Real Time Classification of Vehicle Types and Modes using Image Analysis and Data Fusion

Final Report

by

Nathan Huynh[1], Robert Mullen[2], Yohanna Mejia[3]
[1] huynhn@cec.sc.edu, 8037778947, 300 Main St C211
[2] rlm@sc.edu, 8037770524, 300 Main St C118
[3]mejiacru@email.sc.edu, 300 Main St B122

University of South Carolina

March 2019

Center for Connected Multimodal Mobility (C^2M^2)

CLEMSON UNIVERSITY

SC State University | The Citadel, The Military College of South Carolina | University of South Carolina | Benedict College

200 Lowry Hall, Clemson University
Clemson, SC 29634

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated in the interest of information exchange. The report is funded, partially or entirely, by the Center for Connected Multimodal Mobility (C^2M^2) (Tier 1 University Transportation Center) Grant, which is headquartered at Clemson University, Clemson, South Carolina, USA, from the U.S. Department of Transportation's University Transportation Centers Program. However, the U.S. Government assumes no liability for the contents or use thereof. Non-exclusive rights are retained by the U.S. DOT.

ACKNOWLEDGMENT

The authors acknowledge the Research Computing Center at the University of South Carolina for providing the computing resources needed to perform this research project.

Technical Report Documentation Page

1. Report No.	**2. Government Accession No.**	**3. Recipient's Catalog No.**
4. Title and Subtitle Real Time Classification of Vehicle Types and Modes using Image Analysis and Data Fusion		**5. Report Date** March, 2019 **6. Performing Organization Code**
7. Author(s) Nathan Huynh, Ph.D.; ORCID: https://orcid.org/0000-0002-4605-5651 Robert L. Mullen, Ph.D., PE; ORCID: https://orcid.org/0000-0002-4321-5939 Yohanna Mejia, Ph.D.; ORCID: https://orcid.org/0000-0003-3657-4850		**8. Performing Organization Report No.**
9. Performing Organization Name and Address University of South Carolina Department of Civil & Environmental Engineering 300 Main St, Columbia SC 29208		**10. Work Unit No.** **11. Contract or Grant No.** 69A3551747117
12. Sponsoring Agency Name and Address Center for Connected Multimodal Mobility (C^2M^2) Clemson University 200 Lowry Hall Clemson, SC 29634		**13. Type of Report and Period Covered** Final Report January-December 2018 **14. Sponsoring Agency Code**
15. Supplementary Notes NA		

16. Abstract

Several methodologies to count and classify vehicles from video recordings were explored in this research to provide transportation agencies with a safer alternative to collect these data which are needed for traffic engineering and planning studies. In this study, three approaches for detecting and classifying vehicles were investigated: Haar-like features, Local Binary Patterns (LBP), and Convolutional Neural Networks (CNN). Recordings from the South Carolina Department of Transportation (SCDOT) were used for the study. By means of image subtraction and contouring, specific areas of the video stream were extracted as independent images. The resulting database was used for training and testing the Cascade classifiers and the CNN. Results indicate that using a data set consisting of 5,500 car and 2,800 truck images, the CNN achieved higher accuracy (97%) in vehicle classification compared to LBP and Haar cascade.

17. Keywords Computer vision, Cascade Classifiers, Neural networks, Transportation modes, Local Binary Patterns, Haar-like features.	**18. Distribution Statement** No restrictions.		
19. Security Classif. (of this report) Unclassified	**20. Security Classif. (of this page)** Unclassified	**21. No. of Pages** 32	**22. Price** NA

Table of Contents

List of Tables

List of Figures

EXECUTIVE SUMMARY

Object detection and classification from videos have been studied and applied in many fields. The primary objective is to develop techniques to gather and interpret visual information through different discriminant functions. This research area is particularly relevant in transportation for agencies need to obtain and report the traffic volume and types of vehicles that travel on their networks. Also, these data are necessary for traffic engineering studies as well as planning studies. Obtaining traffic counts and types of vehicles via videos is more cost effective and safer for the transportation agencies compared to traditional methods such as pneumatic tubes.

The assembly of a discriminant function for detection and classification of vehicles is addressed in this report. The objective is to combine detection and classification in a single algorithm to classify cars and trucks. To accomplish this, a database containing images of cars and trucks was created using recordings from the South Carolina Department of Transportation (SCDOT). Specific areas of the video stream were extracted as independent images by applying background subtraction using Gaussian mixture-based algorithms.

The set of images was then used to train three different models for classification, namely Local Binary Patterns (LBP), Haar Cascade Classifier (HCC), and a Convolutional Neural Network (CNN). Image extraction and model training were implemented in Python and utilized the OpenCV (Open source Computer Vision) Library.

The results obtained from the analysis showed that the detection scheme using image subtraction and contouring was 93% accurate in detecting all moving objects in each of the frames. The trained models showed that the CNN classifier had a better classification accuracy rate (97%) compared to LBP and HCC algorithms.

CHAPTER 1

Introduction

1.1 Motivation

Traffic data collection is an elementary and necessary function performed by all transportation agencies. Traffic data is essential to understanding the current travel demand and pattern and are needed for the assessment of future developments. Traffic counting is often performed using pneumatic tubes due to their low cost; however, this method is unsafe for DOT staff personnel.

There are many non-intrusive technologies (e.g., passive acoustic detectors, ultrasonic detectors, microwave and radar detection systems, and video detection systems) that can be used for counting traffic and for classifying vehicles. These non-intrusive technologies have been investigated before due to their advantage to collect data off the roadway, and thereby, minimize or eliminate some of the safety and maintenance issues associated with road tubes. The SCDOT collects some of its traffic data via videos; however, this work is outsourced and presents a costly recurring expense.

This research explores three different methods to count and classify vehicles simultaneously: Local Binary Patterns (LBP), Haar Cascade Classifier (HCC), and a Convolutional Neural Network (CNN). The goal is to identify and fine tune a technique that can be used by state DOTs and other transportation agencies to obtain traffic data more safely and at a lower cost.

To accomplish the stated goal, a database with two vehicle classes, namely, cars and trucks was created using video recordings from the South Carolina Department of Transportation. The database was then used to train three classifiers developed using the different techniques mentioned. A discriminant function was developed using OpenCV, a computer vision library for processing video images, which combined the preprocessing methods and the classifiers for the simultaneous detection and classification of cars and trucks. Lastly, the accuracy of the classifiers was measured using the training dataset.

This report includes a literature review of the techniques that were used to train the classifiers, a review of the method used for detecting and extracting objects from the video recordings, and a description about the training procedure and testing of the classifiers.

CHAPTER 2

Literature Review

Video object detection

Automated Vehicle Classification (AVC) systems have been developed to assist human operators in system operations and surveillance (Boukerche et al., 2017). Typically, the AVC algorithms are incorporated in cameras to help identify different vehicles using the plate number, color, or type (Harlow, 2001). This practice has received a great deal of attention in recent years, and its ultimate purpose is to develop robust systems for real-time classification.

With the development of tools for human face recognition (Jones & Viola, 2001), the algorithms used in AVC systems have improved. However, these techniques still face many challenges regarding vehicle diversity, multiplicity, heterogeneous views, and lighting conditions, among others. Authors have proposed various approaches to classify vehicles. Some of these works are focused on geometry-based characteristics. For instance, Gupte et al. (2002) developed one of the first works on vehicle classification. Their work consisted of using a stationary camera to monitor highway scenes. To detect vehicles, the authors used segmentation or separation of the vehicles from the background by separating the object from the background and converting the difference image in a binary object mask. The classification was made by using the length, the width and the velocity of the detected regions.

Avery et al. (2004) proposed another geometry-based approach for vehicle detection. In this work, the authors presented an image processing algorithm to classify vehicles using the length. The classification was made by comparing the different length of the various vehicle classes. However, this technique is limited to the classification of vehicles with a notable change in length, but not different shape characteristics.

Mithun et al. (2012), proposed a detection and classification method using multiple time special images. In this paper, the authors developed a multiple virtual detection line (MVDL) based method that established if the moving vehicles were merged or disjointed, which is a common problem in imaging subtraction and other techniques used to separate the moving objects from the background. The authors used a two-step classification algorithm known as k-nearest neighbors, and reduced context dependencies using different shape and texture-based features.

Mishra et al. (2013) proposed a framework for detecting and classifying vehicles using adaptive background modeling with an SVM (Support Vector Machine) based histogram intersection kernel. The authors evaluated the algorithms in vehicle counting at different times of day. It had an accuracy rate of 89% when analyzing heavy motor vehicles, light motor vehicles, two-wheelers and three wheelers.

Sowjanya & Chakravarthy (2013) recommended a technique that deals with dynamic textures when subtracting the background. Their algorithm, called Fuzzy Color Histogram,

attenuates color variations caused by background motion and uses shape-based features, namely, aspect ratio and compactness. The authors tested the algorithm in different weather conditions, and the results were invariant regarding accuracy.

There many challenges associated with vehicle detection. Some of the algorithms mentioned above may provide give good results in terms of detection, but not in classification. Boukerche et al., 2017 summarized the challenges with vehicle detection and classification as follows.

- *Diversity:* Different vehicle types.
- *Multiplicity:* Vehicles of the same class but with different shape or designs. This is a disadvantage for algorithms developed to classify vehicles using length only.
- *Ambiguity:* Different vehicle classes have similar shapes or designs.
- *Heterogenous views:* variations in camera angle, scale, and viewpoints.
- *Light conditions:* Different illumination conditions or time of day.
- *Environmental conditions:* Different weather conditions.
- *Occlusions:* Closed vehicles, hidden views, etc.

Many of today's ITS systems already make use of the advantages of video-based systems to acquire data. Their applications include: traffic detection for signal operation, traffic management, enforcement, toll collection and surveillance. Perhaps the most advanced implementation of video-based detection system in the transportation field is for autonomous vehicles. These vehicles are outfitted with multiple cameras which are used to create a 3D map of their surroundings. They need to be able to detect lane markings, road signs, other vehicles, pedestrians, and bicyclists. Many car manufacturers have developed working prototypes of autonomous vehicles (e.g., Tesla).

CHAPTER 3

Methods

The Federal Highway Administration (FHWA) classifies vehicles by their axle configuration. As shown in Figure 1, there are a total of 13 vehicle classes. In this project, only two classes, cars, and trucks, are considered.

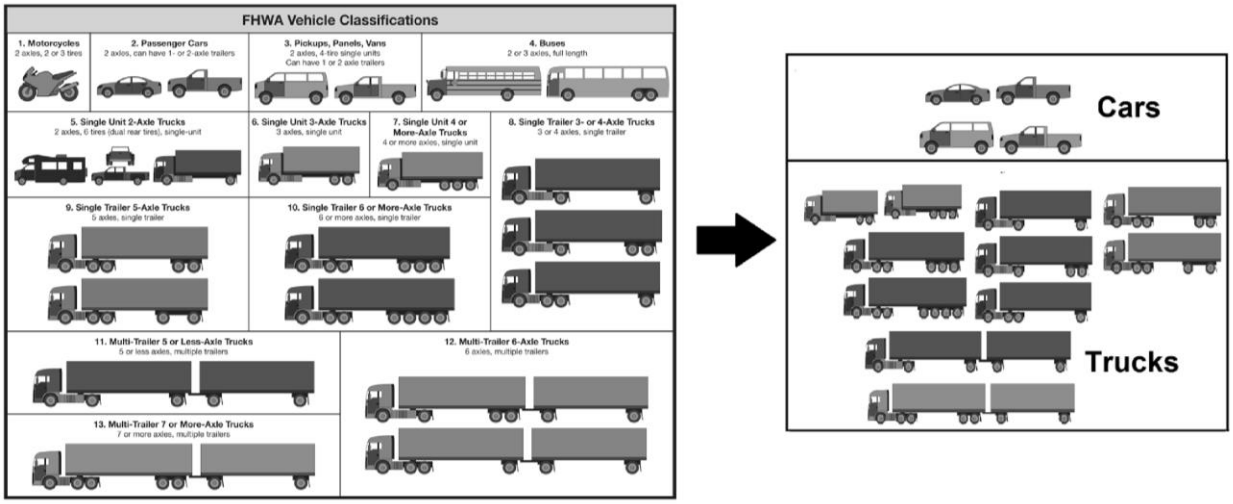


Figure 1. FHWA 13 Vehicle Classes (left: FHWA) and This Project's Simplified Two Vehicle Classes (right)

In this work, we used a cross-platform library called OpenCV in Python, for both computer vision and image processing applications. By image processing, we refer to the background subtraction and contouring after image transformations; this will be discussed in the next section. By computer vision, as illustrated in Figure 2, we refer to the methods coded in OpenCV for image processing, detection, and classification.

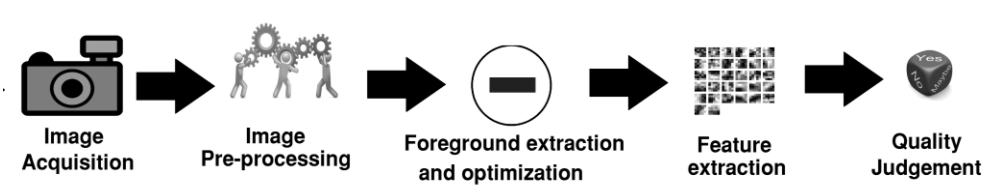


Figure 2. Computer vision system

In this project, we performed background subtraction using the Gaussian mixture for the detection of moving vehicles. For classification, three techniques were used, namely, Haar-like features, local binary patterns, and convolutional neural networks.

3.1 Background subtraction (BS) using Gaussian Mixture Model (GMM)

Background subtraction (BS) is a commonly used technique for pre-processing in video-based applications. BS is the first step in the detection of moving objects, given that it can provide information about their location without any prior knowledge of the objects

(Sobral & Vacavant, 2014). The technique segments the foreground or moving objects from the background in a sequence of frames. These algorithms for foreground extraction can be found in the literature (Friedman & Russell, 1997; Wren et al., 1997), but the process between all of them is very similar and consist of the following steps:

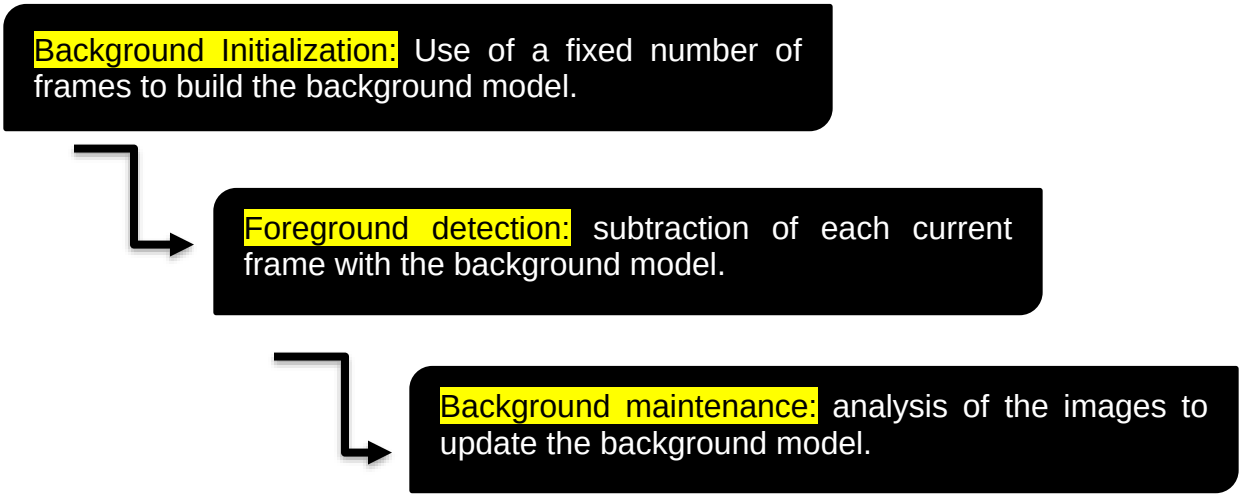


Figure 3. General background subtraction scheme

The background subtraction methods can be divided into five categories that includes a statistical method, a neural network method and a non-parametric method (Nurhadiyatna et al, 2013). The statistical method was proposed by Wren et al. (1997). In their work, they used a Gaussian function to express the background pixel intensity. A single function, however, was not enough when representing a complex environment, so additional methods have been proposed, which include the use of a Gaussian Mixture.

In this work, we used a Gaussian Mixture Model (GMM), following the algorithm proposed by Zivkovic (2004, 2006). The advantage of this algorithm is that it can estimate the parameters of the GMM and provide an optimum number of Gaussian distributions.

Pixel-based background subtraction is, in general, a decision-making problem. If the pixel is a background pixel (BG) or a pixel of a foreground object (FG), it is established by:

$$\frac{\mathrm{p}\left(\mathrm{BG}\middle|\overrightarrow{\mathrm{x}^{(\mathrm{t})}}\right)}{\mathrm{p}\left(\mathrm{FG}\middle|\overrightarrow{\mathrm{x}^{(\mathrm{t})}}\right)} = \frac{\mathrm{p}\left(\overrightarrow{\mathrm{x}^{(\mathrm{t})}}\middle|\mathrm{BG}\right)\mathrm{p}(\mathrm{BG})}{\mathrm{p}\left(\overrightarrow{\mathrm{x}^{(\mathrm{t})}}\middle|\mathrm{FG}\right)\mathrm{p}(\mathrm{FG})} \qquad (1)$$

where $\overrightarrow{\mathrm{x}^{(\mathrm{t})}}$ represents the pixel value at time t.

Given that no information is known about the background, the probabilities p(BG) and p(FG) are equal, and a uniform distribution is assumed by the foreground object $\mathrm{p}\left(\overrightarrow{\mathrm{x}^{(\mathrm{t})}}\middle|\mathrm{FG}\right) = cFG$. The pixel is part of the background if $\mathrm{p}\left(\overrightarrow{\mathrm{x}^{(\mathrm{t})}}\middle|\mathrm{BG}\right) > cFG$, thus cGF is the threshold value. The training set to estimate the background model is assumed as

independent and represented by $\mathcal{X}$, the background model is then expressed as $\bar{p}(\vec{x}|\mathcal{X}, BG)$.

Zivkovic modified the work proposed by Stauffer & Grimson (1999). The later authors created a robust adaptive tracking system to handle variations in lighting due to illumination. The modified algorithm updates the training set. At time t, $\mathcal{X} = \{X^{(t),\ldots,x^{t-T}}\}$, for an adaptation period T. The estimated density for M components is then:

$$\bar{p}(\vec{x}|\mathcal{X}, BG + FG) = \sum_{m=1}^{M} \overline{\pi_m \mathcal{X}}\left(\vec{x}; \overrightarrow{\overline{\mu_m}}, \overrightarrow{\overline{\sigma_m}} I\right) \qquad (2)$$

where $\overrightarrow{\overline{\mu_1}} \ldots \overrightarrow{\overline{\mu_M}}$ and $\overrightarrow{\overline{\sigma_1}} \ldots \overrightarrow{\overline{\sigma_M}}$ are estimated means and variances, I is the identity matrix and $\overline{\pi_m}$ are the mixing weights.

Figure 4 shows the separation of the foreground from the background using GMM.

Figure 4. Background subtraction using Gaussian Mixture Model

3.1.1 Contouring

Contours in image processing represent the curve joining the points that are continuous and have the same intensity or color (see Figure 5). In this work, we used contouring to delineate the objects that were separated from the background using the GMM.

Figure 5. Background subtraction and contouring

In OpenCV, the contour method stores the coordinates of the boundaries of the image with the same intensity (x, y). Different approximation methods can be used to generate the contours from the image after BS.

To reduce computer memory usage, we employed OpenCV "CHAIN_APPROX_SIMPLE." This approximation method does not save all the coordinates that belong to the boundary; it removes redundant points to compress the contour. Figure 6 shows the endpoints of the horizontal and vertical segments.

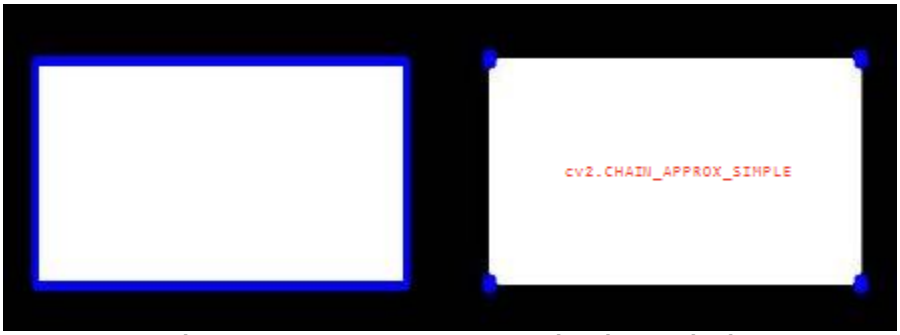


Figure 6. OpenCV Contour approximation method

We used the bounding rectangles to create the contours using the top-left coordinate *(x,y)*, width and height.

3.1.2 Database creation

The detection of the moving objects from the video streams allowed us to construct a database of images of cars and trucks. After applying background subtraction and contouring, the coordinates of the bounding rectangle of the detected objects where subtracted from the video stream. Each contour was collected as an independent image in the database. Afterwards, a manual classification was made to separate the objects in their respective category (see Figure 7).

A total of 2432 images were collected for cars and 1598 for trucks.

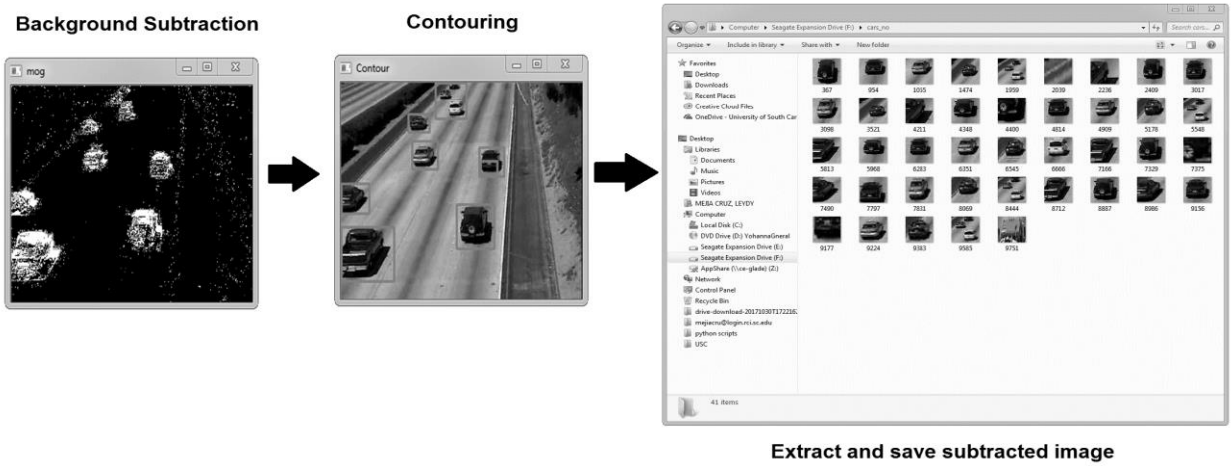


Figure 7. Database generation after background subtraction and contouring

3.2 Vehicle classification

Our classification framework is presented in Figure 8. The images used for training were collected from:

1. *3D Object Representations for Fine-Grained Categorization Jonathan Krause, Michael Stark, Jia Deng, Li zei-Fei 4th IEEE Workshop on 3D Representation and Recognition, at ICCV 2013 (3dRR-13). Sydney, Australia. Dec. 8, 2013.*

2. *MIOvision Trac Camera Dataset (MIO-TCD): http://tcd.miovision.com/, under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.*

3. *Images from our database (as discussed in subsection 3.1.2).*

In total, we used 11,000 images for cars and 5,602 images for trucks. We used half of the images for training and the other half for evaluation of the accuracy of the trained models.

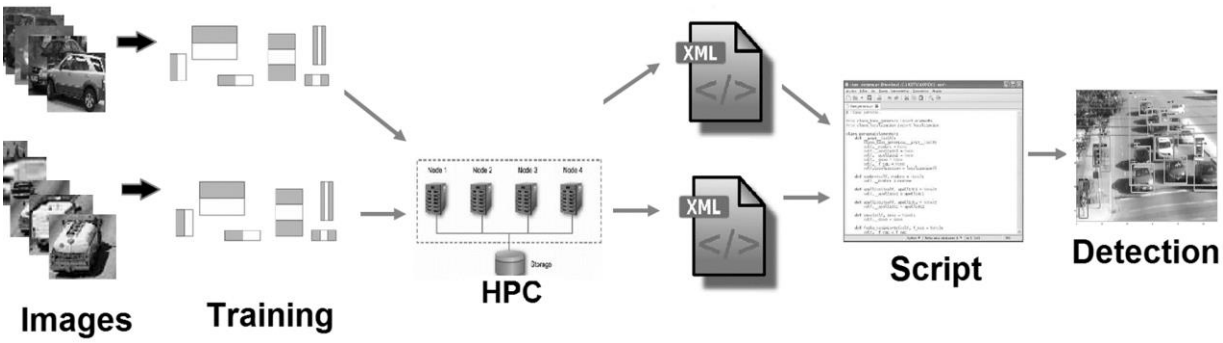


Figure 8. Classification scheme

To train the classifiers, high-performance computers were required. The Hyperion cluster with a Big Data node with 1.5 TB RAM at the University of South Carolina was used.

3.2.1 Cascade classifiers

Cascade classifiers (CC) are a case of ensemble learning using several classifiers, where the information of one classifier is used to inform the next classifier. Each of the stages is composed of a set of weak learners (Heitz et al., 2009).

Object detection in CC is made by locating a window over an image and using a set of features in each stage and establishing if the location of the window corresponds to a positive or a negative. A positive indicates that the object was found in the image and a negative indicates the contrary. If the window is classified as a negative, the classification of the current region ends; if there is a positive classification, the specific area passes to

the next stage of classification. Figure 9 shows the cascade system in the classifier. The final positive decision of the classifier is given when all the stages come to a positive result, which means that the object was detected.

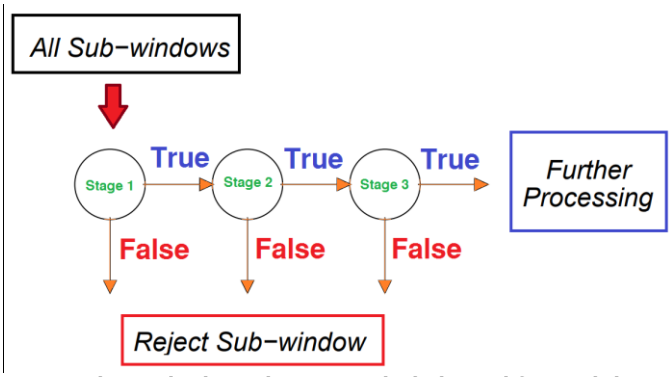


Figure 9 Schematic detection cascade (adapted from Viola, 2001)

Each stage in the cascade should have a low false negative rate to work well, and at the same time, have a relatively high false positive rate. False positive means that the cascade has labeled a window as positive that does not contain the object of interest.

Cascade methodology uses a set of positive and negative images. The number of images was selected in a way such that they can provide a sufficient number of images to the selected number of stages. A low number of stages could result in a weak trained classifier, whereas a high number of stages would require a higher number of images for training.

Both positive and negative images for each of the vehicle classes (cars and trucks) were used to train the classifiers. Negative images do not contain the object of interest (see Figure 10). A total of 15,157 negative images were used.

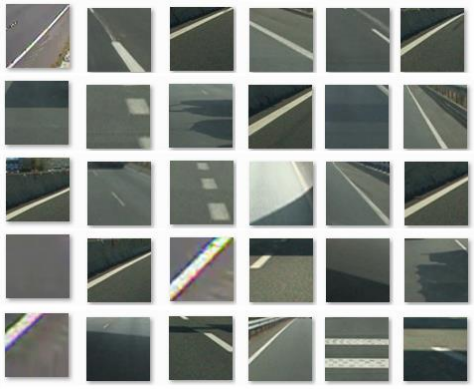


Figure 10. Negative images for cascade classifiers

In this project, two cascade-based classifiers were trained, Haar-like features and Local Binary Patterns. The schemes of these classifiers are the same; however, the construction of the integral image, or features, is different.

Haar-like features

Haar-like features are digital image characteristics developed by Viola & Jones (2001) to detect objects from images by using specific features or structures of the object to be detected. This technique analyzes rectangular regions surrounding a specific pixel location and adds the pixel intensities in each area (see equation 3). The rectangular features are obtained using a version of the actual image, which is called integral Image. The integral image *II(x,y)* is obtained by adding the pixels above and to the left of the coordinates (x,y), as given by the relation:

$$\mathrm{II}(x, y) = \sum_{x' \le x;\, y' \le y} \mathrm{I}(x', y') \tag{3}$$

where *II(x', y')* is the original image. Using:

$$\begin{aligned} s\,(x, y) &= S(x,y\text{-}1) + I(x, y) \\ \mathrm{II}(x, y) &= \mathrm{II}(x-1, y) + S(x, y) \end{aligned} \tag{4}$$

the integral image can be computed. S(x, y) is the row sum, s (x, -1) = 0 and II(-1, y) = 0. Figure 11 shows the general scheme of the integral image computation.

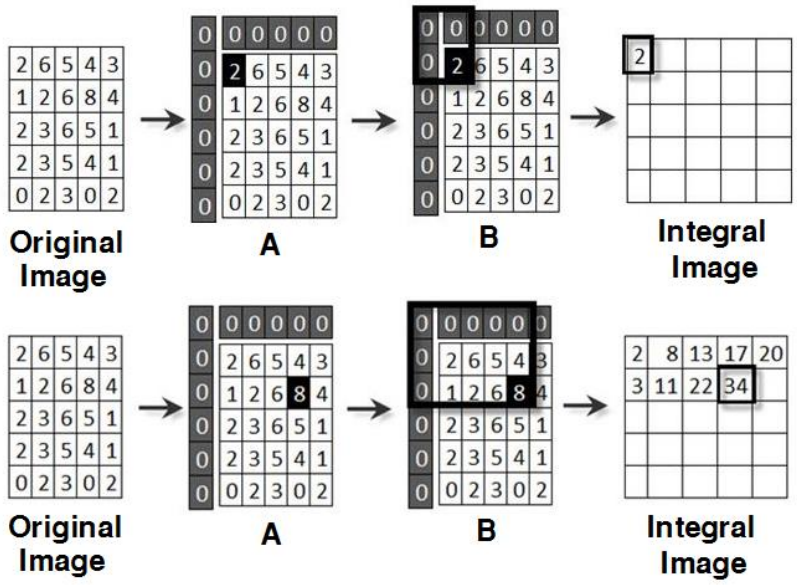


Figure 11. Integral image in Haar-like classifiers, (adapted from Johnson 2015)

Figure 12 shows an example of the rectangular features in car detection. Considering that some areas of the vehicle are darker than others, the Haar-like feature is composed of two rectangular areas.

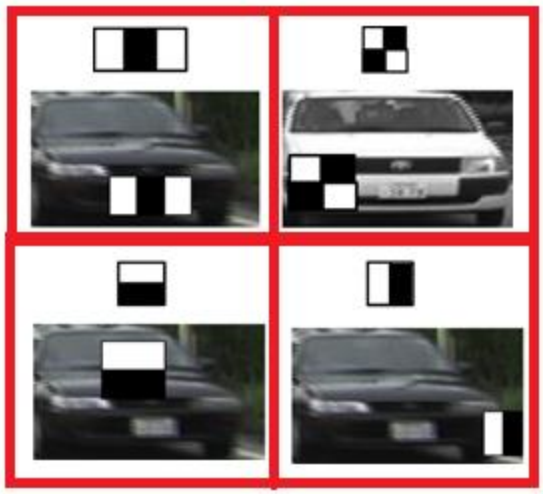


Figure 12. Haar Features in car detection (adapted from Wen et al., 2015)

A cascade classifier using Haar-like features has the structure presented in Figure 13. The generated features are created and evaluated inside the stages of the cascade.

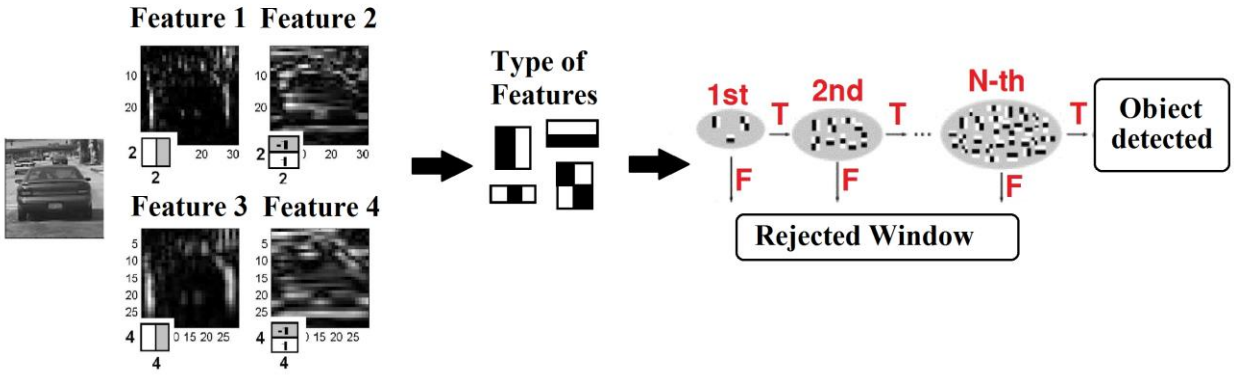


Figure 13. Cascade classifier using Haar-like features (left image adapted from Negri et al., 2007)

Local Binary Patterns

The Local Binary Patterns (LBP) is an operator that summarizes the structure of an image by comparing the value of each pixel with the surrounding pixel values. When the current pixel value is equal or higher to the value of its neighbor, the operator assigns a value of 1. When it is not, the value assigned is 0 (Ojala et al., 2000, 2002). The mathematical representation of the LBP operator is the following:

$$\mathrm{LBP}(x_c, y_c) = \sum_{P=0}^{P=1} 2^P S(i_p - i_c) \qquad (5)$$

where:

$$S = \begin{cases} 1 \rightarrow x \geq 0 \\ 0 \rightarrow else \end{cases}$$

Figure 14 summarizes the steps in the construction of the integral image using LBP.

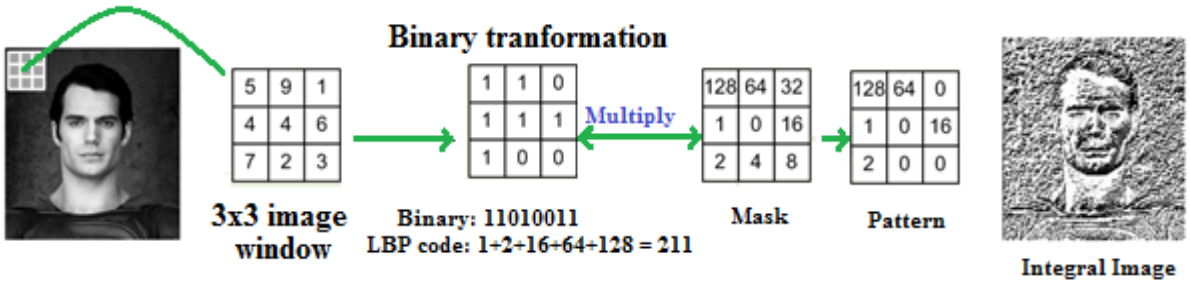


Figure 14. Local Binary Patterns Transformation (adapted from Kyrkou 2017)

The LBP code computed the LBP function over different areas of the image. In this manner, the final operator detects microstructures in the image, such as edges, or spots. The integral image has a histogram for each one of the n regions that divide it. The final feature vector was obtained by concatenating the histograms of all regions (OpenCV, 2014).

3.2.2 Convolutional Neural Networks

Artificial Neural Networks (NN) are computing systems that mimic the connections of the human brain. Biological neurons are simulated using different Activations Functions (AF) whose primary purpose is to activate or "switch on" different states when an input value is given. Similar to the human brain, the neural networks have hierarchical connections, where the outputs of the neurons become the input of other neurons. The different levels of bonds, in more technical terms, can be referred to as layers, which are composed of multiple nodes as illustrated in Figure 15.

The weighted inputs of the NN are added and evaluated in the AF inside the nodes. The weights are real values multiplying the input values, which change with the learning process to determine the output of the nodes. The sum of the weighted inputs also contains a "bias" term that adds flexibility to the AF.

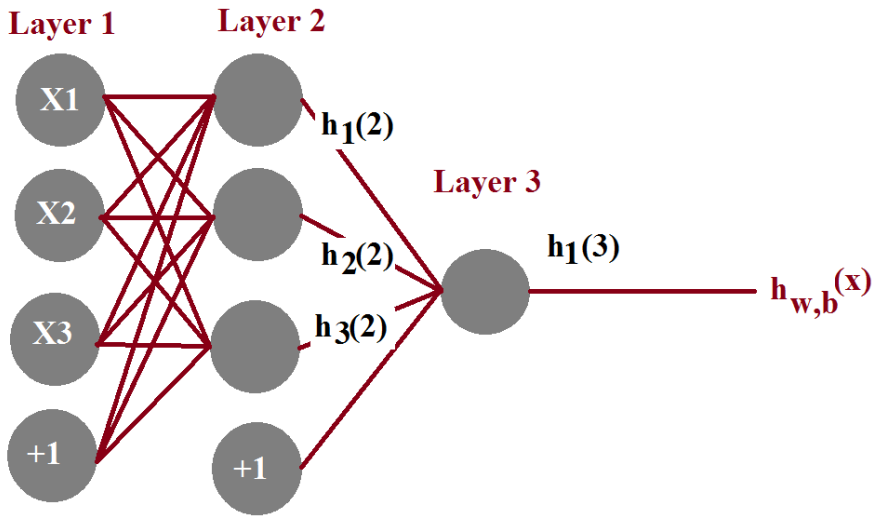


Figure 15. Structure of a three-layer NN

Like Haar-like features and LBP, different types of NN have been used in computer vision systems for object recognition using feature-based approaches (Egmont-Petersen et al., 2002; Rowley et al., 1998; Li et al., 2015)

Convolutional Neural Networks (CNN) is a subclass of NN. The input layer in the NN contains information about the image, and small regions of these are connected to additional layers inside the NN. Feature maps of these regions are created in a process called convolution that occurs in the hidden layers (Figure 16). Similar to the feature-based techniques already explained, the feature map in a NN represent key characteristics of the images such as lines, spots, or edges.

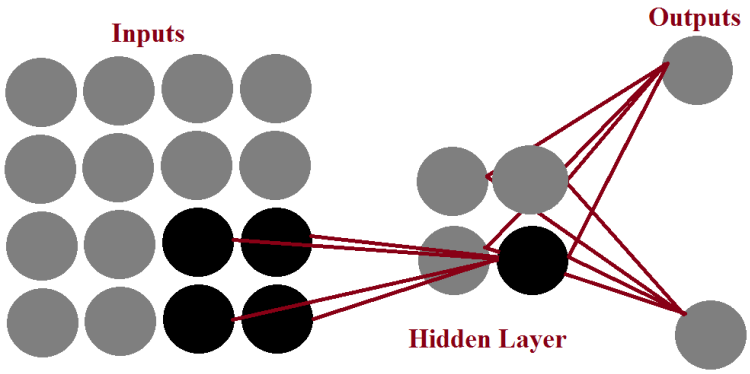


Figure 16. Structure of a CNN (adapted from Patel & Pingel, 2017).

In a CNN, the convolution is also known as the moving filter. For example, in Figure 17, the moving filter is mapping a 2x2 region of the input image. The output node is obtained by multiplying 0.5 to each node value (weights) and adding the results.

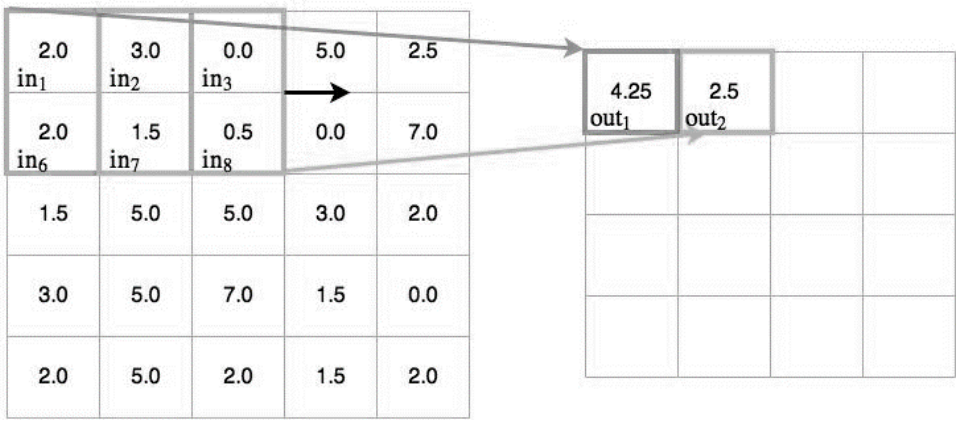


Figure 17. Moving filter in a CNN (adapted from "Adventures in machine learning", 2017)

Pooling is also a moving window technique that can help to reduce the size of the CNN and to make the features invariant to scale or orientation. During the pooling, instead of using weights, a statistical function is applied to the values of the window. For instance, max pooling refers to the selection of the maximum value. Figure 18 shows a full CNN structure.

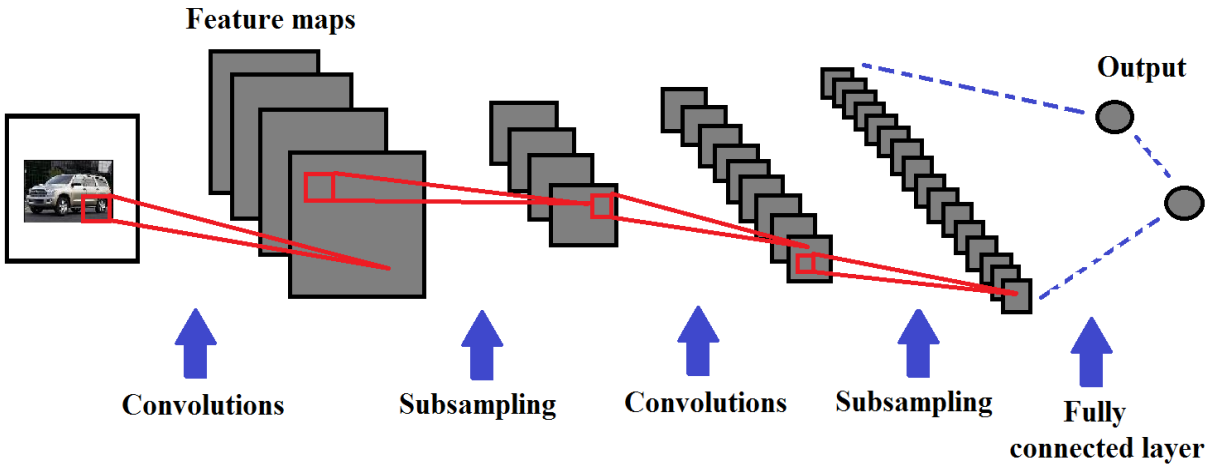


Figure 18. Full CNN structure

CHAPTER 4

Implementation and Results

The image preprocessing was implemented using Python v3.6 and the OpenCV library. The training of the classifiers was also performed using Python v3.6. The CNN was developed using TensorFlow library (Abadi et al., 2016).

4.1 Background subtraction and Contouring

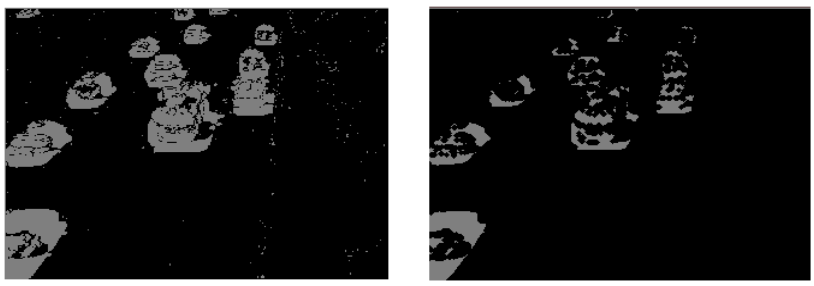
Before BS, Gaussian blur was applied to the frames extracted from the video recordings. The purpose was to smooth out the images and reduce, as much as possible, the noise generated by the camera movement. The Gaussian filter combines the input points with a Gaussian kernel, adds them and produces an output array (Szeliski, 2010). The Gaussian kernel in two dimensions is expressed as

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (6)$$

After smoothing the image, the background subtraction procedure (see Sec 3.1) was applied. The resulting foreground images of the moving vehicles were subject to two additional morphological transformations, namely, erosion and dilation.

Using erosion, the boundaries of the foreground object were eroded using a 2D convolutional kernel. The original image (after background subtraction) was transformed, by assigning a value of 1 to the areas that under the kernel (window) have at least one pixel value of 1. If not, the value is eroded (give a zero value).

Applying this technique, small white dots in the original image (left image in Figure 20), associated with noise were removed (right image in Figure 19), and objects that appeared to be joined are detached.



(a) (b)

Figure 19. Background subtraction (a), Erosion (b)

Dilation is a technique that increases the white region of the foreground object. A pixel value is 1 if, under the kernel, at least one pixel value is 1. In this manner, we assembled separate parts of the objects by increasing the area of the separated regions and without

the presence of the noise. Figure 20 shows the effect of increasing dilation of the foreground objects.

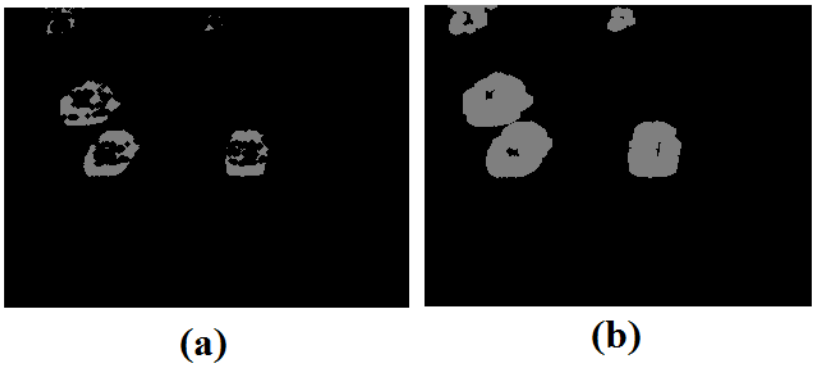


Figure 20. Erosion (a), Dilation (b)

The contouring of the dilated objects was performed using the methods described in section 3.1.1, after dilation of the foreground objects. Given that the foreground shapes are connected to, or may be inside others, the contours have a hierarchy that determines if it is a child or a parent. This consideration was accounted for in our contour retrieval.

Table 1. Detection in the first 20 frames of video recording

Frame No	Number of objects	Contours detected	% Detection
1	5	4	80
2	5	5	100
3	5	5	100
4	6	6	100
5	6	6	100
6	6	6	100
7	7	6	86
8	7	7	100
9	7	7	100
10	7	6	86
11	7	6	86
12	7	6	86
13	7	6	86
14	7	6	86
15	7	6	86
16	8	8	100
17	8	7	88
18	6	5	83
19	6	6	100
20	6	6	100
		Average % of Detection	93

We evaluated the detection scheme for the first 20 frames of a video recording as shown in Figure 21. It is important to note that the detection is likely of the same foreground objects, given that the number of frames is very small. For the total number of elements in each frame, the average percentage detection rate was of 93% (Table 1).

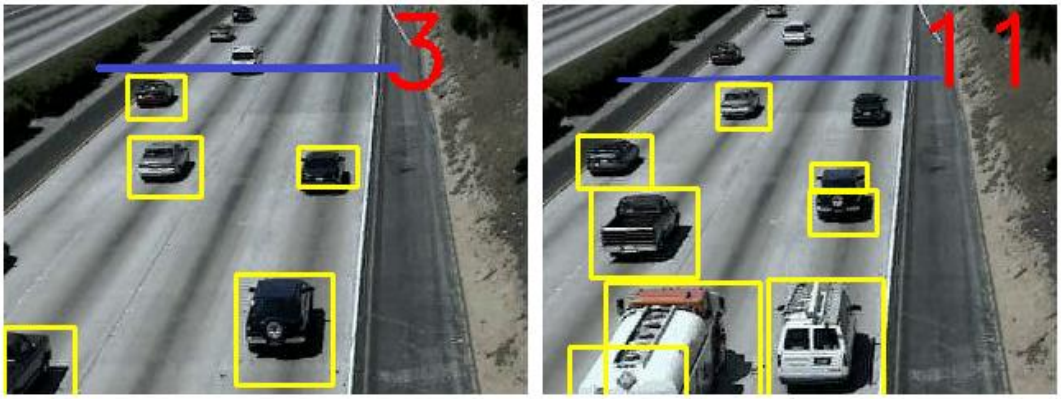


Figure 21. Contours in frames 3 and 11

The algorithm developed for background subtraction and contouring follows the general procedure presented in Figure 22.

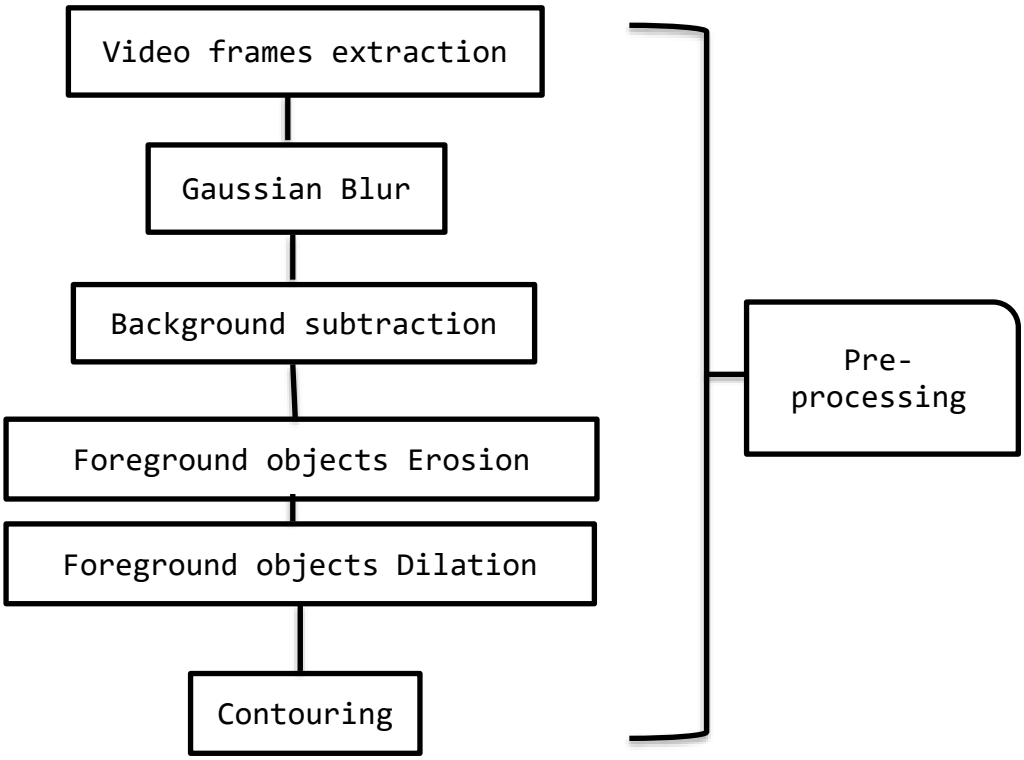


Figure 22. Pre-processing scheme

4.2 Classifiers

4.2.1 Haar-like and LBP input parameters

The images used to train the Boosted classifiers (LBP and Haar) have the following sizes per class:

Cars: 40x20 pixels 

Trucks: 60x20 pixels 

The parameters of the Haar-like and LBP algorithms are the following:

```
PARAMETERS:
Type of boosted classifier: GAB (Gentle AdaBoost)
Minimal desired hit rate for each stage of the classifier: 0.995
Maximal desired false alarm rate for each stage of the classifier: 0.5
Specifies whether trimming should be used and its weight: 0.95
Maximal depth of a weak tree: 1
Maximal count of weak trees for every cascade stage: 100
Mode (Only for Haar): ALL (Uses upright and 45-degree rotated feature set)
```

4.2.2 CNN input parameters

The images used to train the CNN are required to have the same dimension, independent of its classification. Considering that it is convenient to maintain the aspect-ratios of the different vehicles as a property for classification, the images were cropped as follows:

Cars: 56x56 pixels 

Trucks: 56x56 pixels 

We created 32, 5x5 convolutional filters + ReLU activations. Our CNN was composed of two layers. The first layer had 56 x 56 nodes, and a 2X2 window was used to apply max pooling operations using a stride of 2. The second layer had the same structure but with

64 filters. The fully connected layer contained 12,544 nodes, and the hidden layer 1,000 nodes. The output layer contained two classes of vehicles, cars and trucks.

The test of the accuracy of our training model is presented in Figure 23. The accuracy achieved with CNN was 97%.

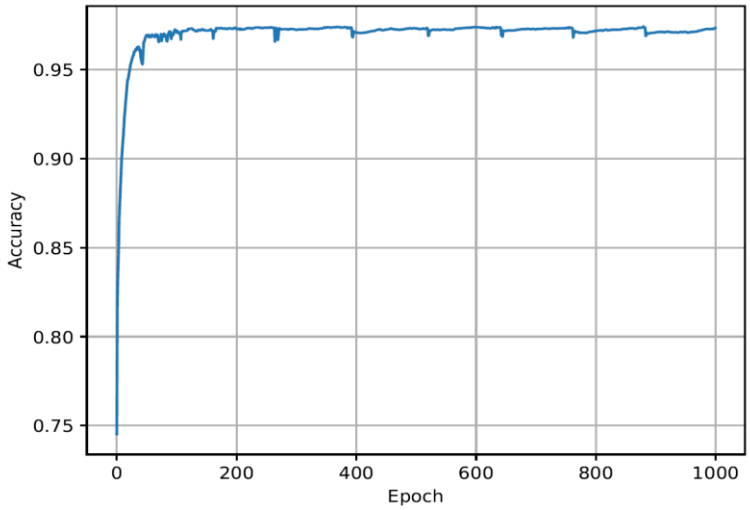


Figure 23. CNN Accuracy graph

4.2.3 Evaluation of the classifiers

The Haar-like, LBP and CNN models were tested in a batch containing 5,500 images for cars and 2,801 images for trucks. The results are presented in Table 2. Note that the sum of the "Cars" row is 5,500 and the sum of the "Trucks" row is 2,801. Among 5,500 cars, the LBP classifier identified 5,162 of them correctly, Harr-like classifier 5,418 and the CNN classifier 5,437. Similarly, among 2,801 trucks, the LBP classifier identified 1,478 correctly, Haar-like classifier 1,711 and the CNN classifier 2687.

Table 2. Accuracy evaluation of the different classifiers

LBP Classifier	Cars	Trucks
Cars	5162	338
Trucks	1323	1478
Accuracy	93.9%	52.8%
Haar-like Classifer	**Cars**	**Trucks**
Cars	5418	82
Trucks	1090	1711
Accuracy	98.5%	61.1%
CNN Classifer	**Cars**	**Trucks**
Cars	5437	63
Trucks	114	2687
Accuracy	98.9%	95.9%

As shown in Table 2, the CNN classifier yielded the highest accuracy rate among the trained classifiers. The ability of the CNN classifier to correctly identify cars and trucks is demonstrated in Figure 24.

Figure 24. Demonstration of CNN classifier on SCDOT-recorded video

CHAPTER 5

Conclusions

The results of this research showed that background subtraction using Gaussian mixture had an accuracy rate of 93% in detection, which indicates that the algorithm successfully detected most of the moving objects. The high accuracy rate is also due to the use of erosion and dilatation. Moreover, by tracking objects through successive frames, counting accuracy is further improved.

The LBP classifier had a 93.9% accuracy rate in classifying cars. However, the accuracy rate for trucks was only 52.8%. The results obtained for Haar-like feature classifier also showed an excellent classification rate for 98.5%. Its classification of trucks (61.1%) is better than the LBP classifier. It is very likely that the cascade classifiers may have a higher classification rate for cars not because the critical features for cars were correctly developed, but rather it is due to the misclassification of trucks as cars.

Contrary to the cascade classifiers, CNN was able to achieve a remarkable accuracy rate for both classes, 98.9% for cars and 95.9% for trucks. These results indicate that CNN is the best classifier among those evaluated and has the best potential to detecting and classifying multiple vehicle classes and other transportation modes (i.e., pedestrians, bicyclists).

REFERENCES

Boukerche, A., Siddiqui, A.J. and Mammeri, A., 2017. Automated Vehicle Detection and Classification: Models, Methods, and Techniques. ACM Computing Surveys (CSUR), 50(5), p.62.

Harlow, C. and Peng, S., 2001. Automatic vehicle classification system with range sensors. Transportation Research Part C: Emerging Technologies, 9(4), pp.231-247.

Jones, M.J. and Viola, P., 2003. Face recognition using boosted local features. Technical Report MERL-TR-2003-25, MitsubishiElectric Research Laboratory.

Gupte, S., Masoud, O., Martin, R.F. and Papanikolopoulos, N.P., 2002. Detection and classification of vehicles. IEEE Transactions on intelligent transportation systems, 3(1), pp.37-47.

Avery, R.P., Wang, Y. and Rutherford, G.S., 2004, October. Length-based vehicle classification using images from uncalibrated video cameras. In Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on (pp. 737-742). IEEE.

Mithun, N.C., Rashid, N.U. and Rahman, S.M., 2012. Detection and classification of vehicles from video using multiple time-spatial images. IEEE Transactions on Intelligent Transportation Systems, 13(3), pp.1215-1225

Mishra, P.K., Athiq, M., Nandoriya, A. and Chaudhuri, S., 2013. Video-based vehicle detection and classification in heterogeneous traffic conditions using a novel kernel classifier. IETE journal of research, 59(5), pp.541-550.

Sowjanya, K. and Chakravarthy, G., 2013. Vehicle Detection and Classification using Consecutive Neighbouring Frame Difference Method. Industrial Science, 1(2).

Randall, J.L., 2012. Traffic Recorder Instruction Manual. Texas: Texas Department of Transportation.

Sobral, A. and Vacavant, A., 2014. A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos. Computer Vision and Image Understanding, 122, pp.4-21.

Friedman, N. and Russell, S., 1997, August. Image segmentation in video sequences: A probabilistic approach. In Proceedings of the Thirteenth Conference on Uncertainty in artificial intelligence (pp. 175-181). Morgan Kaufmann Publishers Inc.

Nurhadiyatna, A., Jatmiko, W., Hardjono, B., Wibisono, A., Sina, I. and Mursanto, P., 2013, October. Background subtraction using Gaussian mixture model enhanced by hole filling algorithm (gmmhf). In Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on (pp. 4006-4011). IEEE.

Wren, C.R., Azarbayejani, A., Darrell, T. and Pentland, A.P., 1997. Pfinder: Real-time tracking of the human body. IEEE Transactions on Pattern Analysis & Machine Intelligence, (7), pp.780-785.

Zivkovic, Z., 2004, August. Improved adaptive Gaussian mixture model for background subtraction. In Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on (Vol. 2, pp. 28-31). IEEE.

Zivkovic, Z. and Van Der Heijden, F., 2006. Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern recognition letters, 27(7), pp.773-780.

Stauffer, C. and Grimson, W.E.L., 1999, June. Adaptive background mixture models for real-time tracking. Proceedings of the 999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (p. 2246).

Heitz, G., Gould, S., Saxena, A. and Koller, D., 2009. Cascaded classification models: Combining models for holistic scene understanding. In Advances in Neural Information Processing Systems (pp. 641-648).

Viola, P. and Jones, M., 2001. Rapid object detection using a boosted cascade of simple features. In Computer Vision and Pattern Recognition. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Vol. 1, pp. I-I).

Johnson, A. (2015). [online] https://slideplayer.com/slide/8715285/ [Accessed 21 Jun. 2018].

Patel, S. and Pingel, J. (2017). [online], Introduction to Deep Learning: What Are Convolutional Neural Networks?, https://www.mathworks.com/videos/introduction-to-deep-learning-what-are-convolutional-neural-networks--1489512765771.html, [Accessed 30 Nov. 2018].

Wen, X., Shao, L., Fang, W. and Xue, Y., 2015. Efficient Feature Selection and Classification for Vehicle Detection. IEEE Trans. Circuits Syst. Video Techn., 25(3), pp.508-517.

Negri, P., Clady, X. and Prevost, L., 2007, May. Benchmarking haar and histograms of oriented gradients features applied to vehicle detection. In ICINCO-RA (1) (pp. 359-364).

Ojala, T., Pietikäinen, M. and Mäenpää, T., 2000, June. Gray scale and rotation invariant texture classification with local binary patterns. In European Conference on Computer Vision (pp. 404-420). Springer, Berlin, Heidelberg.

Ojala, T., Pietikainen, M. and Maenpaa, T., 2002. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Transactions on pattern analysis and machine intelligence, 24(7), pp.971-987.

Opencv dev team (2011-2014). Face Recognition with OpenCV [online] https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html#local-binary-patterns-histograms [Accessed 15 Nov. 2018].

Kyrkou, C. (2017) Object Detection Using Local Binary Patterns [online] https://medium.com/@ckyrkou/object-detection-using-local-binary-patterns-50b165658368. [Accessed 18 Oct. 2018].

Egmont-Petersen, M., de Ridder, D. and Handels, H., 2002. Image processing with neural networks—a review. Pattern Recognition, 35(10), pp.2279-2301.

Rowley, H.A., Baluja, S. and Kanade, T., 1998. Neural network-based face detection. IEEE Transactions on pattern analysis and machine intelligence, 20(1), pp.23-38.

Li, H., Lin, Z., Shen, X., Brandt, J. and Hua, G., 2015. A convolutional neural network cascade for face detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5325-5334).

Andy, (2017). Adventures in machine learning [online] http://adventuresinmachinelearning.com/convolutional-neural-networks-tutorial-tensorflow/ [Accessed 15 Oct. 2018].

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. and Kudlur, M., 2016, November. Tensorflow: a system for large-scale machine learning. In OSDI (Vol. 16, pp. 265-283).

Szeliski, R., 2010. Computer vision: algorithms and applications. Springer Science & Business Med.