contours to produce the best results. A threshold or canny edge detection can be applied.

The different features of a contour include moments, area, perimeter and approximation among others (OpenCV Documentation, n.d.). The moments of a contour can be used to calculate other features such as center of mass and area. OpenCV also offers several functions natively that automatically determines features such as cv.contourArea() to get the area or cv.arcLength() to get the perimeter. Aside from features, contours also have various properties such as aspect ratio, extent, solidity, etc. Aspect ratio can be thought of as the inverse of the slope and is determined by dividing the bounding rectangle’s width by its height.

#### 1.6.3 Perspective Manipulation

Perspective manipulation is generally done by determining points from the target image and matching its position with the corresponding points on the desired plane of projection.

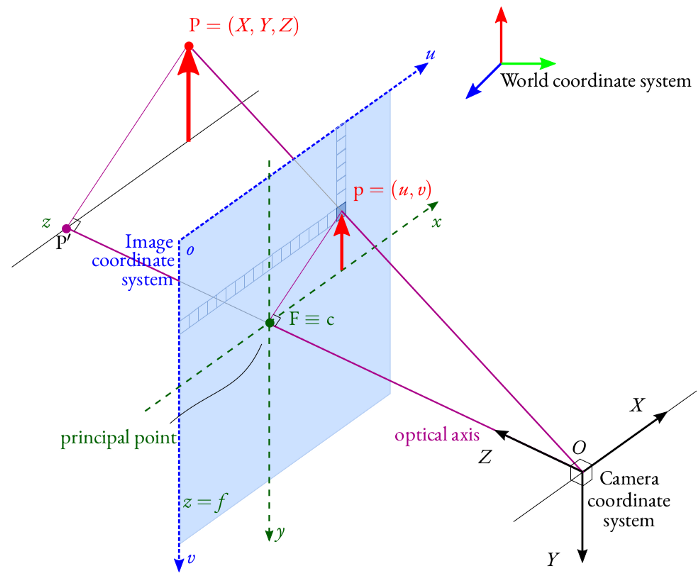


Fig. 1.1 World to Camera Projection (Source: OpenCV Documentation)

Using trigonometry, an object’s coordinates in the 3d world can be converted into a 2D image plane (Vijayrania, 2020). To obtain the coordinates in the image plane, the 3D coordinates are multiplied by a 3x4 matrix called the perspective matrix. Using this matrix, the point (x, y, z) in the 3D world coordinate system is projected to the point (u, v) on the 2D camera image plane.

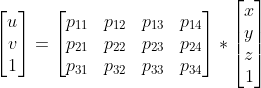


Fig. 1.2 3D points to 2D conversion (Source: Nilesh Vijayrania, 2020, *Camera Image Perspective Transformation to different plane using OpenCV*)

## Chapter 2

**LITERATURE REVIEW**

**Contents**

[2.1 Traffic Management in the Philippines](#_b2qaib7p00au) [14](#_b2qaib7p00au)

[2.2 OpenCV](#_jhgcr48hsmnj) [16](#_jhgcr48hsmnj)

[2.3 Background Subtraction](#_f3ir4zcfcj6x) [16](#_f3ir4zcfcj6x)

[2.3.1 Background Modeling Methods](#_r80twpqixcip) [17](#_r80twpqixcip)

[2.3.1.1 GMG](#_zfq04cftl6tz) [17](#_zfq04cftl6tz)

[2.3.1.2 Mixture of Gaussians (MOG)](#_vy4bg3rj7iar) [17](#_vy4bg3rj7iar)

[2.4.1.3 Mixture of Gaussians 2 (MOG2)](#_g6o5vws0g87m) [17](#_g6o5vws0g87m)

[2.4 Intelligent Traffic Systems with Computer Vision](#_p9leln71nt7e) [18](#_p9leln71nt7e)

[2.5 Perspective Manipulation](#_lnz0d8mr8tlo) [21](#_lnz0d8mr8tlo)

[2.5.1 Inverse Perspective](#_z4yttnw3ucd1) [21](#_z4yttnw3ucd1)

[2.5.2 Homography](#_546pnemaimjs) [21](#_546pnemaimjs)

### 2.1 Traffic Management in the Philippines

Lidasan et al (2009) published a study titled “A Needs Assessment of Transport Planning and Traffic Management of Local Cities: The Case of the Philippines.” The study assessed the technical capabilities of various cities on transport planning and traffic management. They attempted to collect data from all cities in the Philippines. However, not all cities participated. Cities were divided into three categories: large, mid-size and small. They also gathered information on the basic tools used for transport planning and traffic management. The paper identified five key areas that need to be addressed. The following are:

(i) Management and planning of public transportation operations;

(ii) Management of automobiles and trucks in cities;

(iii) Management of traffic flow at intersections;

(iv) Planning of transport networks; and

(v) Project funding.

Traffic congestion was ranked as the sixth overall among the top issues being faced by cities of various sizes. Moreover, it ranked second for large cities and fourth for mid-size cities.

The study assessed the know-how of the participating cities in Transport Planning and Traffic Management. Specifically, the participating cities self-assessed their Level of Knowledge in Transport Planning Concepts. The following are some of the notable concepts and respective scores:

| Transport Planning Concept | Small | Mid-size | Large | All |
| --- | --- | --- | --- | --- |
| Planning, execution, and analysis of transport surveys to generate data for planning | 1.63 | 1.43 | 1.39 | 1.48 |
| Forecasting transport demand and analyzing how it impacts the present network | 1.41 | 1.36 | 1.30 | 1.36 |

Source: TPTM Self-assessment Survey, 2008. Note: 1=limited knowledge, 2=some knowledge, 3=good knowledge. Figures are averages of cities in the group.

The cities’ Level of Knowledge in Traffic Engineering and Management Concepts was also assessed. Some of the relevant concepts and respective scores are:

| Traffic Engineering and Management Concept | Small | Mid-size | Large | All |
| --- | --- | --- | --- | --- |
| Analysis of traffic accidents and concepts to enhance road safety | 2.00 | 1.74 | 1.71 | 1.81 |
| Concepts in traffic flow management | 2.20 | 1.85 | 1.83 | 1.95 |
| Use of information and technology in traffic management | 1.83 | 1.46 | 1.52 | 1.59 |

Source: TPTM Self-assessment Survey, 2008. Note: 1=limited knowledge, 2=some knowledge, 3=good knowledge. Figures are averages of cities in the group.

The results of the needs assessment of the paper showed the equal importance of coming up with academic research in transport and traffic management.

### 2.2 OpenCV

OpenCV is a computer vision library that includes hundreds of ready-to-use imaging and vision functions (Garcı́a et al., 2015). It offers a mix of low-level image-processing functions and high-level algorithms that can be used for plenty of applications such as face detection, pedestrian detection, feature matching, and tracking (Pulli et al., 2012). The basic OpenCV architecture was designed mostly with CPUs in mind. However, OpenCV also supports both CPU and GPU processing. Developers can also try different combinations of these two processing modules. The ideal approach when developing applications with OpenCV would be to develop it using the CPU module and then accelerate it with the GPU module.

A paper by Guennouni et al (2014) proposed an OpenCV-based solution for multiple object detection. The study also compared the performances in a regular platform and an embedded device. Results showed that object detection can be deployed in different platforms as needed. This system is appropriate to adapt for surveillance cameras with object detection notification. The system can also be trained for any type of object to be detected for different situations. The researchers then recommended enhancing the embedded platform performance. To achieve this, several processors can be used to run separate tasks simultaneously in order to enhance performance and response time.

### 2.3 Background Subtraction

Footages consist of two parts: the foreground and background. The foreground is made up of the so-called “objects of interest”. The remaining objects that are not of interest are collectively regarded as the background. By using segmentation and tracking, these two areas can be differentiated from each other to isolate the “objects of interest” automatically. There are several different approaches that can be implemented to achieve this.

#### 2.3.1 Background Modeling Methods

##### 2.3.1.1 GMG

GMG is an algorithm that uses a combination of Bayesian Inference and statistical background image estimation. It extracts the first several frames of the footage. Weighted values for each pixel are accumulated in the first step. These values depend on how long a color stays on that position. The colors that stay static for a certain amount of time are then inferred to be part of the background. To adapt with illumination changes, newer estimates are given heavier weights. Several morphological filtering operations are also applied to reduce noise.

##### 2.3.1.2 Mixture of Gaussians (MOG)

This algorithm is a Gaussian Mixture-based segmentation algorithm. This method uses a mixture of K Gaussian distributions within 3 to 5 to model the background. These different distributions each correspond to a different background/foreground color. The weight of the distribution is proportional to the duration that the color stays on the pixel. The probable background colors are then determined by analyzing the weight of the pixel distribution. If the weight of the pixel distribution is high, it denotes that the pixel remains static longer and is then classified as background. Pixels with low distribution weights are classified as foreground. It also has the capability to detect shadows.

##### 2.4.1.3 Mixture of Gaussians 2 (MOG2)

MOG2, which is an iteration of MOG, is also a Gaussian Mixture-based segmentation algorithm. What sets it apart is that an appropriate number of gaussian distributions for each pixel is selected instead of just a fixed amount K. Because of this, it can create a better representation of the color complexity in each frame. Just like the previous algorithm, it can also detect shadows very well.

A study by Marcomini and Cunha (2018) showed that MOG2 has an overall better performance boasting superior precision rate and lower processing times. This paper compared three background-modeling methods available in OpenCV - GMG, MOG, and MOG2, in their Python implementation and assessed their performance in vehicle segmentation. Results showed that the three methods showed very similar accuracies which were mostly concentrated at 100% with one outlier. MOG and MOG2 showed significantly better results for the precision with near 100% ratings, ie, with pixels correctly classified as foreground. MOG2 recorded the fastest processing time of the three with 10,000 frames processed in 150 seconds, resulting in a rate of 64 frames per second. It showed that MOG2 performance results were 3 times better than MOG, and 10 times better then GMG.

### 2.4 Intelligent Traffic Systems with Computer Vision

Classical approaches such as the background subtraction-based Gaussian Mixture Model are widely used in these types of traffic monitoring systems (Yang, 2017). The advantages of this algorithm include fast and accurate detection. Background subtraction-based detection methods localize objects by analyzing their motion (Chandan et al., 2018).

In a paper by Maqbool et al. (2018), a GMM was used to compute the variance, covariance and mean of every pixel in a frame. As a new frame arrives, these parameters are calculated again. The foreground is determined when the difference between the values for the two frames is larger than the product of actual value and standard deviation. In this method, a model for each pixel is created using multiple, adaptive Gaussian distributions (Yang, 2017).

In a project by Memon et al (2018), a video-based system was created using OpenCV for detection, counting and classification of vehicles. The system uses a supervised algorithm to categorize vehicles into three classes: LTV, HTV or MTV. The algorithm uses background subtraction to extract the vehicles for detection. The OpenCV algorithm used in the implementation of the system is BackgroundSubtractorMOG2. The algorithm uses an automated approach and selects an appropriate number of Gaussian mixtures for the pixel. The authors also pointed out that it is better at handling illumination changes in the scene. After the foreground is extracted, the contours are determined using cv2.findContours(). For the classification, the contour properties are extracted and compared with already assumed values to determine whether the vehicle is LTV, HTV or MTV. The researchers also tested another method of classification which uses BoF along with SVM algorithm. The results showed that the Contour Comparison method performed significantly better. However, the system needs human supervision during the determination of Region of Interest. Thus, the researchers recommended the creation of an unsupervised alternative to the system.

Li et al (2018) presented a solution to moving vehicle detection, tracking and counting using an adaptive background subtraction technology. The system was implemented using Visual C++ code with OpenCV. The footage is processed using binarization to subtract the foreground. Unlike other similar systems, the researchers proposed a method to also remove shadows of the vehicles to create a more efficient system. To count the vehicle, a technique called “Virtual Detector” was used. Contrary to similar systems, each lane is assigned a rectangular region of interest (ROI) as the virtual detector. Vehicles are detected by monitoring changes in the area of the virtual detector. Another method was also proposed to detect movement, count and trajectory. In this method, blob tracking technology was used to track and count the vehicles moving in the field of view. The results of the project showed that the accuracy rate can be up to 97.1% for the virtual detector method, and 98.4% for the blob tracking method.

Another paper by Seenouvong (2016) proposed a similar method of vehicle detection and counting system. Like other algorithms previously mentioned, the system also used background subtraction and a defined region of interest. However, a single region of interest and virtual detection zone were defined for multiple lanes. The region of interest was also divided into five zones and each zone was processed differently. This was implemented to avoid false positives in the detection of vehicles. The system was tested with seven input footages. Results showed that 646 out of 667 vehicles were detected, thus demonstrating an accuracy of 96.85%. The accuracy for each input footage also varied only from 95% to 99% which shows the great reliability of the system.

### 2.5 Perspective Manipulation

There are various differences in camera structures and specifications. The camera angle for instance affects the perspective of the captured image. Due to this, results of image processing algorithms for different footages can vary. According to Yaghoobi Ershadi (2020), the removal of this perspective effect provides many advantages in road scenarios, such as better detection of vehicles and lanes in the scene. Thus, it is an important aspect of the system that has to be considered. As such, we can employ perspective manipulation techniques

#### 2.5.1 Inverse Perspective

This technique is used to generate a bird’s eye view scene of an image. If the plane the target point (x, y) is level with the reference point of the system (such it is at z=0 of the system), it is known that the points in the floor plane will have coordinates <x, y, 0> the effect of the third column of the perspective matrix can then be disregarded (Vijayrania, 2020). Therefore, the perspective matrix P is reduced to a 3x3 matrix and hence becomes invertible. To get the corresponding (x, y) from (u, v), only the inverse of the 3x3 perspective matrix P is needed to determine the corresponding point on the floor plane from the image plane.

#### 2.5.2 Homography

Homography is used to map one image to the other by relating the transformation to each other. (Chellappa et al, 2005). If given the point <u, v> in one plane and <x, y> in another plane, then the transformation from <u, v> to <x, y> could be written as.

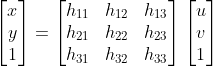


Fig. 2.1 Homography Matrix (Source: Nilesh Vijayrania, 2020, *Camera Image Perspective Transformation to different plane using OpenCV*)

where H is the 3x3 homography matrix. To get the homography matrix, at least four points can be used to specify it. The points in one plane and its corresponding mapping point in the other plane have to be specified. Given the perspective matrix P shown above, the 4 corresponding points between any 3d world plane (assuming z is fixed) and 2d plane can be determined using the matrix multiplication defined earlier.

According to Nieto et al (2018), properties of the homographies are used to change the position of the objects detected from the plane of each camera to a common plane. The homography matrix was obtained using 4 points from each camera viewpoint and each point correspondence in an image extracted from a top view.

For Perspective transformation from the image plane to a fixed plane in world coordinates, either of the two methods can be used. However, for any other arbitrary plane, the homography matrix should be used.

Unzueta conducted a project titled “Adaptive Multicue Background Subtraction for Robust Vehicle Counting and Classification” in 2012. The perspective correction in this study was achieved by rectifying the road plane using a planar homography. The four points were selected manually in the system. Based on elements in the environment with known length, the relation between pixel size and real world distances (in the longitudinal dimension of the road) was determined manually. The lane lines were also marked to detect the lanes in which the vehicle projections lie.

## Chapter 3

**METHODOLOGY**

**Contents**

[3.1 System Specifications](#_5y6cyoo73wsh) [24](#_5y6cyoo73wsh)

[3.2 Supervised Initialization](#_68xkqhacv0fn) [26](#_68xkqhacv0fn)

[3.2.1 Selection of Region of Interest (ROI)](#_1dmsswkpf24t) [26](#_1dmsswkpf24t)

[3.2.2 Perspective Transformation](#_hmpbwawy8ju6) [26](#_hmpbwawy8ju6)

[3.3 Detection](#_uq6pzpnojlvk) [27](#_uq6pzpnojlvk)

[3.3.1 Background Modeling](#_krot2kwh40uc) [27](#_krot2kwh40uc)

[3.3.2 Foreground Extraction](#_j1e8glxzrx8) [27](#_j1e8glxzrx8)

[3.3.3 Contour Detection](#_erl4e45hy7rn) [28](#_erl4e45hy7rn)

[3.4 Virtual Line](#_l70dp4m9cgff) [28](#_l70dp4m9cgff)

[3.4.1 Vehicle Counting](#_yoy1ttpzftep) [28](#_yoy1ttpzftep)

[3.4.2 Contour Analysis](#_s2x3iqwnnr41) [28](#_s2x3iqwnnr41)

[3.4.3 Vehicle Classification](#_7w2b9cmho0z1) [29](#_7w2b9cmho0z1)

[3.5 Real-time Data Dashboard](#_z2xn273zzbgm) [29](#_z2xn273zzbgm)

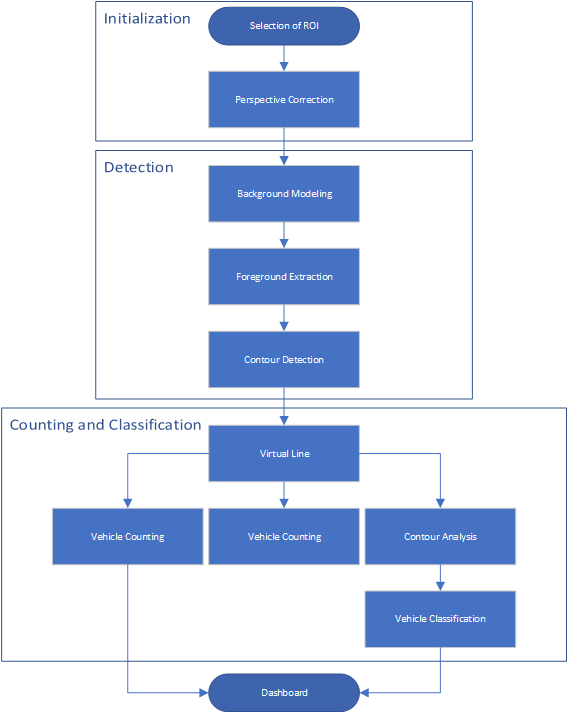


Fig. 3.1 Methodology Flowchart

### 3.1 System Specifications

The system will be implemented using the Python programming language along with the OpenCV library. The footage to be used will be collected from overhead road surveillance cameras. The system will be tested both with recorded and real-time video feeds. The video feed will just be transmitted remotely from the camera to the computing system.

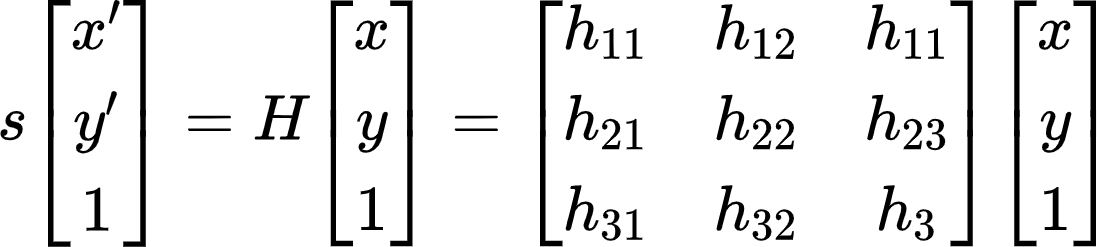
### 3.2 Supervised Initialization

#### 3.2.1 Selection of Region of Interest (ROI)

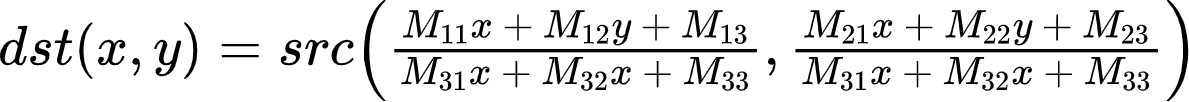
To save on resources, we can crop the footage and extract only the particular region which contains the objects of interest. This will remove the unnecessary parts of the image and will reduce the chances of false positives in the tests. This is a supervised step and the user will have to define the borders of the region of interest manually. After initializing the system, the user will select two points on the video - the upper left corner and the lower right corner. Using the coordinates of the two points (, ) and (, ), we can get the four corners of the ROI: (, ), (, ), (, ), (, ). After the selection, the feed will be updated and will show the cropped video.

#### 3.2.2 Perspective Transformation

The differences between the lenses and the angles of the cameras that will be used can affect the accuracy of the classification algorithm which is based on the vehicle dimensions. Thus, the different perspectives might introduce inconsistencies in the system. To deal with this, perspective correction will be executed to keep things consistent. The perspective correction is achieved using homography so that images can be moved to an arbitrary plane. This can be done in OpenCV by using the function cv2.findHomography(). The planar homography relates the transformation between two planes (up to a scale factor):

 (eq. 1)

This is also a supervised step. The user will have to define four points corresponding to the corners of the flat vertical plane of the road. These four points will also be used as a reference for the virtual line. After coordinates of the source corners transformed using homography, the image is warped into the desired perspective using cv2.warpPerspective(). The function warpPerspective transforms the source image using the specified matrix:

 (eq. 2)

docs.opencv.org

### 3.3 Detection

#### 3.3.1 Background Modeling

The background modeling method that will be used for this system is MOG2. OpenCV natively provides this algorithm with BackgroundSubtractorMOG2 and can be applied using cv2.createBackgroundSubtractorMOG2(). The function provides the option to detect shadows or not. The parameters are set to detect shadows by default. This algorithm is also good at dealing with lighting changes in the environment. It is also a GMM-based method which requires less storage capacity as they store significantly lesser preceding frames.

#### 3.3.2 Foreground Extraction

The pixels are binarized. The foreground mask is calculated. The white pixels represent the foreground and black pixels consist of the background. The white pixels from the binarized image are extracted and used as the mask. It is then used to subtract between the current frame and the background model.

#### 3.3.3 Contour Detection

The closed shape formed by a cluster of pixels that represent an object is regarded as the contour. This step will detect the borders of the isolated contours of each vehicle. The contours will be determined using the OpenCV built-in function cv.findContours(). A rectangular bounding box will be put around the borders to enclose the vehicle. The bounding box will stay around the vehicle until it reaches the virtual line. The moments of the contour will be calculated to determine the centroid and will be represented by a dot.

### 3.4 Virtual Line

Depending on the four points defined during the perspective manipulation step, a virtual line will be drawn and will serve as the threshold point that will trigger the counting of the vehicle.

#### 3.4.1 Vehicle Counting

Once the centroid of the vehicle crosses the virtual line, the total vehicle count will be incremented. If the centroid is detected in the range of the defined virtual line, the system will proceed to counting. Otherwise, the system will be redirected to the start of the contour detection stage.

#### 3.4.2 Contour Analysis

The classification will also start at this point. The contour properties of the vehicle at the moment will be extracted and analyzed. Specifically, the system will collect the width and height of the bounding box. After this, the bounding box will be removed from the vehicle.

#### 3.4.3 Vehicle Classification

The contours will be classified according to its width and height. The values extracted will be compared with the assumed values. The vehicles will be classified into four classes according to their size: two-wheeled vehicle, small vehicle, medium vehicle, and heavy vehicle. After classification, the count of the class to which the vehicle belongs will also be updated.

### 3.5 Real-time Data Dashboard

Aside from showing the count and classifications in the video feed, the real-time data will be reflected and published real-time in a web application/dashboard. Several visualizations such as bar (reflecting the vehicle count) and time-series (reflecting the vehicle count trend over time) charts will be published in the dashboard.

## References

Bouwmans, T. (2012). Background subtraction for visual surveillance: A fuzzy approach. *Handbook on Soft Computing for Video Surveillance*, *5*, 103–138.

Chandan, G., Jain, A., Jain, H., & Mohana. (2018). Real Time Object Detection and Tracking Using Deep Learning and OpenCV. *2018 International Conference on Inventive Research in Computing Applications (ICIRCA)*, 1305–1308. <https://doi.org/10.1109/ICIRCA.2018.8597266>

Garcı́a, G. B., Suarez, O. D., Aranda, J. L. E., Tercero, J. S., Gracia, I. S., & Enano, N. V. (2015). *Learning image processing with OpenCV*. Packt Publishing Birmingham.

Guennouni, S., Ahaitouf, A., & Mansouri, A. (2014). Multiple object detection using OpenCV on an embedded platform. *2014 Third IEEE International Colloquium in Information Science and Technology (CIST)*, 374–377. <https://doi.org/10.1109/CIST.2014.7016649>

Lidasan, H. S., Espada, I. C., & Leon, M. R. C. D. (2009). A Needs Assessment of Transport Planning and Traffic Management of Local Cities: The Case of the Philippines. *Proceedings of the Eastern Asia Society for Transportation Studies*, *2009*, 3. <https://doi.org/10.11175/eastpro.2009.0.3.0>

Li, D., Liang, B., & Zhang, W. (2014). Real-time moving vehicle detection, tracking, and counting system implemented with OpenCV. *2014 4th IEEE International Conference on Information Science and Technology*, 631–634. <https://doi.org/10.1109/ICIST.2014.6920557>

Marcomini, L., & Cunha, A. L. (2018). A comparison between background modelling methods for vehicle segmentation in highway traffic videos. *arXiv Preprint arXiv:1810.02835*.

Memon, S., Bhatti, S., Thebo, L. A., Talpur, M. M. B., & Memon, M. A. (2018). A video based vehicle detection, counting and classification system. *International Journal of Image, Graphics and Signal Processing*, *11*(9), 34.

Nieto, R. M., Garcı́a-Martı́n, Á., Hauptmann, A. G., & Martı́nez, J. M. (2018). Automatic vacant parking places management system using multicamera vehicle detection. *IEEE Transactions on Intelligent Transportation Systems*, *20*(3), 1069–1080.

Pulli, K., Baksheev, A., Kornyakov, K., & Eruhimov, V. (2012). Real-time computer vision with OpenCV. *Communications of the ACM*, *55*(6), 61–69.

Seenouvong, N., Watchareeruetai, U., Nuthong, C., Khongsomboon, K., & Ohnishi, N. (2016). A computer vision based vehicle detection and counting system. *2016 8th International Conference on Knowledge and Smart Technology (KST)*, 224–227. <https://doi.org/10.1109/KST.2016.7440510>

Unzueta, L., Nieto, M., Cortés, A., Barandiaran, J., Otaegui, O., & Sánchez, P. (2011). Adaptive multicue background subtraction for robust vehicle counting and classification. *IEEE Transactions on Intelligent Transportation Systems*, *13*(2), 527–540.

Vijayrania, N. (2020). *Camera Image Perspective Transformation to different plane using OpenCV*. <https://nilesh0109.medium.com/camera-image-perspective-transformation-to-different-plane-using-opencv-5e389dd56527>

Yaghoobi Ershadi, N., Menéndez, J. M., & Jiménez, D. (2018). Robust vehicle detection in different weather conditions: Using MIPM. *PLoS One*, *13*(3), e0191355.

Yang, Z., & Pun-Cheng, L. S. (2018). Vehicle detection in intelligent transportation systems and its applications under varying environments: A review. *Image and Vision Computing*, *69*, 143–154.

## Tentative Project Timeline

