Automatic Extraction of Vehicle, Bicycle and Pedestrian Traffic from Video Data

FINAL REPORT

Prepared by:

Nathan Huynh, Ph.D.
Robert Mullen, Ph.D.
John Rose, Ph.D.
Quentin Eloise

Department of Civil and Environmental Engineering
University of South Carolina

FHWA-SC-21-09

December 2021

Sponsoring Agencies:

South Carolina Department of Transportation
Office of Materials and Research
1406 Shop Road
Columbia, SC 29201

Federal Highway Administration
South Carolina Division
Strom Thurmond Federal Building
1835 Assembly Street, Suite 1270
Columbia, SC 29201

Technical Report Documentation Page

1. Report No
FHWA-SC-21-09

2. Government Accession No.

3. Recipient's Catalog No.

4. Title and Subtitle
Automatic Extraction of Vehicle, Bicycle, and Pedestrian Traffic from Video Data

5. Report Date
December 31, 2021

6. Performing Organization Code

7. Author/s
Nathan Huynh, Robert Mullen, John Rose, and Quentin Eloise

8. Performing Organization Report No.

9. Performing Organization Name and Address
University of South Carolina
Department of Civil and Environmental Engineering
300 Main St.
Columbia, SC 29208

10. Work Unit No. (TRAIS)

11. Contract or Grant No.
SPR No. 742

12. Sponsoring Organization Name and Address
South Carolina Department of Transportation
Office of Materials and Research
1406 Shop Road
Columbia, SC 29201

13. Type of Report and Period Covered
Final Report

14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

This project investigated the use of traffic cameras to count and classify vehicles. The intent is to provide an alternative approach to pneumatic tubes for collecting traffic data at high volume locations and to eliminate safety risks to SCDOT personnel and contractors. The objective is to develop algorithms to post-process the 48-hour videos to determine the number of vehicles in each one of four categories: motorcycles, passenger cars and light trucks, buses/campers/tow trucks, and small to large trucks. To this end, background subtraction and foreground detection algorithms were implemented to detect moving vehicles, and a Convolutional Neural Network (CNN) model was developed to classify vehicles using thermal images obtained from a custom-built thermal camera and solar-powered trailer. Additionally, to overcome false detection of vehicles due to either camera motion or erratic light reflection from the pavement surface, an algorithm was developed to keep track of each vehicle's trajectory and the vehicle trajectories were used to determine the presence of an actual vehicle. The developed algorithms and CNN model were incorporated into a Windows-based application, named DECAF (detection and classification by functional class) to enable users to easily specify the folder that contains the video files to be processed, specify the region for which traffic should be analyzed, specify the time interval for which the data should be aggregated, and view the detection and classification results in two report formats: 1) a spreadsheet with vehicle-by-vehicle information, and 2) a PDF summary report with totals aggregated for the user-specified interval. DECAF was tested using videos collected from five different sites in Columbia, SC, and the overall detection and classification accuracy for the hours evaluated was found to be 95% or higher.

17. Key Words
Video-based traffic data collection, convolutional neural network, thermal imaging.

18. Distribution Statement
No restrictions. This document is available to the public through the National Technical Information Service, Springfield, VA 22161.

19. Security Classification (of this report)
Unclassified

20. Security Classification (of this page)
Unclassified

21. No. Of Pages

22. Price

Reproduction of form and completed page is authorized

DISCLAIMER

The contents of this report reflect the views of the author who is responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the South Carolina Department of Transportation or the Federal Highway Administration. This report does not constitute a standard, specification, or regulation.

The State of South Carolina and the United States Government do not endorse products or manufacturers. Trade or manufacturer's names appear herein solely because they are considered essential to the object of this report.

ACKNOWLEDGMENTS

The project team greatly appreciate the guidance and assistance from the following Project Steering and Implementation Committee members:

- Wise, Stacy (Chair)
- Anderson, Todd (*Formerly with SCDOT*)
- Morris, Yolanda (FHWA)
- Heaps, Meredith
- Swygert, Terry

EXECUTIVE SUMMARY

The objectives of this research were to: 1) develop image processing algorithms to automatically extract vehicle counts and classifications as well as counts of motorcycles, bicycles, and pedestrians from videos, and 2) incorporate the developed algorithms into a stand-alone application with an easy-to-use interface to enable the SCDOT staff to process traffic videos in house.

A survey of State Departments of Transportation (DOTs) was conducted to obtain information regarding the use of video-based systems for traffic counting and classification to guide the research. Responses from 19 DOTs indicated that the video-based system is the most commonly used non-intrusive method for vehicle counting and classification. The next three most popular non-intrusive methods are: 1) side-fire radar, 2) infrared axle detector, and 3) radar detector. Among the 11 DOTs that use the video-based systems, two use them to collect traffic data during the day only and nine use them to collect traffic data both during daytime and nighttime. Only three out of the 11 DOTs that use video-based systems have the vehicle classification task performed in-house; the rest outsource this work to a vendor. Regarding the desired classification accuracy rate, two indicated 90% or better, five indicated 95% or better and three indicated 98% or better. From the responses, it was determined that the video-based system developed in this project should have the capability to process traffic data during both the daytime and nighttime and produce counting and classification accuracy at 95% or better.

To accomplish the first objective, background subtraction and foreground detection algorithms were implemented to detect moving vehicles, and a Convolutional Neural Network (CNN) model was developed to classify vehicles. CNN is a deep learning approach that learns features and patterns directly from the input images. Its performance is affected by the quality of the provided images, which was found to be an issue during the nighttime when the visible camera was used. To overcome the issue of poor image quality at night with the traditional visible traffic camera, the FLIR TrafiSense2 Dual (visible and thermal) camera was purchased and custom-built to allow for recording of thermal images to an external drive. To power the thermal camera, the SCDOT solicited bids for a portable solar trailer with specific power requirements, height of crank up mast, and storage capacity. To overcome false detection of vehicles due to either camera motion or erratic light reflection from the pavement surface, an algorithm was developed to keep track of each vehicle's trajectory and the vehicle trajectories were used to determine the presence of an actual vehicle.

To accomplish the second objective of this project, a Windows-based application, named DECAF (detection and classification by functional class) was developed to enable users to easily specify the folder containing the video files to be processed, specify the region for which traffic should be analyzed, and specify the time interval for which the data should be aggregated. This application uses the thermal CNN model to classify each vehicle at every frame of a running video while the vehicles are within the user-specified region of interest. The final classification of each vehicle is the one most frequently classified. To speed up the processing of the video, the detection and classification is performed for every other frame. It was found that if the traffic to be processed is light to medium, the ratio of processing time to actual video time is about 0.85. That is, DECAF can process a 100-minute video in 85 minutes. When the traffic to be processed

is heavy, the ratio could be as high as 1.15; this is due to the higher number of vehicles that need to be detected and tracked, and the higher number of images that need to be classified. The application allows the user to view the detection and classification results in two report formats: 1) a spreadsheet with vehicle-by-vehicle information, and 2) a summary report with totals aggregated for the user-specified interval.

A 95% accuracy or higher in both detection and classification can be achieved with DECAF for roadways with up to four lanes in each direction. Therefore, it is recommended that the SCDOT consider using the thermal camera and the developed DECAF application to collect traffic data at high-volume areas. Deploying the trailer and thermal camera on the side of the road will be safer for the SCDOT personnel than deploying MetroCount's pneumatic tubes across multiple lanes. The level of effort required for deploying the trailer and thermal camera is similar to that of deploying the Miovision Scout which the SCDOT has done in the past. The advantage of using the in-house equipment and software is that it will save the SCDOT the video processing cost, approximately $500.00 per 48-hour count. There are two situations when the deployment of the thermal camera should be avoided. The first is windy conditions; video quality deteriorates due to noticeable camera motion even when the camera is anchored with two cables. The second is when the air temperature exceeds 90 degrees Fahrenheit. At these temperatures, the thermal camera has to recalibrate due to the temperature changes which results in periodic blackouts. During such blackouts, the foreground images (vehicles) are indistinguishable from the background (pavement and other fixed objects), and therefore, vehicles that appear during blackouts will not be detected and classified.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

The Federal Highway Administration's (FHWA's) Office of Highway Policy Information maintains national programs to track traffic trends, vehicle distributions, and weight to meet data needs specified in federal highway legislations. Activities include the development of guidelines, regulations, direct data collection, data processing, research, analysis, and professional conferences (FHWA, 2021a). For the data collection component, the FHWA created the Highway Performance Monitoring System (HPMS) in 1978 that stores data on the extent, condition, performance, use and operating characteristics of the nation's highways (FHWA, 2021b). The traffic data themselves are collected by State Departments of Transportation (DOTs) and reported to the FHWA on an annual basis. To this end, the South Carolina Department of Transportation (SCDOT) annually collects and provides the FHWA traffic volume and vehicle classification data on SCDOT-maintained roads using a combination of Continuous Count Stations (CCS) and Short Duration Count Stations (SDCS).

Traffic volume is reported as Annual Average Daily Traffic (AADT), which is the total volume of vehicle traffic on a road for a year divided by 365 days. The AADT traffic data are used by many units within the SCDOT not only to meet HPMS data reporting requirements, but also to make informed decisions, and support various studies related to planning, design, operations, safety, maintenance, and environmental analysis. Examples of projects that require AADT data are corridor operations and/or safety studies, pavement design, traffic signal control improvement, and pavement rehabilitation/reconstruction. AADT data are also used to make work zone lane closure decisions. Therefore, the reported AADT values must be accurate.

The FHWA developed a standardized vehicle classification system in the mid-1980s. This system classifies vehicles as one of 13 class groups, as shown in Figure 1-1 (FHWA, 2014c). It was developed to meet the needs of different traffic data users and to support various studies that require detailed vehicle class groups. For example, the application of the SCDOT Pavement Design Guide requires knowledge of the number of Class 5, 6, 8, and 9 vehicles on the road to be designed or rehabilitated, the application of the SCDOT Roadway Design Manual requires knowledge of projected vehicle types for they affect lane widths, corner radii, and level of service, and the application of the SCDOT Traffic Calming Guidelines requires knowledge of the expected percentage of vehicles with a long wheelbase. As is the case with AADT, the SCDOT needs to obtain vehicle classification data not only for HPMS reporting purposes but also to support various in-house studies and statewide projects.

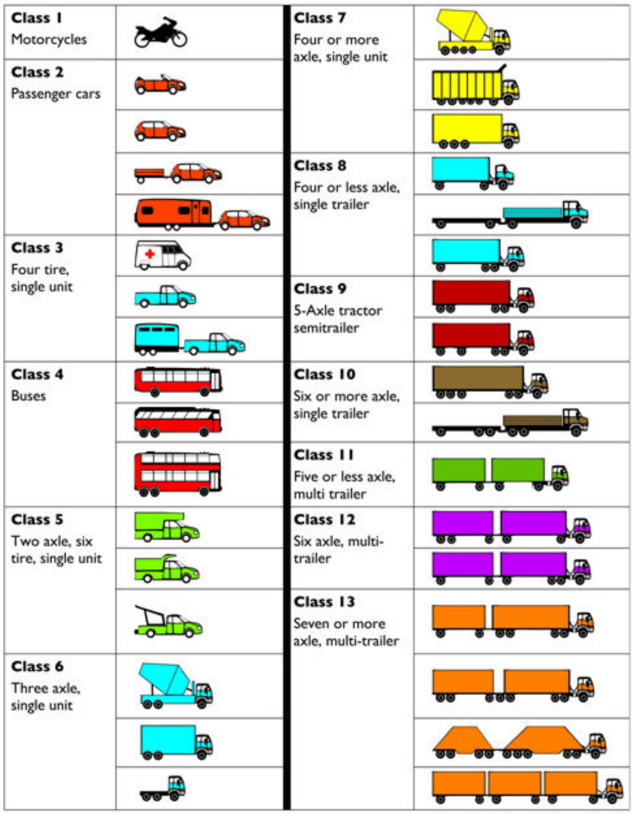

Class 1
Motorcycles
Class 2
Passenger cars
Class 3
Four tire, single unit
Class 4
Buses
Class 5
Two axle, six tire, single unit
Class 6
Three axle, single unit
Class 7
Four or more axle, single unit
Class 8
Four or less axle, single trailer
Class 9
5-Axle tractor semitrailer
Class 10
Six or more axle, single trailer
Class 11
Five or less axle, multi trailer
Class 12
Six axle, multi-trailer
Class 13
Seven or more axle, multi-trailer

Figure 1-1. FHWA's 13 vehicle category classification
(source: FHWA, 2014c)

To collect traffic volume and vehicle classification data, the SCDOT uses a combination of Peek ADR-2000 and TDC-EMU 3 traffic counters/classifiers at CCS, and MetroCount RoadPod VT counters and Miovision Scout cameras at SDCS. The Peek ADR and TDC-EMU traffic counters/classifiers provide the most accurate estimate of AADT because they collect traffic data continuously for 24 hours per day and 365 days per year. The CCS AADT is computed using the AASHTO method, which is an average of averages developed by the American Association of State Highway Transportation Officials. The AASHTO method is recommended according to the Traffic Monitoring Guide (2013) because “it allows factors to be computed accurately even when a considerable number of data is missing from a year at a site, and because it works accurately under a variety of data conditions (both with and without missing data).” The installation of Peek ADRs and TDC-EMUs throughout the state to count and classify vehicles is not economically feasible. Currently, the SCDOT has about 187 CCS and weigh-in-motion stations located throughout the entire state; these stations are primarily on interstates. To provide sufficient statewide coverage, the SCDOT has about 12,000 SDCS where 24-hour, 48-hour, or one week of data are collected on an annual, biennial, or triennial basis, depending on the functional class. The goal of SDCS is to collect data that can be adjusted by factoring and creating an AADT estimate that represents a typical traffic volume any time or day of the year.

Currently, the primary method being used by the SCDOT to collect 48-hour traffic data at SDCS is the MetroCount Counter with pneumatic tubes as shown in Figure 1-2. This method is considered ‘intrusive’ because its use requires placing the rubber tubes across the roadway. This intrusive method is problematic on high-volume roads. Specifically, it is not safe for the data collection crew to be in the roadways, it is difficult to secure tubes on roads with multiple lanes, and high-volume roads tend to have a higher percentage of classification errors due to multiple vehicles passing the tubes at the same time. At locations where it is clearly unsafe to use the MetroCount counters, the SCDOT would use the Miovision Scout camera to record traffic and then send the video to the vendor to process at a cost. This method is both inconvenient and costly for the SCDOT.

Building on current practice and the benefits of a video-based traffic data collection system, the aim of this research project is to develop models, algorithms, and software to provide the SCDOT with the means to count and classify vehicles to reduce safety risks to its personnel and contractors and to improve vehicle count and classification. The specific objectives of this project are to: 1) develop image processing algorithms to automatically extract vehicle counts and classifications as well as counts of motorcycles, bicycles, and pedestrians from videos, and 2) incorporate the developed algorithms into a stand-alone application with an easy-to-use interface to enable the SCDOT staff to process traffic videos in house.

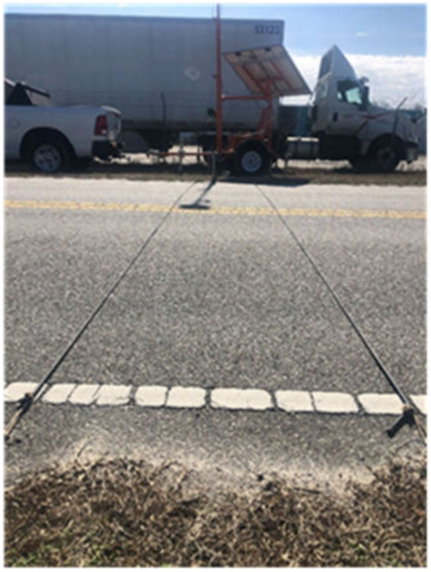


Figure 1-2. Pneumatic tubes deployed on Walter Price Road, Columbia, SC

CHAPTER 2: BACKGROUND AND LITERATURE REVIEW

The literature review conducted in this project indicates that among the different object recognition methods, Convolutional Neural Networks (CNNs) are the most widely used for visual imagery. For this reason, this project focused exclusively on applying CNN to classify vehicles. There are two different approaches to applying CNNs: supervised and unsupervised. Supervised learning is an approach that is defined by its use of labeled datasets. That is, the datasets are designed to train or "supervise" algorithms into classifying data or predicting outcomes accurately. On the other hand, unsupervised algorithms discover hidden patterns in data without the need for human intervention (Delua, 2021). The supervised CNN approach is the more popular of the two and one that is utilized in this project. A prerequisite for applying the CNN model is object detection. A review of previous work on vehicle detection and classification is provided below, followed by a brief background on object recognition and CNNs.

2.1 Background on object recognition

Object recognition involves identifying objects in digital photographs or videos. It is accomplished via a series of tasks: localization, classification, and detection. Object localization refers to the task of identifying the location of one or more objects in an image and drawing a bounding box around their extents. Image classification involves assigning a class label to an object. Object detection combines these two tasks - drawing a bounding box around each object of interest in the image and assigning a class label to them. Collectively, these tasks are referred to as object recognition. Object recognition is a trivial task for humans; however, the algorithms and models that are needed to have computers perform this task automatically are rather complex (Karpathy, 2016). For this project, the ultimate purpose of object recognition is to locate vehicles in a video, draw rectangular bounding boxes around them, and then determine their vehicle groups.

The challenges that all automated systems faced in object recognition are variations in viewpoints, scale, object deformation, occlusion, illumination, background clutter, and intra-class variation (Liu et al., 2016; Boukerche et al., 2017). In the past, researchers utilized a two-stage approach where in the first stage, features using human-designed heuristics are extracted, and in the second stage, a classifier is used to recognize the objects (Rawat et al., 2017 and LeCun et al., 1998). A key limitation of this approach is that the recognition accuracy is largely determined by the ability of the designer to come up with an appropriate set of features for the feature extractor module in the first stage. Moreover, the feature extractor must be modified for each new set of objects that need to be detected. To overcome these limitations, recently researchers have developed deep learning models that exploit multiple layers of nonlinear information processing for feature extraction and transformation, as well as for pattern analysis and classification (Rawat et al., 2017). Among these deep learning models, convolutional neural networks (CNN) are the leading architecture for image recognition, classification, and detection.

2.2 Background on convolutional neural networks

Convolutional neural networks are a subclass of Artificial Neural Networks (ANNs) which are computing systems that are loosely modeled after the neural structures of human brains. Similar to the human brain, ANNs have hierarchical connections, where the output of a neuron becomes the input of other neurons. The network forms a directed, weighted graph. As such, an ANN consists of a collection of simulated neurons, where each neuron is a node that is connected to other nodes via links that correspond to biological axon-synapse-dendrite connections. Each link has a weight, which determines the strength of one node's influence on another. Biological neurons are simulated using different activations functions, whose primary purpose is to activate different states when an input value is given.

In a typical ANN, each node in a layer is connected to all nodes in the next layer. However, in a CNN, small regions of the input layers are connected to nodes in the next layer. This region is known as the local receptor field and is translated into an image to create a feature map from the input layer to the hidden layer. This process is called convolution (Rawat et al, 2017 &, LeCun et al, 2015). The CNN architecture has many variations, but in general, it consists of a convolutional and pooling layer, a fully connected layer, and an output class layer. Figure 2-1 shows the general structure of a CNN.

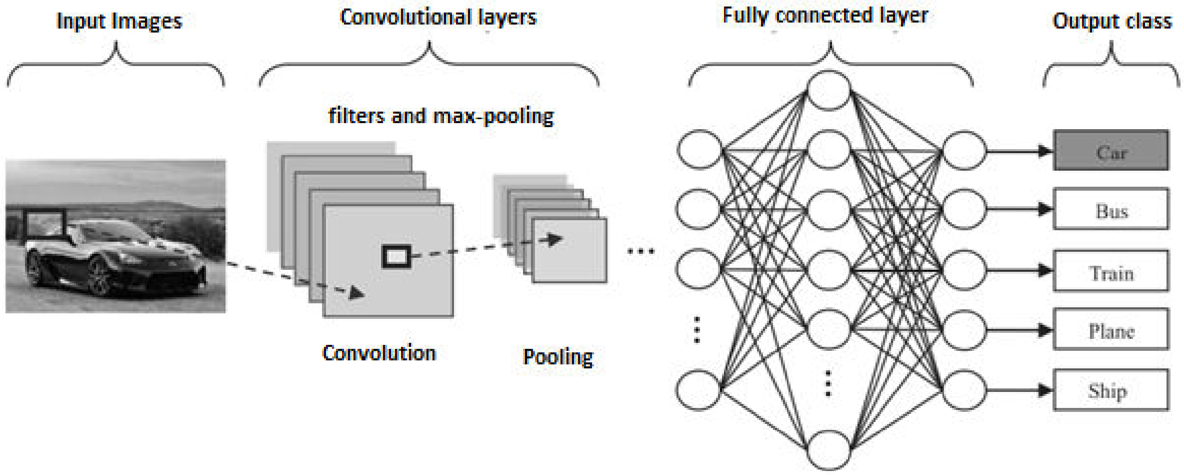


Figure 2-1. CNN general structure (source: Rawat et al., 2017)

The convolutional layers serve as feature extractors, and thus they learn the feature representations of their input images. This is accomplished by having neurons in the convolutional layers arranged to form feature maps. Each neuron in a feature map has a receptive field, which is connected to a neighborhood of neurons in the previous layer via a set of trainable weights. Inputs are then convolved with the learned weights to compute a new feature map, and the convolved results are sent to a nonlinear activation function, which allows for the extraction of nonlinear features (Rawat et al, 2017).

The purpose of the pooling layers is to reduce the spatial resolution of the feature maps to have spatial invariance to input distortions and translations (LeCun et al., 1989a, 1989b; LeCun et al., 1998, 2015). That is, the output feature maps may be sensitive to the location of the features in the input. To eliminate this sensitivity, pooling is used to down sample the feature maps. This has the effect of making the resulting down-sampled feature maps more robust to changes in the position of the feature in the image, referred to as “local translation invariance.” The two most used pooling methods are average pooling and max pooling. Average pooling calculates the average value for each patch on the feature map, whereas max-pooling calculates the maximum value for each patch of the feature map.

Several convolutional and pooling layers are usually stacked on top of each other to extract more abstract feature representations in moving through the network. The fully connected layers that follow these layers interpret these feature representations and perform the function of high-level reasoning. After passing through the fully connected layers, the final layer uses the softmax activation function which is used to get probabilities of the input being in a particular class (classification).

2.3 Vehicle detection and classification using non-CNN methods

There are many challenges associated with vehicle detection and classification. They include (Boukerche et al., 2017):

- Diversity: There are 13 different vehicle classes.
- Multiplicity: Vehicles of the same class have different shapes or designs.
- 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.

Jazayeri et al. (2011) developed a comprehensive approach to localizing target vehicles in videos under various environmental conditions. They continuously extracted vehicle geometry features from the video, projected them onto a 1-D profile, and constantly tracked the vehicles' trajectories. The authors used the temporal information of the features and their motion behaviors for vehicle identification, which compensates for the complexity in recognizing vehicle shapes, colors, and types. They probabilistically modeled the motion in the field of view according to the scene characteristics and the vehicle motion model. The hidden Markov model (HMM) was used to separate target vehicles from the background and track them probabilistically. The authors evaluated the effectiveness of their approach using daytime and nighttime videos of different road types. Experimental results showed an overall detection accuracy of 86.6% for cars and 85.9% for background. The authors concluded that their proposed approach is effective and given that it was implemented in real-time, it would be easy to embed it into hardware for real-time in-car video analysis to detect and track vehicles ahead for safety, auto-driving, and target tracing

Bhaskar et al. (2014) aimed to develop an automatic counting system that counts the number of vehicles passing by a spot during a particular period. They proposed the use of a Gaussian mixture model and blob detection methods. They first differentiated the foreground from the background in frames. Then they developed a foreground detector to detect vehicles and drew rectangular boxes around every detected object. To detect the moving objects correctly, they applied morphological operations. They obtained higher than 91% average accuracy in detection and counting.

Gupte et al. (2002) developed algorithms for vision-based localization and classification of vehicles in monocular image sequences of traffic scenes recorded by a stationary camera. Their proposed system consists of six stages. Stage 1 is segmentation, which separates vehicles from the background in the scene. Stage 2 is region tracking which tracks regions over a sequence of images using a spatial matching method. Stage 3 is the recovery of vehicle parameters, which uses information about the camera's location and makes use of the fact that in a traffic scene, all motion is along the ground plane. Stage 4 is vehicle identification which assumed that a vehicle may be made up of multiple regions and groups the tracked regions from the previous stage into vehicles. Stage 5 is vehicle tracking. Recognizing that a vehicle may consist of multiple regions and a single region might correspond to multiple vehicles, their system tracks vehicles at two levels: the region level and the vehicle level. Lastly, stage 6 is vehicle classification; vehicles that have been detected and tracked are classified. Their system was able to track and classify most vehicles successfully. In a 20-minute sequence of freeway traffic, 90% of the vehicles were correctly detected and tracked. Of these correctly tracked vehicles, 70% of the vehicles were correctly classified.

Avery et al. (2004) developed an image processing algorithm for length-based vehicle classification using an image stream captured by an uncalibrated video camera. The basis of their algorithm is to relatively compare vehicle lengths to each other to estimate truck volumes and eliminate the need for complicated system calibration. Like other studies, their system consists of two distinct processes: background extraction and vehicle detection. Their vehicle detection scheme utilizes a registration line. When a vehicle exits the registration line corresponding to the longitudinal line (that is, the first frame a registration line is unoccupied after being occupied for at least one frame), the length classification algorithm is run to measure the length of the vehicle in pixels. This makes the lengths of all the vehicles in a lane be measured at almost the same starting point so that the measured lengths are comparable. The length algorithm merely steps along the longitudinal line counting the number of different pixels. This length is the length in a projected plane; it does not represent the actual length of the vehicle. Results showed their system was able to classify vehicles with 98% accuracy and trucks with 92% accuracy.

Mithun et al. (2012) proposed a novel detection and classification method using multiple time-spatial images (TSIs), each obtained from a virtual detection line on the frames of a video. The authors found that the use of multiple TSIs provided the opportunity to identify the latent occlusions among the vehicles and to reduce the dependencies of the pixel intensities between the still and moving objects to increase the accuracy of detection performance as well as to achieve an improved classification performance. To identify the class of a particular vehicle, they used a two-step k-nearest neighborhood classification scheme. In the first step, vehicles are grouped into one of the four broad classes, namely 2W, 3W, 4W, and 6W. These classes roughly indicate the relative size of the vehicles. In the second step, each of these broad classes is further grouped into a particular type of vehicle among those available in the traffic. Extensive experimentations were carried out in vehicular traffics of varying environments to evaluate the detection and classification performance of the proposed method, as compared with the existing methods. Experimental results demonstrated that their proposed method provides a significant improvement in vehicle counting and classification accuracy as compared to other methods.

Mishra et al. (2013) developed a real-time algorithm for detecting and classifying different categories of vehicles in a heterogeneous traffic video. The processing of the video was done in four steps: camera calibration, vehicle detection, speed estimation, and classification. Vehicle detection was achieved by using background subtraction and blob tracking method. The speed of the detected vehicle was estimated by utilizing virtual start and stop line markers and calibration parameters. Vehicle classification was done by extracting multiple features of the detected vehicles which serve as input to a support vector machine-based classifier. A histogram-based nonlinear kernel was used in the classifier. The proposed kernel-based classifier was compared to other kernels, namely Gaussian radial basis function kernel and polynomial kernel. Their model development and testing used 50% of 9,360 images for training and the rest for testing. The classification results indicate that their proposed kernel achieved the highest accuracy rate (89%) for heavy motor vehicles and light motor vehicles.

Sowjanya and Chakravarthy (2013) aimed to develop a robust traffic surveillance system for vehicle counting and classification. They adopted a clustering-based feature, called a fuzzy color histogram, which has the ability to greatly attenuate color variations generated by background motions while still highlighting moving objects. Their system first used the consecutive neighboring frame difference to detect the moving regions from the highway scene. Then morphological operations are used to remove the shadow noise and to detect the moving objects. After vehicle detection, a region-based vehicle tracking method is used for building the correspondence between vehicles detected at different time instants. Parameters such as aspect ratio and compactness were used to classify vehicles. Experimental results demonstrated the effectiveness of their approach for background subtraction in dynamic texture scenes compared to several other methods proposed in the literature.

Salvi (2012) proposed an automated vehicle counting system based on blob analysis for traffic surveillance. Their proposed algorithm is composed of five steps: background subtraction, blob detection, blob analysis, blob tracking, and vehicle counting. A vehicle was modeled as a rectangular patch and classified via blob analysis. Meaningful features were extracted by analyzing the blob of vehicles. Tracking of moving targets was achieved by comparing the extracted features and measuring the minimal distance between consecutive frames. The system was tested under different situations (bidirectional and unidirectional flow). The evaluation consists of comparing the automatic count of vehicles in videos against the manual count (ground truth). The detection accuracy was found to be 98.9% on average. It should be noted that their system does not classify vehicles.

Betke et al. (2000) proposed a real-time system that analyzes color videos taken from a forward-looking video camera in a car driving on a highway. Their system used a combination of color, edge, and motion information to recognize and track the road boundaries, lane markings and other vehicles on the road. Cars are recognized by matching templates that are cropped from the input data online and by detecting highway scene features and evaluating how they relate to each other. Cars are also detected by temporal differencing and by tracking motion parameters that are typical for cars. Experimental results using thousands of image frames demonstrated that their system is able to produce robust, real-time car detection and tracking. The authors noted that their system did not perform well in situations when there is low contrast between the cars and the background. Also, it did not perform well at night on city expressways when there are many city lights in the background. In these situations, their system has problems finding vehicle outlines and distinguishing vehicles on the road from obstacles in the background.

Chantakamo (2015) proposed a method for vehicle recognition using video data from surveillance cameras. First, they extracted a set of images from videos using a top-front view. Their algorithm extracts RGB data from images and sets the initial threshold value for classification. Then their developed vision-based algorithm is applied to recognize and classify the size and color of vehicles in the images. Blob analysis was used as feature detection to categorize the vehicle type. The Euclidean distance was applied to extract the color of the car body. Their system can recognize several types of vehicles such as cars, pickups, and trucks. Experimental results indicated that their vehicle detection accuracy was 82% and their vehicle classification accuracy was 80%.

2.4 Vehicle detection and classification using CNN methods

Adu-Gyamfi et al. (2017) proposed a video-based vehicle detection system that classified a vehicle as one of seven groups using a Deep Convolutional Neural Network (DCNN); some of the FHWA’s 13 vehicle classes were combined. Their proposed system decoupled object recognition into two main tasks: localization and classification. The localization task generated class-independent region proposals for each video frame, and the classification task used the DCNN to extract feature descriptors for each proposed region. Lastly, their system scored and classified the proposed regions by using a linear support vector machines template on the feature descriptors. The accuracy of their system was found to vary by vehicle class. Passenger cars and SUVs were classified correctly 95% of the time, and single-unit, single-trailer, and double-trailer trucks were classified correctly between 92% and 94% of the time.

Hu et al. (2018) proposed a Scale-Insensitive convolutional neural Network (SINet) for fast detection of vehicles with a large variance of scales. They first used a context-aware region of interest pooling to maintain the contextual information and original structure of small-scale objects. Then they used a multi-branch decision network to minimize the intra-class distance of features. The authors stated that these lightweight techniques bring zero extra time complexity but significant detection accuracy improvement. Their SINet achieved state-of-the-art performance in terms of accuracy and speed (up to 37 FPS) on the KITTI benchmark and a new highway dataset, which contained a large variance of scales and extremely small objects; the KITTI dataset is a widely used benchmark for vehicle detection algorithms with 7,481 images of various scales of vehicles in different scenes and 7,518 images for testing. The average accuracy was 89% for classifying three vehicle classes, namely cars, busses, and vans.

Zhou et al. (2016) sought to use a Deep Neural Network (DNN) for vehicle detection and classification using rear-view images, captured by a static road camera from a distance along a multi-lane highway. They proposed a combination of approaches to use DNN architectures, building around using the higher layers of a DNN trained on a specific large, labeled dataset. For detection, they fine-tuned a DNN detection model, and for classification, they used both fine-tuning and feature-extraction methods (i.e., AlexNet). Additionally, they proposed methods to use scene transformation and late fusion techniques for classification in poor lighting conditions and achieved promising results without changing the classification model. The authors stated that their classification results outperformed state-of-the-art.

Guo et al. (2017) proposed a model that recognizes motion-blurred vehicle images. The dataset used consisted of 40,749 images combining side, frontal, and rear views. It was divided into 66 categories. The classification accuracy of the GoogleNet model using the blurred images resulted in a classification accuracy of 94.99%, 38.77%, 2.1%, and 0.93% for blur kernel sizes of 0, 5, 11 and 21. To improve the results with blurred images, the authors modified the input data layer of the GoogleNet by generating random motion blur to the images in the training process. The results for the same kernel sizes were notably better at 98.37%, 85.22%, 84.98% and 86.5%, respectively. Similarly, the authors tested the effectiveness of incorporating blur into the training data for different input image sizes. The accuracies obtained were 9.55%, 66.71%, and 94.99% for 64x64,128x128 and 256x256 image sizes, respectively. The results with blur incorporated into training images had accuracies of 91.85%, 98.02%, and 98.37%, respectively. These results showed that their proposed method outperformed the traditional approach of training directly with CNN.

Kim and Lim (2017) proposed a new vehicle type classification scheme for the traffic surveillance system. They proposed four concepts to improve the performance of CNN on images that have different resolutions from multi-viewpoints. These concepts include Deep Learning method, bagging method, data augmentation, and post-processing. They combined these schemes to build a novel vehicle type classification system. Their system showed 97.84% classification accuracy on the 103,833 images in the classification dataset.

Ji et al. (2020) proposed a new Faster R-CNN model with Domain Adaptation (DA) to improve vehicle detection at nighttime. The key element of their work is to make maximum use of labeled daytime images (Source Domain) to help the vehicle detection in unlabeled nighttime images (Target Domain). To evaluate their model, they created a new dataset, named CAU-UTRGV Benchmark, and manually labeled the images. The results indicated that the traditional Faster R- CNN obtained an F-measure of 82.84% on the nighttime vehicle detection, while their proposed method (Faster R-CNN+DA) achieved an F-measure of 86.39% on the nighttime vehicle detection.

Chen et al. (2018) proposed a novel model based on the AdaBoost algorithm and deep convolutional neural networks (CNNs) to classify five distinct groups of vehicles. Their deep CNN model was inspired by VGGNet and AlexNet to directly extract the features of vehicle images. The output layer of their CNN model was taken as the base learner of the AdaBoost algorithm. The results showed that their proposed model attained a classification accuracy of 99.50% on the test dataset and took only 28 milliseconds to classify an image. In addition to being fast, their proposed deep CNN-based feature extractor has fewer parameters, and thereby, uses much smaller storage resources as compared with the state-of-the-art CNN models.

Hasnat et al. (2018) proposed a new vehicle classification method for automatic toll collection. Their method used a set of CNN models, followed by the Gradient Boosting based classifier to fuse the continuous class probabilities with the discrete class labels. The evaluation of their method used a dataset collected from the toll collection cameras at various points of the VINCI Autoroutes French network, and the results showed that their method outperformed the existing automatic toll collection system: 99.02% accuracy compared to 52.77%.

Jo et al. (2018) proposed a transfer learning-based vehicle classification from the CNN pre-trained on a large-scale dataset. Transfer learning is a technique that applies previously learned knowledge to new datasets. The authors stated that transfer learning can achieve better performance with a relatively small dataset. This is because a major assumption of CNN models is that the training and future data must have the same feature space. However, in many real-world applications, this assumption may not hold. In such cases, knowledge transfer, if done successfully, would greatly improve the performance of learning by avoiding expensive data-labeling efforts. Jo et al.'s proposed system has two stages. In the first stage, the vehicle area is detected on the roadway using Haar-like features. In the second stage, the transfer-learning based vehicle classification model classifies vehicles. In the transfer learning, the earlier layers of GoogLeNet pre-trained on ILSVRC-2012 (ImageNet Large Scale Visual Recognition Challenge 2012) are fixed and gradients are backpropagated only through the higher-level portion with the vehicle dataset. Experimental results showed that their proposed system achieved a classification accuracy of 98.3%, which is 32.6% higher than that of the support vector machine without transfer learning.

Yu et al. (2017) proposed a model for fine-grained vehicle classification based on deep learning to handle complicated transportation scenes. Their model consists of two parts: vehicle detection and vehicle fine-grained detection and classification model. Faster Region-based CNN (R-CNN) method was adopted in the vehicle detection model to extract single-vehicle images from an image with a cluttered background that may contain serval vehicles. Then an image that contains only one vehicle was fed into a CNN model to produce a feature, and lastly, a joint Bayesian network was used to implement the fine-grained classification process. Experiments showed that their system was able to recognize a vehicle's make and model from transportation images, with a detection accuracy of 85% and classification accuracy of 89%.

Table 2-1 lists those studies with a similar objective as this study, which is to use a CNN model to classify vehicles traversing on a highway as captured on a video. The number of layers, filter size and optimizer were used to guide the development of this project's CNN models. That is, various combinations of these parameters were evaluated, and the combination that yielded the highest training classification accuracy was selected for evaluation. The chosen parameters are provided in the last row of Table 2-1. Note that this study did not explore different numbers of vehicle groups. The use of four vehicle groups was a requirement specified by the SCDOT.

Table 2-1. Summary of prior studies that used CNN for vehicle classification

Author (year)	Number of Layers	Filter Size	Number of Vehicle Groups	Classification Accuracy
Adu-Gyamfi et al. (2017)	5 layers	11x11	7	94%
Kim and Lim (2017)	5 layers	3x3	11	98%
Chen et al. (2018)	12 layers	3x3	5	99%
Hasnat et al. (2018)	42 layers	7x7	5	99%
Yu et al. (2017)	5 layers	3x3	73 vehicle makes and 208 vehicle models	89%
This project	4 layers	9x9	4	96%

2.5 State DOTs Survey

As part of this study, an online survey was conducted to understand how state DOTs are using video-based systems to collect vehicle and bike/ped data. The survey was distributed to other state DOTs on May 15, 2019, and a response was requested by June 14, 2019. A total of 19 state DOTs responded to the survey, including the SCDOT. Two departments within Nevada DOT completed the survey, and New Mexico's response was not included as they do not use video-based systems. Therefore, there are a total of 19 responses. The following summary first lists the questions in *italics* followed by a summary of the responses.

1. *Indicate the non-intrusive method(s) your agency uses to obtain vehicle classification and/or bike/ped data (check all that apply).*

Table 2-2. Non-intrusive methods used to obtain vehicle classification and/or bike/ped data

Non-intrusive methods	No. of Responses	Percent of Responses
Side-fire radar	9	47.4%
Video based system	12	63.2%
Infrared axle detector	4	21.1%
Radar detector	4	21.1%
Peek data collectors	1	5.3%
Inductive loops (CLR Analytics, loop signatures)	1	5.3%
Infrared pedestrian detectors	1	5.3%

As shown in Table 2-2, the video-based system is the most commonly used non-intrusive method (63.2%) among the respondents, followed by side-fire radar (47.4%). Infrared axle detector and radar detector are used by four respondents each. Three respondents indicated “Other” and specified their specific technologies; these are listed in the last three rows of Table 2-2.

2. *What percentage of vehicle data is obtained via video-based systems (e.g., Miovision Technologies)?*

Table 2-3. Percentage of vehicle data collected using video-based systems

Responses	No. of Responses	Percent of Responses
None	8	42.1%
< 20%	10	52.6%
20% to 40%	1	5.3%
40% to 60%	0	0%
60% to 80%	0	0%
> 80%	0	0%
Total	**19**	**100%**

As shown in Table 2-3, 10 respondents (52.6%) indicated that they were collecting less than 20% of their vehicle data using video-based systems, while eight respondents (42.1%) indicated none. Only one respondent was collecting more than between 20% and no respondents collected more than 40%.

3. *What is the video-based system that your agency uses to collect vehicle data (manufacturer name and model number)?*

Table 2-4. Video-based system used to collect vehicle data

Video-based systems	No. of Responses	Percent of Responses
MioVision	11	57.9%
Gridsmart	2	10.5%
COUNTCam2	1	5.3%
No response	8	42.1%

As shown in Table 2-4, 11 respondents (57.9%) indicated that they were using the MioVision Scout to collect vehicle data, with two (10.5%) using the Gridsmart system, and one using the COUNTCam2 system. Eight respondents (42.1%) did not respond to this question. Note that some state DOTs used more than one video-based system; thus, the total number of responses is greater than 19.

4. *At how many sites are vehicle data being collected via video-based systems?*

Table 2-5. Number of sites where video-based systems are used

Responses	No. of Responses	Percent of Responses
< 20	7	36.8%
20 to 100	1	5.3%
>100	3	15.8%
No response	8	42.1%
Total	**19**	**100%**

As shown in Table 2-5, seven respondents (36.8%) indicated they were collecting vehicle data using video-based systems at less than 20 sites, with three (15.8%) collecting between 20 and 100, and three collecting at more than 100 sites. Eight respondents (42.1%) did not respond to this question.

5. *How often are video-based systems used to collect vehicle data at each site?*

Table 2-6. Frequency of videos-based systems used at each site

Responses	No. of Responses	Percent of Responses
As needed for special situations	4	21.1%
Annually	1	5.3%
Continually	2	10.5%
3-to-4-year cycle	3	15.8%
No response	9	47.3%
Total	**19**	**100%**

As shown in Table 2-6, four respondents (21.1%) indicated that they were using video-based systems to collect vehicle data at each site "as needed for special situations," with one (5.3%) collecting annually, two (10.5%) collecting continually, and three (15.8%) collecting on a 3-to-4-year cycle. Nine respondents (47.3%) did not respond to this question.

6. *Once the data collection is completed via the video-based system, is the vehicle classification performed in-house or outsourced?*

Table 2-7. Method used for classification of collected vehicle data

Responses	No. of Responses	Percent of Responses
In-house	3	15.8%
Outsourced	8	42.1%
No response	8	42.1%
Total	**19**	**100%**

As shown in table 2-7, once the vehicle data collection is completed using a video-based system, three respondents (15.8%) indicated that they were using an in-house system to perform the vehicle count and classification, whereas eight (42.1%) outsourced this work. Eight respondents (42.1%) did not respond to this question.

7. *What is your agency's target accuracy rate for vehicle classification?*

Table 2-8. Target accuracy for vehicle classification

Responses	No. of Responses	Percent of Responses
No criteria	1	5.3%
90% or better	2	10.5%
95% or better	5	26.3%
98% or better	3	15.8%
No response	8	42.1%
Total	**19**	**100%**

As shown in table 2-8, one respondent (5.3%) indicated that it has no specific criteria for classification accuracy, with two (10.5%) having a 90% or better criterion, five (26.3%) having a 95% or better criterion, and three (15.8%) having a 98% or better criterion. Eight respondents (42.1%) did not respond to this question.

8. *When collecting traffic data by video-based systems, how many vehicle groups or bins are used by your agency? Please specify the groups/bins (e.g., bin 1 includes FHWA vehicle classes 1, 2 and 3).*

Table 2-9. Number of vehicle groups used for classification

Responses	No. of Responses	Percent of Responses
3	5	26.3%
4-6	5	26.3%
No response	9	47.4%
Total	**19**	**100%**

As shown in Table 2-9, five respondents (26.3%) indicated that they were using three vehicle groups, and with five (26.3%) using four to six groups. Nine respondents (47.4%) did not respond to this question.

9. Is the video-based system collecting vehicle data at night?

Table 2-10. Usage of video-based system at night

Responses	No. of Responses	Percent of Responses
Yes	9	47.4%
No	2	10.5%
No response	8	42.1%
Total	**19**	**100%**

As shown in Table 2-10, nine respondents (47.4%) indicated that they were using the video-based system to collect vehicle data at night, and two (10.5%) did not. Eight respondents (42.1%) did not respond to this question.

10. What percentage of bike/ped data are obtained via video-based systems (e.g., Miovision Technologies)?

Table 2-11. Percentage of bike/ped data obtained via video-based systems

Responses	No. of Responses	Percent of Responses
<20%	7	36.8%
20% to 40%	0	0%
40% to 60%	0	0%
60% to 80%	0	0%
Not used	3	15.8%
No response	8	42.1%
Total	**19**	**100%**

As shown in Table 2-11, seven respondents (36.8%) indicated that they were collecting less than 20% of their bike/ped data using video-based systems. Three respondents (15.8%) that they did not use their video-based systems to collect bike/ped data. Eight respondents (42.1%) did not respond to this question.

11. What is the video-based system that your agency uses to collect bike/ped data (manufacturer name and model number)?

Table 2-12. Video-based system used to collect bike/ped data

Video-based systems	Response Count	Percent of Responses
MioVision	7	36.8%
Gridsmart	1	5.3%
Prototype system	1	5.3%
No response	11	57.9%

As shown in Table 2-12, seven respondents (36.8%) indicated that they were using the MioVision Scout to collect bike/ped data, with one using the Gridsmart system, and one using the Prototype system. Eleven respondents (57.9%) did not respond to this question. Note that some state DOTs used more than one video-based system; thus, the total number of responses is greater than 19.

12. At how many sites are bike/ped data being collected via video-based systems?

Table 2-13. Number of bike/ped sites where video-based systems are used

Responses	No. of Responses	Percent of Responses
< 20	6	31.6%
< 200	2	10.5%
No response	11	57.9%
Total	**19**	**100%**

As shown in Table 2-13, six respondents (31.6%) indicated they were collecting bike/ped data at less than 20 sites using their video-based systems, with two (10.5%) collecting at more than 20 but less than 200 sites. Eleven respondents (57.9%) did not respond to this question.

13. How often are video-based systems used to collect bike/ped data at each site?

Table 2-14. Frequency of videos-based systems used at each bike/ped site

Responses	No. of Responses	Percent of Responses
As needed for special situations	5	26.2%
Annually	1	5.3%
Continually	1	5.3%
3-to-4-year cycle	1	5.3%
No response	11	57.9%
Total	**19**	**100%**

As shown in Table 2-14, five respondents (26.2%) indicated that they were using their video-based systems to collect bike/ped data at each site "as needed for special situations," with one (5.3%) collecting annually, one (5.3%) collecting continually, and one (5.3%) collecting on a 3-to-4-year cycle. Eleven respondents (57.9%) did not respond to this question.

14. Once the data collection is completed via the video-based system, is the bike/ped video processing done in-house or outsourced?

Table 2-15. Method used for classification of collected bike/ped data

Responses	No. of Responses	Percent of Responses
In-house	5	26.3%
Outsourced	3	15.8%
No response	11	57.9%
Total	**19**	**100%**

As shown in table 2-15, once the bike/ped data collection is completed using a video-based system, five respondents (26.3%) indicated that they were using an in-house system to perform the bike/ped count and classification, whereas three (15.8%) outsourced this work. Eleven respondents (57.9%) did not respond to this question.

15. What is your agency's target accuracy rate for bike/ped count?

Table 2-16. Target accuracy for bike/ped count

Responses	No. of Responses	Percent of Responses
No criteria	1	5.3%
90% or better	1	5.3%
95% or better	4	21%
98% or better	1	5.3%
No response	12	63.1%
Total	**19**	**100%**

As shown in table 2-16, one respondent (5.3%) indicated that it has no specific criteria for bike/ped counting and classification accuracy, with one (5.3%) having a 90% or better criterion, four (21%) having a 95% or better criterion, and one (5.3%) having a 98% or better criterion. Twelve respondents (63.1%) did not respond to this question.

16. Is the video-based system collecting bike/ped data at night?

Table 2-17. Usage of video-based system to collect bike/ped data at night

Responses	No. of Responses	Percent of Responses
Yes	7	36.8%
No	1	5.3%
No response	11	57.9%
Total	**19**	**100%**

As shown in Table 2-17, seven respondents (36.8%) indicated that they were using their video-based systems to collect bike/ped data at night, and one (5.3%) did not. Eleven respondents (57.9%) did not respond to this question.

CHAPTER 3: METHODOLOGY

3.1 Introduction

The method utilized in this project consists of both hardware and software. The hardware consists of either a visible camera mounted onto a road-side pole, or an infrared camera mounted onto a solar-powered trailer that is positioned approximately six feet from the edge of the road. The developed software takes the visible or thermal videos as input and outputs the time and type of vehicle detected in the video, as well as an overall count of each vehicle group. The identification of each vehicle was made more accurate by mapping object positions (at every two frames) into a track for determination of whether such a trajectory resembles one from a vehicle. The classification of each vehicle was made more accurate by recording the vehicle's classification at every two frames and then selecting the one that has the highest count (i.e., mode).

3.2 Hardware

In the first part of the project, visible (referred to as RGB hereafter) videos collected previously by the SCDOT using the Miovision Scout were used to train the CNN model. When it became apparent that the RGB videos are unsuitable for nighttime use, a thermal camera was procured, assembled, and deployed at various locations to obtain the necessary data.

3.2.1 Miovision camera set up

The Miovision Scout as shown in Figure 3-1 was mounted onto telephone poles, about 20 feet above the ground, by the SCDOT to collect traffic data. The camera was aimed in the direction of traffic, and thus, provided both the rear and side view of vehicles as they passed the observation point. Given that the battery life of the Miovision Scout lasts up to seven days, no external power was needed for the typical 48-hour data collection. The resolution of the Scout videos is 720 by 480 pixels, and the frame rate is 30 frames per second.

Figure 3-1. Miovision Scout (source: Miovision.com)

3.2.2 FLIR thermal camera set up

The TraficSense2 Dual camera was procured from FLIR midway through the project. Unlike other off-the-shelf cameras, the FLIR TrafiSense2 Dual camera (referred to as thermal camera hereafter) required connections to other components in order for it to work. As shown in Figure 3-2, the thermal camera is connected to a PoE (Power over Ethernet) module. It is also connected to a Raspberry Pi via a network switch; the Raspberry Pi runs a recorder software that stores the video data onto an external hard drive. The recorder software is launched and configured using a laptop. Once configured, the laptop does not need to be connected to the thermal camera. Instructions for using the recording software for data collection are provided in Appendix A.

To power the thermal camera via the PoE, a solar-powered trailer was procured as part of this project. The following are the specifications of the trailer that was purchased.

- 20 feet tall crank up mast
- Lead-acid batteries to provide ≥ 100 watts
- Solar panels and power regulator to provide continuous power
- Extendable jack stands
- Standard trailer hitch for towing behind a light-duty pick up or SUV
- Lockable metal battery box
- On-board battery charger
- Large plastic NEMA Box large enough to house thermal camera components
- Exterior dimensions of pelican case (in inches): length=21.96, width=13.97, height=8.98
- A sturdy platform near the top of the mast to mount and secure FLIR TrafiSense2 Dual camera

Figure 3-3 shows how the thermal camera was mounted onto the trailer during one of the field deployments. For this particular deployment, the camera was mounted at 18 feet above the ground and 6 feet away from the edge of the road. To minimize camera motion due to wind, the platform to which the camera is affixed was secured by two cables, one on each side to the base (see Figure 3-3). Instructions for setting up the trailer are provided in Appendix B. Although the FLIR TrafiSense2 Dual camera can record traffic data using both of its visible and thermal cameras simultaneously, only the thermal camera was used in this project to allow for the longer recording of the video. Each thermal video segment is 10 minutes long. Thus, for a 48-hour count, there will be 288 10-minute video files. The resolution of the thermal videos is 640 by 480 pixels, and the frame rate is 30 frames per second.

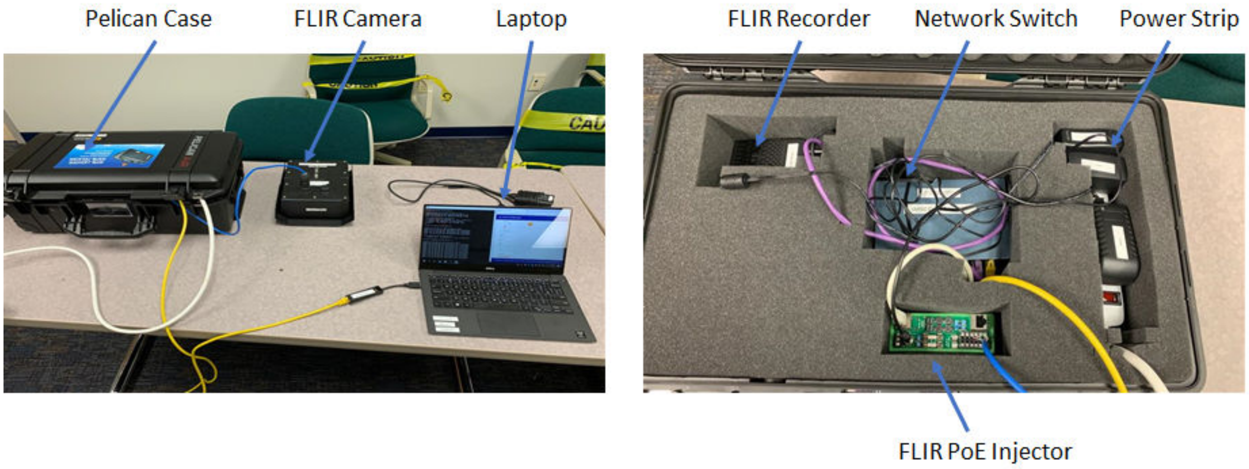


Figure 3-2. FLIR TrafiSense2 Dual camera components and assembly

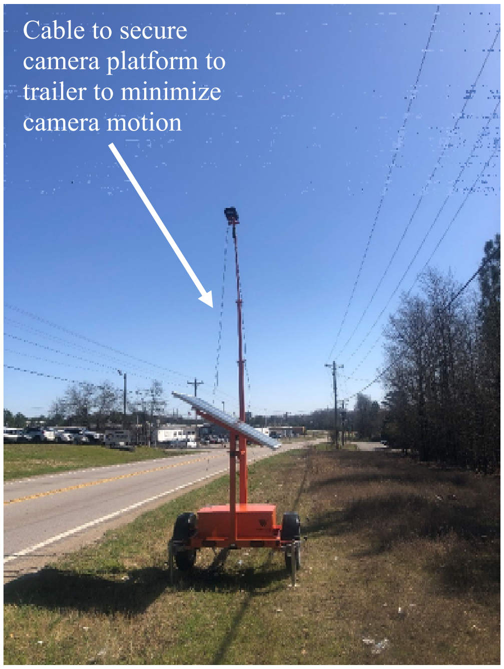

Cable to secure camera platform to trailer to minimize camera motion

Figure 3-3. Thermal camera and trailer at data collection site

3.3 Video Processing

The goal of video processing is for an observer, human or machine, to extract useful information about the scene being imaged (Kang, 2007). In this project, the interest was to have the machine automatically count and classify vehicles in a video. To accomplish this, several video processing operations were implemented. The technical details of these operations are provided in subsequent sections.

3.3.1 Vehicle detection

The term "vehicle detection" used in this report and hereafter refers to the task of detecting moving vehicles in a video. In the literature, this task is sometimes referred to as localization and sometimes as detection. Since the term detection is more easily understood, it is used here instead of localization. In this project, vehicle detection is accomplished using a standard procedure as shown in Figure 3-4. The procedure requires an input sequence which is a color or grayscale traffic video. The first step in detecting the vehicles is to perform motion object segmentation which involves determining whether individual pixels are part of the background or the foreground. The background of a typical traffic video generally consists of roads, buildings, trees, and poles. These fixed objects have the same pixel positions from frame to frame or have very slight changes to them from frame to frame. Camera motion, due to high winds, can cause background objects to appear to have motion. Any foreground objects that are deemed not to be a vehicle due to invalid trajectories are removed by the developed tracking algorithm (discussed in Section 3.3.5).

3.3.2 Motion object segmentation

The sequence of operations used to identify background versus foreground objects is as follows. First, background initialization is performed using a fixed number of frames to build a statistical model of the background pixels. Next, foreground detection is performed by subtracting each current frame from the background model. Those pixels that are statistically different from the background are deemed as foreground. The last step is background maintenance, which involves analyzing the current frame to update the background statistical model. The second and third steps are repeated for each frame in the video. In this project, MATLAB's Foreground Detection function (MathWorks, 2021) was used to perform these two steps. The specific model used by MATLAB for this procedure is the Gaussian Mixture Model (GMM). Given an input sequence shown in Figure 3-5(a), the outcome of using the GMM to separate the foreground from the background is shown in Figure 3-5(b).

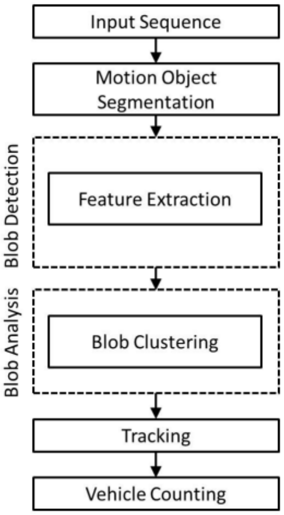


Figure 3-4. Detection system overview (Source: Salvi, 2012)

3.3.3 Blob detection

The motion object segmentation step identified foreground pixels in the current frame. In the next step, blob detection, these foreground pixels are grouped together by a contour detection algorithm. In this project, MATLAB's Deep Learning Object Detector algorithm was used. The contour detection algorithm groups the individual pixels into disconnected classes, and then finds the contours, in the form of a bounding box, surrounding each class. Each class is marked as a candidate blob, and these candidate blobs are then checked for their sizes. Blobs that have areas less than 200 squared pixels or greater than 32,000 squared pixels are removed; these parameter values were identified as near optimal through a trial-and-error process.

3.3.4 Blob analysis

The blob analysis step takes the remaining blobs with their positions from the previous step, as input, and identifies which blobs in the current frame belong to the same vehicle. This step involves comparing the positions of the blobs in the current frame to those in the previous frame using the k-Means clustering technique using the centroid of each blob's bounding box. In this project, MATLAB's Blob Analysis function was used to accomplish both the blob detection and blob analysis steps. The visual output of the blob analysis step is shown in Figure 3-5(c). In this project, the frame numbers, blob IDs, bounding boxes' widths and lengths, bounding boxes' centroid coordinates, and vehicle group classifications are stored at every two frames. This information is used in the tracking step to improve the counting of vehicles.

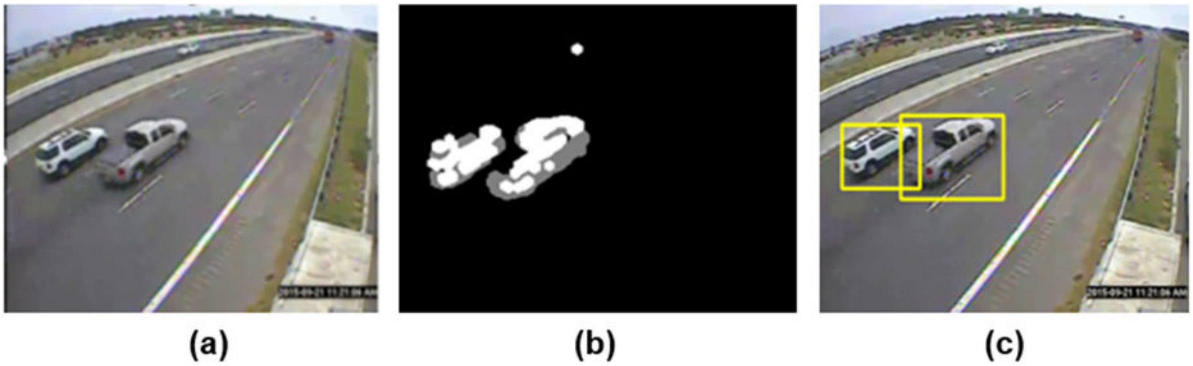


Figure 3-5. (a) Video frame, (b) background subtraction, and (c) blob analysis

Lastly, the tracking algorithm is run after the video is processed. The tracking algorithm constructs object tracks to enable the determination of whether a detected foreground object is a vehicle or not. An object's trajectory is simply a plot of its centroid coordinates from the frame it entered the region of interest to the frame that it exited the region of interest. The centroids' pixel coordinates represent the physical locations of the objects. These coordinates are represented as nodes, and if two nodes belong to the same object then they are connected by a link. Thus, an object's trajectory indicates the path or track of the object. Figure 3-6 shows a sample set of tracks. Note that the plotted tracks do not correspond with the movement of vehicles shown in Figure 3-5. Specifically, the tracks in Figure 3-6 move in the south-east direction, whereas vehicles in Figure 3-5 move in the north-east direction. The reason is due to the difference in origin. In Figure 3-5, the origin is located on the upper-left corner, whereas in Figure 3-6, the origin is located in the bottom-left corner.

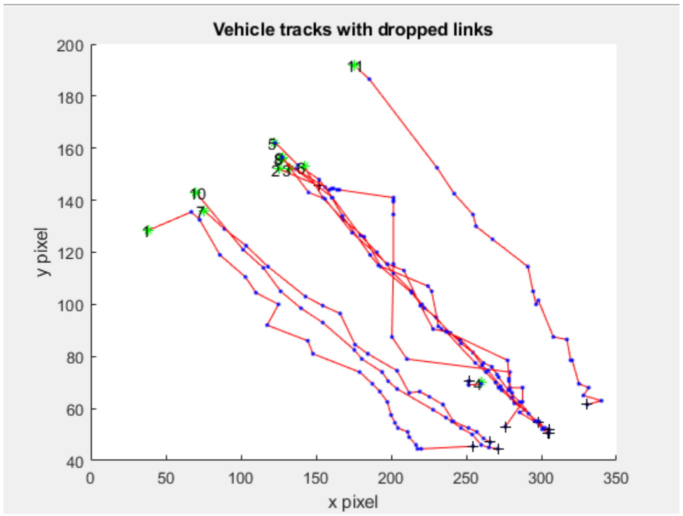


Figure 3-6. Vehicles tracks

The logic for determining whether an object is a vehicle or not is algorithmically performed by examining its track. If a track remains after the following operations, then it is a valid track, and therefore, that object must be a vehicle.

- **Operation 1**: If a track has a link that goes against the direction of traffic flow, then that link is removed. As a result, the track is split into two. This process is repeated for all links in a track.
- **Operation 2**: If the starting node of a track is within 15 frames of another track's starting node and these two tracks have similar angles, then the two tracks are joined. This process is repeated for all tracks.
- **Operation 3**: If a track's total number of nodes is lower than the 35th percentile of the numbers of nodes in all tracks, then it is removed. The reason for this operation is that short tracks are unlikely to be those of a vehicle. It should be noted that the k-means clustering algorithm was also evaluated, but it did not yield a better detection accuracy.
- **Operation 4**: If a track's starting node is not in the upper half of the region of interest, then it is removed. The reason for this operation is that a valid track cannot suddenly appear in the middle of the region of interest.

The remaining number of tracks is the number of vehicles. The classification of the vehicle group for each track is determined by using the mode of the classification of the image of each node in the track. In other words, the vehicle group that is classified most often for the object, among the nodes in the track, is selected as its vehicle group for the track.

3.3.5 <u>Vehicle classification</u>

Figure 3-4 shows the standard steps typically taken for vehicle counting applications. In this project, the classification of vehicles was also needed. Thus, steps in Figure 3-4 were modified slightly. To determine which of the four vehicle groups a detected foreground object is, a CNN model is used for classification. This step is performed after Blob Detection. Four types of layers were used to build the convolutional network architecture: Convolutional + ReLU Layer, Pooling Layer, Fully Connected Layer, and SoftMax Layer. MATLAB was used to implement the CNN model. Figure 3-7 provides an overview of the CNN model building steps.

For this project, the input to the CNN model is a 72x72x3 image; that is an image with a width of 72 pixels, height of 72 pixels, and three-color channels: Red, Green, and Blue. These are images within the bounding boxes shown in Figure 3-5(c). If an image's size is smaller than 72x72, then white spaces are padded around the image. Next, a convolutional layer with 20 filters is applied. This is the first step of the Feature Learning component shown in Figure 3-7. Each filter, 9x9 in size, activates certain features from the images (e.g., edge detect, headlights detect, tires detect). For this reason, the filter is also known as Feature Detector or Kernel. The result of the convolution process is 20 feature maps because 20 filters are specified. Next, the Rectified Linear Unit (ReLU) activation function is applied to break up any linear progression of pixel colors because images are highly non-linear. Moreover, ReLU allows for faster and more effective training by mapping negative values to zero and maintaining positive values. Through this activation function, only the activated features are carried forward into the next layer. Next, the pooling function is applied. This project uses max pooling with a size of 2x2. This step involves taking a box of 2x2 pixels from the Feature Map created previously, finding the maximum value, and outputting it to the Pooled Feature Map. Then the box is moved to the next 2x2 portion of the Feature Map to find its maximum pixel value, and this step is repeated for the entire feature map. A benefit of pooling is that it reduces the dimensionality of the image and reduces the number of parameters that the network needs to learn. The sequence of operations (convolution, RELU, and pooling) is repeated four times. The only difference in these iterations is that the fourth convolutional layer has 40 filters instead of 20. After the model learned the image features, the architecture of the CNN shifts to classification. The fully connected layer is applied to output a vector of K dimensions where K is the number of classes that the network will be able to predict. In this project, K = 5 (four vehicle groups and background). The K vector contains the probabilities for each class of any image being classified. The final layer of the CNN architecture uses the softmax function to provide the classification output. The softmax function is a generalization of the logistic function, and it is used to ensure that the predicted probabilities of all classes add up to 1.

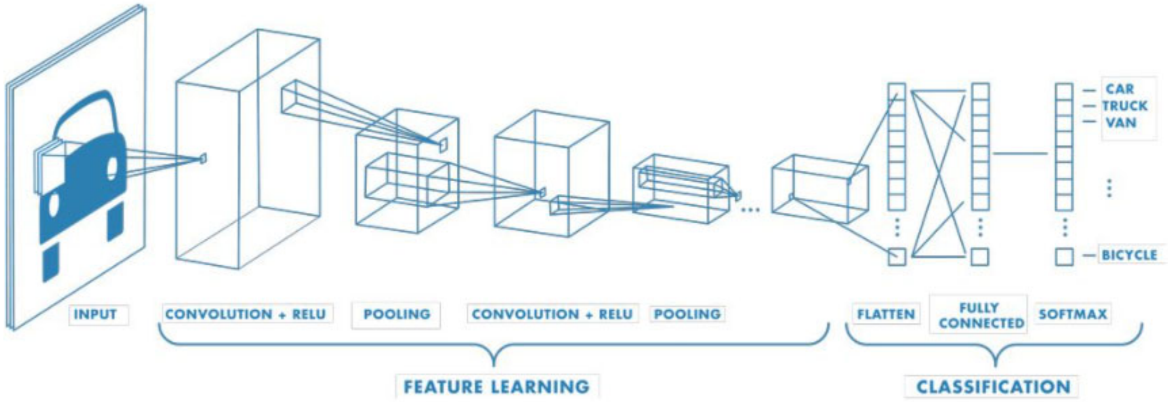


Figure 3-7. CNN architecture (source: MathWorks, 2021)

3.3.6 <u>CNN model training</u>

In developing the convolutional network architecture discussed in the previous section, various parameters were evaluated to produce the highest possible classification accuracy. These include the number of layers, number of filters, and filter size. For our datasets, the best performing combination required 4 layers, 20 filters for the first three convolutional layers, 40 filters for the fourth convolutional layer, and a 9x9 pixel filter size. The CNN model was then trained using manually labeled images (discussed in Section 3.5). MATLAB allows the user to define the global training parameters to converge faster and/or obtain higher classification accuracy. In general, a model's convergence speed and accuracy are dependent on the following aspects.

- The architecture of the network
  - The number of layers
  - The number of parameters in each layer
  - Activation functions used
  - Other architectural details
- The dataset and the complexity of the problem
- Learning algorithm
- Hyperparameters
  - Learning rate

    - Dropout rate
    - Weight decay
- Loss function
    - Weight initialization
    - Random
- Pre-trained model

In this project, the focus was on increasing the accuracy of the CNN model. The model training time was not a concern. Two parameters were found to have a significant impact on the CNN model classification accuracy: filter size and learning rate. Three different filter sizes were evaluated: 3x3, 6x6, and 9x9 pixels. Similarly, three different learning rates were evaluated: 0.1, 0.01, 0.001. The optimal values for these parameters are 9x9 pixels for filter size and 0.001 for learning rate.

3.4 Software

To facilitate the use of the developed code for video processing, a stand-alone application named DECAF (detection and classification by functional class) was developed. Essentially, a Graphical User Interface (GUI) was built around the video processing functions explained in Sections 3.3.1 to 3.3.6. The GUI, with a pull-down menu and toolbar, allows the user to easily identify the folder where the to-be-processed video files are stored, draw the area (referred to as the region of interest) where vehicles should be detected and classified, process the videos, and generate reports. As shown in Figure 3-8, the use of DECAF involves three steps. The first step requires the user to provide the input videos, region of interest and desired time interval for data to be aggregated. The second step utilizes the developed video processing functions to detect and classify vehicles. The final step enables the user to view the results, either in a PDF report with summarized statistics or in a CSV file with vehicle-by-vehicle information. A user's manual for DECAF is provided in Appendix C.

3.5 Data generation

A simplified version of DECAF was used to generate images to train and validate the developed CNN model. Specifically, at every frame of the video, the detected foreground objects within the bounding boxes are outputted as 72x72 pixel images. If an image's size is smaller than 72x72, then white spaces were padded around the image. These images were outputted into their respective categories/folders based on their classified vehicle groups. In this project, four vehicle groups are used as specified by the SCDOT. These four vehicle groups as they relate to the FHWA's 13 vehicle category classification are as follows.

- Vehicle group 1: FWHA Class 1 vehicles
- Vehicle group 2: FHWA Classes 2-3 vehicles
- Vehicle group 3: FHWA Classes 4-5 vehicles
- Vehicle group 4: FHWA Classes 6-13 vehicles

Before the images were used for training and validating the CNN model, they were manually reviewed and placed into the correct categories/folders. That is, if an image was classified by the model as group 1, but it is actually a group 2 vehicle, then that image was moved from the category/group 1 to the category/group 2 folder. This process is repeated for every image generated from a video. Images which the researchers could not determine their vehicle groups, due to poor image quality, were removed. Images that contain only the background were placed in a separate category/folder called background; background images are sometimes detected as foreground objects due to camera motion or changing lighting conditions.

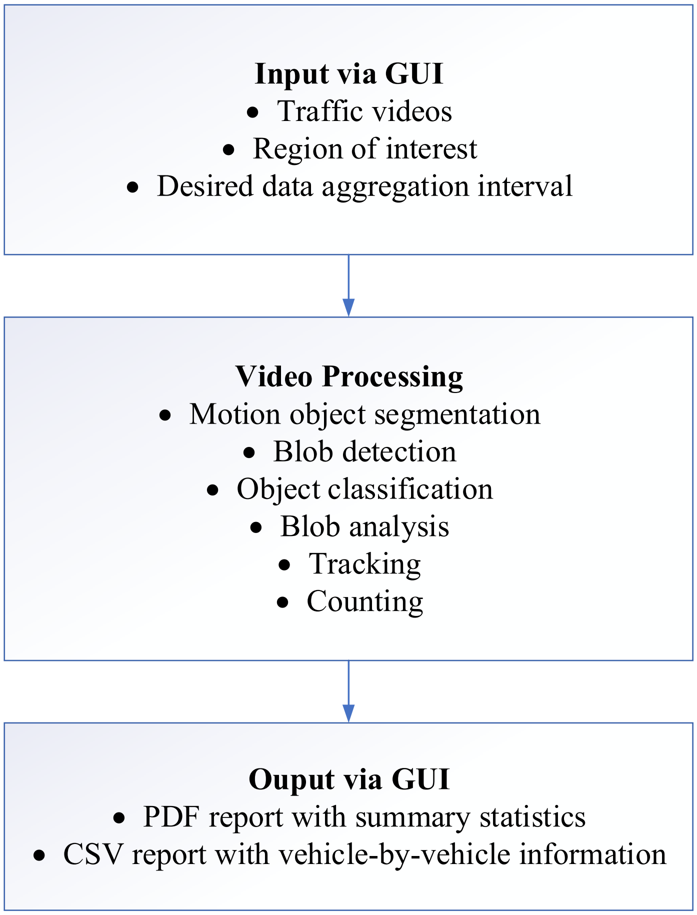


Figure 3-8. DECAF primary elements and functionalities

CHAPTER 4: FINDINGS AND DISCUSSIONS

4.1 Introduction

This project developed three different CNN models. At the start of the project, it was envisioned that two models will be needed, one for daytime and one for nighttime, using the RGB traffic video data that the SCDOT had collected. As shown below, the classification results of the nighttime CNN model were poor despite attempts to enhance the nighttime images prior to using them for model training, as well as attempts to enhance the convolutional network architecture for the nighttime model. For these reasons, thermal traffic video data were collected and used to develop the thermal CNN model. The results, when used within DECAF, yielded a counting and classification accuracy of at least 95%, which was the aim of this project. Where the detection region is drawn, relative to the camera position, was found to have a significant effect on the detection and classification results. The trailer offset distance was also found to have a significant effect on the results. Their results and guidelines are provided in subsequent sections.

4.2 Effect of placement of the region of interest on counting accuracy

Due to the camera angle and height, it was noticed that the location and size of the region of interest (ROI), relative to the camera position, affected the counting and classification accuracies. Three commonly used approaches by the researchers were evaluated. The first approach is to draw the ROI to cover as large an area as possible as shown in Figure 4-1(a). The benefit of this approach is to provide more frames to detect and track the vehicles. A second approach is to draw the ROI much further downstream as shown in Figure 4-1(b). The benefit of this approach is that it provides a vantage point where vehicles in each lane are clearly visible (i.e., vehicles in the left-most lane are not blocked by a larger vehicle in the middle lane). The third approach is to draw the ROI to cover the middle third of the lanes as shown in Figure 4-1(c). The benefit of this approach is that it provides the greatest change in the angle/slope of the tracks. To determine which approach produces the highest counting accuracy, ten videos were randomly selected for the evaluation, with each video containing at least 10 vehicles. The results are shown in Table 4-1.

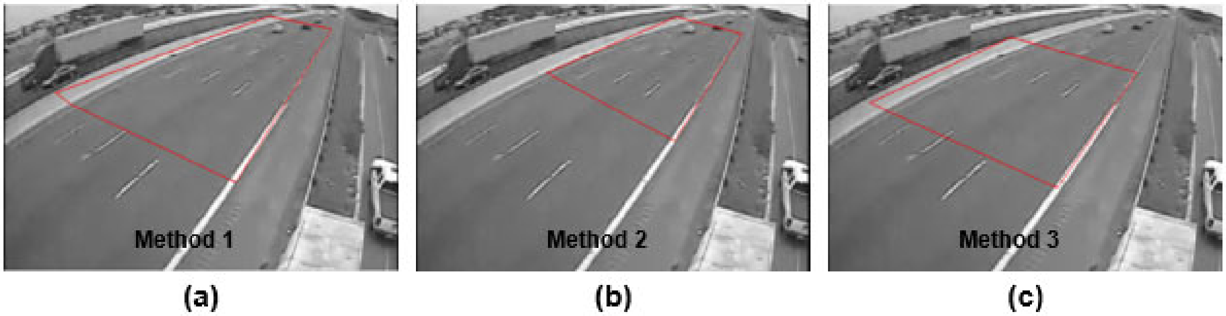


Figure 4-1. Different methods to draw the region of interest

Table 4-1. Counting accuracies for different ROI-drawn methods

Videos	Method 1 accuracy (%)	Method 2 accuracy (%)	Method 3 accuracy (%)
Video 1	41.2	88.2	47.1
Video 2	80	90	80
Video 3	78.9	89.5	94.7
Video 4	65.5	96.6	72.4
Video 5	24.1	93.5	15.7
Video 6	25	92.1	60.5
Video 7	78.9	89.5	63.2
Video 8	84.4	97.8	82.2
Video 9	88.9	94.4	83.3
Video 10	95.2	95.2	95.2
Average	66.2	92.7	69.4

Using the results in Table 4-1, a paired t-test was performed to determine if it can be concluded at the 95% confidence level that the mean counting accuracy of method 2 is greater than that of method 1. The null and alternative hypotheses are as follows.

H_0: $\mu_1 = \mu_2$
H_A: $\mu_2 > \mu_1$

The paired t-test was performed using the R software (R Core Team, 2021), and it returned a p-value of 0.005125. Since the p-value is less than 0.05, the null hypothesis can be rejected. Thus, it can be concluded that method 2 is better than method 1.

Similarly, a paired t-test was performed to determine if it can be concluded at the 95% confidence level that the mean counting accuracy of method 2 is greater than that of method 3. The null and alternative hypotheses are as follows.

H_0: $\mu_2 = \mu_3$
H_A: $\mu_2 > \mu_3$

A p-value of 0.006498 was obtained. Therefore, it can be concluded that method 2 is better than method 3.

The above results indicated that method 2 is superior to methods 1 and 3. Thus, it is the recommended method. All results reported below used method 2 to draw the ROI in DECAF.

4.3 DECAF with RGB daytime CNN model results

Using the RGB traffic video data collected by the SCDOT using Miovision cameras on multilane highways in 2015, a total of 314,684 daytime images was generated, reviewed, and manually labeled according to their actual vehicle groups. Of these, 172,223 were selected for the training dataset and the remainder (142,461) were used for the test dataset. To avoid bias in the training and testing of the CNN model, the images from one video or site were not included in both datasets. That is, images from one set of videos were used for training and images from another set of videos were used for testing. This approach ensured that the CNN model does not see images in testing similar to those it was trained with. Table 4-2 shows the number of images per class in the training and test datasets.

Table 4-2. RGB daytime datasets

Class	Training	Test
Background	114,244	88,396
Class 1	109	90
Class 2-3	32,944	29,997
Class 4-5	809	715
Class 7-13	22,117	23,263
Total	**172,223**	**142,461**

The RGB daytime CNN model trained with the data shown in Table 4-2 yielded a classification accuracy of 92.70%; note that this is the training accuracy. To test the performance of the model, it was utilized within DECAF. Three RGB traffic videos with at least 50 vehicles in each were used for testing. The locations where these were recorded are unknown. Therefore, these videos are labeled as “Site A”, “Site B”, and “Site C.” Table 4-3 shows testing results for Site A. In the first column, instead of presenting the vehicle groups (1 to 4), their corresponding FHWA classes are listed since this information is more well known to readers. The second and third columns show the results of DECAF without the tracking component. The fourth and fifth columns show the results of DECAF with the tracking component. The reason for showing the results with and without the tracking component is because it was the most challenging task intellectually and most time-consuming task to implement. It is also the most unique contribution of this project. The last column shows the actual count, which was obtained by manually reviewing the video and manually counting the number of vehicles in each classifications.

Table 4-3. DECAF results with RGB daytime CNN model for Site A

Classes	**Without tracking**		**With tracking**		**Actual Count**
	Count	Over/Under	Count	Over/Under	
Class 1	0	0	0	0	0
Class 2-3	61	+12	51	+2	49
Class 4-5	0	0	0	0	0
Class 6-13	0	-1	0	-1	1
Total	61	+11	51	+1	50
Total misclassified		13	0	3	

The results in Table 4-3 indicates that without the tracking component, DECAF yielded a count of 61 vehicles compared to an actual count of 50. Thus, it overcounted by 11 vehicles, resulting in an error of 22%. In terms of classification, it misclassified 13 vehicles, an error of 26%. The misclassification error is computed by dividing the total number of vehicles misclassified by the actual total number of vehicles. With the tracking component, the count is over by 1 vehicle (2% error) and the misclassification is reduced to 3 (6% error).

Table 4-4. DECAF results with RGB daytime CNN model for Site B

Classes	Without tracking		With tracking		Actual Count
	Count	Over/Under	Count	Over/Under	
Class 1	0	0	0	0	0
Class 2-3	81	-4	68	-17	85
Class 4-5	2	-1	0	-3	3
Class 6-13	65	+45	37	+17	20
Total	148	+40	105	-3	108
Total misclassified		50	0	37	

The results in Table 4-4 indicate that without the tracking component, DECAF yielded a count of 148 vehicles compared to an actual count of 108. Thus, it overcounted by 40 vehicles, resulting in an error of 37%. In terms of classification, it misclassified 50 vehicles, an error of 46%. With the tracking component, the count is under by three vehicles (3% error) and the misclassification is reduced to 37 (34% error). Note that the reported misclassification does not reflect the fact that of the 68 vehicles classified as Class 2-3, some could actually be Class 6-13. Similarly, some of the 37 vehicles classified as Class 6-13, some could actually be Class 2-3. In other words, the researchers did not manually verify the classification of each vehicle reported by DECAF. The comparison is based on total actual counts and not a vehicle-by-vehicle comparison.

Table 4-5. DECAF results with RGB daytime CNN model for Site C

Classes	Without tracking		With tracking		Actual Count
	Count	Over/Under	Count	Over/Under	
Class 1	0	0	0	0	0
Class 2-3	74	+16	53	-5	58
Class 4-5	1	-1	1	-1	2
Class 6-13	47	+31	23	+7	16
Total	122	+46	77	+1	76
Total misclassified		48	0	13	

The results in Table 4-5 indicates that without the tracking component, DECAF yielded a count of 122 vehicles compared to an actual count of 76. Thus, it overcounted by 46 vehicles, resulting in an error of 60.5%. In terms of classification, it misclassified 48 vehicles, an error of 63%. With the tracking component, the count is over by one vehicle (1% error) and the misclassification is reduced to 13 (17% error).

4.4 RGB nighttime CNN model results

The RGB nighttime images, under poor lighting conditions, presented an insurmountable challenge. It was difficult to determine, even to the human eye, how many axles a truck has, as shown in Figure 4-2(a) and whether the vehicle is of Class 2 or 3 as shown in Figure 4-2(b). Therefore, after the manual review and sort process, only a small number of images remain in the RGB nighttime dataset (less than 5,000 images). Due to the small sample size, the classification accuracy of the nighttime CNN model was well below the target 95%. To overcome this issue, four different low-light image enhancement methods were explored: 1) histogram equalization

(HE), 2) adaptive histogram equalization (AHE), 3) contrast limited adaptive histogram equalization (CLAHE), and 4) gamma correction.

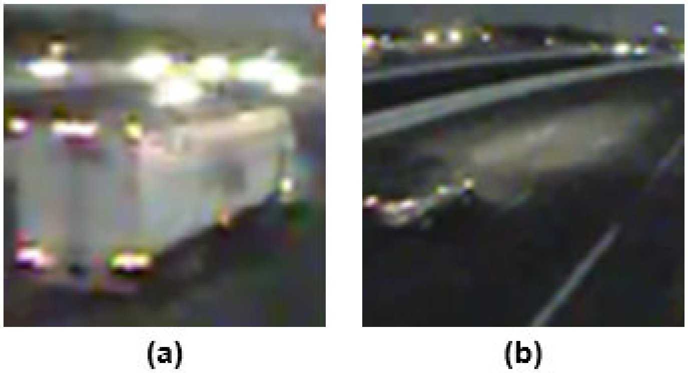


Figure 4-2. RGB nighttime images

The HE method aims to produce an output image with pixel values evenly distributed throughout the range. The AHE method operates on small regions of the image, called tiles. Each pixel is transformed based on the histogram of a square surrounding the pixel. The AHE method tends to overamplify noise in relatively homogeneous regions of an image. CLAHE is a variant of AHE designed to prevent this by limiting the amplification. The gamma correction method controls the overall brightness of an image. Figure 4-3 shows the effect of applying these four image enhancement methods. The intent was to use a combination of these methods to enhance the nighttime images to make their vehicle groups easier to distinguish for the researchers and subsequently the CNN model. Due to the low quality of the original images, none of the combinations of enhancement methods applied resulted in a better image. For this reason, in consultation with the SCDOT, it was decided that thermal traffic video data should be used.

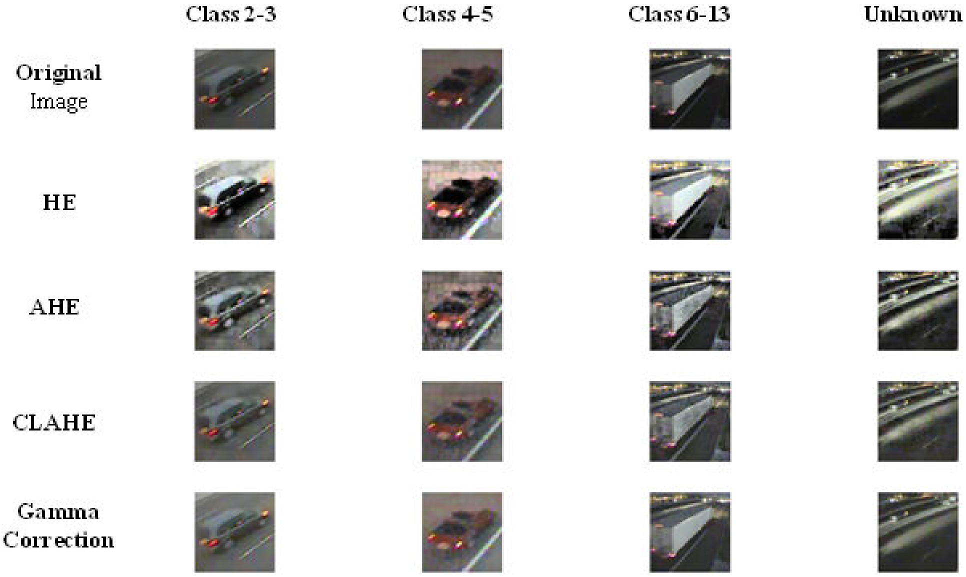


Figure 4-3. RGB nighttime images after enhancement

Figure 4-4 shows thermal images of traffic as recorded by the FLIR TrafiSense2 Dual camera. Figure 4-4(a) shows an image of a car taken at 9 a.m. EST and Figure 4-4(b) shows an image of a car taken at 11 p.m. EST. Notice that although one image was captured during the day and one was captured at night, there is very little difference between them visually. Both of these images were taken when the temperature was cooler (less than 70 degrees). Figure 4-4(c) shows an image of a vehicle taken when the temperature was much warmer (93 degrees). Notice that the change in the grayscale of the pavement, from dark to light. Collectively, Figure 4-4 shows that while thermal images are not affected by lighting conditions, they are affected by temperature.

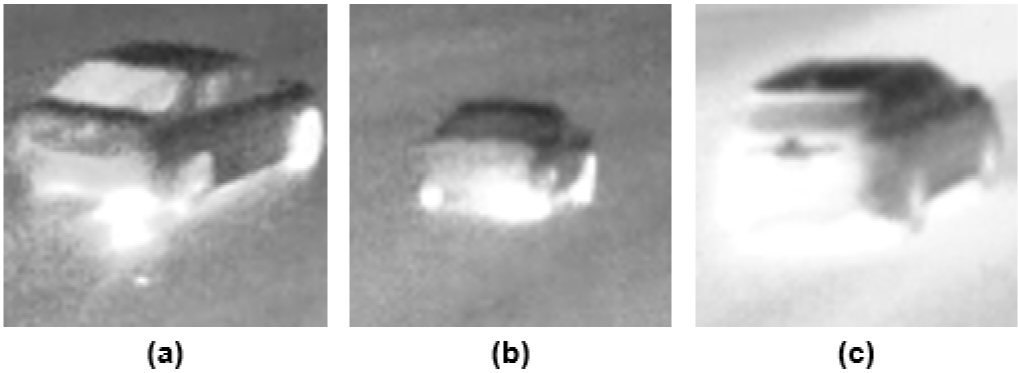


Figure 4-4. Thermal images recorded by FLIR TrafiSense2 Dual camera

4.5 DECAF with thermal CNN model results

The trailer and thermal camera were deployed 11 times by the research team and SCDOT staff. Each deployment recorded at least 24 hours of traffic. The locations of the deployments were selected jointly with the SCDOT to provide a representative sample of roads where the thermal camera would most likely be deployed in the future. As a result, the thermal camera was deployed on a primary road two times, on a secondary road seven times, and on a local road two times. Both of the primary roads have 6 lanes (3 lanes in each direction). One secondary road has four lanes, and the rest have two lanes. Both of the local roads have two lanes. To increase the number of Class 1 (motorcycles) vehicles in the datasets, the research team and SCDOT staff made one deployment in Myrtle Beach, SC during the annual "Bike week" near a motorcycle shop located on Satchelford Road. Also, to increase the number of Class 4-5 vehicles, particularly school buses, the team made two deployments near Airport High School in West Columbia, SC. Table 4-6 provides a summary of the deployments regarding location, time, route type, direction, and weather conditions.

Table 4-6. Summary of thermal camera deployments

Deployment number	Location	Deployment period	Route type	Direction of traffic	Weather condition
1	Charleston Highway	12/15/2020 12/17/2020	6-lane primary	N	Rainy/windy
2	Walter Price Road	02/17/2021 02/19/2021	2-lane local	NE	Sunny/windy
3	Old Dunbar	03/04/2021 03/05/2021	4-lane secondary	S	Sunny/windy

Deployment number	Location	Deployment period	Route type	Direction of traffic	Weather condition
4	Rosewood Drive	03/24/2021 03/26/2021	2-lane secondary	N	Rainy/windy
5	Boston Avenue	04/14/2021 04/16/2021	2-lane secondary	E	Sunny
6	Myrtle Beach	05/06/2021 05/11/2021	6-lane primary	N	Rainy and windy
7	Boston Avenue II	05/24/2021 05/26/2021	2-lane secondary	W	Sunny
8	Motorcycle shop	05/28/2021 06/01/2021	2-lane local	NE	Sunny
9	Pineview Road	08/27/2021 08/30/2021	2-lane secondary	N	Sunny
10	Old Dunbar Road	09/15/2021 09/18/2021	2-lane secondary	S	Light rain/ fog
11	Boston Avenue	09/30/2021 10/02/2021	2-lane secondary	E	Sunny

A total of 162,824 thermal images was generated, reviewed, and manually labeled according to their actual vehicle groups. Of these, 109,867 were selected for the training dataset and the remainder (52,957) were used for the test dataset. As was done with the RGB datasets, to avoid bias in the training and testing of the thermal CNN model, the images from one video or site were not included in both datasets. That is, images from one set of videos were used for training and images from another set of videos were used for testing. This approach ensured that the thermal CNN model does not see images in testing similar to those it was trained with. Table 4-7 shows the number of images per class in the thermal training and test datasets.

Table 4-7. Thermal datasets

Class	Training	Test
Background	36,201	26,192
Class 1	961	53
Class 2-3	66,146	25,964
Class 4-5	1,156	713
Class 7-13	12,403	6,077
Total	**109,867**	**52,957**

The thermal CNN model trained with the data shown in Table 4-7 yielded a training classification accuracy of 96.3%. To test the performance of this model, it was utilized within DECAF. Three different sites were used for testing: Old Dunbar Road, Rosewood Drive, and Pineview Road. Table 4-2 shows the testing results for Old Dunbar Road during the three highest volume hours. The reported DECAF results are with the tracking component. The presented information is similar to those presented previously with the exception of DECAF without tracking results being replaced with MetroCount results. The intent of comparing the DECAF results against MetroCount results is to gain insight into situations where the use of video is better than the use of pneumatic tubes, and vice-versa.

Table 4-8. Comparison of DECAF and MetroCount on Old Dunbar Road

Classes	MetroCount		DECAF		Actual Count
	Count	Over/Under	Count	Over/Under	
Class 1	3	+2	0	-1	1
Class 2-3	512	+4	513	+5	508
Class 4-5	37	+16	1	-20	21
Class 6-13	29	-53	78	-4	82
Total	581	-31	592	-20	612
Total misclassified		75		30	

The Old Dunbar Road test results shown in Table 4-8 indicate that MetroCount yielded a count of 581 vehicles compared to an actual count of 612. Thus, MetroCount undercounted by 31 vehicles, an error of 5%. In terms of classification, MetroCount misclassified 75 vehicles, an error of 12%. DECAF (with tracking) yielded a count of 592 vehicles. Thus, it undercounted by 20 vehicles, an error of 3%. The misclassification by DECAF is 30 vehicles, an error of 5%. It can be concluded from these results that DECAF outperformed MetroCount and that it produced estimates within 5% of the actual values.

Table 4-9. Comparison of DECAF and MetroCount on Rosewood Drive

Classes	MetroCount		DECAF		Actual Count
	Count	Over/Under	Count	Over/Under	
Class 1	2	+1	0	-1	1
Class 2-3	160	+16	139	-5	144
Class 4-5	14	+11	0	-3	3
Class 6-13	176	-22	194	-4	198
Total	352	+6	333	-13	346
Total misclassified		50		13	

The Rosewood Drive test results shown in Table 4-9 indicate that MetroCount yielded a count of 352 vehicles compared to an actual count of 346. Thus, it overcounted by 6 vehicles, an error of 2%. In terms of classification, MetroCount misclassified 50 vehicles, an error of 14%. DECAF (with tracking) yielded a count of 333 vehicles. Thus, DECAF undercounted by 13 vehicles, an error of 4%. The misclassification by DECAF is 13 vehicles, an error of 4%. It can be concluded from these results that MetroCount outperformed DECAF in counting but underperformed DECAF in classification. DECAF produced estimates within 5% of the actual values.

Table 4-10. Comparison of DECAF and MetroCount on Pineview Road

Classes	MetroCount		DECAF		Actual Count
	Count	Over/Under	Count	Over/Under	
Class 1	2	+1	1	0	1
Class 2-3	384	-64	427	-21	448
Class 4-5	76	+74	0	-2	2
Class 6-13	18	-7	25	0	25
Total	480	+4	453	-23	476
Total misclassified		146	0	23	

The Pineview Road test results shown in Table 4-10 indicate that MetroCount yielded a count of 480 vehicles compared to an actual count of 476. Thus, it overcounted by 4 vehicles, an error of 1%. In terms of classification, MetroCount misclassified 146 vehicles, an error of 30%. DECAF (with tracking) yielded a count of 453 vehicles. Thus, DECAF undercounted by 23 vehicles, an error of 5%. The misclassification by DECAF is 23 vehicles, an error of 5%. It can be concluded from these results that DECAF outperformed MetroCount in counting but underperformed DECAF in classification. DECAF produced estimates within 5% of the actual values.

4.6 Effect of offsets on thermal CNN model results

The purchased trailer and custom-made platform allowed the camera to be mounted at a maximum height of 18 feet. Mounting it lower to the ground is not desirable since it will primarily provide a side view instead of a top and side view. Mounting it higher is not only infeasible but also not desirable because of the amount of camera motion likely to be generated by wind. To determine the optimal offset distance (i.e., distance from the edge of the road to the left tire of the trailer, with the hitch arm positioned parallel to the roadway) for deploying the trailer and thermal camera, three different distances were evaluated (6, 12, and 18 feet) at 2 different locations (Old Dunbar Road and Boston Avenue). At each offset distance, one hour and thirty minutes of traffic data were used for the evaluation of counting and classification accuracy. To avoid sampling bias, each offset distance was evaluated at two different times of the day. Lastly, each offset was evaluated at the same times for three consecutive days. Table 4-11 provides a summary of the offset evaluation setup, along with weather and wind conditions.

Table 4-11. Offset evaluation setup

Date	Time	Location	Offset (ft)	Weather	Wind (MPH)
9/15/2021 – 9/17/2021	7:30-9:00	Old Dunbar Road	6	Light rain & fog 70-72° F	2
9/15/2021 – 9/17/2021	16:00-17:30	Old Dunbar Road	6		4
9/15/2021 – 9/17/2021	7:30-9:00	Old Dunbar Road	12	Light rain & fog 73-80° F	3
9/15/2021 – 9/17/2021	16:00-17:30	Old Dunbar Road	12		3
9/15/2021 – 9/17/2021	7:30-9:00	Old Dunbar Road	18	Cloudy 73-81° F	1
9/15/2021 – 9/17/2021	16:00-17:30	Old Dunbar Road	18		1
9/30/2021 - 10/2/2021	7:30-9:00	Boston Avenue	6	Sunny 72-88° F	0
9/30/2021 - 10/2/2021	16:00-17:30	Boston Avenue	6		4
9/30/2021 - 10/2/2021	7:30-9:00	Boston Avenue	12	Fog & partly sunny 75-82° F	2
9/30/2021 - 10/2/2021	16:00-17:30	Boston Avenue	12		3
9/30/2021 - 10/2/2021	7:30-9:00	Boston Avenue	18	Sunny 72-83° F	4

Date	Time	Location	Offset (ft)	Weather	Wind (MPH)
9/30/2021 - 10/2/2021	16:00-17:30	Boston Avenue	18		2

Table 4-12 shows the results of the offset evaluation. At 6 feet offset, DECAF has a counting error of 1% and classification error of 6% at Old Dunbar Road, compared to 2% and 11% respectively at Boston Avenue. At 12 feet offset, DECAF has a counting error of 12% and classification error of 13% at Old Dunbar Road, compared to 4% and 28% respectively at Boston Avenue. At 18 feet offset, DECAF has a counting error of 2% and classification error of 12% at Old Dunbar Road, compared to 4% and 20% respectively at Boston Avenue. These findings suggest that six feet offset is best. It should be noted the CNN model was trained with images that have an offset distance closer to six feet than 12 and 18. That is, at the 11 deployed sites, the most practical offset distance was around six feet. The reason for the high classification error at Boston Avenue when the offset distance was at 6 feet is due to the misclassification of class 4-5 (i.e., school buses). The high misclassification is due to the thermal CNN model being trained with a very low number of class 4-5 images; only deployment 5 at the Airport High School provided images of school buses. Therefore, a higher misclassification rate can be expected at locations where there is a high percentage of school buses.

Table 4-12. Offset evaluation results

Location	Offset (ft)	Counting Error	Classification Error
Old Dunbar Road	6	1%	6%
Old Dunbar Road	12	12%	13%
Old Dunbar Road	18	2%	12%
Boston Avenue	6	2%	11%
Boston Avenue	12	4%	28%
Boston Avenue	18	4%	20%

4.7 Effect of windy conditions on thermal CNN model results

As discussed in Appendix A, the procedure for deploying the trailer is to secure the platform to which the thermal camera is affixed with two cables to the trailer to minimize camera motion. Even with these measures implemented, due to the height and weight of the camera, a noticeable camera motion can be seen visually on windy days, and the results from DECAF clearly show their undesired effects. Table 4-13 shows the results in breezy conditions (≥ 15 MPH). The data are from the Rosewood Drive deployment on March 25, 2021. Between 1 to 3 p.m. when the conditions were breezy, DECAF overcounted by one vehicle, an error of 1%, and it misclassified nine vehicles, an error of 5%. Between 3:30 and 5:30 p.m. when there were gusts in excess of 30 MPH, DECAF overcounted by 34 vehicles, an error of 27%, and it misclassified 40 vehicles, an error of 32%. Therefore, the use of the trailer and thermal camera is not recommended on windy days.

Table 4-13. Effect of wind results

Date and Time	Wind Conditions	Counting Error	Classification Error
3/25/2021, 1:00 – 3:00 p.m.	Moderate	+1 (1%)	9 (5%)
3/25/2021, 3:30 – 5:30 p.m.	Strong	+34 (27%)	40 (32%)

CHAPTER 5: CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

This project investigated the use of traffic cameras, specifically the Miovision Scout and FLIR TrafiSense2 Dual, to count and classify vehicles via post-processing algorithms. Given a series of traffic videos in MP4 format that span 24 or 48 hours, the goal was to provide a count of the number of vehicles in each of the following four categories: motorcycles, passenger cars and light trucks, buses/campers/tow trucks, and small to large trucks. To achieve this goal, this project implemented standard video processing steps for counting vehicles, which include motion object segmentation, blob detection, blob analysis, and tracking. These steps were augmented with a classification step (after blob detection) enabled by the development of a CNN model unique to this project. To our knowledge, the manually labeled images of vehicles with a top-side view used to train the CNN models, is the first of its kind. Additionally, the developed tracking algorithm is unique in that it uses the trajectories of objects to determine those that are most likely created by a vehicle. Moreover, to facilitate the use of the developed algorithms, a Windows application named DECAF, with a graphical user interface, was developed to allow users to easily select video files, draw the region of interest where vehicles should be counted and classified, and view the results.

While the performance of DECAF using the Miovision Scout-collected daytime videos produced satisfactory performance in terms of counting and classification, the nighttime videos did not. This was primarily due to poor lighting conditions which led to poor quality images. Many of the nighttime images had to be removed from the dataset during the review and sorting process because the researchers could not determine to which vehicle group they belonged. Four different low-light image enhancement methods were explored, but none of the combinations tested produced a significantly better image. The small nighttime training dataset led to poor CNN model performance, and changes made to the convolutional network architecture for the nighttime model did not show improvement. For these reasons, in consultation with the SCDOT steering committee, it was determined that thermal imaging should be used to overcome the nighttime lighting issue. The FLIR TrafiSense2 Dual camera was selected, purchased, and assembled. Since this camera does not come with a built-in power source, a solar-powered trailer was also purchased.

The convolutional network architecture developed previously using visual images was also optimal for the thermal images. The thermal CNN model, when used within DECAF, yielded a counting and classification accuracy of at least 95%, which was the aim of this project. Compared to MetroCount, DECAF produced higher classification accuracy and comparable count accuracy, when deployed in recommended conditions.

5.2 Recommendations

Based on this project’s findings, it is recommended that the SCDOT consider using the purchased solar-powered trailer and thermal camera to collect traffic data on high-volume roads and using the developed Windows application, DECAF, to obtain vehicle counts by categories. Deploying the trailer and thermal camera on the side of the road will be safer for the SCDOT personnel than deploying MetroCount's pneumatic tubes across multiple lanes. The level of effort required for deploying the trailer and thermal camera is similar to that of deploying the Miovision Scout which the SCDOT has done in the past. The advantage of using the in-house equipment and software is that it will save the SCDOT the video processing cost, approximately $500.00 per 48-hour count. There are two situations where the use of the thermal camera and DECAF are not recommended over MetroCount's counters: 1) breezy conditions (over 15 MPH) with gusts over 30 MPH for portions of the 48-hour period, and 2) extreme heat with temperatures above 90 $^{\circ}$F for portions of the 48-hour period. When selecting a suitable day for deploying the thermal camera, the SCDOT staff was always mindful of the weather and avoided rainy days. It is recommended that this practice be maintained in the future. Cloudy days do not pose any power problem for the thermal camera.

5.3 Implementation plan

The hardware purchased and assembled as part of this project, solar-powered trailer and thermal camera has already been delivered to the SCDOT. This equipment is stored at the SCDOT storage facility on Shop Road and is secured with two locks for which the SCDOT has keys. Transporting the trailer to the site requires a truck with a hitch. An SCDOT truck has been used for this purpose. Guidelines for deploying the trailer are provided in Appendix A. Once the trailer is positioned with the left tire between six to eight feet from the edge of the white pavement marker and the thermal camera is raised via the mast to 18 feet, the platform for which the camera is affixed needs to be secured with two cables to the trailer to minimize camera motion. The offset distance can be measured by pacing, or more accurately, by using a tape measure. In positioning the trailer, due to its weight, two staff members are recommended to avoid injury. The process of setting up the trailer and camera can be performed without the need for traffic control. Once the trailer and camera are set up, then the recording software needs to be configured; step-by-step instructions for setting up the recording software are provided in Appendix B.

A Windows-based stand-alone application was provided to the SCDOT along with the final project report. This application will need to be installed by SCDOT IT staff on the computer which is reserved for video processing. It is recommended that this computer not be used while the video is being processed. On average, a 24-hour video will take about 24 hours to process, and a 48-hour video will take about 48 hours to process. Figure 5-1 shows a screenshot of DECAF, with the primary steps outlined. Step-by-step instructions for using DECAF are provided in Appendix C.

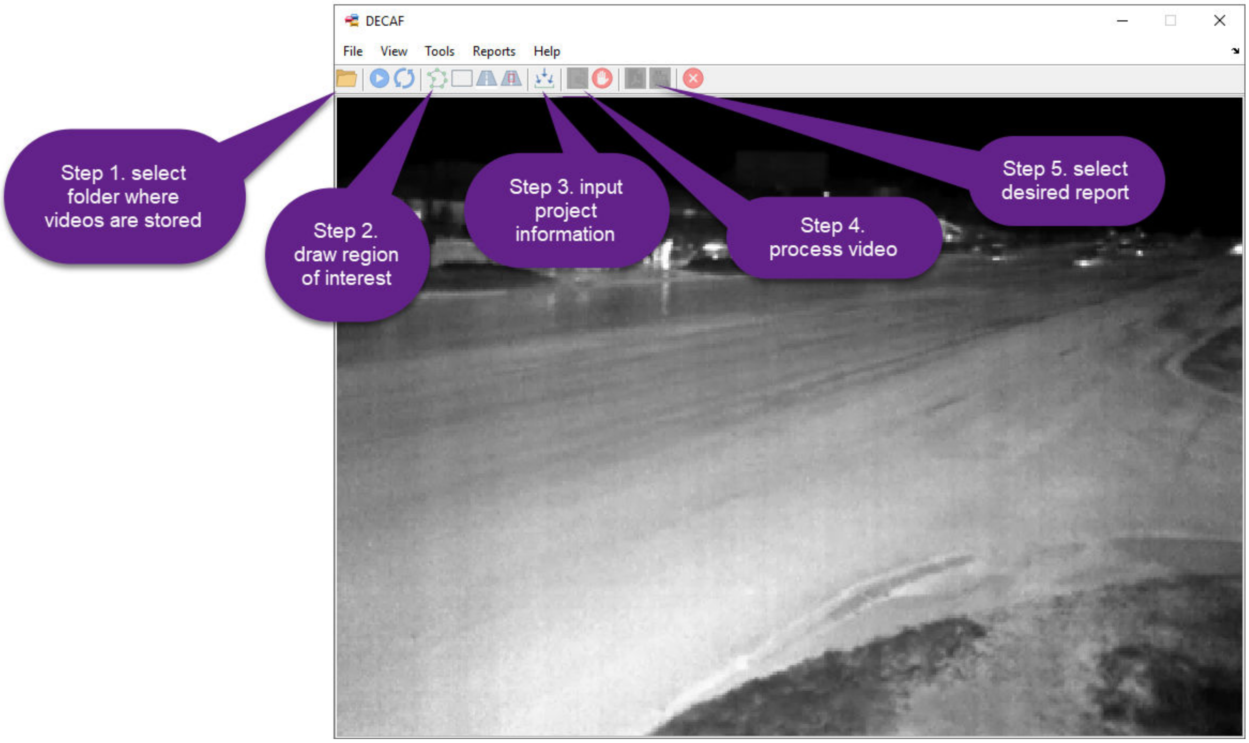

DECAF
File View Tools Reports Help
Step 1. select folder where videos are stored
Step 2. draw region of interest
Step 3. input project information
Step 4. process video
Step 5. select desired report

Figure 5-1. Screenshot of DECAF with primary steps outlined

As illustrated in Figure 5-1, the use of DECAF to post-process one or more videos requires the user to follow the following five steps. These steps are demonstrated as part of the training provided to SCDOT staff at the completion of the project.

- Step 1. Click on the folder icon on the toolbar to select the folder where the video files are stored. It is recommended that the video files are stored on the computer's hard drive and not an external hard drive or flash drive.
- Step 2. Click on the polygon icon on the toolbar to draw the region of interest (ROI). The ROI is the area where the user wants vehicles to be detected and classified. The ROI should be positioned further downstream, with respect to the camera location.
- Step 3. Click on the input icon on the toolbar to input project information and desired aggregation interval for summary statistics.
- Step 4. Click on the magnifying glass icon on the toolbar to process the video. The videos will be processed at or slightly faster than in real-time. The computer should not be used to perform other tasks while DECAF is running.
- Step 5. Once the processing is finished, click on either the PDF icon or CVS icon to open the report in the desired format.

REFERENCES

Adu-Gyamfi, Y.O., Asare, S.K., Sharma, A. and Titus, T. (2017). Automated vehicle recognition with deep convolutional neural networks. *Transportation Research Record*, 2645(1), 113-122.

Adventures in Machine Learning, Retrieved September 2021, from http://adventuresinmachinelearning.com/.

Avery, R., Wang, Y., & Scott Rutherford, G. (2004). Length-based vehicle classification using images from uncalibrated video cameras. *Proceedings. The 7th International IEEE Conference on Intelligent Transportation Systems (IEEE Cat. No.04TH8749)*, 737-742.

Bhaskar, P.K., and Yong, S. (2014). Image processing-based vehicle detection and tracking method. *2014 International Conference on Computer and Information Sciences*, 1-5.

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

Chang, J., Wang, L., Meng, G., Xiang and C. Pan. (2018). Vision-Based Occlusion Handling and Vehicle Classification for Traffic Surveillance Systems, in *IEEE Intelligent Transportation Systems Magazine*, vol. 10, no. 2, 80-92.

Chen W., Sun Q., Wang J., Dong J., and Xu J. A Novel Model Based on AdaBoost and Deep CNN for Vehicle Classification, *IEEE Access*, vol. 6, 60445-60455.

Chen, B.H. and Huang, S.C. (2015). Probabilistic neural networks based moving vehicles extraction algorithm for intelligent traffic surveillance systems. *Information Sciences*, 299, 283-295.

Cho, S.W., Baek, N.R.; Kim, M.C.; Koo, J.H.; Kim, J.H.; Park, K.R. Face Detection in Nighttime Images Using Visible-Light Camera Sensors with Two-Step Faster Region-Based Convolutional Neural Network. *Sensors, 18*, 2995.

Dong, Z., Wu, Y., Pei, M. and Jia, Y. (2015). Vehicle type classification using a semisupervised convolutional neural network. *IEEE transactions on intelligent transportation systems*, 16(4), 2247-2256.

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

Girshick, R. (2015). Fast R-CNN. *Proceedings of the IEEE international conference on computer vision* 1440-1448.

Girshick, R., Donahue, J., Darrell, T. and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 580-587.

Guo, Q., Liang, Z. and Hu, J. (2017). Vehicle Classification with Convolutional Neural Network on Motion Blurred Images. *DEStech Transactions on Computer Science and Engineering*, doi:10.12783/dtcse/aiea2017/14912.

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), 37-47.

Han, Y., Jiang, T., Ma, Y. and Xu, C. (2018). Pretraining Convolutional Neural Networks for Image-Based Vehicle Classification. *Advances in Multimedia*, 1-10.

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

Hasnat, A., Shvai, N., Meicler, A., Maarek, P. and Nakib, A. (2018). New Vehicle Classification Method Based on Hybrid Classifiers. *The 25th IEEE International Conference on Image Processing (ICIP)*, 2018, 3084-3088.

He, D., Lang, C., Feng, S., Du, X. and Zhang, C. (2015). August. Vehicle detection and classification based on convolutional neural network. *Proceedings of the 7th International Conference on Internet Multimedia Computing and Service*, 3.

He, K., Zhang, X., Ren, S. and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770-778.

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*, 641-648.

Hinton, G.E. and Salakhutdinov, R.R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786), 504-507.

Hu, X., Xu, X., Xiao, Y., Chen, H., He, S., Qin, J. and Heng, P.A. (2018). SINet: A scale-insensitive convolutional neural network for fast vehicle detection. *IEEE Transactions on Intelligent Transportation Systems*, 20(3), 1010-1019.

Huang, S.C. and Do, B.H. (2013). Radial basis function based neural network for motion detection in dynamic scenes. *IEEE transactions on cybernetics*, 44(1), 114-125.

Ji, J., Xu, Z., Yu, H., Fu, L., & Zhou, X. (2020). Domain Adaptation for Vehicle Detection In Traffic Surveillance Images From Daytime To Nighttime. *Transportation Research Board the 99th Annual Meeting.*

Jo, N. Ahn, Y. Lee, Y., and Kang S. Transfer Learning-based Vehicle Classification. (*2018*) *International SoC Design Conference (ISOCC)*, 127-128.

Johnson, A. (2015). *Automated Solar Cavity Detection*. Retrieved September 2021, from https://slideplayer.com/slide/8715285/

Jones, M.J. and Viola, P. (2003). Face recognition using boosted local features. *Technical Report MERL-TR-2003-25, Mitsubishi Electric Research Laboratory*.

Kang, Byeong. (2007). A Review on Image & Video Processing. *International Journal of Multimedia and Ubiquitous Engineering.* 2.

Karpathy, A. (2016). *Convolutional neural networks for visual recognition.* Retrieved September 2021, from http://cs231n.github.io/classification/

Kavukcuoglu, K., Sermanet, P., Boureau, Y.L., Gregor, K., Mathieu, M. and Cun, Y.L. (2010). Learning convolutional feature hierarchies for visual recognition. *Advances in neural information processing systems*, 1090-1098.

Khan M., Nawaz M., Nida-Ur-Rehman Q., Masood G., Adnan A., Anwar S., Cosmas J. (2019). Multiple Moving Vehicle Speed Estimation Using Blob Analysis. In: Rocha Á., Adeli H., Reis L., Costanzo S. (eds) New Knowledge in Information Systems and Technologies. WorldCIST'19 2019. *Advances in Intelligent Systems and Computing, vol 931.*

Kim, P.K. and Lim, K.T. (2017). Vehicle type classification using bagging and convolutional neural network on multi view surveillance image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 41-46.

Kyrkou, C. (2017). *Object Detection Using Local Binary Patterns* Retrieved September 2021, from https://medium.com/@ckyrkou/object-detection-using-local-binary-patterns-50b165658368.

LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. *Nature*, 521(7553), 436-444.

LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W. and Jackel, L.D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4), 541-551.

LeCun, Y., Boser, B.E., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W.E. and Jackel, L.D. (1990). Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems* 396-404.

Liu, W., Wen, Y., Yu, Z. and Yang, M. (2016). Large-margin softmax loss for convolutional neural networks. *ICML*, Vol. 2, No. 3, 7.

Matlab GUI, MATLAB & Simulink, Retrieved September 2021, from https://www.mathworks.com/discovery/matlab-gui.html.

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), 541-550.

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), 1215-1225.

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), 359-364.

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 Systems, Man, and Cybernetic*, 4006-4011.

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

Patel, S. and Pingel, J. (2017). *What Are Convolutional Neural Networks?*, Retrieved September 2021, from https://www.mathworks.com/videos/introduction-to-deep-learning-what-are-convolutional-neural-networks--1489512765771.html.

R: A language and environment for statistical computing. *R Foundation for Statistical Computing*, Vienna, Austria. URL https://www.R-project.org/.

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

Rawat, W. and Wang, Z. (2017). Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation*, 29(9), 2352-2449.

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

Salvi, G. (2012). An automated vehicle counting system based on blob analysis for traffic surveillance. *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)*, 1.

Yu S., Wu, Y., Li W., Song H., Zeng W. (2017). A model for fine-grained vehicle classification based on deep learning, *Neurocomputing*, Volume 257, 97-103.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

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, 4-21.

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

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

Supervised vs. unsupervised learning: What's the difference? IBM. (n.d.). Retrieved September 7, 2021, from https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning.

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

Tairi, H. (2015). Motion detection based on the combining of the background subtraction and spatial color information. 10.1109/ISACV.2015.7105548.

Tayara, H., Soo, K.G. and Chong, K.T. (2017). Vehicle detection and counting in high-resolution aerial images using convolutional regression neural network. *IEEE Access*, 6, 2220-2230.

Traffic Monitoring Guide. U.S. Department of Transportation/Federal Highway Administration. (n.d.). Retrieved September 7, 2021, from https://www.fhwa.dot.gov/policyinformation/tmguide/tmg_2013/vehicle-types.cfm.

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).

Yang, L., Luo, P., Change Loy, C. and Tang, X. (2015). A large-scale car dataset for fine-grained categorization and verification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 3973-3981.

Zhang, F., Li, C. and Yang, F. (2019). Vehicle detection in urban traffic surveillance images based on convolutional neural networks with feature concatenation. *Sensors*, 19(3), 594.

Zhou, Y., Nejati, H., Do, T.T., Cheung, N.M. and Cheah, L. (2016). Image-based vehicle analysis using deep neural network: A systematic study. *2016 IEEE International Conference on Digital Signal Processing*, 276-280.

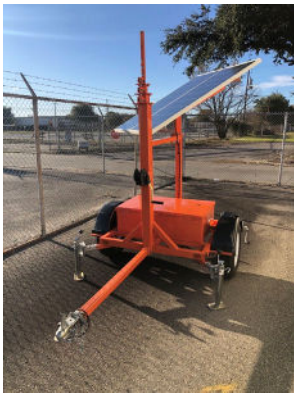
Zivkovic, Z. (2004). Improved adaptive Gaussian mixture model for background subtraction. In Pattern Recognition, 2004. ICPR 2004. *Proceedings of the 17th International Conference on Pattern Recognition,* Vol. 2, 28-31.

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), 773-780.

APPENDIX A

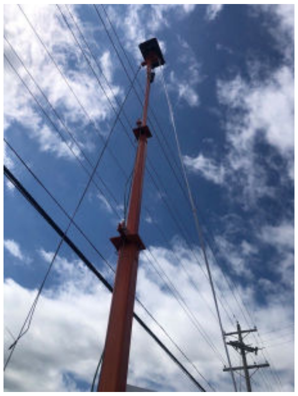
Instructions for storing the solar-power trailer

When not in use, the solar-powered trailer should be stored at the SCDOT storage facility on Shop Road, between the fence and the shed, and the platform with the thermal camera attached should be stored in the office at SCDOT headquarters. During storage or at a deployment site, the solar panel should be oriented to face south (shown below); doing so will ensure the panel will receive the highest amount of sunlight to charge the batteries. While in storage, the trailer should be secured with two locks. The first lock should be placed on the trailer hitch to prevent the trailer from being mounted onto a vehicle. The second lock should be placed on the NEMA box to prevent it from being opened; the NEMA box provides housing for the batteries and all of the FLIR TrafiSense2 Dual camera components.

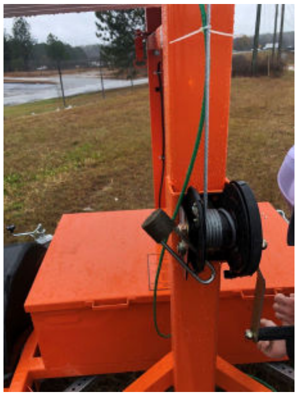


Instructions for transporting and deploying the solar-power trailer and thermal camera

1. Unlock the hitch lock and mount the trailer onto the SCDOT truck's hitch.
2. Orient the solar panel parallel to the ground to minimize drag during transport.
3. Transport the trailer to the site and position the vehicle and trailer to the right side of the road. Two staff members are recommended for the following tasks, which can be done without the need for traffic control.
4. Position the trailer with the hitch arm parallel to the road and the left tire of the trailer 6 feet from the edge of the road.
5. Level the trailer.
6. Orient the solar panel to face south.
7. Mount the platform which has the thermal camera attached onto the mast. Position the camera to point in the direction of traffic to provide a top-side view of vehicles.
8. Attach two cables, one on the right side of the camera platform and one on the left side of the camera platform.
9. Wind the crank clockwise to raise the camera up to 18 feet high; this height is marked and labeled on the mast.
10. Secure the two cables that are attached to the camera platform to the trailer to minimize camera motion as shown below.



11. Set up the recording software following the instructions provided in Appendix B.
12. Before leaving the deployment site: 1) put a lock on the crank to prevent the mast from being lowered (shown below), 2) put a lock on the hitch to prevent the trailer from being stolen, and 3) put a lock on the NEMA box to prevent anyone from accessing the batteries and camera components during deployment.

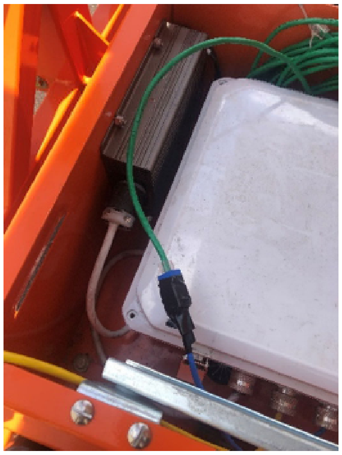


APPENDIX B

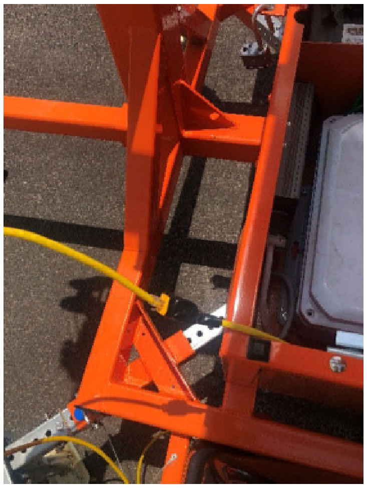
Instructions for setting up the recording

Prerequisite: a laptop with Windows and FLIR camera and recording software, which have been made available on ProjectWise. The following instructions assume the software has been installed on a laptop with two shortcuts labeled "FLIR Camera" and "FLIR ITS Record."

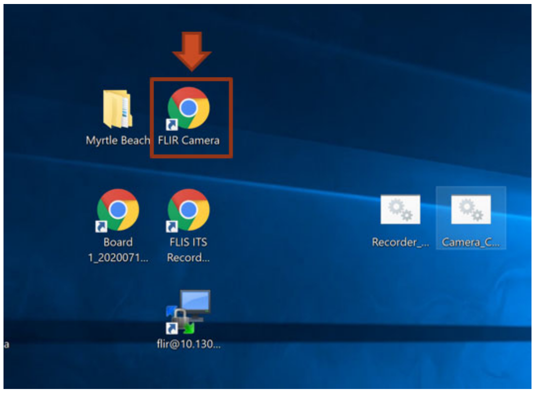
1. Turn on the power strip/surge protector.
2. Plug the green cable (connected on one end to the thermal camera) to the blue connector as shown below. This step connects the camera to the POE injector to enable the camera to receive power.



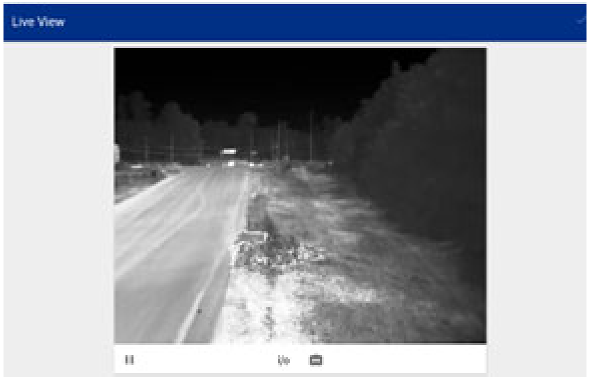
3. Plug one end of a Cat 5 cable to the yellow connector as shown below and the other end of the Cat 5 cable to the Cat 5-to-USB converter.



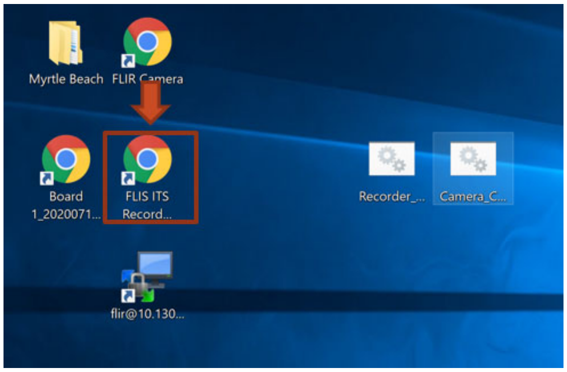
4. Plug the Cat5-to-USB converter into the laptop's USB port. This step and the previous one connects the laptop to the switch to enable the laptop to communicate with the Raspberry Pi which runs the recorder software.
5. Turn on the laptop and login.
6. Double-click on the shortcut labeled "FLIR Camera" to check the thermal camera view.



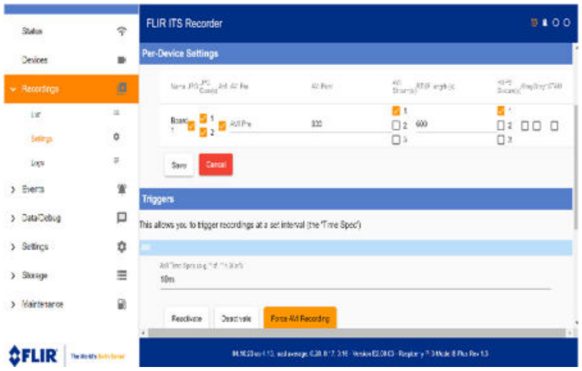
7. Once the camera software is up and running, the user should see the following Live View window. It may take a couple of minutes for the connections from the laptop to the Raspberry Pi, and from the Raspberry Pi to the thermal camera to establish.



8. A window with the message "No Internet" will appear if there is a connection issue. In this case, double check all cable connections or reconnect them. Wait for the connections to establish. Once connected, a Live View window will show what is being seen by the thermal camera.
9. Change the angle of the thermal camera, if needed, to point in the direction of traffic to provide a top-side view of vehicles. If the camera angle needs to be adjusted, then the mast will need to be lowered.
10. Double click on the shortcut labeled "FLIR ITS Record" to run the recorder software as shown below.

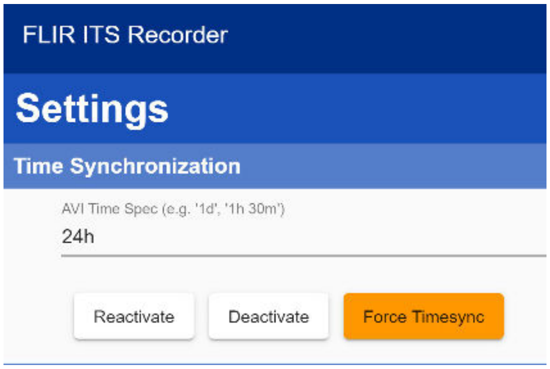


11. Once the recorder software is up and running, click on the “Recordings” link as shown below.



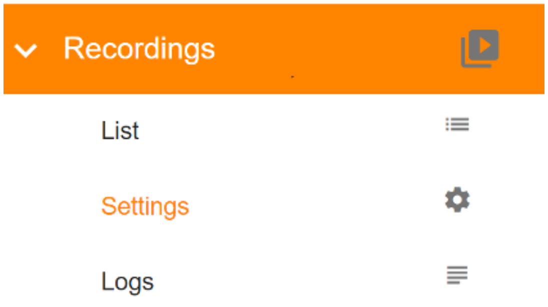
This allows you to trigger recordings at a set interval (the 'Time Spec')

12. Under Time Synchronization, select “Force Timesync” as shown below.

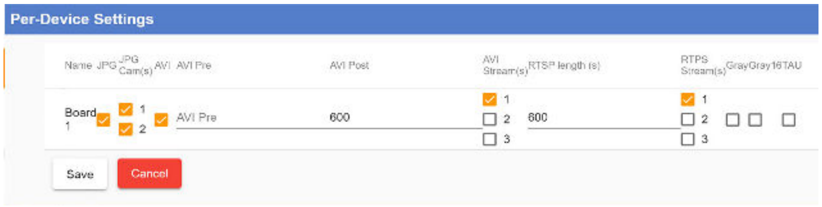


AVI Time Spec (e.g. '1d', '1h 30m')
24h

13. Click on the “Settings” option under Recordings as shown below.



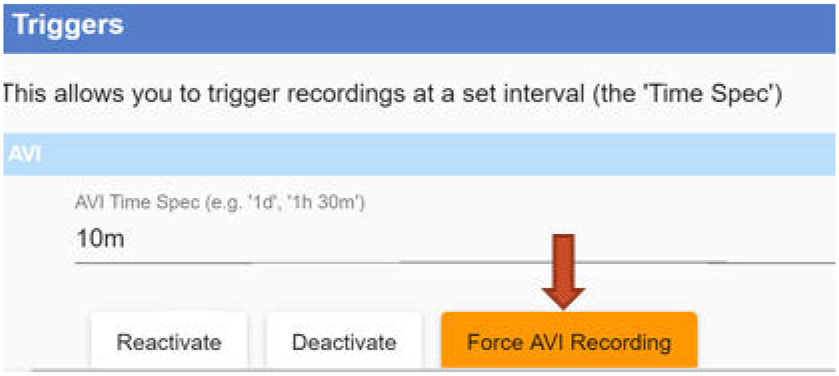
14. Input 600 (seconds) for the video duration as shown below.



Per-Device Settings

Name	JPG	JPG Cam(s)	AVI	AVI Pre	AVI Post	AVI Stream(s)	RTSP length (s)	RTPS Stream(s)	Gray	Gray16	TAU
Board 1	☑	☑ 1 ☑ 2	☑	AVI Pre	600	☑ 1 □ 2 □ 3	600	☑ 1 □ 2 □ 3	□	□	□

15. Input 10 (minutes) for the recording interval as shown below. This step and the previous one will create a recording every 10 minutes, with each video being 10 minutes long. Select “Force AVI Recording” as shown below to start recording.



Triggers

This allows you to trigger recordings at a set interval (the 'Time Spec')

AVI

AVI Time Spec (e.g. '1d', '1h 30m')

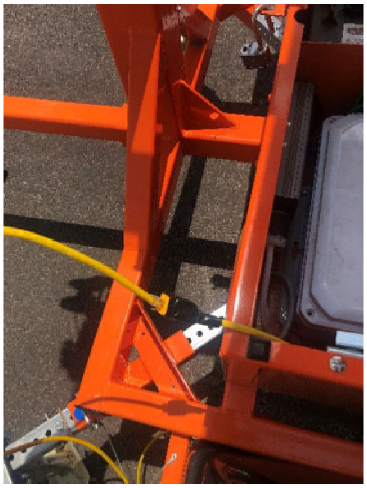
10m

16. Disconnect the Cat 5 cable from the Cat5-to-USB converter.
17. Disconnect the Cat5-to-USB converter from the laptop’s USB port.
18. Shutdown the laptop.

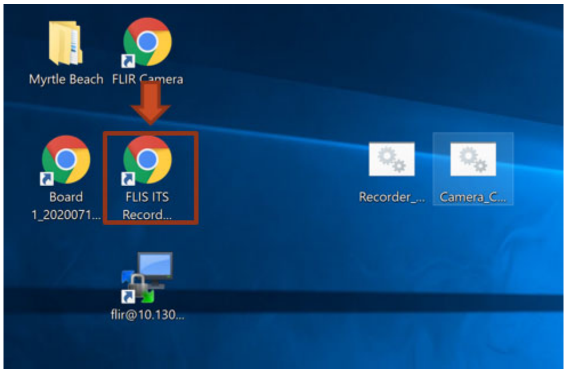
Instructions for stopping the recording

To stop the recording after the desired data collection period (24 or 48 hours), return to the site where the trailer was deployed and perform the following steps.

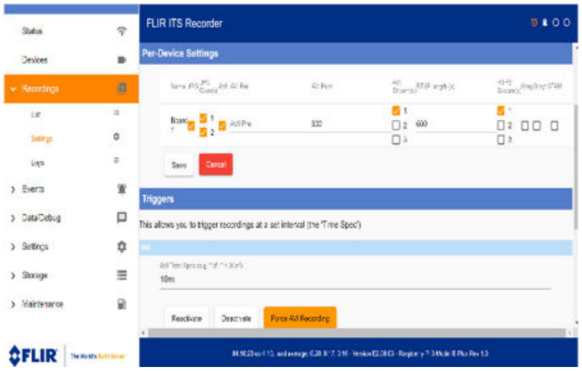
1. Plug one end of a Cat 5 cable to the yellow connector as shown below and the other end of the Cat 5 cable to the Cat 5-to-USB converter.



2. Plug the Cat5-to-USB converter into the laptop's USB port.
3. Turn on the laptop and login.
4. Double click on the shortcut labeled "FLIR ITS Record" to run the recorder software as shown below.



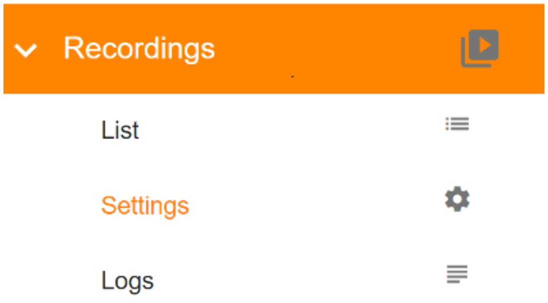
5. Click on the "Recordings" link as shown below



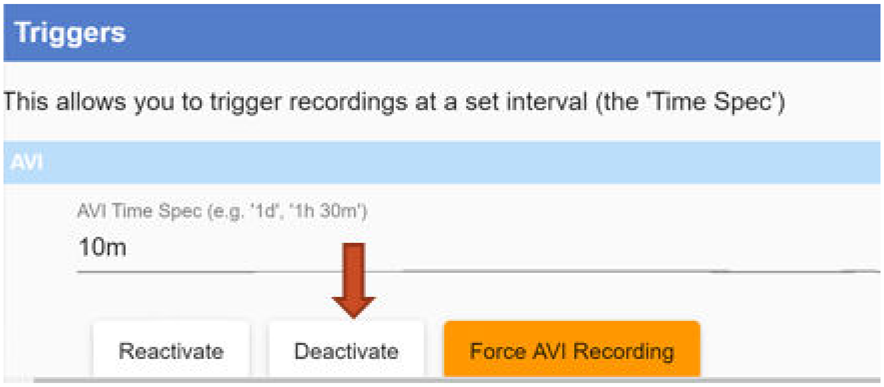
This allows you to trigger recordings at a set interval (the 'Time Spec')

10m

6. Click on the “Settings” option under Recordings as shown below.



7. Under Triggers, select “Deactivate” to stop recording as shown below.



This allows you to trigger recordings at a set interval (the 'Time Spec')

AVI Time Spec (e.g. '1d', '1h 30m')

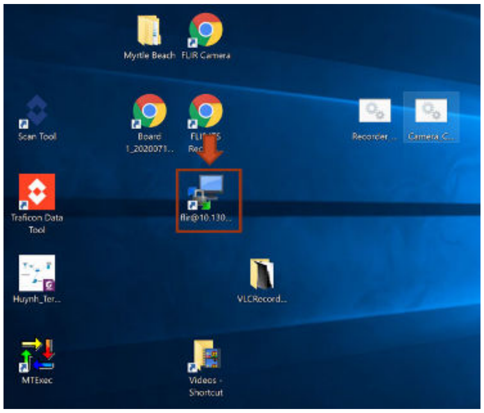
10m

8. If recorded data need to be downloaded, follow the “Instructions for downloading recorded data” provided in the next section. If not, disconnect the Cat 5 cable from the Cat5-to-USB converter and remove the Cat5-to-USB converter from the laptop.
9. Shutdown the laptop.
10. Turn off the power strip/surge protector.
11. Lower the mast.
12. Remove the platform with the thermal camera attached to it. Put the camera in the front seat of the truck and store it in the office at headquarters.
13. Orient the solar panel parallel to the ground to minimize drag during transport.
14. Transport the trailer to the storage facility.

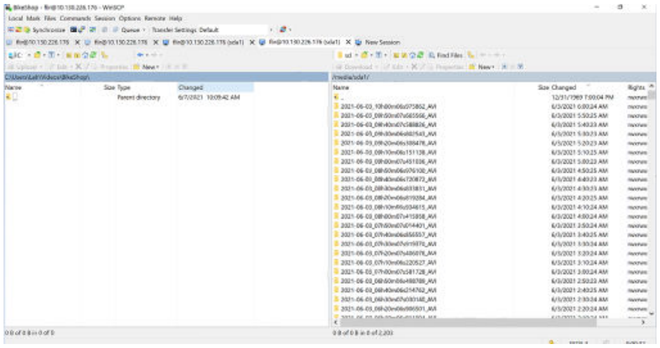
Instructions for downloading recorded data

The following instructions convert recorded data in Linux format to Windows format and transfer the recorded videos from an external hard drive to the laptop.

1. With the laptop turned on and connected to the switch via the Cat 5 cable, double-click on the file transfer shortcut as shown below.



2. Create a folder on the laptop where the videos should be stored. Use location and date of the deployment for the folder name.
3. Select the folders containing the videos to be copied (right pane of the window shown below). Multiple folders can be selected using either Ctrl or Shift key. Each folder contains a 10-minute video and is named by the date and starting time of the recording.
4. Drag the selected folders to the left pane of the window shown below. The file transfer process will take approximately one hour to complete for 24 hours of recording.



BikeShop - flir@10.130.226.176 - WinSCP
Local Mark Files Commands Session Options Remote Help
Synchronize
Queue
Transfer Settings Default
flir@10.130.226.176
flir@10.130.226.176
flir@10.130.226.176 (sda1)
flir@10.130.226.176 (sda1)
New Session
C:
Upload
Edit
Properties
New
sd
Find Files
Download
Edit
Properties
New

C:\Users\Lab\Videos\BikeShop\

Name	Size	Type	Changed
..		Parent directory	6/7/2021 10:09:42 AM

/media/sda1/

Name	Size	Changed	Rights
..		12/31/1969 7:00:04 PM	rwxrwxrwx
2021-06-03_10h00m06s075862_AVI		6/3/2021 6:00:24 AM	rwxrwxrwx
2021-06-03_09h50m07s565566_AVI		6/3/2021 5:50:25 AM	rwxrwxrwx
2021-06-03_09h40m07s588826_AVI		6/3/2021 5:40:23 AM	rwxrwxrwx
2021-06-03_09h30m06s602543_AVI		6/3/2021 5:30:23 AM	rwxrwxrwx
2021-06-03_09h20m06s308478_AVI		6/3/2021 5:20:23 AM	rwxrwxrwx
2021-06-03_09h10m06s151138_AVI		6/3/2021 5:10:25 AM	rwxrwxrwx
2021-06-03_09h00m07s451036_AVI		6/3/2021 5:00:23 AM	rwxrwxrwx
2021-06-03_08h50m06s076100_AVI		6/3/2021 4:50:25 AM	rwxrwxrwx
2021-06-03_08h40m06s730873_AVI		6/3/2021 4:40:23 AM	rwxrwxrwx
2021-06-03_08h30m06s033831_AVI		6/3/2021 4:30:23 AM	rwxrwxrwx
2021-06-03_08h20m06s819284_AVI		6/3/2021 4:20:25 AM	rwxrwxrwx
2021-06-03_08h10m05s034615_AVI		6/3/2021 4:10:24 AM	rwxrwxrwx
2021-06-03_08h00m07s415858_AVI		6/3/2021 4:00:24 AM	rwxrwxrwx
2021-06-03_07h50m07s014401_AVI		6/3/2021 3:50:24 AM	rwxrwxrwx
2021-06-03_07h40m06s856557_AVI		6/3/2021 3:40:25 AM	rwxrwxrwx
2021-06-03_07h30m07s919970_AVI		6/3/2021 3:30:24 AM	rwxrwxrwx
2021-06-03_07h20m07s406076_AVI		6/3/2021 3:20:24 AM	rwxrwxrwx
2021-06-03_07h10m06s220527_AVI		6/3/2021 3:10:24 AM	rwxrwxrwx
2021-06-03_07h00m07s581728_AVI		6/3/2021 3:00:24 AM	rwxrwxrwx
2021-06-03_06h50m06s488788_AVI		6/3/2021 2:50:22 AM	rwxrwxrwx
2021-06-03_06h40m06s214762_AVI		6/3/2021 2:40:25 AM	rwxrwxrwx
2021-06-03_06h30m07s030140_AVI		6/3/2021 2:30:24 AM	rwxrwxrwx
2021-06-03_06h20m06s906501_AVI		6/3/2021 2:20:24 AM	rwxrwxrwx

0 B of 0 B in 0 of 0

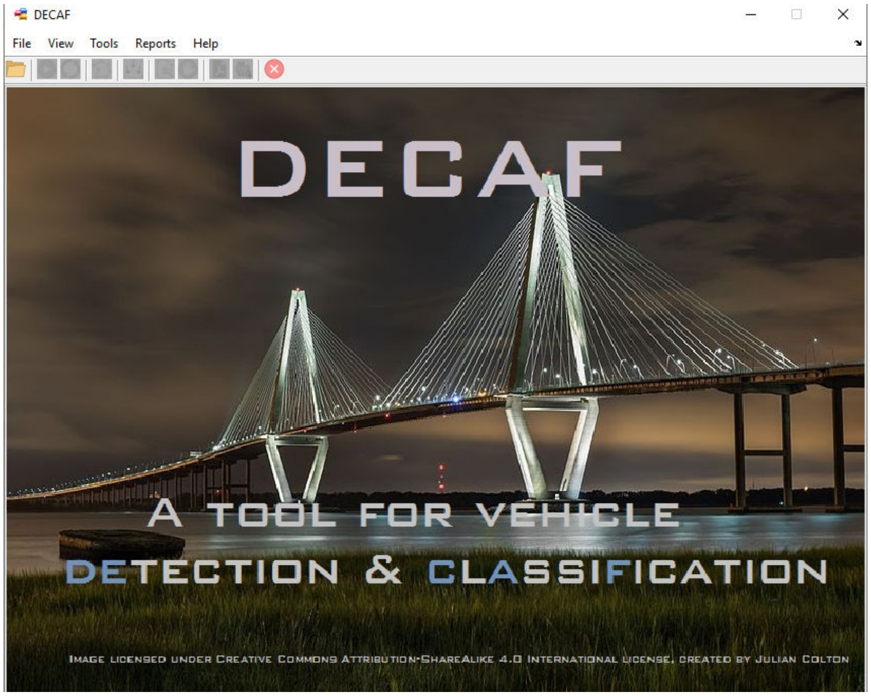
0 B of 0 B in 0 of 2,203

APPENDIX C

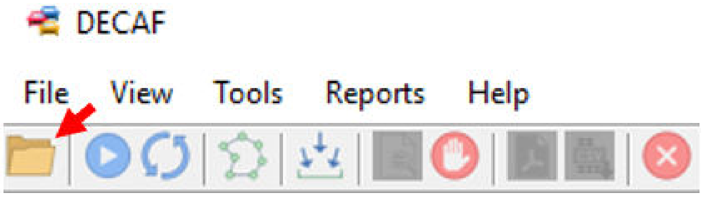
The following instructions assume that DECAF has been installed on a computer reserved for processing videos.

Instruction for using DECAF

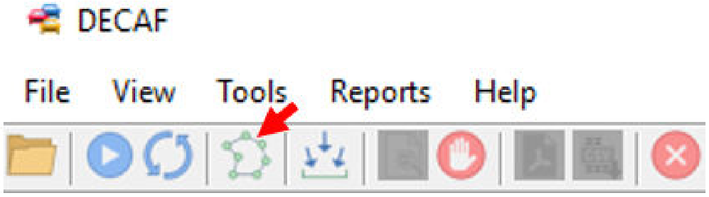
1. Double click on the DECAF icon/shortcut to run it. The following window will appear.


DECAF
File View Tools Reports Help
DECAF
A TOOL FOR VEHICLE
DETECTION & CLASSIFICATION
IMAGE LICENSED UNDER CREATIVE COMMONS ATTRIBUTION-SHAREALIKE 4.0 INTERNATIONAL LICENSE. CREATED BY JULIAN COLTON

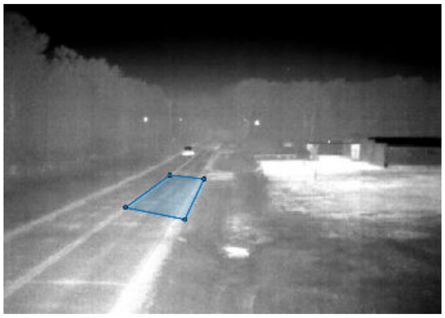
2. On the toolbar, select the folder icon as shown below and select the folder that contains the videos to be processed.



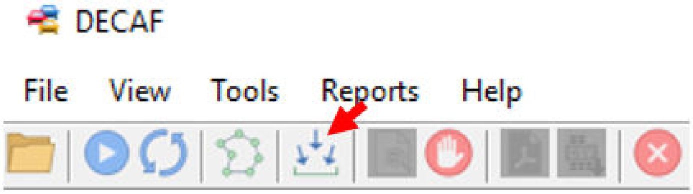
3. On the toolbar, click on the polygon icon as shown below to draw the region of interest (ROI), where vehicles should be detected and classified.



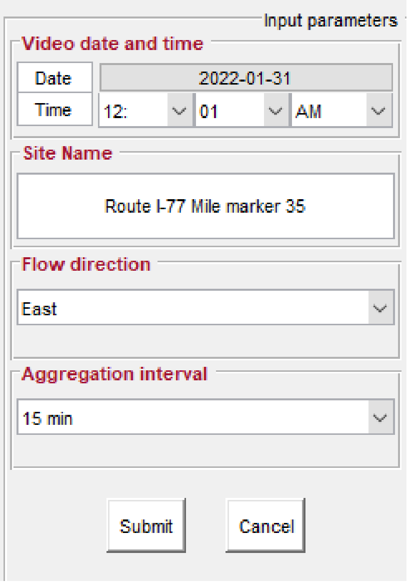
4. Draw the ROI by establishing vertices via the mouse's left button. To close the polygon, double-click on the starting vertex with the left button. The ROI should be drawn further downstream on the road with respect to the camera position as shown below.



5. Right-click inside the drawn ROI and select add the selected “Add region.”
6. On the toolbar, select the input icon as shown below to input project information and desired aggregation interval for the report.



7. Provide the date, time, location, and direction of the collected traffic data. By default, the aggregation interval is set to 15 minutes (i.e., the PDF report will provide counts by categories for every 15 minutes), but it could be changed to 30 minutes or 1 hour. Click on the Submit button upon completion of entry.



Input parameters

Video date and time

Date	2022-01-31
Time	12: 01 AM

Site Name

Route I-77 Mile marker 35

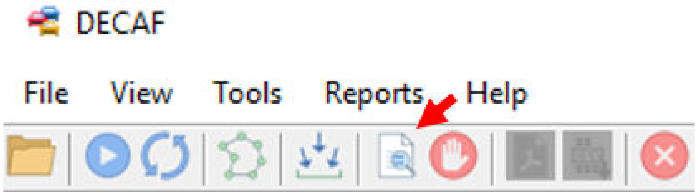
Flow direction

East

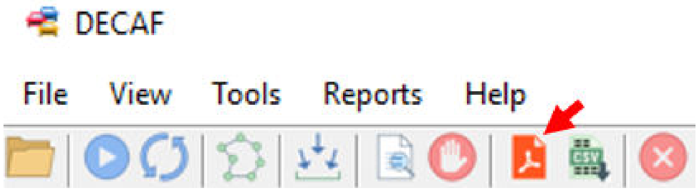
Aggregation interval

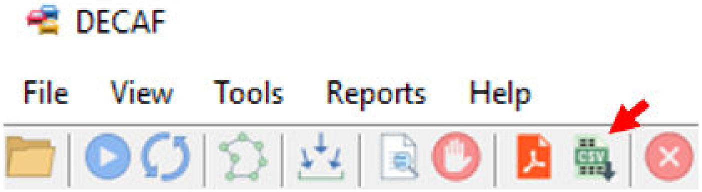
15 min

8. On the toolbar, click on the magnifying glass icon as shown below to process the videos. The videos will be processed at or slightly faster than in real-time, depending on the traffic volume. The computer should not be used to perform other tasks while DECAF is running.



9. Once the videos are processed, the PDF and CVS report options will be available. To open the PDF report, click on the PDF icon and to open the CSV report, click on the CSV icon as shown below.





10. To exit DECAF, click on the X icon on the toolbar, select File and then Exit, or close the DECAF window.