

**DRONE-BASED COMPUTER VISION-ENABLED VEHICLE
DYNAMIC MOBILITY AND SAFETY PERFORMANCE
MONITORING**

FINAL PROJECT REPORT

by

**Guohui Zhang, Ph.D., P.E.; Runze Yuan; Hao Yu; Ph.D., Panos D. Prevedouros,
Ph.D.; and Tianwei Ma, Ph.D.**

**Department of Civil and Environmental Engineering
University of Hawaii at Manoa**

for

**Center for Safety Equity in Transportation (CSET)
USDOT Tier 1 University Transportation Center
University of Alaska Fairbanks
ELIF Suite 240, 1764 Tanana Drive
Fairbanks, AK 99775-5910**

**In cooperation with U.S. Department of Transportation,
Research and Innovative Technology Administration (RITA)**



DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the U.S. Department of Transportation's University Transportation Centers Program, in the interest of information exchange. The Center for Safety Equity in Transportation, the U.S. Government and matching sponsor assume no liability for the contents or use thereof.

1. Report No.				2. Government Accession No.		3. Recipient's Catalog No.	
4. Title and Subtitle Drone-based Computer Vision-Enabled Vehicle Dynamic Mobility and Safety Performance Monitoring				5. Report Date January 7, 2023		6. Performing Organization Code	
				7. Author(s) and Affiliations Guohui Zhang, Ph.D., P.E., Runze Yuan, Hao Yu, Ph.D., Panos D. Prevedouros, Ph.D., and Tianwei Ma, Ph.D. Department of Civil and Environmental Engineering University of Hawaii at Manoa			
9. Performing Organization Name and Address Center for Safety Equity in Transportation ELIF Building Room 240, 1760 Tanana Drive Fairbanks, AK 99775-5910				10. Work Unit No. (TRAIS)		11. Contract or Grant No.	
				12. Sponsoring Organization Name and Address United States Department of Transportation Research and Innovative Technology Administration 1200 New Jersey Avenue, SE Washington, DC 20590			
15. Supplementary Notes Report uploaded to:				14. Sponsoring Agency Code			
16. Abstract This report documents the research activities to develop a drone-based computer vision-enabled vehicle dynamic safety performance monitoring in Rural, Isolated, Tribal, or Indigenous (RITI) communities. The acquisition of traffic system information, especially the vehicle speed and trajectory information, is of great significance to the study of the characteristics and management of the traffic system in RITI communities. The traditional method of relying on video analysis to obtain vehicle number and trajectory information has its application scenarios, but the common video source is often a camera fixed on a roadside device. In the videos obtained in this way, vehicles are likely to occlude each other, which seriously affects the accuracy of vehicle detection and the estimation of speed. Although there are methods to obtain high-view road video by means of aircraft and satellites, the corresponding cost will be high. Therefore, considering that drones can obtain high-definition video at a higher viewing angle, and the cost is relatively low, we decided to use drones to obtain road videos to complete vehicle detection. In order to overcome the shortcomings of traditional object detection methods when facing a large number of targets and complex scenes of RITI communities, our proposed method uses convolutional neural network (CNN) technology. We modified the YOLO v3 network structure and used a vehicle data set captured by drones for transfer learning, and finally trained a network that can detect and classify vehicles in videos captured by drones. A self-calibrated road boundary extraction method based on image sequences was used to extract road boundaries and filter vehicles to improve the detection accuracy of cars on the road. Using the results of neural network detection as input, we use video-based object tracking to complete the extraction of vehicle trajectory information for traffic safety improvements. Finally, the number of vehicles, speed and trajectory information of vehicles were calculated, and the average speed and density of the traffic flow were estimated on this basis. By analyzing the acquiesced data, we can estimate the traffic condition of the monitored area to predict possible crashes on the highways.							
17. Key Words Drone, Convolutional Neural Network, Transfer Learning, Speed Estimation, Crash Prediction, Traffic Flow Parameter Estimation				18. Distribution Statement			
19. Security Classification (of this report) Unclassified.		20. Security Classification (of this page) Unclassified.		21. No. of Pages 56		22. Price N/A	

SI* (Modern Metric) Conversion Factors

APPROXIMATE CONVERSIONS TO SI UNITS				
Symbol	When You Know	Multiply By	To Find	Symbol
LENGTH				
in	inches	25.4	millimeters	mm
ft	feet	0.305	meters	m
yd	yards	0.914	meters	m
mi	miles	1.61	kilometers	km
AREA				
in ²	square inches	645.2	square millimeters	mm ²
ft ²	square feet	0.093	square meters	m ²
yd ²	square yard	0.836	square meters	m ²
ac	acres	0.405	hectares	ha
mi ²	square miles	2.59	square kilometers	km ²
VOLUME				
fl oz	fluid ounces	29.57	milliliters	mL
gal	gallons	3.785	liters	L
ft ³	cubic feet	0.028	cubic meters	m ³
yd ³	cubic yards	0.765	cubic meters	m ³
NOTE: volumes greater than 1000 L shall be shown in m ³				
MASS				
oz	ounces	28.35	grams	g
lb	pounds	0.454	kilograms	kg
T	short tons (2000 lb)	0.907	megagrams (or "metric ton")	Mg (or "t")
TEMPERATURE (exact degrees)				
°F	Fahrenheit	5 (F-32)/9 or (F-32)/1.8	Celsius	°C
ILLUMINATION				
fc	foot-candles	10.76	lux	lx
fl	foot-Lamberts	3.426	candela/m ²	cd/m ²
FORCE and PRESSURE or STRESS				
lbf	poundforce	4.45	newtons	N
lbf/in ²	poundforce per square inch	6.89	kilopascals	kPa
APPROXIMATE CONVERSIONS FROM SI UNITS				
Symbol	When You Know	Multiply By	To Find	Symbol
LENGTH				
mm	millimeters	0.039	inches	in
m	meters	3.28	feet	ft
m	meters	1.09	yards	yd
km	kilometers	0.621	miles	mi
AREA				
mm ²	square millimeters	0.0016	square inches	in ²
m ²	square meters	10.764	square feet	ft ²
m ²	square meters	1.195	square yards	yd ²
ha	hectares	2.47	acres	ac
km ²	square kilometers	0.386	square miles	mi ²
VOLUME				
mL	milliliters	0.034	fluid ounces	fl oz
L	liters	0.264	gallons	gal
m ³	cubic meters	35.314	cubic feet	ft ³
m ³	cubic meters	1.307	cubic yards	yd ³
MASS				
g	grams	0.035	ounces	oz
kg	kilograms	2.202	pounds	lb
Mg (or "t")	megagrams (or "metric ton")	1.103	short tons (2000 lb)	T
TEMPERATURE (exact degrees)				
°C	Celsius	1.8C+32	Fahrenheit	°F
ILLUMINATION				
lx	lux	0.0929	foot-candles	fc
cd/m ²	candela/m ²	0.2919	foot-Lamberts	fl
FORCE and PRESSURE or STRESS				
N	newtons	0.225	poundforce	lbf
kPa	kilopascals	0.145	poundforce per square inch	lbf/in ²

*SI is the symbol for the International System of Units. Appropriate rounding should be made to comply with Section 4 of ASTM E380.
(Revised March 2003)

TABLE OF CONTENTS

DISCLAIMER.....	i
Technical Report Documentation Page	ii
SI* (Modern Metric) Conversion Factors.....	iii
LIST of FIGURES.....	vi
LIST OF TABLES.....	viii
EXECUTIVE SUMMARY	1
CHAPTER 1. INTRODUCTION.....	2
1.1 General Background.....	2
1.2 Research Motivation and Objectives	2
1.3 Overview of Research Contributions	3
1.4 Report Organization.....	4
CHAPTER 2. LITERATURE REVIEW	6
2.1 Video-Based Vehicle Detection.....	6
2.1.1 Feature-Based Methods.....	6
2.1.2 Motion-Based Methods	7
2.2 Convolutional Neural Network and Transfer Learning	8
2.2.1 Convolutional Neural Network	8
2.2.2 Transfer Learning	13
2.3 Road Boundary Extraction	14
2.4 Drone-Based Vehicle Tracking	15
CHAPTER 3. RESEARCH METHODOLOGY	17
3.1 System Architecture and Algorithm Design.....	17
3.2 CNN Network Design and Transfer Learning Mechanism.....	17
3.2.1 Modified YOLO v3 Network	17
3.2.2 Transfer Learning Mechanism	20
3.3 Image Sequence Based Self-Calibration Design and Vehicle Tracking	22
3.3.1 Road Boundary Extraction	23

3.3.2 OpenCV-Based Vehicle Tracking	27
3.4 Flow Parameter Estimation.....	30
3.4.1 Perspective transformation	30
3.4.2 Parameter estimation	31
CHAPTER 4. EXPERIMENTAL TESTS	34
4.1 Testing Hardware and Environment	34
4.2 Testing Scenarios and Data Collection.....	35
4.2.1 Testing Scenarios	35
4.2.2 Data Collection.....	37
4.3 Experimental Results and Analysis.....	37
4.3.1 Model Calibration and Metrics of Evaluation	37
4.3.2 Experimental Results.....	40
4.3.3 Analysis of Results.....	41
CHAPTER 5. CONCLUSIONS	43
References	44

LIST of FIGURES

Figure 1 Convolution Calculation Visual Diagram.....	9
Figure 2 Dynamic Illustration of Convolution Operations.	9
Figure 3 Commonly Used Activation Functions.	10
Figure 4 A max pooling operation with filter size (F) =2 and stride (S) = 2.	11
Figure 5 The structure and the output of FC layer.....	11
Figure 6 R-CNN structure from [13].	12
Figure 7 A simple neural network structure with 2 hidden layers.....	13
Figure 8 Learning Process of TL from [23].	13
Figure 9 A network trained using TL method.....	14
Figure 10 The Flow Chart of the Proposed Approach.....	17
Figure 11 YOLO v3 Structure.	18
Figure 12 YOLO v3 Detection and Classification.	19
Figure 13 Examples of Training Dataset.....	20
Figure 14 The Number of Objects Per Image of Dataset.	21
Figure 15 Occlusion Degree of Objects.	21
Figure 16 Device Synchronization Process.....	22
Figure 17 A screenshot of the output video.	22
Figure 18 Left: original frame. Right: adjusted frame.....	23
Figure 19 A simple example of Gaussian mean method in a neighborhood.	24
Figure 20 Example of Dilation and Erosion operations.....	25
Figure 21 Result of adaptive threshold segmentation and morphological operations.....	25
Figure 22 Result of Hough transform.....	27
Figure 23 Detected boundaries.....	27
Figure 24 Intersection over Union (IoU).	28
Figure 25 Left: detection result of NN. Right: NN detection combined with object tracking.	29
Figure 26 The Result of Perspective Transformation.....	31

Figure 27 Center Drift.	32
Figure 28 Configuration of Hardware.	34
Figure 29 Preset Waypoint Information.	35
Figure 30 Actual Traffic Conditions In Congestion.	39
Figure 31 Road Fork on the Test Section.	42

LIST OF TABLES

Table 1 Description of Test Videos.....	36
Table 2 Confusion Matrix.....	37
Table 3 Level of Service for basic freeway sections for 70 km/h design speed [47]	38
Table 4 Preliminary Test Results	39
Table 5 Experimental Results of Vehicle Detection	40
Table 6 Experimental Results of Vehicle Average Speed.....	41
Table 7 Experimental Results of Vehicle Density.....	41

EXECUTIVE SUMMARY

The acquisition of traffic system information, especially vehicle speed and trajectory information, is of great significance to the study of the characteristics and management of the traffic system. The traditional method of relying on video analysis to obtain vehicle number and trajectory information has its application scenarios. Still, the standard video source is often a camera fixed on a roadside device. In the videos obtained this way, vehicles are likely to block each other, seriously affecting vehicle detection accuracy and speed estimation. Although there are methods to get high-view road videos utilizing aircraft and satellites, the corresponding cost will be high. Therefore, since drones can obtain high-definition video at a higher viewing angle and the price is relatively low, we decided to use drones to get road videos to complete vehicle detection.

Our proposed method uses convolutional neural network technology to overcome the shortcomings of traditional object detection methods when facing a large number of targets and complex scenes. We modified the YOLO v3 network structure and used a vehicle data set captured by drones for transfer learning. Finally, we trained a network that can detect and classify vehicles in videos captured by drones. At the same time, a self-calibrated road boundary extraction method based on image sequences was used to extract road boundaries and filter vehicles to improve the detection accuracy of cars on the road. Using the results of neural network detection as input, we use video-based object tracking to complete the extraction of vehicle trajectory information. Finally, the number of vehicles, speed, and trajectory information of vehicles were calculated, and the average speed and density of the traffic flow were estimated.

The proposed model can achieve more than 98% detection accuracy in our experimental results under different weather, lighting conditions, and traffic flow scenarios. Based on the obtained vehicle trajectory data, we can also complete the estimation of the average speed and average density of the traffic flow on the test road. Concerning the Level of Service (LOS) measurement index for ordinary expressways and the actual speed limit of the test road, the average density of vehicles calculated can accurately reflect the degree of vehicle congestion on the road, which can provide a reference for the intelligent traffic management system. The data obtained from the experiment can be used as a reference data set for studying the safety index of freeway traffic systems and connected vehicles after manual calibration.

CHAPTER 1. INTRODUCTION

With the development of intelligent transportation technology, the emergence of new technologies such as connected and driverless vehicles have brought new challenges to the design of modern transportation systems. Considering the data requirements in developing these complex and data-intensive technologies, the past traditional transportation system information acquisition technologies have difficulty meeting the needs of real-time speeds and positions. In the past decade, with the advancement of computer hardware and theoretical research, the research on convolutional neural networks (CNN) has made significant progress. This technology is widely used in natural language processing, object detection, and classification models. This report applies neural networks to vehicle detection and computer vision technology to propose a new method to obtain detailed and macro parameters of vehicles on the road from videos taken by drones. In the first chapter, we focus on the background and motivation of this research, formulate our research goal and summarize our contributions.

1.1 General Background

The dynamic parameters of the transportation system play a vital role in the management of the transportation system. These data are also essential for traffic system analysis. Especially now that connected vehicle research has entered a new stage, high-quality traffic system dynamic parameter data (e.g., speed, density, delay, trajectory data) can provide strong support for developing related control systems. For now, mainstream vehicle dynamic parameters collection methods include video analysis, GPS (Global Positioning System) probe car, Radar and magnetic field sensors, and other methods [1].

In the past few decades, video processing technology has made significant progress. Still, due to the fixed angle of CCTV cameras on the roadside infrastructure, the traditional CCTV system cannot provide video of vehicle driving without mutual occlusion. Therefore, analyzing the classic road surveillance system video still faces limitations in using probe cars equipped with GPS devices to monitor road conditions. However, this method can be used to estimate the vehicle's driving time on a particular road and give information such as the degree of congestion. Due to the limitation on the number of probe vehicles, it is impossible to cover all roads simultaneously. In addition, this method can only obtain fuzzy information about the road system, which cannot provide information such as the number of vehicles and accurate vehicle speed. Radar sensors have been widely used to monitor the number and rate of cars on the road and provide a certain degree of protection for the safety of high-speed roads. But once the number of vehicles is too large, this system cannot provide accurate data. Other methods also have applications in specific situations, such as using magnetic induction coils to estimate the speed and number of vehicles. Still, they also produce obvious errors when vehicles are congested on the road, so their application scenarios are also limited.

1.2 Research Motivation and Objectives

From the above comparison, we can see that traditional traffic parameter acquisition methods have their limitations. We may obtain good results through complex operations, but the related costs will also increase. To this end, we hope to propose a new traffic monitoring method that can obtain vehicle

trajectory extraction, carry out statistics on the number of vehicles and speed estimation, and ensure the accuracy of the results without spending high staffing and material costs.

Among all the methods mentioned above, the video analysis method may be affected by weather and light conditions, but it still can give reliable dynamic traffic system information. However, we continue to use the road video obtained by the traditional fixed-view CCTV system. In that case, we will still need help locating the vehicle and accurately estimating the vehicle speed. Although we can use higher altitude videos taken by helicopters to avoid mutual occlusion between cars, from an economic point of view, we cannot use helicopters and other aircraft to obtain road videos from aerial photography at different periods every day. Considering these factors, we decided to use drones for road video recording. Presently, consumer-grade drones on the market can already record 4K+30fps videos from a high view and achieve a flying height of up to 500 meters and a flying distance of about 2 kilometers. At the same time, the price of a drone is generally one thousand to several thousand dollars. In contrast, a road surveillance camera with a fixed angle of view costs about ten times more, not to mention the high cost of a single helicopter flight.

Based on these conditions, the research goal of this report is to study, design, and implement a system that uses drones to obtain high-view road videos, uses trained neural networks to complete vehicle detection and obtain vehicle trajectories, and then uses computer vision technology to complete the estimation of vehicle speed. Our goal is to explore whether such a system can provide accurate and reliable road macro dynamic parameters, determine what improvements can obtain better efficiency of this system, and propose new goals for future work.

1.3 Overview of Research Contributions

The advantage of our research is that this system uses drones to obtain high-view road videos that cannot be obtained by traditional road monitoring systems, which can effectively reduce the error caused by mutual occlusion of vehicles and improve the accuracy of the results from the root cause. At the same time, unlike the existing articles that use drones to hover over a fixed point to complete vehicle detection on a fixed road segment, our drone uses a pre-set route to obtain a road monitoring video. Therefore, the accurate position and trajectory of the vehicle on a longer road can be obtained. On this basis, our research is combined with neural network technology, which can avoid the traditional object detection process that requires complex parameter adjustment in different environments. Our main contributions are listed below.

- **Transfer-Learning-Based Neural Network for Vehicle Detection.** In this work, we tried to train a neural network using transfer learning (TL) technology according to previous research. Built based on YOLO v3 (You Only Look Once version 3.0) neural network, we modified the structure of the original network and retrained it with a small dataset for drone-based vehicle detection competition. The process was finished on the Google Collaboratory platform, and the network worked well in our experiments. Although the speed of detection on videos with a resolution of 1920×1080 is only about five frames per second, the accuracy on our test videos is above 96% on average. If we reduce the resolution of videos, we can acquire higher process speed.

- **Computer Vision-Enabled Self-Calibrated Road Boundary Detection.** To improve the accuracy of vehicle detection and avoid the interference of roadside vehicles and vehicles on branch roads, we proposed a computer vision-based self-calibrated road boundary detection method. This method combines the Hough transform in digital image processing to complete the preliminary detection of straight lines in the picture and then uses the boundary positions of the several initial frames and the relationship between two consecutive frames to complete the preliminary extraction of the boundaries. On this basis, the precise road boundaries are further determined based on the positions of the vehicles detected in the current picture.
- **Vehicle Tracking and Trajectory Extraction.** Considering the errors that may occur when only using neural networks for vehicle detection, the network incorrectly recognizes objects as vehicles, and some vehicles are not detected. At the same time, to obtain each car's precise trajectory and speed information, we used OpenCV to complete the vehicles' further tracking based on the neural network's detection results. This method can further improve vehicle detection accuracy and complete the recording of each vehicle's trajectory.
- **Traffic Flow Parameter Estimation.** Based on vehicle detection and extracted vehicle position information, we have realized the estimation of macroscopic parameters of traffic flow. In the final output, we can count the total number of vehicles on the current frame and the cumulative total number of vehicles from the beginning of the video, and the speed of each vehicle is also marked on the corresponding vehicle. At the same time, we can count the density and average speed of vehicles over time.
- **Calibrated Vehicle Trajectory Dataset.** Based on the results of our experiments, we processed the videos taken by the drones and manually corrected the results to obtain a data set with accurate vehicle trajectories. This data set can be used for road system safety factor evaluation and further used as a benchmark data set for subsequent system improvements.

1.4 Report Organization

In this introductory chapter, we have outlined the general background of this work, the motivation and goal of our research, and a summary of the work's significant contribution. The rest of this report is focused on constructing our proposed method.

After reviewing related work, we will elaborate on the research methodology of our work. Later, after discussing the details of our proposed method, we will summarize the process of the entire system and conduct a series of experiments under different conditions. Details of the content of the subsequent chapters are listed below.

- **Chapter 2** reviews the pieces of literature related to our research. This chapter includes a background of objective detection, convolutional neural network and transfers learning, road

boundary extraction, vehicle tracking, and traffic flow parameter estimation. By reviewing related work, this chapter provides a theoretical basis for our system design.

- **Chapter 3** gives details of our research methodology. We provide the design and system architecture to give a complete view of our work. Then we dig deeper into how the various parts of the proposed system cooperate closely to complete the detection of vehicles on the road and the estimation of traffic macro parameters.
- **Chapter 4** shows the experiments' results using our proposed model and the analysis of the final results. This chapter details the hardware equipment and software environment used in this research. After using the drone to obtain the road video, the statistics of the number of vehicles on the road and the extraction of the trajectory of the detected vehicles are completed by processing and detecting the video. On this basis, we met the estimation of the detected vehicles' speed and traffic flow parameters.
- **Chapter 5** presents some of the current limitations by reviewing and summarizing the proposed methods. The current algorithm only uses a CPU as the main unit of calculation and cannot reach the speed of the GPU, so real-time detection is not achieved. In addition, we need to continue training the neural network to improve the detection accuracy of different vehicles in different scenarios to improve the algorithm's overall performance further.
- **Chapter 6** makes a summary of this research and gives conclusions about the research results. This research proposes a new method that uses video taken by drones as input, combined with convolutional neural networks and traditional computer vision technology to complete vehicle detection and vehicle trajectory extraction. On this basis, the macro parameters of traffic flow are estimated, and the results obtained by the proposed method can provide detailed and accurate traffic parameter data sets for future research on intelligent vehicle networks.

CHAPTER 2. LITERATURE REVIEW

This chapter reviews some of the works most relevant to our research. These studies provide references and a theoretical basis for our proposed method. These previous works' solid results have provided a theoretical basis and inspired us to use drones to record videos to overcome the inherent disadvantages of traditional vehicle detection systems, such as fixed viewing angles and vehicle occlusion problems. This chapter is divided into four main sections: video-based vehicle detection-related works, convolutional neural network-related works, road boundary extraction-related works, and drone-based vehicle tracking-related research.

2.1 Video-Based Vehicle Detection

Video-based vehicle detection systems can generally be divided into two categories for two stages: hypothesis generation (HG) models and hypothesis verification (HV) models [2]. Typically, HG models extract the regions of the picture that most likely contain the vehicles using the characteristics of the vehicles. The results obtained using HG models must be evaluated using hypothesis verification models. The standard technique in this step is machine learning. Sometimes, the second step models also need to classify the detected vehicles into the correct types. This kind of method is usually considered a two-stage method.

Since most vehicle detection systems are based on video, these methods usually rely on features or motion to complete vehicle detection. Feature-based methods use prior knowledge to detect the vehicles in the frame. Motion-based approaches rely on the differences caused during the motion process of vehicles to distinguish between the vehicle and the background. The result of this step is usually used as input to HV models. Here we mainly review HG models, which can predict the positions of vehicles in the images.

2.1.1 Feature-Based Methods

Feature-based methods need to use information from the frame to decide the regions of interest where vehicles may exist. Here we review several approaches using color space, edges, and vehicle lights.

1) Color

When trying to identify a vehicle from the image, a very intuitive phenomenon is that the color of the vehicle is generally different from the road. Although the vehicle may be challenging to recognize directly due to light, weather, and background texture, its color will always maintain its characteristics compared with the surrounding background [3]. In the pictures captured by most photographic equipment, in the color space composed of three elements of red, green, and blue (RGB space), the colors of all objects can be displayed with different numerical combinations of these three variables. Therefore, when we want to detect a vehicle with a particular color, such as a red car, we need to analyze the value of the red channel in the RGB space.

2) Edges

Consider that the vehicle can be simplified as a three-dimensional structure similar to a cuboid. Therefore, when we observe from the front or rear of the vehicle, we will always notice lines parallel or perpendicular to the coordinate axis of the viewing angle system, such as windows, trunks, and other

parts of the vehicle. As an inherent property of the vehicle, even if the vehicle is in the process of moving, we can always detect these edges to determine the boundary of the vehicle [4].

3) Vehicle Lights

The aforementioned vehicle detection methods have their advantages in different application scenarios, but when the detection environment is at night, these methods will lose their advantages. In the case of poor lighting conditions at night, the outline, shadow, color, and other information about the vehicle will become challenging to obtain. In this scenario, the vehicle's lights can be used to quickly identify the vehicle from a low-brightness or completely black background [5].

2.1.2 Motion-Based Methods

The vehicle detection method mentioned above can be applied separately in each video frame. We only need to use prior knowledge of the vehicle, and we can detect the vehicle from a static picture. However, in the video of road traffic flow, the vehicles are often moving. At this time, in the absence of prior knowledge, we can distinguish them from the static background with the help of the movement of the vehicles. Such methods often require a series of continuous pictures to provide information about vehicle movement and background. These methods can be divided into non-parametric methods and parametric methods.

1) Non-parametric Methods

When it comes to using movement to detect vehicles, the most intuitive method is to compare two consecutive frames [6] and determine the position of the moving vehicle by extracting the pixels whose pixel values have changed in the two frames before and after comparing them with the set threshold. The threshold selection of this method is affected by the lighting conditions, and the segmentation results may appear as gaps as the detection object moves slowly. For this reason, some researchers proposed using the frame to calculate the difference to detect slow-moving vehicles [7]. Still, when the vehicles are moving fast, different vehicles may overlap, resulting in detection errors.

Another detection method is subtracting the background from the current frame to obtain the moving vehicle detection result [8]. However, due to the presence of slow-moving or parked vehicles, the background of this method will be quickly filled up by these relatively static vehicles in practical applications, resulting in incorrect recognition.

2) Parametric Methods

Standard parameterization methods include the optical flow method and background modeling method. The relative movement between the object and the observer causes optical flow. We can perceive and track the vehicle's displacement using an optical flow algorithm. By moving the centroid of the vehicle between consecutive frames and calculating the vehicle's speed, we can estimate the distance of the vehicle [9].

Different from the optical flow method, in the background modeling method, the background is regarded as motionless, and any moving objects are regarded as part of the foreground. To identify moving objects, this method usually includes background initialization, foreground object detection, and background update. In this way, by continuously updating the background, the separation of the moving object and the background can be accurately completed [10].

2.2 Convolutional Neural Network and Transfer Learning

We reviewed some of the common vehicle detection methods in the previous section. In contrast, in computer vision, the model needs to finish the detection and label the corresponding object types of every instance of objects on the screen in the object detection task. At this time, traditional object detection methods may not be able to meet this demand.

At first, when the neural network showed good performance, it was limited by the development of hardware and theory, which restricted further development. After a long period of stagnation, with the development of AlexNet [11], researchers in the field of computer vision discovered the great potential of convolutional neural networks. They applied them to object detection [12], image classification [13], image segmentation [14], and other fields. Unlike traditional object detection and classification methods, neural networks can better extract and learn higher abstract-level object features and accurately complete target contour extraction and object classification. The performance of traditional target detection methods will gradually saturate when faced with a large amount of data; that is, as the amount of data increases, although its performance will be improved, the gain brought by the more significant amount of data is minimal. However, the neural network can perform better by training large amounts of data in the exemplary scenario.

2.2.1 Convolutional Neural Network

By using window sliding detection technology [15], we can reduce the time complexity required for target detection tasks. This method first requires a window with a preset size. It then gradually slides the window across the entire screen and, at the same time, classifies and labels the items in each window to complete the identification of all objects on the screen. To complete the detection and classification of items, we need to train a classifier to recognize different objects. Just like teaching children what a car is, we need to give positive examples that include cars and negative examples that do not. The training of the classifier is completed with the help of the preset loss function and other measures, as the convolutional neural networks have great potential in object detection, classification, and image segmentation. After years of development, we now have art-of-the-state neural networks like R-CNN [16], Faster R-CNN [17], YOLO [18] series, and SSD [19].

CNNs use deep learning algorithms and usually deal with high-dimensional data like images and videos. To obtain a higher level of abstraction of input features, CNNs usually have a multi-layer architecture, which may contain layers like convolutional layer(s) (conv), fully connected layer(s) (FC), etc. [20] Here, we briefly introduce some constituent layers of the convolutional neural network to understand the structure and corresponding functions of CNNs.

1) Convolutional Layer

The convolutional layer is an indispensable structure of CNN. The input picture or other data is operated with the convolution kernel; that is, a two-dimensional activation map is output by calculating the dot product of the data and the convolution kernel. Through this step, the network can learn which filter should be activated when a particular feature appears. Neurons in the same feature map share the same weight parameters, and therefore the network would not need to set weight parameters for each of them, which can reduce the complexity of the whole network [21].

As Figure 1 shows, set a 3×3 sub-block of the input matrix as M with elements $m_{i,j}$ and the kernel as matrix K, then we have

$$M \otimes K = \frac{1}{9} * \sum_{i=0, j=0}^{i=2, j=2} m_{i,j} = 3 \tag{1}$$

The definition of convolution operation in CNN is different from mathematical two-dimensional convolution. The convolution operation here means the elements at the corresponding position of matrix are multiplied and added up [22]. After completing the convolution operation of the first sub-block, move one stride S to the next sub-block and perform the same operation until the entire input matrix completes the convolution operation.

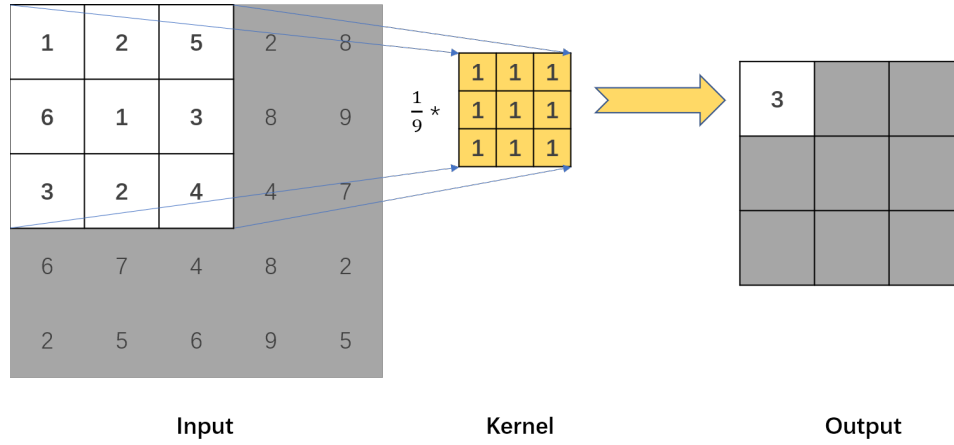


Figure 1: Convolution Calculation Visual Diagram.

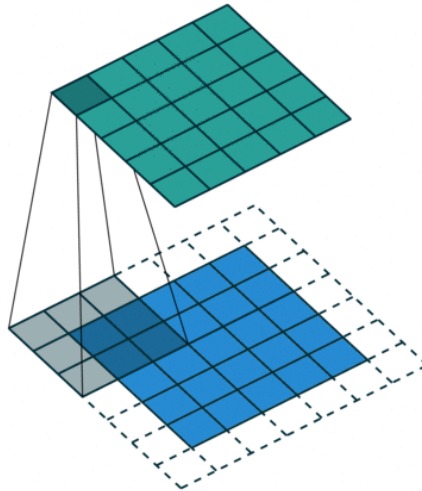


Figure 2: Dynamic Illustration of Convolution Operations

Figure 2 shows the process of a simple convolution operation. A convolutional layer usually has multiple convolution kernels to extract different features¹.

2) Activation Functions

The activation function plays an important role, and the common functions are sigmoid, Tanh, ReLU, etc. [20] The graphs of some functions are shown below in Figure 3. Through the activation of these functions, the input signal is transformed into an output signal, which will be used as the input for the next layer. If there is no activation function, the output y of the entire network will be a linear combination of the input features x .

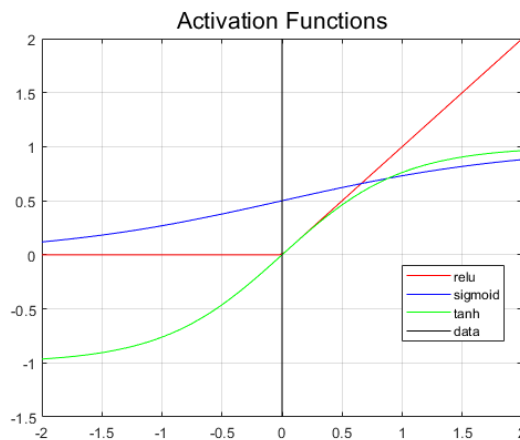


Figure 3: Commonly Used Activation Functions.

3) Pooling Layer

The role of the pooling layer is to down sampling to reduce parameter dimensions, remove redundant information, compress features, simplify network complexity, reduce calculations, reduce memory consumption, and so on. While using the pooling layer to reduce parameters, it can also make the network easier to converge. Common pooling methods include average pooling and maximum pooling. Take maximum pooling as an example here in Figure 4.

¹ <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>

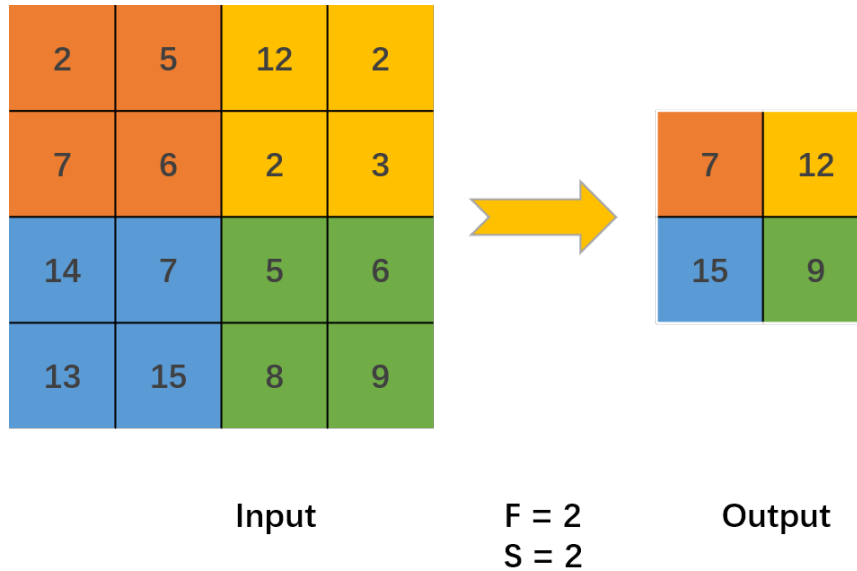


Figure 4: A max pooling operation with filter size (F) =2 and stride (S) = 2.

4) Fully Connected Layer

Each node of the fully connected layer is connected to each node of the previous layer, so that the previously obtained features can be integrated for classification or regression. Because of this feature, this structure needs to consume a lot of memory resources. For example, an FC layer has 1024 nodes, and the output of the upper layer also has 1024 nodes, so 1024×1024 parameters are needed for transmission (without bias parameters). Another example is shown in Figure 5.

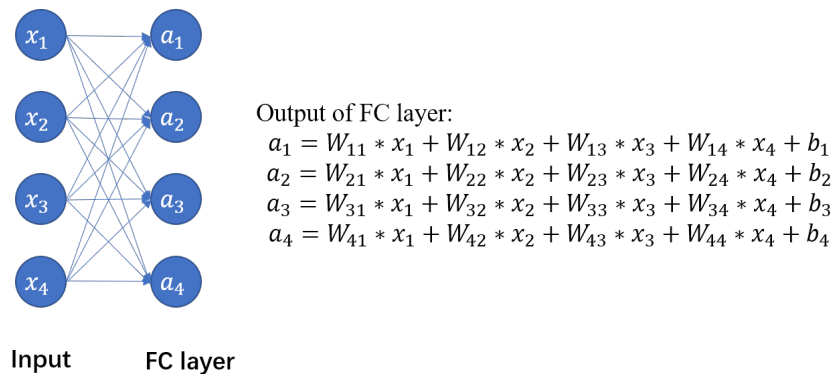


Figure 5: The structure and the output of FC layer. x_i stand for the output of last layer.

$W_{m,n}$ stand for the weight parameters. b_k stand for the bias parameters.

5) Loss Layer

The loss layer can measure the difference between the network prediction result and the ground truth, thereby changing the network training parameters to optimize the network. Common loss functions include Softmax, Cross-entropy, etc.

According to the process of the neural network to complete the target detection, they can be divided into two categories: two-stage methods and one-stage methods.

a) Two-Stage Methods

In the two-stage detection method, the first stage completes the proposal of the regions of interest (ROI) or the locations of the target objects, and then the second stage completes the detection and classification of these proposals and marks the result with bounding boxes.

As a representative of this type of neural network, R-CNN [16] is also the first model to use convolutional neural networks to complete two-stage target detection. The structure diagram of the R-CNN network is shown in Figure 6. The structure of the first stage uses the input picture to propose the ROI to be classified through the algorithm. The second stage of the convolutional neural network consists of 5 convolutional layers and two fully connected layers, which is able to classify the proposed areas from first stage.

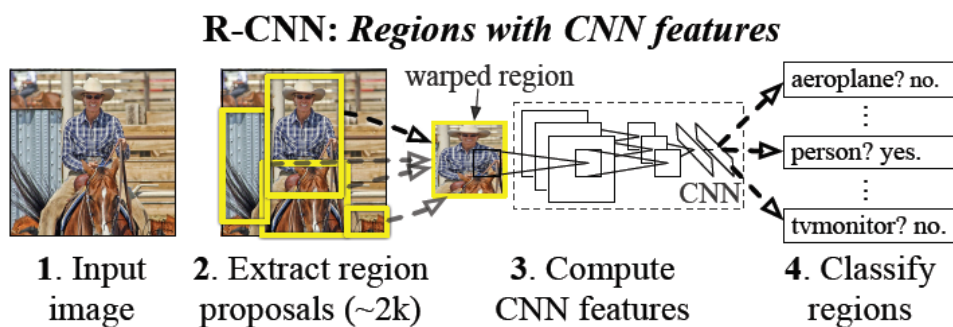


Figure 6: R-CNN structure from [13].

b) One-Stage Methods

As the name suggests, the one-stage method is to merge the ROI proposal and classification into one step. While using the input features for object detection, prediction of the category of the object is also done. The representative of this kind of method is the YOLO [18] series neural network. After processing the characteristics of the input picture, the YOLO network divides the picture into a fixed number of grids, and each grid predicts a set number of bounding boxes with confidence. The confidence score is obtained by multiplying the object detection probability by the intersection ratio of the expected box and the actual box. When the confidence score exceeds a preset threshold, the YOLO network considers that a specific object category has been detected in the area.

2.2.2 Transfer Learning

While training a NN, researchers usually need numerous labeled samples to “feed” the complex network so that it would be able to identify different objectives and achieve high accuracy. Take Figure 7 for example, even a two-hidden-layer NN could contain 8 neurons and 16 connections using the fully connected structure. However, researchers may not be able to acquire big data set containing ten thousand of labeled pictures to train their own models. To solve this, scientists proposed a method called transfer learning (TL) to make it possible to train a network for new tasks using a pre-trained model as shown in Figure 8. Transfer learning relaxes the assumption that the training data must be independently and identically distributed (i.i.d.) with the test data [23]. Therefore, compared with normal neural network training, transfer learning can greatly reduce the computational complexity, data set size, and training time.

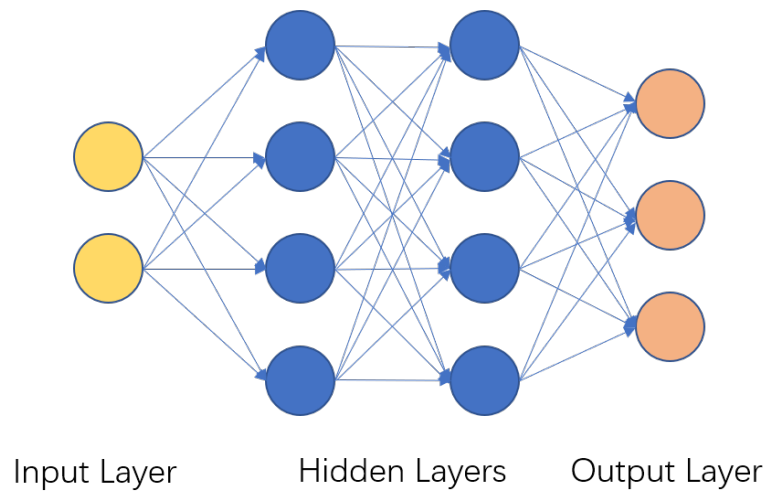


Figure 7: A simple neural network structure with 2 hidden layers.

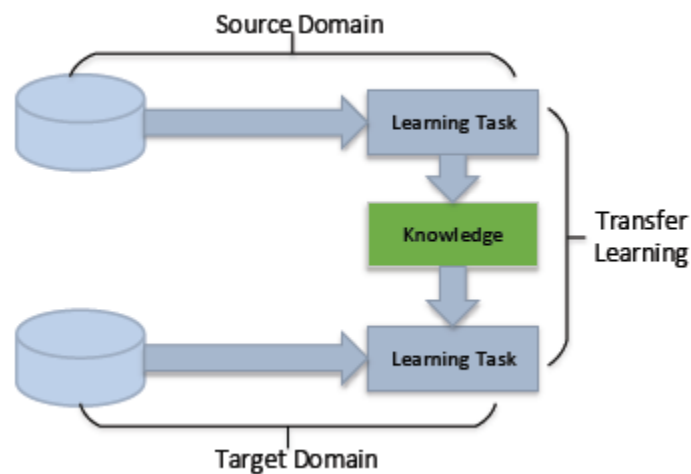


Figure 8: Learning Process of TL from [23].

Deep neural networks have been widely used in various scenarios, so a large amount of transfer learning methods have also been proposed to train networks that meet different needs. Based on the techniques used in TL, transfer learning can be divided into four types: instance-based transfer learning, mapping-based transfer learning, adversarial-based transfer learning and network-based transfer learning [23]. Like Figure 8 shows, the target domain model can be retrained by using a small data set to identify trucks based on the source domain network trained to detect cars. The source domain network could be represented using Figure 7, while the transfer learning applied target domain network could share the same hidden layer structure and adjust its input and output layers to achieve different detection goal as Figure 9 shows.

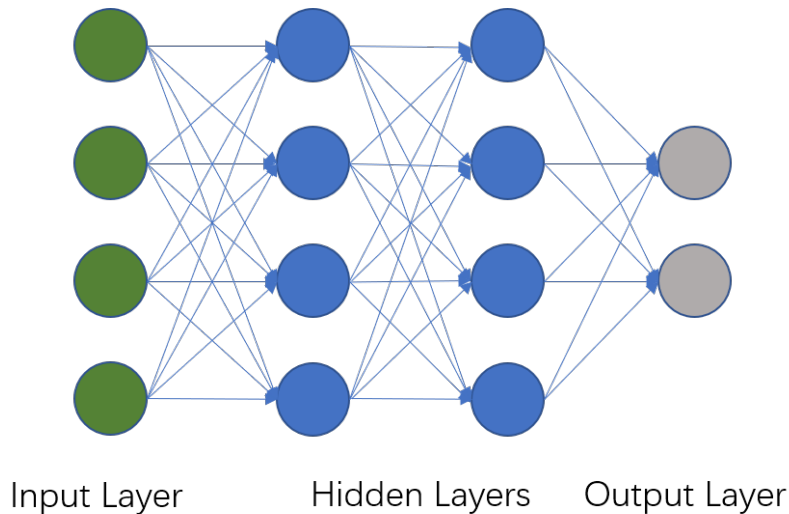


Figure 9: A network trained using TL method. The input and output layers are adjusted to satisfy the need of new task, but the hidden layers structure is the same as the Fig.7's.

2.3 Road Boundary Extraction

Many algorithms have been developed to extract road boundaries under different scenarios. Videos for analysis are usually from three primary sources according to the view difference: a low view from cameras fixed in or on vehicles, a mid-height view from planes like helicopters, and a high view from satellites. However, with the wide application of UAVs, it is easier to acquire mid-height view videos of roads. This report focuses on analyzing videos taken by UAV and then needs to extract road boundaries from mid-height view videos.

Without considering special scene requirements, most road boundary extraction algorithms can be divided into feature-based and model-based methods [24]. Usually, feature-based methods need to separate the road boundary according to the differences between lanes and backgrounds. When processing video, the commonly used features are color space, edges, and patterns that recur in exceptional cases. In [25], the color space of the picture is used for processing. The values of R, G, and B channels are all considered to propose the regions that may contain road areas. This way, the image area with a specific color can be extracted, but the result is easily affected by light, shadow, and camera

angle. While using edge features, since the image pixels on both sides of the edge will change drastically, the standard method is to use this feature to extract the edge and then further filter out the road boundaries [26]. Some would use the Hough transformation method to find candidate lines, and the most likely results as boundaries [27]. In [28] and [29], the method incorporated morphological operations and gradients to find edges as road boundaries. However, the edge-based method may only give promising results once the lanes are clearly shown in the pictures. In general, to obtain better processing results, different methods usually use mixed features as input.

As the model needs a lot of road samples to be built, the researchers usually collect different road samples with different contours, background colors, and shadows. However, if the road has sudden changes in its color, texture, or background, it may lead to wrong detection results. In contrast, model-based methods may be more robust than feature-based approaches. Typically, the model-based methods would represent the boundaries using models with prior knowledge. Model-based methods aim to find out the fittest parameters for the models. Several different strategies for model fitting were proposed in [30, 31]. In [30], a quadratic road surface model was first fitted by a RANSAC approach. Then the primary result was refined by a region-growing-like process driven by the 3D resolution and the uncertainty model of the sensor to give the detected result. In [31], the author proposed Least Square Estimation (LSE) based approach to fit a model for the road surface. Both feature-based and model-based methods would give results of lane boundaries, but the model-based methods are more inclined to express the results as concise mathematical expressions.

In this report, since the UAV usually flies under good weather conditions, we need to pay more attention to the spin of the UAV and the changes in features because the video collected does not have temporal and spatial stability. Due to this reason, model-based methods may not be robust, so hybrid of color; the edge is used as an input feature to extract road boundaries. To make the extraction result more reliable, the relationship between two consecutive frames is also considered to avoid failure of edge detection.

2.4 Drone-Based Vehicle Tracking

For computer vision researchers, moving object tracking has always been challenging. To accurately calculate the number of detected vehicles and the speed of their movements, video-based object tracking technology is applied to this process. In this report, we use videos obtained by drones to detect and track vehicles. Since the drone is also moving along the direction of the road, the estimated speed of vehicles that move in the same direction as the drone should be minus the drone's speed to acquire actual speed estimation results. As for those vehicles in the opposite direction from the drone, their speeds should add the drone's speed to get their estimated speed. In such a scenario, drone-based vehicle tracking is simplified to video-based object tracking that needs to consider the movement of the lens. In its development history, object tracking methods can be divided into two categories: generative model tracking and discriminative model tracking.

A generative model is the research focus of early object-tracking algorithms. Related algorithms include the optical flow method [32], the Mean-shift algorithm [33], the Camshift algorithm [34], and so on. This type of algorithm first uses the characteristics of the target to build a target model and then searches for the target attributes in subsequent frames. If the matching features are found, it is considered that the target object is found, and the entire tracking process is iteratively completed in sequence. This type of method needs to make full use of the background information. Therefore, due to the randomness and

diversity of the target's appearance, when the ambient light changes and the motion causes the target to rotate, blur, or be blocked, the accuracy of the tracking result will be reduced.

With the development of deep learning technology, researchers have begun to use neural networks to complete target detection and tracking. In contrast to the generative model, the discriminant model considers both target features and background information and uses the difference between the target model and background information to extract the target [35]. This method has great potential through researchers' testing and evaluation of data sets. After the filtering method in the communication field was introduced into the practice of object tracking, a technique called correlation filtering showed surprising results in terms of speed and accuracy [36, 37, 38].

This research uses the results of neural network detection as input for object tracking. It uses the discriminative correlation filter tracker with channel and spatial reliability to build a tracker list to track one or more target vehicles in the picture. This type of tracker can accurately track the vehicles detected in the picture and save the position of each vehicle in each frame, which is used to calculate the moving speed of the vehicle and extract the trajectory data set.

CHAPTER 3. RESEARCH METHODOLOGY

This chapter gives details in each step of our drone-based computer vision-enabled self-calibrated traffic flow parameter estimation model. The system's module composition, key algorithm design and parameter calculation process are provided in detail.

3.1 System Architecture and Algorithm Design

As Shown in Figure 10, the proposed method has five key modules to process the videos taken by a drone: (1) vehicle detection CNN (2) road boundary extraction (3) vehicle tracking (4) perspective transformation (5) parameter estimation.

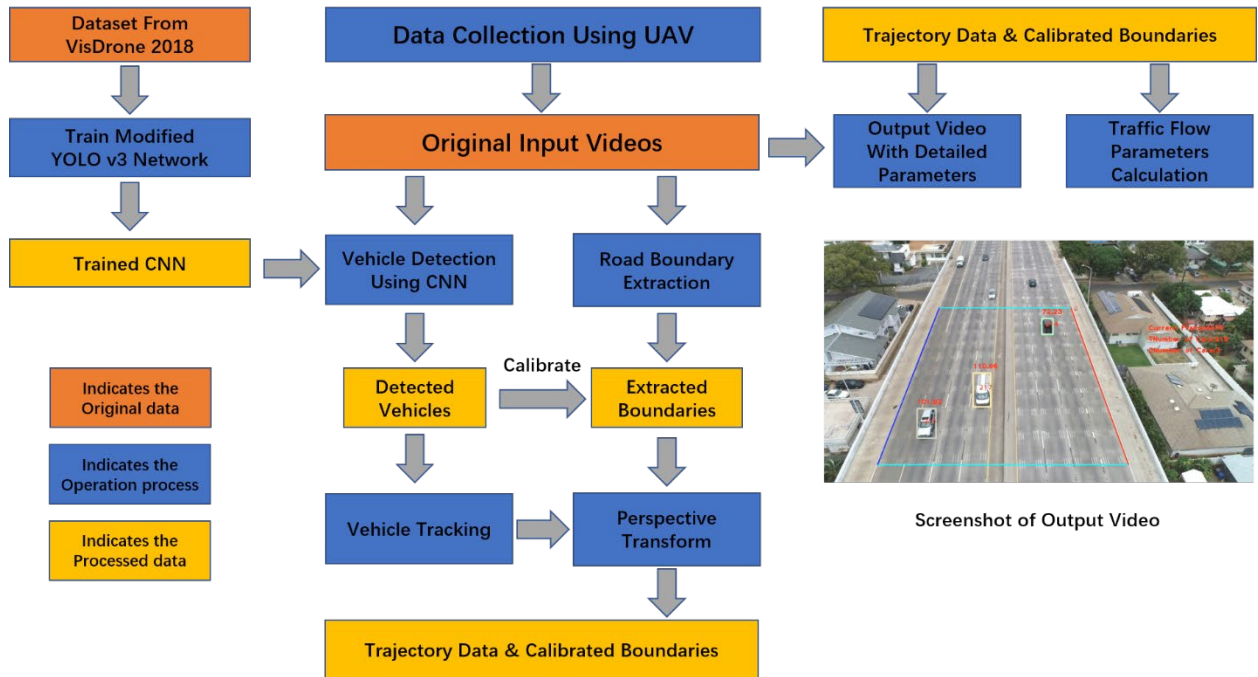


Figure 10: The Flow Chart of the Proposed Approach.

3.2 CNN Network Design and Transfer Learning Mechanism

3.2.1 Modified YOLO v3 Network

For this project, the model used for transfer learning is You Only Look Once version 3 (YOLO v3) [39]. You only look once (YOLO) is a state-of-the-art, real-time object detection system. The version used is YOLO v3, which is able to classify about 80 different kinds of objects. The key architecture of YOLO v3 is darknet53, which is a convolutional neural network comprises of 53 convolutional layers.

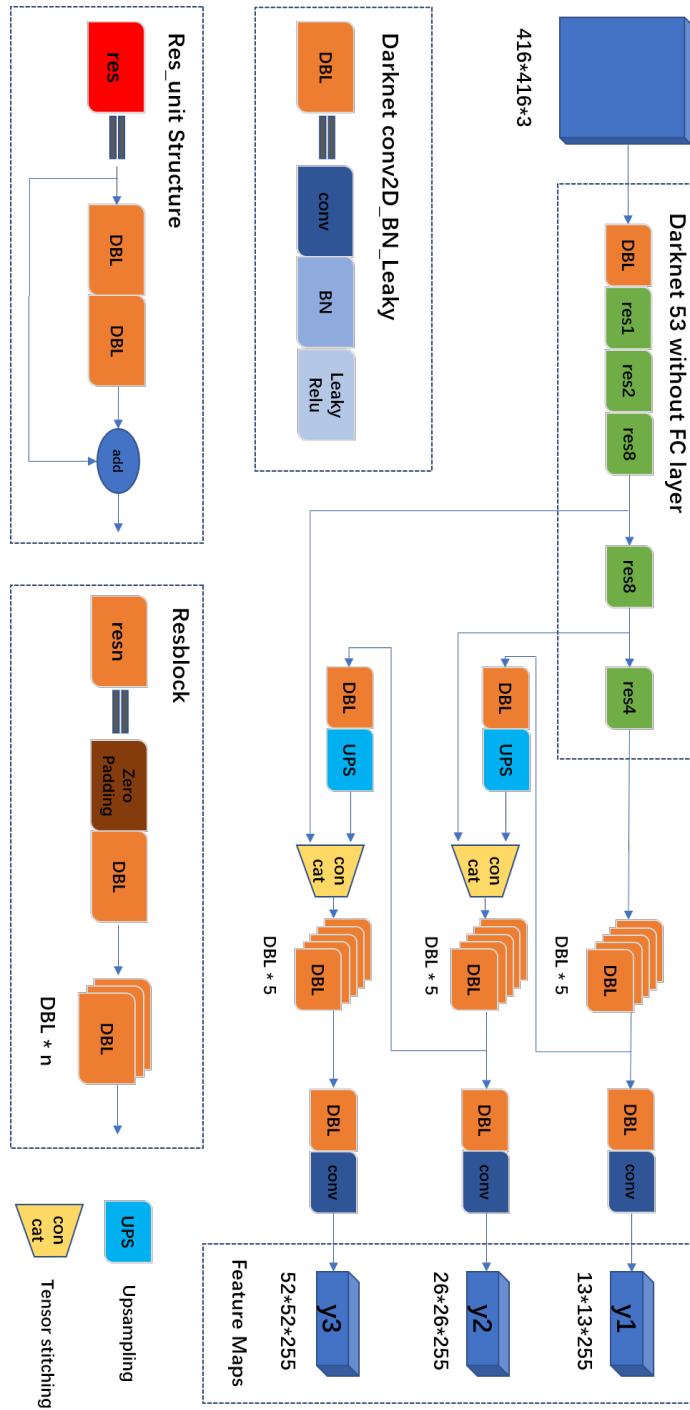


Figure 11: YOLO v3 Structure.

Figure 11 shows the detailed structure of YOLO v3. The structure used is the YOLO v3 pre-trained structure. The input picture is processed and resized to a size of $416 \times 416 \times 3$. After the processing by Darknet 53, in order to perform feature extraction and detection of objects of different sizes, the subsequent network structure uses tensor stitching to stitch together the results of different processing

stages to extract feature maps of different scales. Finally, in the output y_1, y_2, y_3 , we got three feature maps of different scales respectively. Since there are 80 types of objects to be recognized, the last dimension of the feature map, 255, is equal to $3 \times (5 + 80)$. We can see that the three outputs have different scales. This is because the network needs to detect objects of different sizes. Since the smaller feature map has a larger receptive field, the 13×13 size feature map is used to detect large-sized objects, the 26×26 size feature map is used to identify medium-sized objects, and the 52×52 size feature map is used to identify small-sized objects. In this way, objects of different sizes can all be considered, and the detection ability of the network is enhanced.

In Figure 12, we show how YOLO v3 detects and classifies objects in the picture. The network divides the picture into $S \times S$ grids, and each grid is responsible for predicting B bounding boxes, the corresponding confidences and C classes probabilities. Each box needs to have five basic parameters ($x, y, w, h, \text{confidence}$). Therefore, the output dimension of the feature map we get is $S \times S \times B \times (5 + C)$. After obtaining the feature maps of the picture, the subsequent fully connected (FC) layer is used to predict the classification probability of the object and give the coordinates of the bounding boxes.

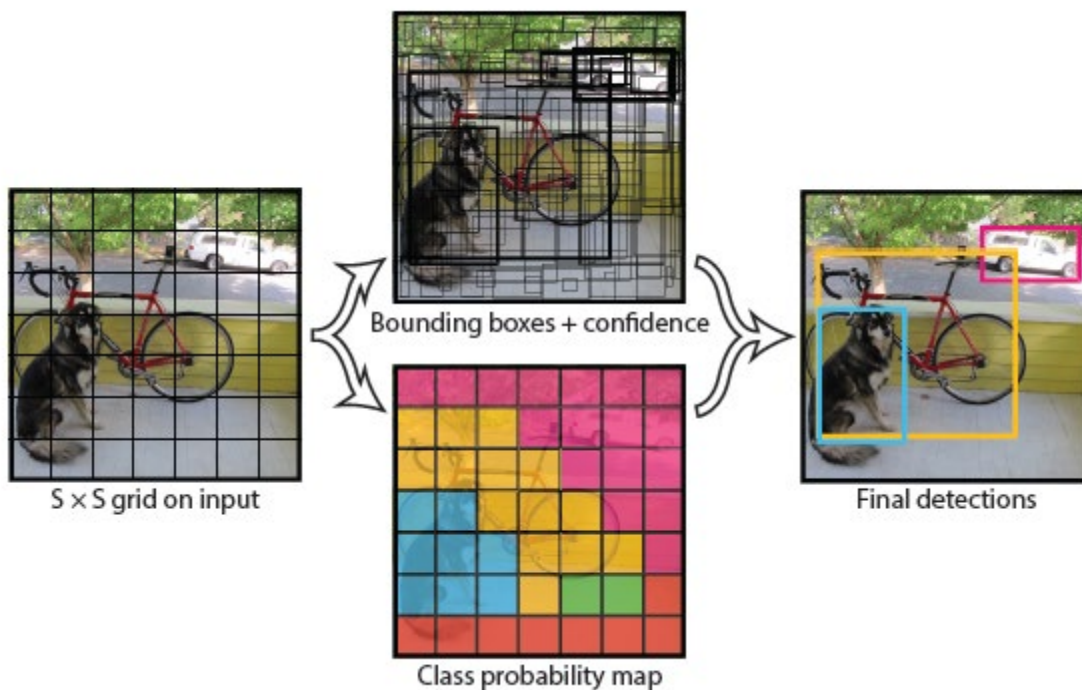


Figure 12: YOLO v3 Detection and Classification.

In our research, the types of vehicles that we need to consider only include cars, trucks, buses, vans, and motorcycles. Therefore, the dimension of the output tensor we get in actual training and detection is actually 30.

3.2.2 Transfer Learning Mechanism

Based on YOLO v3, to train our own CNN for vehicle detection, the transfer learning was achieved using a data set from a drone video detection contest named as “VisDrone 2018” [40]. The training data set contains 6287 labeled pictures (total 1.44 GB) with several cars in each of them. Figure 13 shows a selection of images in the training dataset. Figure 14 shows the statistics of object per image in this dataset, while Figure 15 gives the number of objects in each category and the occlusion degrees of objects for each category. After setting the parameters, the model was trained for about 30 hours for 18000 epochs and the final average training loss is 7.7.

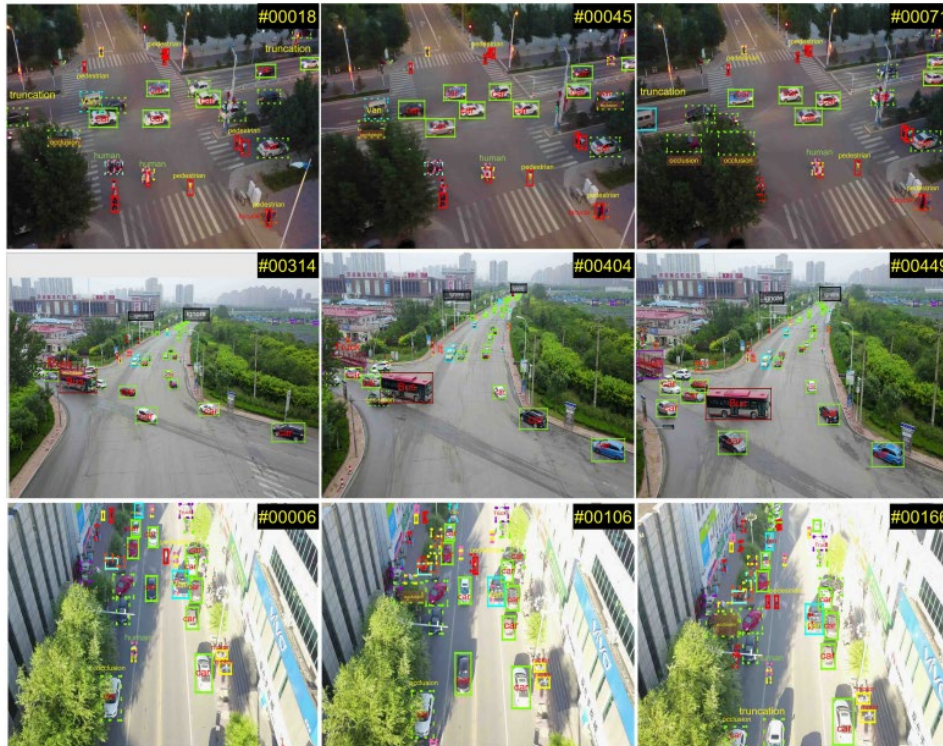


Figure 13: Examples of Training Dataset.

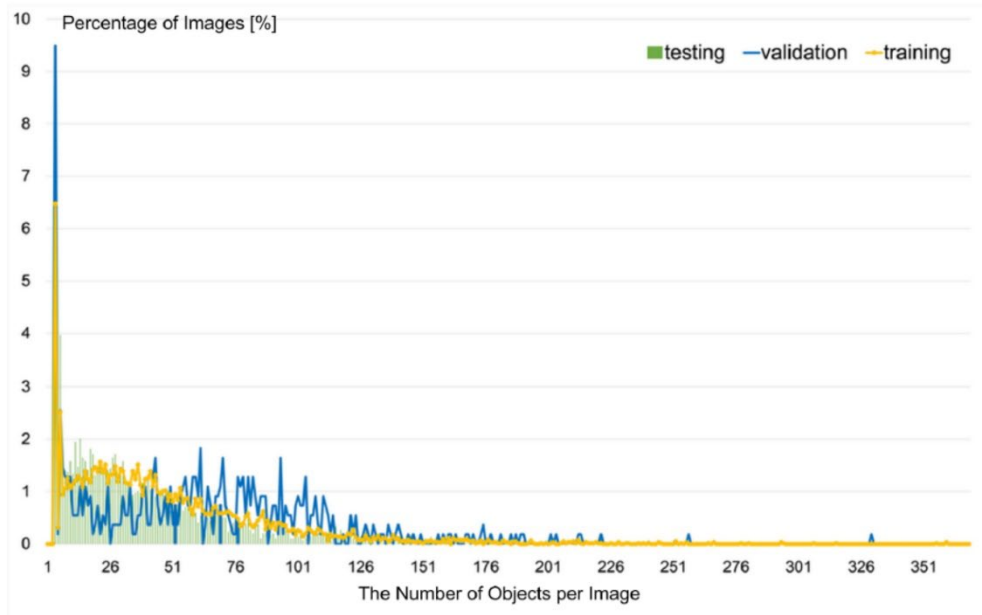


Figure 14: The Number of Objects Per Image of Dataset.

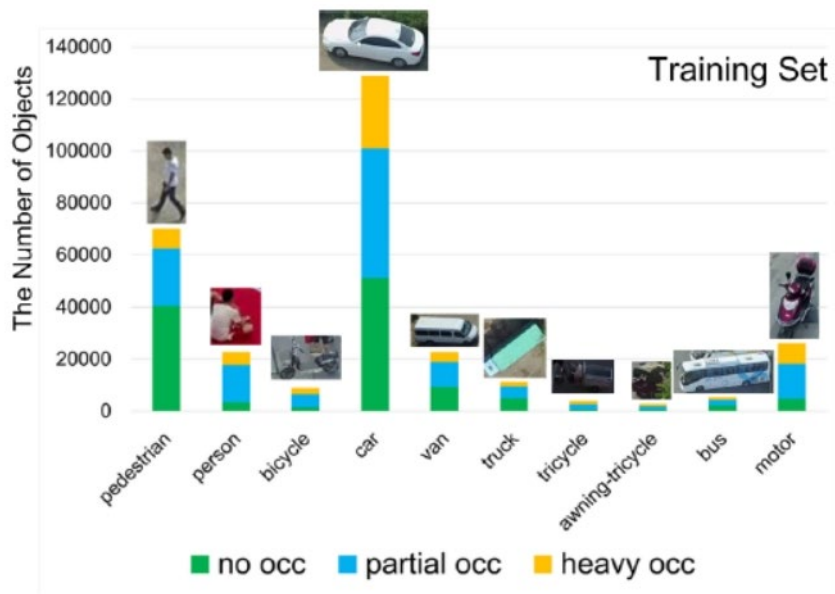


Figure 15: The Number of Objects in Each Category and The Occlusion Degree of Objects.

The training process is completed on Google Collaboratory. The structure of our training platform is shown in Figure 16. By uploading training data to Google Driver and synchronizing the folder of the laptop with Google Driver, the training results can be synchronized to the laptop in time. The weight parameters of each training 1000 epochs neural network will be saved once.



Figure 16: Device Synchronization Process.

After the training process, the network was used to detect vehicles in the video taken by a drone and saved their bounding box coordinates and centers into .txt files. Figure 17 shows one frame of the output video and the boxes that mark vehicle locations in the frame. In the test drone videos, the trained network can achieve an average accuracy of 94%.

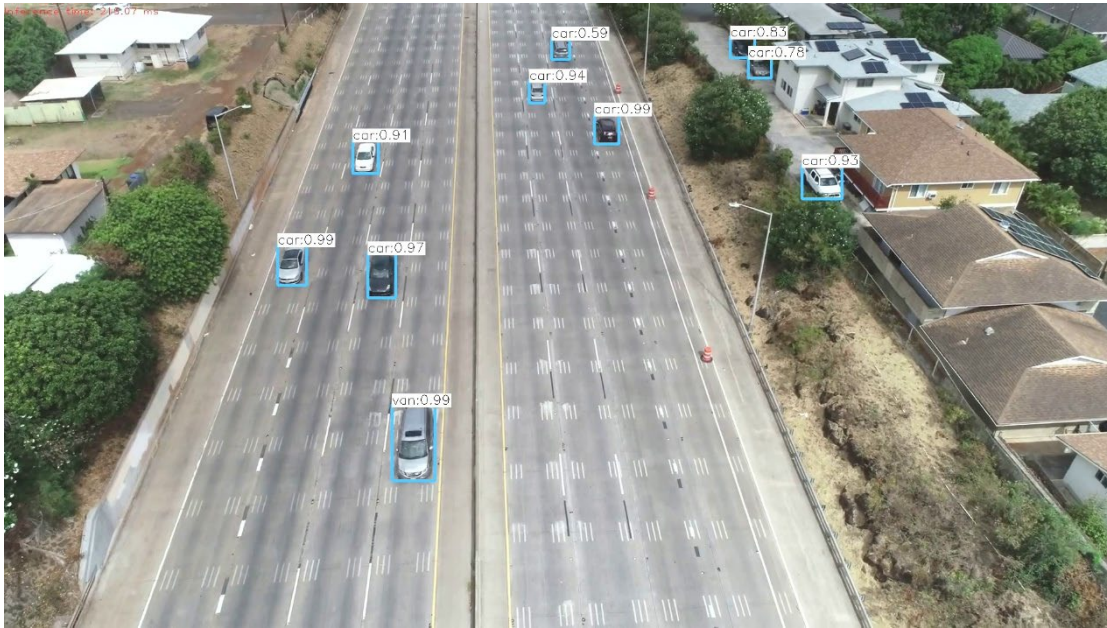


Figure 17: A screenshot of the output video. The markers show positions of the detected vehicles, the predicted vehicle type and the corresponding confidence level.

3.3 Image Sequence Based Self-Calibration Design and Vehicle Tracking

To calculate the dynamic parameters of traffic system, the model must acquire the boundaries of the road to distinguish the vehicles in the road from the vehicles in the background of the video. We propose a self-calibration road boundary extraction method based on image sequence. After completing the rough road boundary extraction, we need to filter out the vehicles on the road and further complete their tracking to extract the trajectory data of each vehicle for subsequent calculations. Section 3.3.1 illustrates the steps to extract precise boundaries from each frame of a video. It is worth noting that the road boundary extraction was actually carried out twice, the first time was to extract directly from the

picture using digital image processing technology, and the second time was to use the obtained vehicle position information as a reference for estimation.

3.3.1 Road Boundary Extraction

In general, this process is completed in two significant steps. The first step is based on digital image processing technology, using OpenCV to complete the preprocessing of the image and extract all the boundary lines. Under ideal conditions, we assume that five standard road boundary lines can be detected in each frame. Then the leftmost and rightmost boundary lines are the outer boundaries of the road. However, considering the interference of light intensity and shadows, and the road boundary line may become inconspicuous due to wear and tear, the boundary line cannot be detected. If the drone is not flying at high speed, the background of the two consecutive frames will stay the same. We can use the result of the previous frame as a substitute. Still, due to the interference of shadows from trees and roadside buildings, the outer boundary obtained in this step cannot be guaranteed to be accurate and may affect our subsequent calculations. Therefore, we need to get more accurate inner boundary lines.

Taking the above factors into account, the second step is to use the outer boundary lines to complete the preliminary filtering process of the vehicles detected by the neural network. In the subsequent process, we will decide the locations of boundaries according to the positions of the leftmost and rightmost vehicles on the road. Similarly, the relationship between two consecutive frames is also applied here. After completing the above two steps, we can get precise road inner boundary lines. The detailed process and principle of the algorithm are given below.

- (1) Read and turn a frame into gray scale picture, then adjust the brightness of the picture to lighten the dark part of the road boundaries. As Figure 18 shows, the brightness of the processed frame is obviously improved and the details of the shadow area are easier to identify.



Figure 18: Left- original frame. Right- adjusted frame.

- (2) Perform adaptive threshold segmentation on the frame after Gaussian noise reduction and morphological operations on the segmented picture.

Different from the global methods that use all pixel values to determine the segmentation threshold, the adaptive threshold segmentation method divides the screen into small neighborhoods for

processing. In Figure 19, for each neighborhood (a 19×19 size for the neighborhood is used in this study), the average value of the neighborhood's pixel values is calculated using the mean method or the Gaussian mean method as the threshold. All pixels greater than this threshold are set to 255, and those lower than this threshold are set to 0.

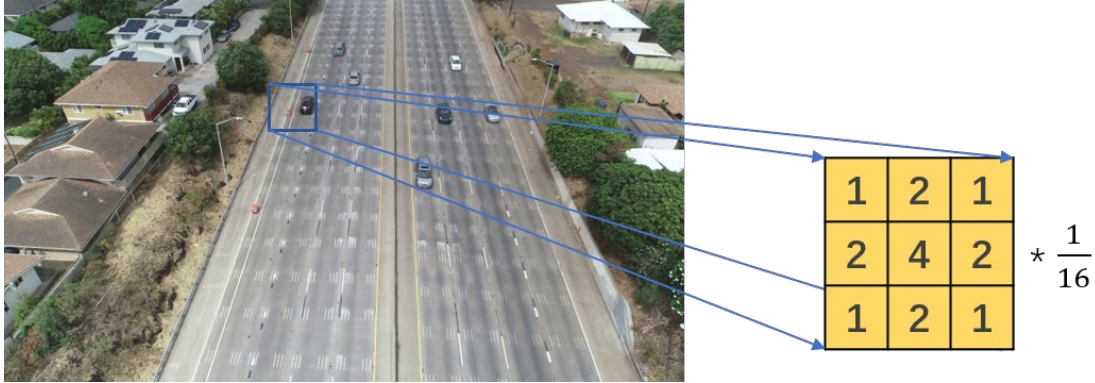


Figure 19: A simple example of Gaussian mean method in a neighborhood.

After completing the segmentation operation, we need to use the closing operation of the morphological operation to complete the connection of the discontinuous lines in the frame to facilitate the subsequent straight lines extraction. The closing operation needs to perform a dilation operation on the frame first and then a corrosion operation to smooth the boundary of the object and connect the adjacent objects. Here we use a cross structure like Figure 20 shows to perform all the operations. A processed frame is given in Figure 21 to demonstrate the processed result. The processed frames are binarized to make it easier to extract the lines.

Dilation: If there is an overlapping area with structure A in the process of moving structure B, record the position. The set of all positions that overlap with structure A when moving structure B is the result of structure A's expansion under structure B.

$$A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \emptyset\} \quad (2)$$

Erosion: If the intersection of structure B and structure A completely belongs to the area of structure A, save the location point, and all points that meet the conditions constitute the result of structure A being corroded by structure B.

$$A \ominus B = \{z \mid B_z \subseteq A\} \quad (3)$$

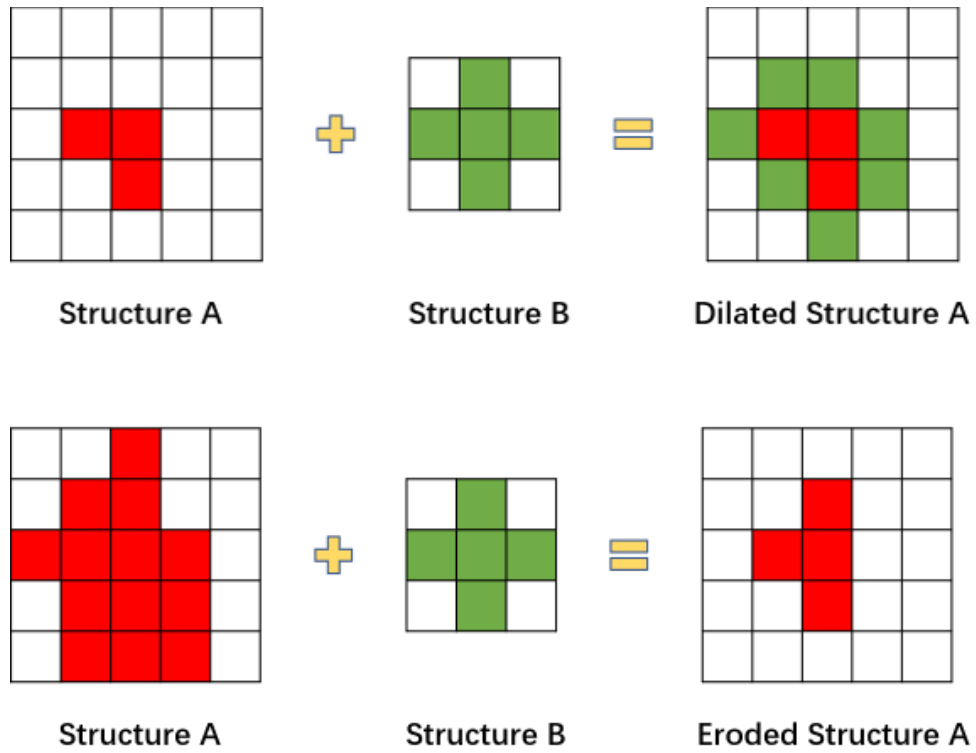


Figure 20: Example of Dilation and Erosion operations.

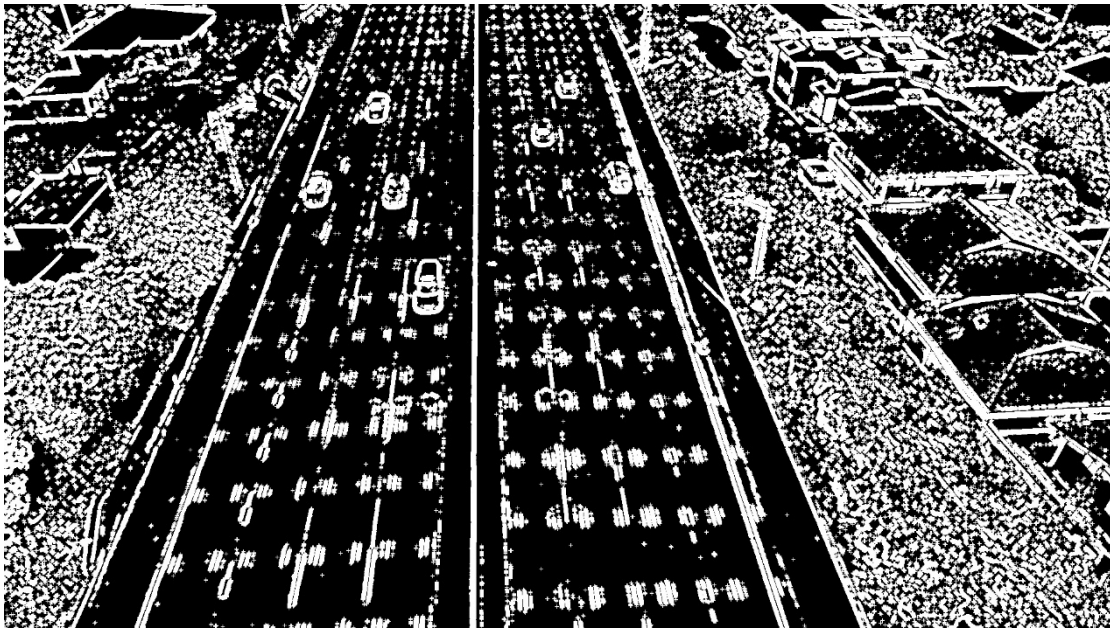


Figure 21: Result of adaptive threshold segmentation and morphological operations

(3) Use Progressive Probabilistic Hough Transform (PPHT) [41] to detect straight lines in the video frame.

The PPHT has been commonly accepted as one of the best line detection methods based on Hough

transform theory. In polar coordinates, a straight line can be expressed as

$$\rho = x \cos(\theta) + y \sin(\theta). \quad (4)$$

In this way, even a straight line perpendicular to the coordinate axis can be represented by a pair of parameters (ρ, θ) . The steps show how to get straight lines with a length larger than a preset minimum value:

- a) Randomly select a new point to update the accumulator array, with contributions to all available bins (bin stands for a pair of (ρ, θ)). Then remove the selected pixel from the input image.
- b) Check if the highest peak that was modified by the new pixel in the updated accumulator is greater than a pre-defined threshold. If not then go to Step a).
- c) Find the longest segment from all lines with the parameter (ρ, θ) which was specified by the peak in Step b). The selected line can be expressed using the beginning point A and ending point B.
- d) Remove all the points of the longest line from the input image.
- e) "Unvote" from the accumulator all the pixels from the line that have previously voted. Then these points do not attend any other voting process.
- f) If the selected segment is longer than a pre-defined minimum length, then the point pair of the beginning and ending points will be recorded as one of the output results.
- g) Go to Step a).

Finally, we can have an output list containing all the segments that longer than the min length value. Figure 22 gives an example of the extracted line segments. The list is saved as:

$LineList = \{(x_1, y_1), (x_2, y_2)\}, \dots, \{(x_{n-1}, y_{n-1}), (x_n, y_n)\}$,
the first point in each pair is the beginning point, and we assume there are n lines detected.

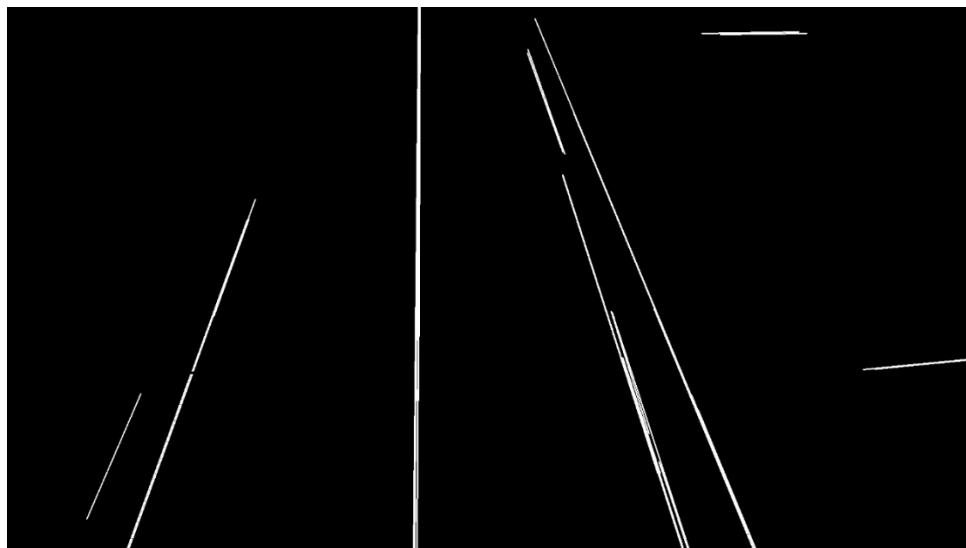


Figure 22: Result of Hough transform.

- (4) A mask designed based on prior knowledge was used to filter out the detected line clusters on the road. The filtered lines can be saved as:

$$LineList = \{(x_1, y_1), (x_2, y_2)\}, \dots, \{(x_{m-1}, y_{m-1}), (x_m, y_m)\}.$$

- (5) Use the K-means clustering method [42] to filter out the center points of each cluster, and use the average values of the slopes of the clustered lines as the actual slopes.

Now the clustered lines will be saved as:

$$LineList = \{(x_1, y_1, k_1), \dots, (x_s, y_s, k_s)\}, (x_i, y_i) \text{ stands for the beginning point and } k_i \text{ is the slope of corresponding line.}$$

- (6) Estimate the boundaries of the next frame considering that the boundaries of two consecutive frames will not change drastically, and compare with the detected boundaries to determine the actual boundaries of the current frame. The final estimated boundaries are shown in Figure 23. Different colors indicate boundaries at different positions on the road plane.

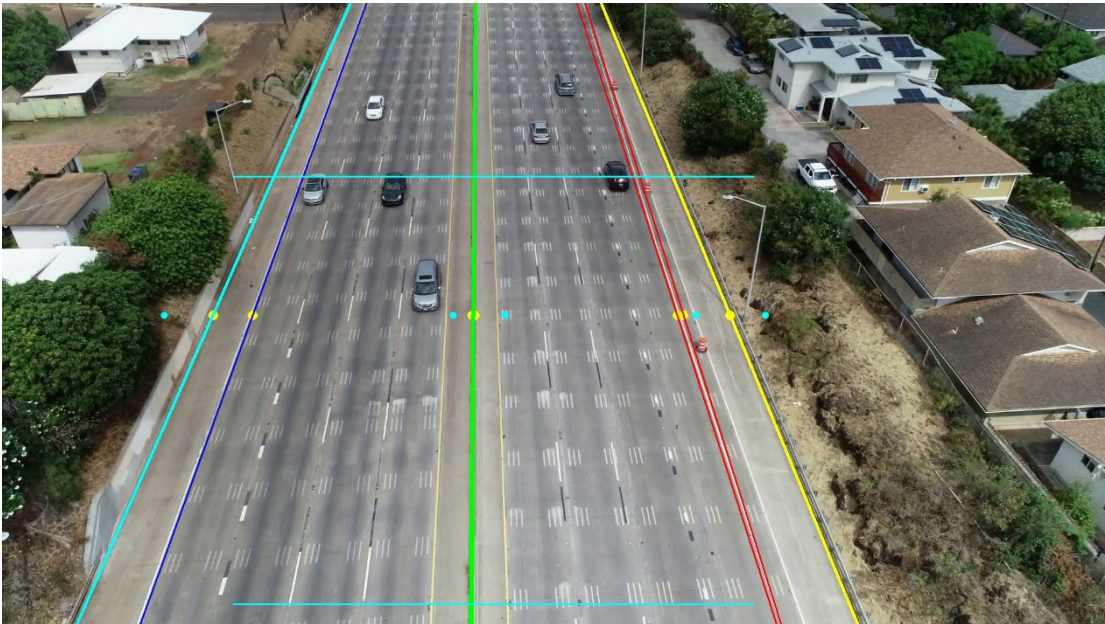


Figure 23: Detected boundaries. Straight lines of different colors indicate that edges are at different positions on the road plane. Yellow points are the result of K-means clustering.

- (7) Save all the detected road boundaries after K-means clustering (including road center lines) for later steps to determine more accurate locations of the road boundaries.

3.3.2 OpenCV-Based Vehicle Tracking

Based on the detected vehicle bounding box files, the vehicle trajectories are extracted using OpenCV

CSRT tracker [43]. The tracking process is performed with the following five steps:

Step 1: Tracker Initialization

For the first frame Fr0 of the video, read the corresponding bounding box file “frame0.txt” and initialize the tracker list with bounding boxes.

The bounding boxes are saved in this form:

$\{x, y, w, h, cx, cy\}$, (x, y) is the top left corner coordinate;
 w is the width of bounding box; h is the height;
 (cx, cy) denotes the center of bounding box.

Step 2: Update Trackers

Assume the current frame is Fri, update the trackers to get predicted bounding boxes using the current frame.

Step 3: Match Prediction and Ground Truth

Compare the predicted boxes with the detected boxes using CNN to find the boxes corresponding to the vehicles in the scene. If a predicted box is not matched, maybe it is because that the NN did not detect the corresponding vehicle, the tracker will also be saved.

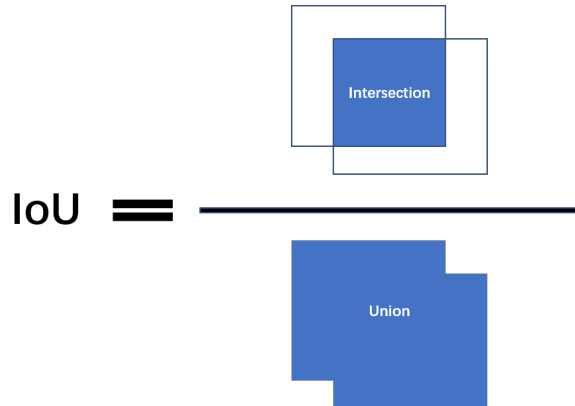


Figure 24: Intersection over Union (IoU).

In order to find the prediction box corresponding to the actual vehicle position, we use the Intersection over Union (IoU) [44] indicator for measurement as shown in Figure 24. The predicted boxes are compared with the vehicle positions of the frame one by one, and their IoUs are calculated at the same time. If the preset minimum threshold is reached, then the corresponding bounding box is considered to be found. In the UAV's perspective, there is generally no mutual occlusion between vehicles, which can simplify the judgment process. The generated trajectories would be combined with the NN detection result to improve the detection accuracy as shown in Figure 25.

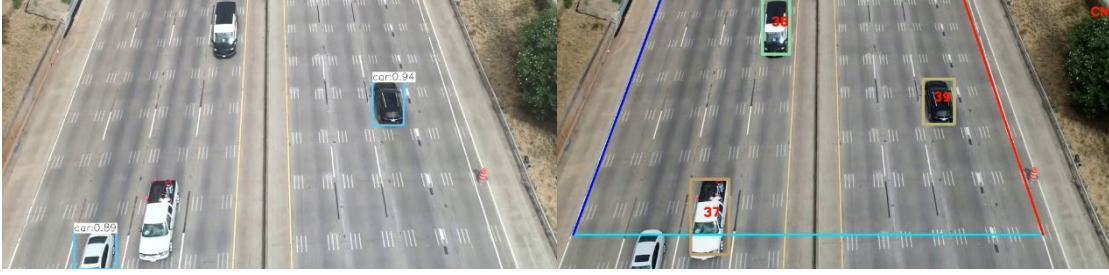


Figure 25: Left- detection result of NN. Right- NN detection combined with object tracking. The cars marked as 37 and 38 are detected using our proposed method. At the same time, the white car in the lower left corner is no longer marked because it exceeds the ROI range.

Step 4: Test Boundary Conditions

If the boxes reach the boundary conditions, the corresponding tracker is set to an unusable state and would not be updated in next frames.

Assuming the center of a predicted bounding box is (x_p, y_p) , now that we have ensured that all the vehicles in the picture are in the ROI, we only need to consider the case where the ordinate exceeds the given range, that is:

$$\begin{aligned} y_p < THR_{ul} \text{ or } y_p > THR_{ll}, \text{ if } x_p < MID; \\ y_p < THR_{ur} \text{ or } y_p > THR_{lr}, \text{ if } x_p > MID \end{aligned}$$

Where THR_{ul}, THR_{ll} denotes the upper left and lower left thresholds of the left part of ROI respectively; THR_{ur}, THR_{lr} denotes the upper right and lower right thresholds of the right part of ROI respectively; MID denotes the center line's x coordinate. The preset thresholds on the left and right sides may be different. This is because the camera angle of the drone may affect the heading of the vehicle during the flight, and thus affect the relative position of the center of the vehicle.

Step 5: Determine the Final Positions of Road Boundaries

According to the detected road boundaries and the coordinates of the left-most and right-most vehicle center points in the current frame, the road boundaries that actually need to be drawn in the current screen are determined.

Assume that in the current frame, the detected vehicles are arranged in order from left to right. The center coordinate of the leftmost vehicle is (x_{left}, y_{left}) , and the center coordinate of the rightmost vehicle is (x_{right}, y_{right}) . Now that we have saved all the detected borders, if the borders on the left and right sides are all present, then by comparing with the leftmost and rightmost vehicle center positions, we can get the candidates for the precise borders on the left and right of the current frame.

In the next step, we need to compare the candidates with the precise boundaries of the previous frame. The leftmost and rightmost boundaries of the previous frame are

$$(x_{left_{Fr_{i-1}}}, y_{left_{Fr_{i-1}}}, k_{left_{Fr_{i-1}}}, b_{left_{Fr_{i-1}}}) \text{ for the left side and}$$

$(x_{right_{Fr_{i-1}}}, y_{right_{Fr_{i-1}}}, k_{right_{Fr_{i-1}}}, b_{right_{Fr_{i-1}}})$ for the right side.

If the result of this frame is closer to the central area, then use the current candidates as the accurate boundary. At last, the leftmost and rightmost boundaries of the road obtained in the current frame are $(x_{left_{Fr_i}}, y_{left_{Fr_i}}, k_{left_{Fr_i}}, b_{left_{Fr_i}})$ for the left side and $(x_{right_{Fr_i}}, y_{right_{Fr_i}}, k_{right_{Fr_i}}, b_{right_{Fr_i}})$ for the right side.

If there is a left (right) side boundary that is not detected in the previous boundary extraction step, then we use the slope of the left (right) side boundary of the previous frame and the center coordinates of the vehicle in the outermost lane on that side to estimate the boundary.

The missing leftmost or rightmost boundaries of the road obtained in the current frame would be $(x_{left} + offset_x, y_{left}, k_{left_{Fr_{i-1}}}, b_{left_{Fr_{i-1}}})$ for left side or $(x_{right} + offset_x, y_{right_{Fr_i}}, k_{right_{Fr_{i-1}}}, b_{right_{Fr_{i-1}}})$ for the right side.

3.4 Flow Parameter Estimation

3.4.1 Perspective transformation

When the human eyes receive the view of the real world, objects in the vicinity will look larger than those in the distance. This is often referred to as the perspective phenomenon. Because the camera has optical perspective characteristics similar to the human eye, two parallel straight lines on both sides of the road in real life will converge into a point in the distance in the picture. Before proceeding further, the model needs to perform perspective transformation [45] as shown in Figure 26.

Assume the original point coordinate in the frame is (x, y) , the transformation matrix is M and the corresponding coordinate in the transformed picture is (x', y') , we have

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = M \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (5)$$

$$x' = \frac{u}{w} = \frac{a_{11}x + a_{12}y + a_{13}}{a_{31}x + a_{32}y + a_{33}} \quad (6)$$

$$y' = \frac{v}{w} = \frac{a_{21}x + a_{22}y + a_{23}}{a_{31}x + a_{32}y + a_{33}} \quad (7)$$

Based on the boundaries extracted, the algorithm transforms the ROI in the original frame into a rectangular picture to estimate the distance and then the speeds of vehicles move between two consecutive frames.

Suppose the center coordinate of a car is (x_1, y_1) in the current frame, and (x'_1, y'_1) in the transformed coordinate system. In the next frame, its center coordinate becomes (x_2, y_2) , and the transformed

coordinate becomes (x'_2, y'_2) . Since we assume that the picture will not change drastically between two consecutive frames, using the same actual ROI area size as a measure, the displacement of the center can be considered as the moving distance of the vehicle in two consecutive frames.

$$D = \sqrt{(x'_2 - x'_1)^2 + (y'_2 - y'_1)^2} \quad (8)$$

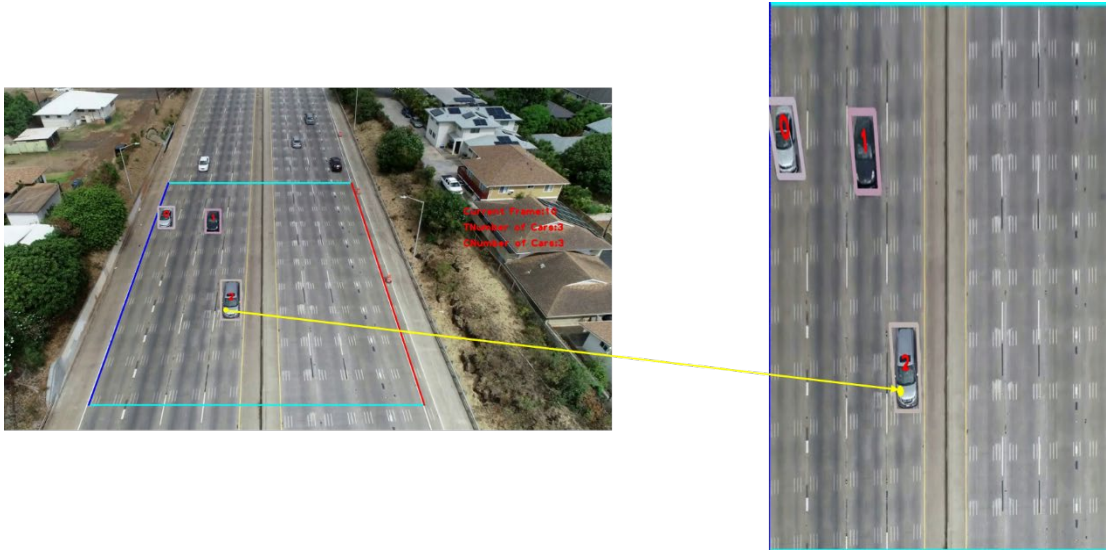


Figure 26: The Result of Perspective Transformation. The ROI in the left image was transformed into the picture on the right. The yellow point coordinates can be converted using matrix M mentioned above.

3.4.2 Parameter estimation

a) Vehicle number

After using neural network to complete the detection of vehicles, the model needs to calculate the number of vehicles running in the current frame and the total number of vehicles based on the detection results. Since the neural network may not recognize some vehicles during the process of detection, the following situations may occur while calculating the number of vehicles:

- 1) All vehicles in the screen are detected by the neural network, and the result that should be displayed is the number of vehicles detected;
- 2) Some vehicles in the picture are not detected by the neural network, but in the previous frame, some trackers were used to complete the tracking of these vehicles, and these trackers are still within the ROI range. At this time, the number of vehicles is the number of vehicles detected by the neural network plus the number of available trackers;
- 3) All the vehicles in the picture are not detected by the neural network. If the trackers used in the previous frame are still within the ROI range, the number of vehicles in the picture is the number of available trackers.

When calculating the total number of detected vehicles, after obtaining the number of vehicles in the current frame, the model only needs to count and add the number of new vehicles that appear in each frame. However, it should be noted that the neural network may detect a car in the previous frame, but may not detect the same car in the subsequent frames. Under such circumstances, this car should not be treated as a new car. In fact, in order to deal with this situation, the model looks for the matching results of the vehicles detected in the current frame from the prediction results of the trackers in the previous frame. If the corresponding tracker is found, the model will no longer add a new tracker for this car. Instead, the model uses its current position to update the corresponding tracker.

b) Vehicle speed

After the detection and tracking of the vehicle is completed, the speed of the vehicles needs to be calculated and marked on the corresponding vehicle in the output video. When the neural network marks the position of the vehicle, the bounding box is marked with a rectangle that can include the outline of the vehicle (possibly including its shadow). However, in the tracking process, the tracker estimates the corresponding position of the next frame according to the size of the frame of the vehicle given in the previous frame, so the position of the center point of the vehicle may have a large drift. Figure 27 gives an example of center drift caused by the inaccurate detection result.

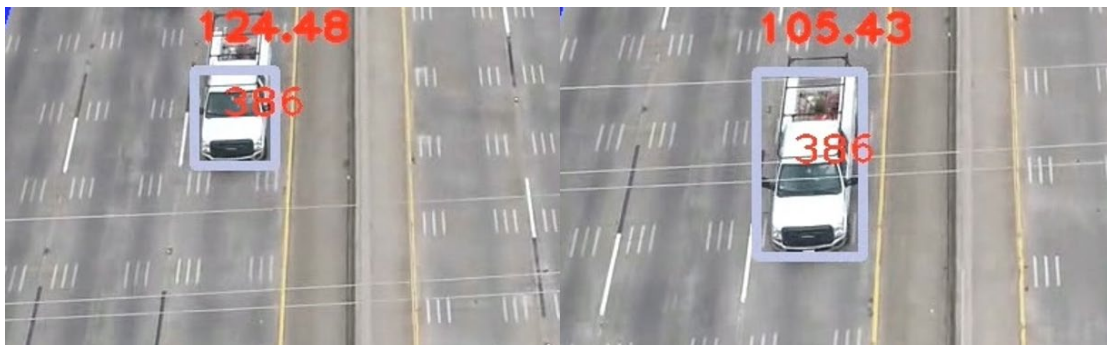


Figure 27: Center Drift. The detection result of the neural network may not be accurate enough, which may cause the center to drift when tracking.

To reduce the error caused by the drift of the vehicle's center point, the moving distance of 5 consecutive frames is averaged, and the speed calculated using this value is regarded as the moving speed of the vehicle in these five frames. If the total number of frames that the vehicle appears is not an integer multiple of 5, then the moving distance of the last few structures is also averaged to estimate the speed. In addition, after the vehicle's speed is estimated, considering that the drone is also flying in the direction of the road, vehicles traveling in the same direction as the drone needs to subtract the drone's speed. Correspondingly, the drone's flight speed needs to be added to the vehicle driving in the opposite direction to the drone. In this way, the true speed of the vehicle can be estimated.

It is worth noting that in the first frame, when a vehicle appears, the speed is 0 by default (because the position of the previous frame cannot be obtained). However, to avoid affecting the display result in the video, the speed of the second frame is used as the speed of the first frame considering that the speed of the vehicle generally does not change drastically when running at high speed.

c) Vehicle Average Speed

The average speed of vehicles indicates the degree of congestion on the road. When the average speed of vehicles is higher than a certain value, it means that there is almost no congestion on the road and the traffic flow is almost free flow. When the road is heavily congested, the average speed of the vehicle becomes very close to zero.

Assuming that there are N vehicles on the road at time T , the average speed of each vehicle can be calculated using the detected vehicle trajectory data, denoted as $V_i, i = 1, 2, \dots, N$. Then the average speed of vehicles on the road at this time can be simply estimated as the average of their speeds:

$$V_{avg.} = \frac{1}{N} \sum_{i=1}^N V_i, \quad i = 1, 2, \dots, N \quad (9)$$

d) Vehicle Density

Generally speaking, we can calculate the density of vehicles per unit distance by dividing the total number of vehicles detected in a certain period of time by the total distance (usually 1 km is used as the unit distance). On this basis, we can further divide this result by the number of lanes to get the density of vehicles on each lane. In the case of this report, we only use the former as the parameter to measure traffic flow.

Suppose that the starting time is $T_0 = 0$, and it becomes T_i after a period of time. During this time, the total number of detected vehicles is N , the number of lanes is L , and the average flying speed of the drone is V_d , and the flying distance is $S_d = V_d \times T_i$. Considering the detection distance S_{ROI} of the drone's ROI, we can calculate the average density of vehicles at time T_i :

$$Density = \frac{N/L}{S_d + S_{ROI}} \quad (10)$$

CHAPTER 4. EXPERIMENTAL TESTS

4.1 Testing Hardware and Environment

The hardware used in this study is shown in Figure 28. The UAV used for recording videos of the freeway is DJI Phantom 4 Pro. A remote controller controls the drone. An iPad mini installed with flight control software is used as the flight control panel and the display screen. The drone is equipped with a GPS device and can be automatically stabilized against wind disturbances. According to the user manual, the drone could reach a height of 500 m and a max control range of 7000 m. The flight process can be controlled manually or autonomously with a GPS system, which makes it possible to follow preprogrammed waypoints as shown in Figure 29. The camera is fixed on the gimbal at the bottom of the drone and can achieve a shooting angle from -90° to $+30^{\circ}$. In the recording mode, it can record 4K (3840×2160 30fps) videos and save them as a .mov to a MicroSD card. Some screenshots are shown in the following section, which was recorded by the drone on the freeway following the preprogrammed waypoints.



Figure 28: Configuration of Hardware.

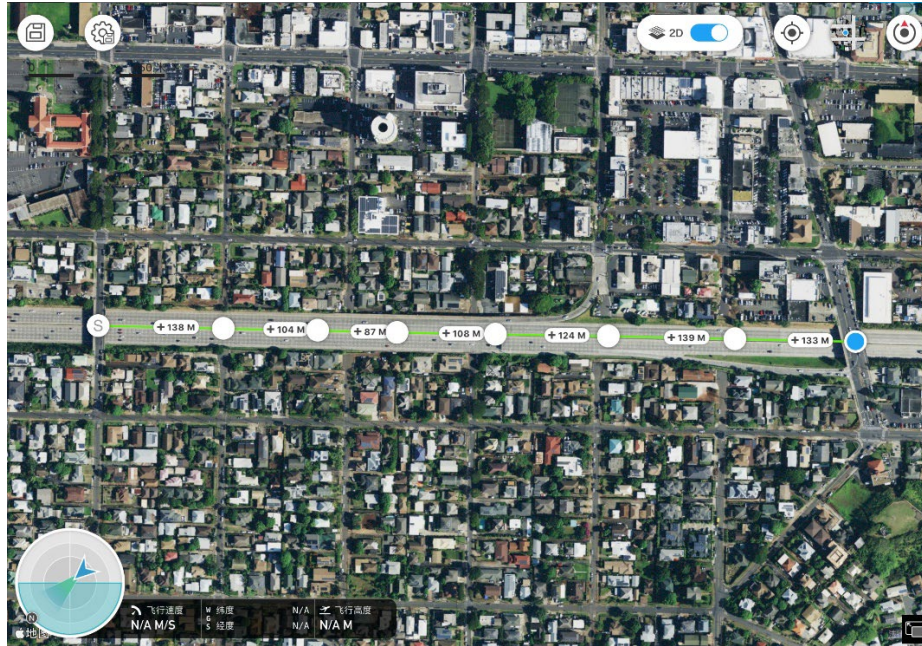


Figure 29: Preset Waypoint Information.




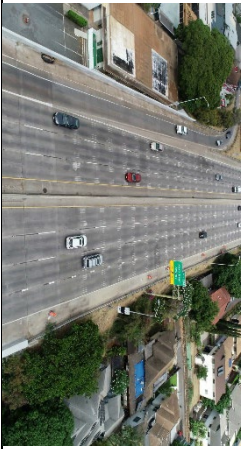
As for the code running environment, the neural network was constructed under Python 3.7 in Ubuntu 18.04 system frame. The vehicle tracking module uses OpenCV 3.4.2 to read and process video and images. The final codes run on a laptop computer that uses an I7 6700HQ model CPU and 16 GB RAM.

4.2 Testing Scenarios and Data Collection

4.2.1 Testing Scenarios

To test the performance of our proposed model in different scenarios, we used a drone to obtain videos of the same road at other times. When recording videos by drone, one must consider that the UAV is small in size and cannot resist strong wind. Therefore, we chose days in the field when the winds are calm, the weather is sunny or cloudy, and different lighting conditions are also considered. The road section selected in this experiment is on the H1 freeway in Honolulu, Hawaii, with a total length of about 880 meters on approximately a straight alignment. Table 1 provides a detailed description.

Table 1: Description of Test Videos

Number	Time	Weather	Light Condition	Screenshot	Road Type	Length
1	Morning (8:00-9:00)	Clear	High ambient lighting (Objects with shadows)		Freeway	00:27
2	Noon (12:00-13:00)	Clear	High ambient lighting (Objects with shadows)		Freeway	05:27
3	Afternoon (17:00-18:00)	Windy	Moderate ambient lighting (Objects with shadows)		Freeway	06:00
4	Evening (18:00-18:30)	Windy	Low ambient lighting (No obvious shadows)		Freeway	05:40

Note: The videos used in the experiment were not shot on the same day, but we tried our best to choose suitable weather and lighting conditions to test the performance of the model in different scenarios.

4.2.2 Data Collection

The hardware devices used in this experiment have been shown and explained in detail in the previous chapters. To avoid drone flight accidents that may be caused by manual operation, try to keep the drone's viewing angle as stable as possible. With the help of specialized software, we have set up a series of fixed waypoints through many tests and adjustments using the GPS function of the drone. After the drone takes off, turn on the automatic flight on the software, and the software will wirelessly transmit the waypoint information to the drone. Then the drone will automatically follow the preset route and fly at a constant speed. In this process, the camera of the UAV is used to complete the collection of road video information, and the videos are stored in the UAV's micro SD card in the 4K+30fps .mov file format.

4.3 Experimental Results and Analysis

4.3.1 Model Calibration and Metrics of Evaluation

To verify the effectiveness of our proposed model and calibrate the parameters, we first used a short video as verification data. This video is also from the H1 freeway and is about 27 seconds long. The detailed description of this short clip is listed above.

Before we start the test, we first need to give definition of metrics to measure performance. Considering that the model we proposed is to detect the number of vehicles on the road, extract vehicle trajectory data then estimate vehicle speed, traffic flow density and average speed on this basis, we propose the following measurement standards for these data:

1) Metrics of Evaluation for Vehicle Detection

Table 2: Confusion Matrix

Total Population		Predicted Condition	
		Positive	Negative
True Condition	True	True Positive (<i>TP</i>)	True Negative (<i>TN</i>)
	False	False Positive (<i>FP</i>)	False Negative (<i>FN</i>)

Start from the concept of binary classifier, **Precision**, which is the fraction of relevant instances among the retrieved instances, and **Recall**, which is the fraction of the total amount of relevant instances that have been retrieved, are commonly used in detectors performance evaluation [46]. They are defined as in the following equations:

$$Precision = \frac{TP}{TP+FP} \quad (11)$$

$$Recall = \frac{TP}{TP+FN} \quad (12)$$

In an ideal state, both the accuracy rate and the recall rate should be high, which shows that the model can accurately identify the correct result while ensuring a low error recognition rate. Based on these two parameters, the **F-measure** is the trade-off between Recall and Precision, which have equal importance in the equation below:

$$F = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (13)$$

2) Metrics of Evaluation for Traffic Flow Density

According to the level of service (LOS) criteria table from [47], flow conditions are considered "free" when less than 12 vehicles per mile per lane are on a road. "Stable" is sometimes described as 12–30 vehicles per mile per lane. As the density reaches the maximum mass flow rate (or flux) and exceeds the optimum density (above 30 vehicles per mile per lane), traffic flow becomes unstable, and even a minor incident can result in persistent stop-and-go driving conditions. A "breakdown" condition occurs when traffic becomes unstable and exceeds 67 vehicles per mile per lane.

Table 3: Level of Service for basic freeway sections for 70 km/h design speed [47]

LOS	Flow Condition	Service Volume	Speed	Density
		(veh/h/lane)	(miles/h)	(veh/mile per lane)
A	Free	700	>60	<12
B	Stable	1100	>57	<20
C	Stable	1550	>54	<30
D	High Density	1850	>46	<40
E	Near Capacity	2000	>30	<67
F	Breakdown	Unstable	<30	>67

3) Metrics of Evaluation for Average Speed

Since the freeway section used for the test contains forks and has different speed limits in different sections, we partially refer to the LOS standard table given above and the actual speed limit standards to investigate the average speed of the traffic flow. On this basis, a rough judgment of road congestion is given.

In the test phase, after processing the 27-second video (marked as video No. 1), we got the following test results:

Table 4: Preliminary Test Results

Ground Truth	TP Sample Number	FP Sample Number	FN Sample Number	P	R	F-measure
90	88	2	0	97.78%	100%	98.88%

Since the videos were taken by us using a drone, there is no corresponding data set containing the number and speed of vehicles. Therefore, in the result, the actual total number of vehicles (i.e. ground truth), TP, FP, and FN data are all manually counted.

The speeds of the vehicles are saved in a .csv file. For the convenience of calculation, the speeds of the vehicles used in the calculation are in feet/s. By calculating their average value, we get an average vehicle speed of 14.66 miles/h (23.60 km/h) on the left lanes. Since the road in the test video is close to the fork, the speed limit is 25 miles/h. Considering this actual speed limit, the data we get is within a reasonable range. At the same time, referring to the standards given above, this section of the road is already in severe congestion. The following results will also give a consistent judgment.

As for the traffic density, through estimation, the total length of the 7-lane test section is about 160m (about 0.1 mile), and the total number of vehicles detected by the model is 88, so the traffic density on this road section during this time period 126 veh/mile per lane. According to the LOS standard listed above, this section of the road is actually in a state of congestion, and it has also been verified from the actual picture as shown in Figure 30 that vehicles on the left lane have already started to queue up.



Figure 30: Actual Traffic Conditions in Congestion.

The processed results of this test video prove the effectiveness of our model, and that the model can provide more accurate results.

4.3.2 Experimental Results

After testing and adjusting the parameters of the model, this part gives the processed results of the remaining three longer videos. Here is the result of vehicle detection:

1) Vehicle Detection

Table 5: Experimental Results of Vehicle Detection

Number	Ground Truth	TP Sample Number	FP Sample Number	FN Sample Number	P	R	F
2	516	499	7	10	98.62%	98.04%	98.33%
3	663	647	5	11	99.23%	98.33%	98.78%
4	513	495	2	16	99.60%	96.87%	98.22%

2) Vehicle Average Speed

Table 6: Experimental Results of Vehicle Average Speed

Number	Estimated Average Speed (Left)	Estimated Average Speed (Right)	Flow Condition (Left)	Flow Condition (Right)	Actual Speed Limit
2	53.60 miles/h	41.07 miles/h	High Density	Near Capacity	50 miles/h
3	55.93 miles/h	42.68 miles/h	Stable	High Density	50 miles/h
4	58.64 miles/h	44.49 miles/h	Stable	High Density	50 miles/h

3) Vehicle Density

Table 7: Experimental Results of Vehicle Density

Number	Detected Number of Vehicles	Total Length of Road Section	Number of Lanes	Estimated Vehicle Density (veh/mile per lane)
2	512	900 m	8	114
3	661	1000 m	8	133
4	506	900 m	8	113

4.3.3 Analysis of Results

In the previous parts of this chapter, we have listed the evaluation criteria and experimental results, respectively. Our model performs well in vehicle detection and speed estimation from the experimental results. Our proposed method can achieve a vehicle detection precision rate of over 98% in different scenarios. At the same time, the F value is also maintained above 98%. It shows that our model has an excellent performance in accuracy and recall.

The estimated average vehicle speeds can also provide an intuitive judgment of the degree of road congestion. The average rates of vehicles on the left and right lanes obtained from the test videos are shown in Table 6. The road flow conditions corresponding to them are also given. The results show that the average speeds of the vehicles in the lanes on the right are lower than those on the left. The reason for this result is that on this section of the road, the lanes on the right are uphill, so the average speeds of the vehicles are lower. Therefore, the estimated flow conditions of the right lanes are “near capacity” or “high capacity”, while the flow conditions of the left lanes are “stable” or “high capacity”.

However, the estimation of traffic flow density is quite different from the actual result. Referring to the LOS standard table, we quoted that the road in this scenario should be nearly saturated when the

vehicle density exceeds 67 veh/mile per lane. Vehicles are very likely to fall into long-term congestion on the road due to minor effects, such as emergency braking of a vehicle. Going back to the scene we tested, we can judge from the video we recorded and the estimated average speed that the traffic flow is obviously in a relatively stable state and not close to the maximum capacity. By analyzing the actual situation, the reason for this phenomenon is actually that the road section we used to record the video is not entirely closed. As Figure 31 shows, a fork connects the road, so the total number of vehicles counted by our model includes vehicles entering and leaving the freeway at the fork. In addition, refer to the results obtained using the test video labeled 1, which also exceeds the standard line of 67 veh/mile. Since that section of road is relatively short, it can be approximated as a closed section. Therefore, the estimated result reflects the actual condition of the road, that is, the road is indeed congested.

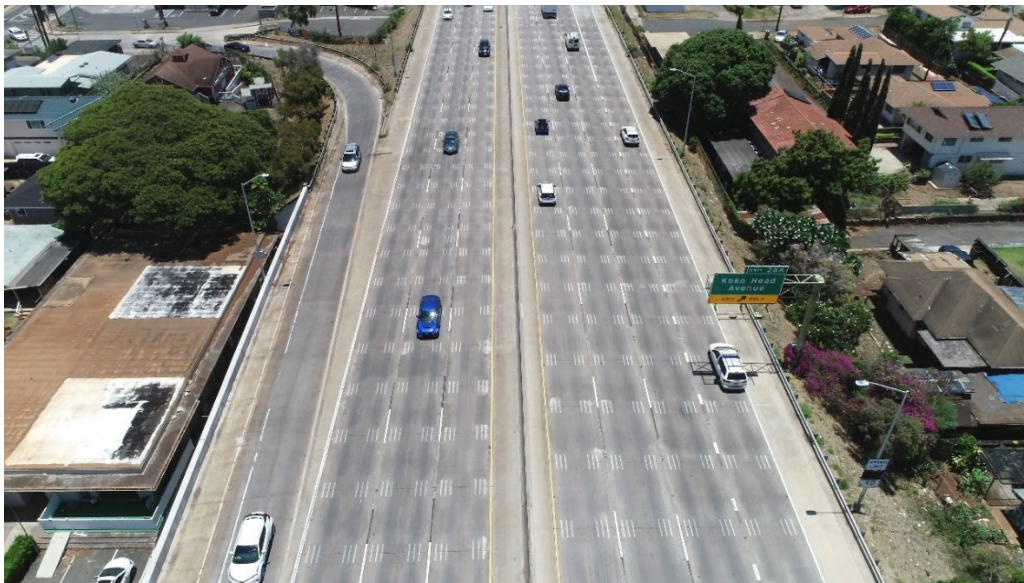


Figure 31: Road Fork on the Test Section.

Based on the reasons listed above, we can explain the result that traffic flow density is higher than the reasonable value in our test in this way: the test road section is not closed, so the estimated density value includes the traffic flow on the branch road, so the result is still reasonable. But this also reminds us that we need to be more rigorous in choosing test scenarios in subsequent tests and try to choose closed roads to avoid the impact of traffic on branch roads.

In summary, our model has achieved a vehicle detection rate of more than 98% under different weather, light, and traffic conditions and can accurately estimate two key macro traffic flow parameters: the average speed of the traffic flow and density. The results obtained by the model can provide powerful support for road congestion analysis. At the same time, we also save detailed detection results, such as vehicle trajectory data and the real-time speed of vehicles, in .csv files. After manual calibration, these data can be used to estimate the safety index of the traffic system and for the research of intelligent traffic management systems.

CHAPTER 5. CONCLUSIONS

This study provides a feasible and relatively accurate method for the dynamic parameter estimation of the transportation system. The proposed method integrates road boundary detection, vehicle detection, tracking, and vehicle speed estimation. It can further provide detailed road macro parameters based on obtaining the basic structure information of the road system. From the experimental results in the obtained freeway videos obtained by the UAV, our proposed method achieves about 98% vehicle detection accuracy on average. It provides highly reliable vehicle speed estimation results. On this basis, we can further calculate the vehicle density on the road and the vehicle's average speed. Thus, the vehicle congestion on the tested road can be evaluated, providing reference information for the intelligent traffic management system.

This approach is proved to be an effective method different from traditional road detection methods. First of all, using a drone to record the road system can achieve complete recording of an entire section of the road, considering a low budget level. The traditional fixed-angle road monitoring system needs to integrate the videos of different road segments to complete the vehicle detection and parameter estimation of the entire road. However, under the proposed model, a single drone can be used to record video of the closed road. And because of the high angle of view, the video taken for vehicle detection will not be affected by the mutual occlusion between vehicles. Second, the proposed model can obtain road boundaries and vehicle trajectories, providing a reliable navigation reference for smart vehicles when the GPS system cannot provide accurate road and surrounding vehicle information. In addition, due to this method, accurate vehicle speed and position information can be obtained. In future research, we can also use this as input data to complete the evaluation of the road safety factor and obtain accurate evaluation results through the statistics of possible vehicle collisions and acceleration and deceleration.

After completing the construction of the entire model, we also noticed some directions for future work in the actual experimental process. First, we need to obtain more training data to train the neural network to improve the detection accuracy of different types of vehicles. Second, since the tracker's center position of the vehicle output may have a significant drift in different frames, the accuracy of the target tracking algorithm needs to be further improved. Currently, in the vehicle detection stage, it takes about 220ms on average to complete the detection of one video frame. In the future, we will improve detection speed by using better hardware and algorithms. Finally, due to the current limitation of computing power, we cannot achieve real-time detection of vehicles and output videos with calculated parameters. Next, we will try to develop more efficient algorithms to reduce resource consumption.

References

- [1] Zhang, Tianya & J, Peter. (2019). A longitudinal scanline-based vehicle trajectory reconstruction method for high-angle traffic video. *Transportation Research Part C Emerging Technologies*. 103. 104-128. 10.1016/j.trc.2019.03.015.
- [2] Sun, Zehang & Bebis, George & Miller, Ronald. (2006). On-Road Vehicle Detection: A Review. *IEEE transactions on pattern analysis and machine intelligence*. 28. 694-711. 10.1109/TPAMI.2006.104.
- [3] Gao, Lei, Chao Li, Ting Fang, and Zhang Xiong, "Vehicle detection based on color and edge information", In *Proceedings of International Conference Image Analysis and Recognition*, Springer, pp. 142-150, 2008.
- [4] Matthews, N. D., P. E. An, D. Charnley, and C. J. Harris, "Vehicle detection and recognition in greyscale imagery", *Control Engineering Practice*, 4(4), pp. 473-479, 1996.
- [5] J. Lee, J. Wu, C. Hsieh and J. Chien, "Close range vehicle detection and tracking by vehicle lights," 2014 11th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Seoul, 2014, pp. 381-386.
- [6] Xia, Yingjie, Chunhui Wang, Xingmin Shi, and Luming Zhang, "Vehicles overtaking detection using RGB-D data", *Signal Processing*, 112, 98-109, 2015.
- [7] Ji, Wenyang, Lingjun Tang, Dedi Li, Wenming Yang, and Qingmin Liao, "Video-based construction vehicles detection and its application in intelligent monitoring system", *CAAI Transactions on Intelligence Technology*, 1(2), pp. 162-172, 2016.
- [8] A. Varghese and G. Sreelekha, "Background subtraction for vehicle detection," 2015 Global Conference on Communication Technologies (GCCT), Thuckalay, 2015, pp. 380-382, doi: 10.1109/GCCT.2015.7342688.
- [9] Horn, B. K. P., and B. G. Schunck, "Determining Optical Flow Artificial Intelligence", Vol. 17, pp. 185-203, 1981.
- [10] Stauffer, Chris & Grimson, W. (2007). Adaptive background mixture models for real-time tracking. *Proceedings of IEEE Conf. Computer Vision Patt. Recog*, vol. 2. 2.
- [11] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* 25 (pp.1097–1105).
- [12] Z. Zhao, P. Zheng, S. Xu and X. Wu, "Object Detection With Deep Learning: A Review," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212-3232, Nov. 2019, doi: 10.1109/TNNLS.2018.2876865.
- [13] Sultana, F., Sufian,A.,&Dutta, P. (2018).Advancements in image classification using convolutional neural network. In *2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)* (pp. 122–129), Nov 2018.
- [14] Rehman, Saad & Ajmal, Hina & Farooq, Umar & Ain, Qurrat & Riaz, Farhan & Hassan, Ali. (2018). Convolutional neural network based image segmentation: a review. 26. 10.1117/12.2304711.

- [15] Lee, J., Bang, J., Yang, S.I.: Object detection with sliding window in images including multiple similar objects. In: 2017 International Conference on Information and Communication Technology Convergence (ICTC), pp. 803–806 (2017).
- [16] Girshick, R.B., Donahue, J., Darrell, I., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In 2014 IEEE Conference on Computer Vision and Pattern Recognition (pp. 580–587).
- [17] Girshick, Ross. (2015). Fast r-cnn. 10.1109/ICCV.2015.169.
- [18] Redmon, Joseph & Divvala, Santosh & Girshick, Ross & Farhadi, Ali. (2015). You Only Look Once: Unified, Real-Time Object Detection.
- [19] Liu, Wei & Anguelov, Dragomir & Erhan, Dumitru & Szegedy, Christian & Reed, Scott & Fu, Cheng-Yang & Berg, Alexander. (2016). SSD: Single Shot MultiBox Detector. 9905. 21-37. 10.1007/978-3-319-46448-0_2.
- [20] N. Aloysius and M. Geetha, "A review on deep convolutional neural networks," 2017 International Conference on Communication and Signal Processing (ICCSP), Chennai, 2017, pp. 0588-0592, doi: 10.1109/ICCSP.2017.8286426.
- [21] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, arXiv preprint arXiv:1207.0580, 2012.
- [22] Yamashita, Rikiya & Nishio, Mizuho & Do, Richard & Togashi, Kaori. (2018). Convolutional neural networks: an overview and application in radiology. Insights into Imaging. 9. 10.1007/s13244-018-0639-9.
- [23] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, Chunfang Liu. (2018). A Survey on Deep Transfer Learning. In the 27th International Conference on Artificial Neural Networks (ICANN 2018).
- [24] Zhang, Li-Ye & Peng, Zhong-Ren. (2014). Road boundary estimation to improve vehicle detection and tracking in UAV video. Journal of Central South University of Technology. 21. 4732-4741. 10.1007/s11771-014-2483-5.
- [25] Kuo-Yu Chiu and Sheng-Fuu Lin, "Lane detection using color-based segmentation," IEEE Proceedings. Intelligent Vehicles Symposium, 2005., Las Vegas, NV, USA, 2005, pp. 706-711, doi: 10.1109/IVS.2005.1505186.
- [26] Sirmacek, Beril & Unsalan, Cem. (2010). Road Network Extraction Using Edge Detection and Spatial Voting. 3113-3116. 10.1109/ICPR.2010.762.
- [27] Parameswaran, Namboodiri & Achan, E. & Shree, Subha & Manjusha, R.. (2019). Road Detection by Boundary Extraction Technique and Hough Transform. 10.1007/978-3-030-00665-5_165.
- [28] Chinnathevar, Sujatha & Dharmar, Selvathi. (2016). FPGA implementation of road network extraction using morphological operator. Image Analysis & Stereology. 35. 10.5566/ias.1493.
- [29] Dai, Jiguang & Zhu, Tingting & Zhang, Yilei & Ma, Rongchen & Li, Wantong. (2019). Lane-Level Road Extraction from High-Resolution Optical Satellite Images. Remote Sensing. 11. 2672. 10.3390/rs11222672.

- [30] Oniga, Florin & Nedeveschi, Sergiu & Meinecke, Marc-Michael & To, Thanh Binh. (2007). Road Surface and Obstacle Detection Based on Elevation Maps from Dense Stereo. 859 - 865. 10.1109/ITSC.2007.4357734.
- [31] Sappa, Angel & Dornaika, Fadi & Ponsa, Daniel & Geronimo, David & López, Antonio. (2008). An Efficient Approach to Onboard Stereo Vision System Pose Estimation. IEEE Transactions on Intelligent Transportation Systems. 9. 476-490. 10.1109/TITS.2008.928237.
- [32] Hua, Shuai & Kapoor, Manika & Anastasiu, David. (2018). Vehicle Tracking and Speed Estimation from Traffic Videos. 153-1537. 10.1109/CVPRW.2018.00028.
- [33] Bedruz, Rhen Anjerome & Sybingco, Edwin & Bandala, Argel & Quiros, Ana Riza & Uy, Aaron & Dadios, Elmer. (2017). Real-time vehicle detection and tracking using a mean-shift based blob analysis and tracking approach. 1-5. 10.1109/HNICEM.2017.8269528.
- [34] Zhong, Kaiyang. (2018). Vehicle Detection and Tracking Based on GMM and Enhanced Camshift Algorithm. Journal of Electrical and Electronic Engineering. 6. 40. 10.11648/j.jee.20180602.11.
- [35] Sharma, Vijay & Mahapatra, Kamalakanta & Acharya, Bibhudendra. (2019). Visual object tracking based on discriminant DCT features. Digital Signal Processing. 95. 10.1016/j.dsp.2019.08.002.
- [36] Zhang, Tianzhu & Xu, Changsheng & Yang, Ming-Hsuan. (2017). Multi-task Correlation Particle Filter for Robust Object Tracking. 4819-4827. 10.1109/CVPR.2017.512.
- [37] Ma, Chao & Huang, Jia-Bin & Yang, Xiaokang & Yang, Ming-Hsuan. (2017). Adaptive Correlation Filters with Long-Term and Short-Term Memory for Object Tracking. International Journal of Computer Vision. 10.1007/s11263-018-1076-4.
- [38] Yang, Tao & Cappelle, Cindy & Ruichek, Yassine & Bagdouri, Mohammed. (2019). Multi-object tracking with discriminant correlation filter based deep learning tracker. Integrated Computer-Aided Engineering. 26. 1-12. 10.3233/ICA-180596.
- [39] Redmon, Joseph & Divvala, Santosh & Girshick, Ross & Farhadi, Ali. (2016). You Only Look Once: Unified, Real-Time Object Detection. 779-788. 10.1109/CVPR.2016.91.
- [40] Zhu, Pengfei & Wen, Longyin & Du, Dawei & Bian, Xiao & Hu, Qinghua & Ling, Haibin. (2020). Vision Meets Drones: Past, Present and Future.
- [41] Matas, Jiri & Galambos, C. & Kittler, J.. (2000). Robust Detection of Lines Using the Progressive Probabilistic Hough Transform. Computer Vision and Image Understanding. 78. 119-137. 10.1006/cviu.1999.0831.
- [42] Yuan, Chunhui & Yang, Haitao. (2019). Research on K-Value Selection Method of K-Means Clustering Algorithm. J. 2. 226-235. 10.3390/j2020016.
- [43] Lukežič, Alan & Vojir, Tomas & Čehovin Zajc, Luka & Matas, Jiri & Kristan, Matej. (2017). Discriminative Correlation Filter with Channel and Spatial Reliability. 4847-4856. 10.1109/CVPR.2017.515.
- [44] Yan, Jiangqiao & Wang, Hongqi & Yan, Menglong & Wenhui, Diao & Sun, Xuewen & Li, Hao. (2019). IoU-Adaptive Deformable R-CNN: Make Full Use of IoU for Multi-Class Object Detection in Remote Sensing Imagery. Remote Sensing. 11. 286. 10.3390/rs11030286.

- [45] Liu, Ziqiong & Wang, Shengjin & Ding, Xiaoqing. (2012). ROI perspective transform based road marking detection and recognition. ICALEP 2012 - 2012 International Conference on Audio, Language and Image Processing, Proceedings. 841-846. 10.1109/ICALIP.2012.6376731.
- [46] S. Agarwal, A. Awan, and D. Roth, "Learning to detect objects in images via a sparse, part-based representation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 11, pp. 1475–1490, 2004.
- [47] John van Rijn. "Road Capacities". INDEVELOPMENT. From: <https://docplayer.net/26200263-John-van-rijn-indevelopment-road-capacities.html>.