©2020

Yi Ge
ALL RIGHTS RESERVED

A Spatial-Temporal-Map-based Traffic Video Analytic Model for Large-Scale Cloud-based Deployment

By

Yi Ge

A thesis submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Master of Science

Graduate Program in Civil & Environmental Engineering

Written under the direction of

Peter J. Jin

And approved by

New Brunswick, New Jersey

Octorber 2020

ABSTRACT OF THE THESIS

A spatial-temporal-map-based traffic video analytic model for large-scale cloud-based deployment

by YI GE

Thesis Director: Peter J. Jin

The CCTV (Closed Circuit Television) traffic surveillance systems are one critical part of traffic operations and management at state and local DOTs (Departments of Transportation). In this thesis, a multi-lane traffic detection model based on the spatial-temporal map (STMap) is proposed, which is a few longitudinal scanlines instead of the entire video frame to detect vehicle trajectories. The proposed model is built based on a prior STMap-based aerial video analytic method but with significant improvement addressing issues with roadside CCTV traffic cameras. A motion-flow-based direction determination method, a bi-section occlusion detection and splitting algorithm, and the lane-changing track method are proposed for determining the directions, splitting vehicles from occlusions, and identifying lane-changes. The model evaluation is conducted by using videos from multiple cameras from the NJDOT (New Jersey DOT)'s 511 traffic video surveillance system. The results show promising performance in both accuracy and computational efficiency for potential large-scale cloud deployment.

Acknowledgement

For research, I would like to pay my special regards to Peter J. Jin, my supervisor, who offered me the chance to contribute myself to transportation-related research, and helped me so much in my research life. I also want to show my gratitude to my committee members for their valuable suggestions and patience. I wish to acknowledge Tianya Zhang, for his previous research, and help in this one, and I also wish to acknowledge Jonathan Martinez for his assistance. For background knowledge and skills, I wish to express my sincere appreciation to Jiaqi Tian, Yong Liu, Yuan Wang, and Yurong He for their help and inspiration in programming. Congratulations on Yizhou Wang's marriage and thanks for all the kindness and help in solving tricky problems. In addition, I want to thank Anjiang Chen for all the assistances in document format editing. As a son, I want to thank my parents for their consistent support. I want to appreciate my best friend Chencheng Xu for all the help in adapting to abroad stuffs. Last and most important, I want to thank my girlfriend, Yuting Wang, for everything, love you.

Table of Contents

List of Tables

List of Illustrations

1. Introduction and Problem Definition

1.1. Research Purpose

In this thesis, a Computer-Vision-based traffic data collection study is presented for vehicle detection using a STMap-based multi-lane traffic detection model.

The CCTV (Closed Circuit Television) traffic surveillance systems are one of the most important assessments of traffic management centers (TMCs) to monitor congestion and incidents in daily operations. Computer vision sensors based on CCTV traffic cameras have been explored over the last few decades. Most of the earlier commercialized systems rely on software programs or hardware computing units fully calibrated for individual CCTV traffic cameras, often preconfigured to fixed angles (Michalopoulos, 1991). Recently, researchers and engineers started to develop easy-to-calibrate computer vision systems for PTZ(Pan-Tilt-Zoom) cameras (Song&Tai, 2016; Birchfield et al., 2010). TMCs can use the existing large number of CCTV traffic cameras to generate speed, flow, occupancy, or even trajectory data to support TSM&O (Transportation Systems Management and Operations) and emerging Connected and Automated Vehicle (CAV) applications. The rapid growth of cloud computing and crowdsourcing technologies provides new opportunities for deploying such high-resolution computer vision systems at the state or regional level for large-scale traffic operations and CAV system needs.

Current commercialized systems can be classified into two major categories, the integrated hardware-software solutions and the generic computer vision solutions. The former focuses on building customized video analytic models fully calibrated with a specific type of camera. Representative products include Autoscope (1984), Gridsmart (2006), etc. Those systems' integrated nature can ensure the software analyzes the video data's full resolution at the “edge” while achieving optimal results with fully calibrated models. However, deploying such systems will need cameras installed at dedicated locations, and the cameras cannot be used for other surveillance purposes. The latter approach emerges with the rapid development of cloud computing technologies and computer vision technologies. The new generation of computer vision systems does not rely on tight integration with specific camera models. The deep learning and latest computer vision models can be used by process traffic video from any scenes with little or no manual input information such as scanlines, detection zones. Representative solutions include CitiLog (1997), TrafficVision (1999), MetroTech (2012), and Good Vision. These solutions can be deployed to existing CCTV video systems without the need for additional hardware installation. However, due to the complexity of the computer vision algorithms used, most of the systems can only process the data offline to achieve high accuracy.

In New Jersey, the main real-time transportation data categories include travel time and traffic event data. Such data are provided through the live data feed from TRANSCOM, a coalition of more than 19 transportation agencies in the tri-state area. TRANSCOM travel time data comes from probe vehicle data collected through EZ-Pass readers (including those for non-toll purposes), Inrix, and HERE(Nokia) probe vehicle travel time data. The EZ-Pass reader data are published to the public, while the full TRANSFusion data integrating all three data sources are only for member transportation agencies.

Nevertheless, the lack of real-time traffic flow data in the dynamic data feed offerings has led to some severe traffic operations and control limitations. Without the flow data, it is difficult to tell whether or not a free-flow segment is in factor free-flow or closed. It is also difficult to tell whether or not the traffic on a free-flow travel time segment has light traffic or saturated flow that may break down at any time. Many regional freeway and arterial traffic control solutions, such as Active Traffic Management (ATM) and adaptive traffic signal control, also heavily rely on dynamic traffic flow data. The coverage of fixed-location detectors such as loop detectors, RTMS (remote traffic microwave sensor), traffic dots, or pucks are still limited and not connected to any real-time dynamic traffic data feed. Meanwhile, more than 400 CCTV traffic cameras were deployed throughout the state and major roadways in NJ. Those cameras can potentially form a large virtual sensor network to generate traffic flow data. Those camera-inferred traffic flow data can be incorporated into the dynamic traffic flow data feed.

This study focuses on developing an efficient cloud-based online video analytic system for generating traffic flow data from large-scale regional CCTV traffic video network. The proposed method can be classified into the generic video analytic solutions since the proposed computer vision algorithms are not designed for specific types of video cameras and can be applied to the video streams from the existing CCTV traffic surveillance cameras. The proposed video analytic algorithms can address the issues, including computational efficiency for online deployment, the automated lane direction determination due to PTZ operations, occlusions in multi-lane traffic video, and lane changes.

STMap-based method (Ardestani et al., 2016) is selected to process large-scale traffic video streams at low computational cost. The proposed solution fixes some issues and improves the accuracy specifically for CCTV traffic videos with several key modules. First, the camera location and direction determination module determines the road name and directions of the lane-by-lane traffic detected by the proposed method. Second, the video analytic algorithms improve an existing model (Zhang, 2019) to generate lane-by-lane traffic flow. Third, the cloud-based parallel computing platform that uses AWS (Amazon Web Services) cloud to simultaneously calculate multiple CCTV traffic video streams at the same time. Fourth, mapping and reporting modules that map the generate lane-by-lane flow data with TRANSFusion link systems.

1.2. Problem Statement

STMap-based traffic video processing methods were proposed for limited traffic data extraction with low computing resources consumption, significantly reducing the required computing resources by only processing the pre-marked lines. The research problems include the following.

- Static noise

In STLine-based CCTV video processing, the vehicles in STMaps always coexist with some static noise, e.g., static objects, shadows, lane markers. The static noise has to be removed; otherwise, confusion may happen.

- Lane direction

In PTZ-supported CCTV camera view, the directions of lanes are necessary for mapping the processed data to traffic links, but the video stream does not supply the information of the directions that the cameras are facing.

- Data flow and computing resources limit of cloud platform

As the chosen cloud platform, AWS has a strict limit in download flow and CPU utilization, which makes it more important to fully utilize the unlimited upload flow (read video stream) and an unlimited number of extension servers. The optimal choice to avoid the extra charges from AWS is reading the CCTV video stream for processing and storing the results into Amazon RDS (Relational Database System).

- Limited features

CPU utilization can be saved by keeping only limited pixels are kept, and processing only limited features. Therefore, traditional vehicle detection and recognition models are not applicable in STMap-based vehicle detection, and new vehicle detection methods need to be developed.

- Occlusion under high traffic volume condition

Limited by the angle of CCTV cameras, high traffic volume may cause occlusion in camera videos.

1.3. Thesis Summary

The structure of the Master thesis is composed of the following chapters: (1) Brief introduction of the Master study; (2) Overview of research background and summary on the existing work related to the video processing methods and systems; (3) Major methodologies; (4) Experiment design, evaluation, and calibration; (5) Result analysis; (6) Deploy recommendation; (7) Conclusion and future work; (8) Reference.

2. Background

2.1. Overview of Video Analytic Technologies and Algorithms

Existing traffic video analytic systems can be classified into three major categories, including integrated camera and analytic solutions, universal virtual sensor solutions, and cloud-based smart city video analytic solutions. Representative products include Autoscope (1984), Citilog (1997), VISATRAM (Zhu, 2000), and GRIDSMART (2006) systems. The analytic software in those systems is often highly customized to fit the features of their OEM cameras. OEM cameras also reduce the need for frequent camera calibration. The key limitation is the closed system making it difficult to integrate other existing agency traffic video resources, and the pricing can be non-competitive with dedicated systems.

2.1.1. Integrated camera and analytic solutions

In the first category, image processing algorithms are developed and fully-calibrated for specific types of video cameras. Many of those systems even encode the image processing algorithms within the processing units directly connected to the cameras. Such tight integration allows those algorithms to take full advantage of the raw video's full resolution and quality to generate needed traffic data. Representative platforms include Autoscope (1984) and Gridsmart (2006).

- **Autoscope Systems**

Image Sensing Systems, Inc. emerged in 1984, which focused on developing and delivering above-ground detection technology, applications, and solutions. Image Sensing Systems combined video, radar, Bluetooth for detection, and supplied wrong way detection solutions and IntellitraffiQ solutions. Now, it has more than 140,000 Autoscope Vision units sold in over 70 countries worldwide.

- **Gridsmart Systems**

GRIDSMART Technologies Inc. was founded in 2006. The company pioneered the world’s first single-camera solution for intersection actuation, traffic data collection, and situational awareness. GRIDSMART uses one Omni-vision camera to monitor an entire intersection. GRIDSMART used a single camera with appropriate video processing to monitor the whole intersection, which has counted and classified more than 216 billion vehicles. The fisheye-camera-based Iconic GRIDSMART Bell Camera has been installed in 1200 cities, 49 states, and 29 countries.

The limitations of the integrated hardware-software platforms are their dependency on the installation and configuration of the hardware systems. Many transportation agencies already have their surveillance video systems, but in order to use integrated hardware-software platforms, the agencies need to install new dedicated fixed-position fixed-view video cameras for those platforms to generate data.

2.1.2. Universal virtual sensor solutions

Universal virtual sensor platforms are essentially hardware independent and developed for typical intersection or roadside traffic scenes often observed in existing CCTV traffic cameras. Some platforms can be deployed to existing traffic data sources even PTZ cameras (e.g., TrafficVision (1999)) with the capability of adjusting the “scanlines” or “detector zones” in PTZ operations. Representative systems include CitiLog (1997), TrafficVision (1999), MetroTech (2012), GoodVision (2017), Miovision (2005), KiwiVision (2011), Aventura (1999). However, significant manual work is still needed to set up virtual detection zones for each camera position and can be difficult for on-demand video sources analytics on-the-fly. The output is simple traffic states derived from vehicle occurrence detection in the manually-drawn detection zone. The occurrence-based detection limits the resolution of the data to support more complicated measures, e.g., advance detection for Purdue diagram analysis. Recently, with the development of AI(Artificial Intelligence) technologies, especially, deep learning models such as Mask RCNN (Region-based Convolutional Neural Networks) (Bharati, 2019; Omeroglu, 2019) and YOLO (Corovic, 2018; Iwasaki, 2018; Lin, 2018), the accuracy of some latest generic computer vision platforms has been significantly improved over the years. Though running those AI models will require significant computational resources and is often due through cloud computing services and cannot be achieved in real-time. Representative video analytic systems are as follows.

- **CitiLog**

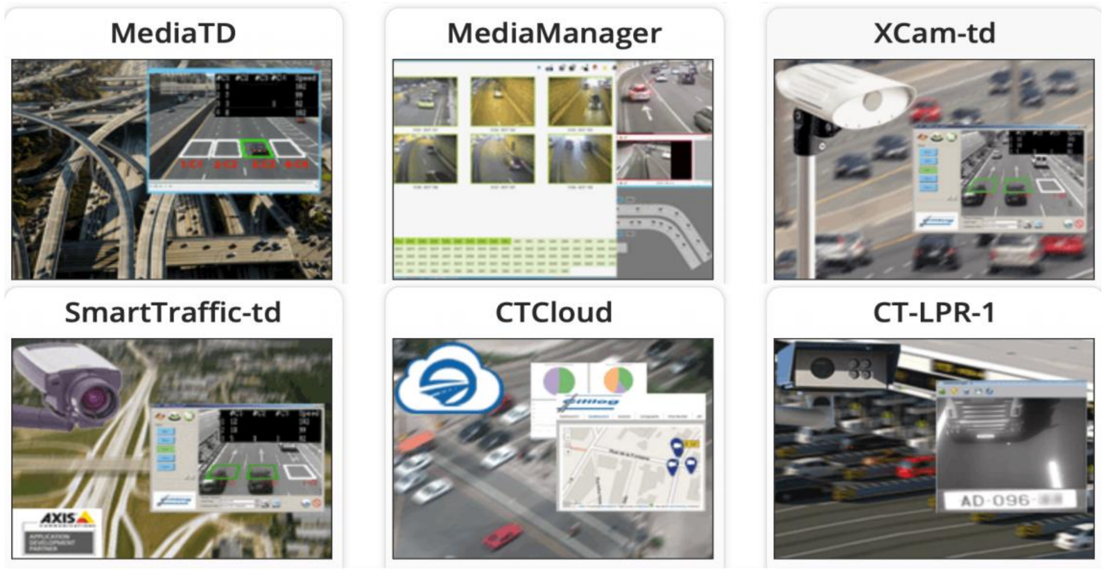
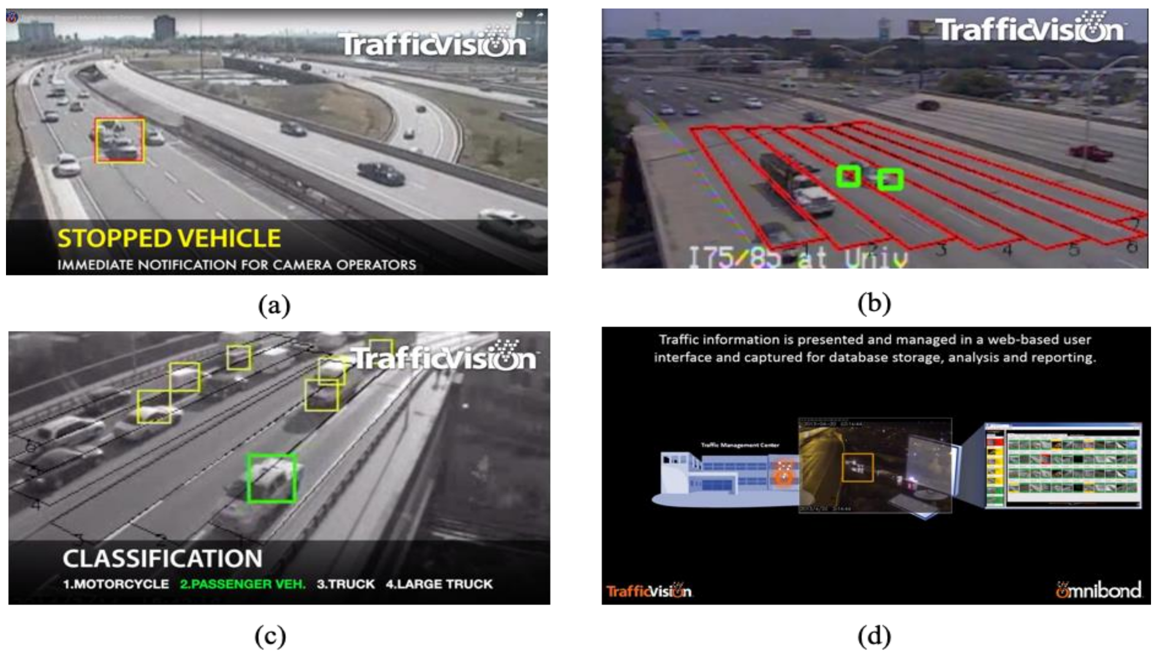


Figure 1 CitiLog Video Analytic Applications

CitiLog was founded in 1997. CitiLog has both integrated hardware and video analytic technologies with its AVIX IP cameras (http://www.citilog.com/product/en/smartcam). The company also has universal detection technologies that can be applied to existing cameras. As Figure 1 shows, CitiLog's traffic detection suites can be used to detect traffic parameters (speed, flow, density) such as MediaTunnel, MediaRoad, MediaTD, MediaManager, intersection flow, and queue lengths such as XCam-p, XCam-ng, XCom, SmartTraffic-p, SmartTraffic-ng, incidents such as VisioPaD, VisioPaD+, SmartTraffic-AID, SmartTraffic-i, SmartTraffic-ww, license plates for generating toll, tracking, and journey travel time information such as CT-LPR-1, CT-LPR-2, CT-HAZ. Its products monitor over 900 sites in 55 countries and have been processing 32,000 video inputs every day.

- **TrafficVision**



Traffic information is presented and managed in a web-based user interface and captured for database storage, analysis and reporting.

Figure 2 Sample Snapshots of Traffic Vision Products

TrafficVision is a division of Omnibond (http://www.trafficvision.com) founded in 1999. It has been focused on real-time traffic monitoring and has high versatility on data collection and incident detection. The company supplies applications for traffic detection such as Incident Detection (Stopped vehicle, wrong way, slowed traffic), Automatic Re-

Calibration, Vehicle Classification. To be mentioned, its real-time incident detection enables immediate alert. Now TrafficVision has helped the Ontario Ministry of Transportation collect over 20,000 hours of traffic data around Toronto.

- **MetroTech**

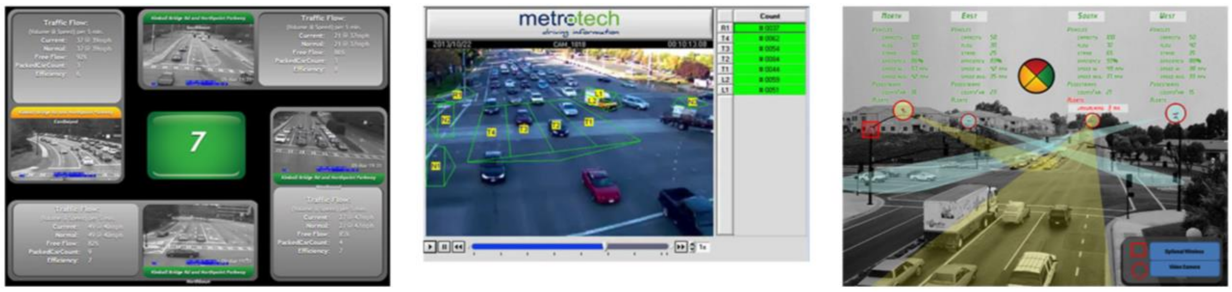


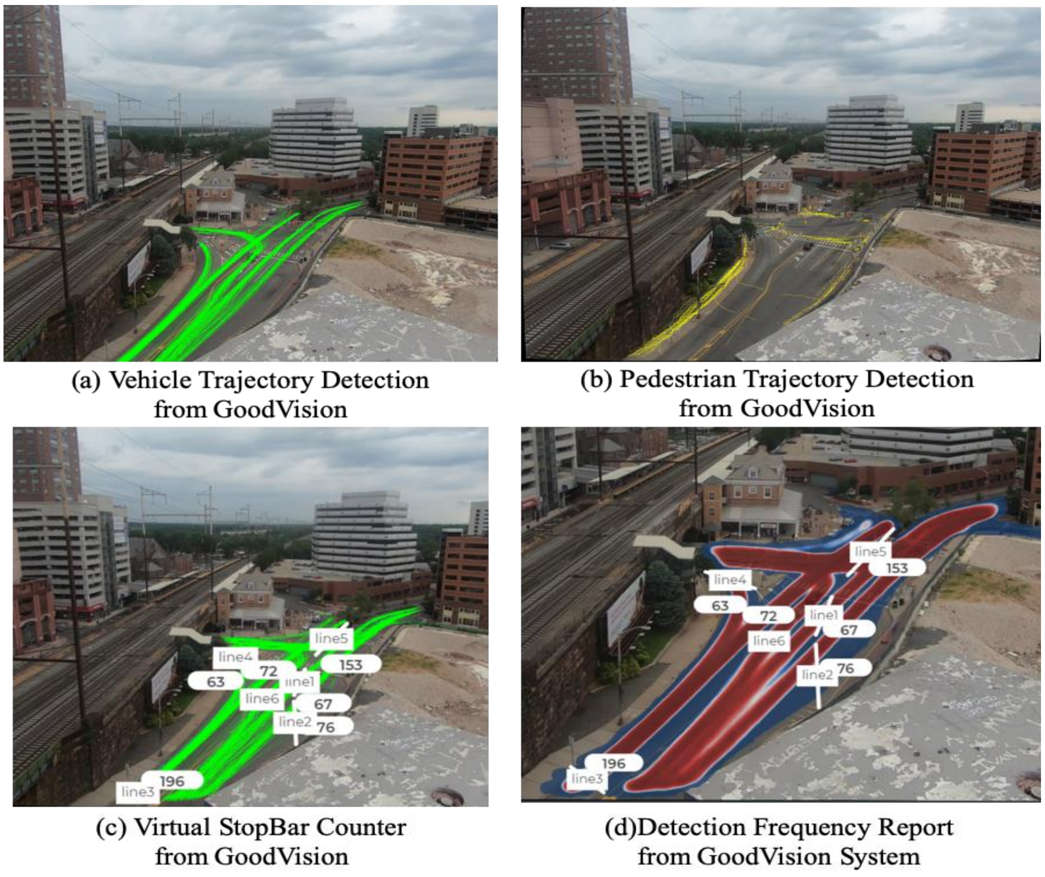
Figure 3 MetroTech Traffic Video Analytic Applications

MetroTech is a company that earned its fame since 2012 (https://metrotech-net.com). It provides precise, cost-effective, and real-time information by aggregating existing sensors and information. The company supplies MetroTech Family of Products, such as IntelliSegment for piece data collection, IntelliSection for data analysis and distribution, MetroTech Digital Streets Fusion Center for data aggregation, and the MetroTraffic Network for global data production. MetroTech can apply their video analytic models to existing traffic cameras with some manual setup of the detection zones. MetroTech was recognized as an up and coming competitor in the industry of smart infrastructure by Government Technology (https://www.govtech.com/100/2019).

- **GoodVision**

GoodVision (2017) is an online deep-learning-based traffic video analytics tool with a price of €15 per video-hour. It has a comprehensive web interface for uploading video and analyzing the results as shown in the figure below.

It is good at dealing with high-resolution traffic video at a middle angle or high angle and can not only classify the moving objects into trucks, vans, motorbikes, pedestrians, etc. but also display the graphical results. It does not require the customers to install any software, but it asks the customers to upload recorded videos and wait patiently for the results. The GoodVision application can also output customized traffic count results by drawing virtual zones or stop-bars on the web interface as shown in the figure below. It also has the limitations of distance from monitored objects, obstacles, lenses quality, lighting conditions, and image resolutions.



(a) Vehicle Trajectory Detection from GoodVision

(b) Pedestrian Trajectory Detection from GoodVision

(c) Virtual StopBar Counter from GoodVision

(d)Detection Frequency Report from GoodVision System

Figure 4 Sample Outputs from the GoodVision Traffic Counting System

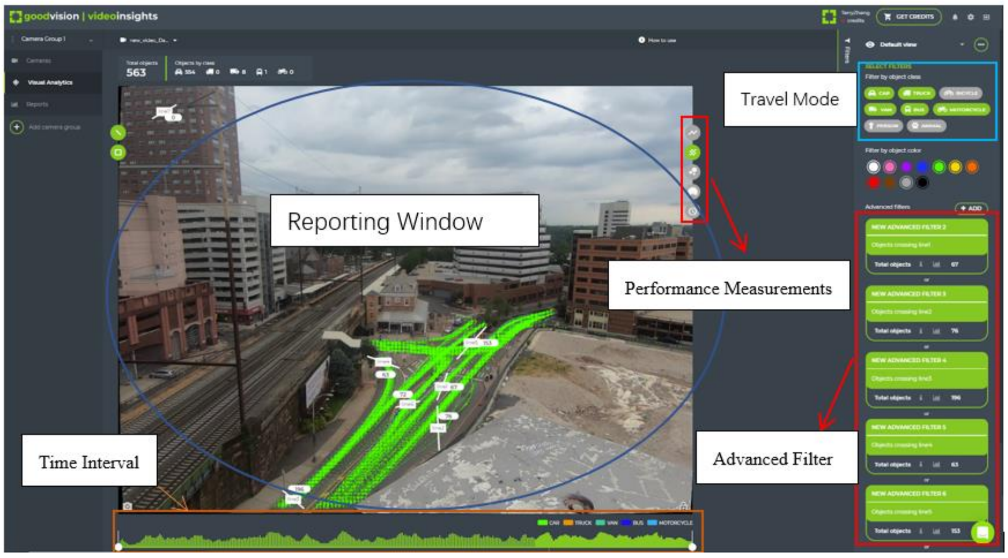


Figure 5 User Interface of GoodVision Video Analytic System

Figure 5 shows the GoodVision application user interface to generate traffic detection and performance metrics. The reporting panel demonstrates the detected trajectory results. The travel mode panel allows users to select different travel modes. The advanced filter can be used to select zone-based or stop-bar based outputs. The time interval panel can be used to select a specific time duration.

Those systems can be deployed to existing traffic data sources even PTZ cameras (e.g., TrafficVision (1999)), and are flexible with existing agency traffic video sources. However, significant manual work is still needed to set up virtual detection zones for each camera position and can be difficult for on-demand video sources analytics on-the-fly. The output is simple traffic states derived from vehicle occurrence detection in the manually-drawn detection zone. The occurrence-based detection limits the resolution of the data to support more complicated measures, e.g., advance detection for Purdue diagram analysis.

2.1.3. Cloud-based smart city video analytic solutions

The advance in central (CPU) and graphics processing unit (GPU) technologies and cloud computing triggers the current wave of cloud-based smart city video analytic solutions. Many platforms such as Placemeter (2012), MicroFocus (ex-HP), IBM Intelligent Video Analytics, Cisco Smart City Cloud, BriefCam, have the capability of crowdsourcing and on-demand video analytics. However, those systems often offer heavy-weight all-inclusive packages of smart city functionalities such as TSM&O (Transportation System Management and Operations), energy and utility management, etc. There have not been light-weight cloud-based crowdsourcing video analytic solutions dedicated to traffic state detection in the market.

2.1.4. Computer-Vision-based Traffic Detection Algorithms

Computer-vision-based traffic detection algorithms can be classified into scanline or scanning box based, model-based, and deep-learning-based algorithms.

Scanline or scanning box based models detect the intensity changes at a localized area in the video to detect vehicles at virtual sensor locations on highways or at intersections. Michalopoulos (1991) created the Autoscope system by defining a “virtual sensor trap” and evaluated the performance with live data from Traffic Management Centers. Beymer, et al. (1997) and Coifman, et al. (1998) developed feature-based real-time computer vision systems separately, both of which could work well during the daytime. Cucchiara, et al. (2000) used rule-based reasoning to develop a traffic monitoring system that could not only work in the daytime but also work at nighttime. Zhu, et al. (2000) and Malinovskiy, et al. (2009) used STMap to develop video-based vehicle detection and tracking systems. Pang, et al. (2007) presented a method for vehicle count in the presence of multiple-vehicle occlusions in traffic images. Hsieh, et al. (2006), Zhang, et al. (1993) and Cheng, et al.

(2005) developed systems for not only detecting and counting vehicles but also classify them using traditional ways. Dong, et al. (2015), Mithun, et al. (2012) and Zhang (2012) developed vehicle classification systems based on AI technology. Bas, et al. (2007) presented a method for automatic vehicle counting from the low-resolution video.

The model-based algorithm explores different types of motion, optical, and time differencing features to detect and track the movement of vehicles in video footage. Kanhere, et al. (1993, 2008) proposed systems that could measure traffic counts and speeds on highways successively. Chen, et al. (2010) developed a motion&contrast-based real-time vision system for nighttime vehicle detection and traffic surveillance. Tai, et al. (2004) presented a real-time image tracking system for automatic traffic monitoring and enforcement applications. Zhao, et al. (2013) proposed to use the Scene structure model, Kanade–Lucas–Tomasi tracker, and GMM for vehicle counting. Ishak, et al. (2016) proposed a method of vehicle counting using Solo Terra Autoscopes, which could not only count vehicles but also monitor if they made turns.

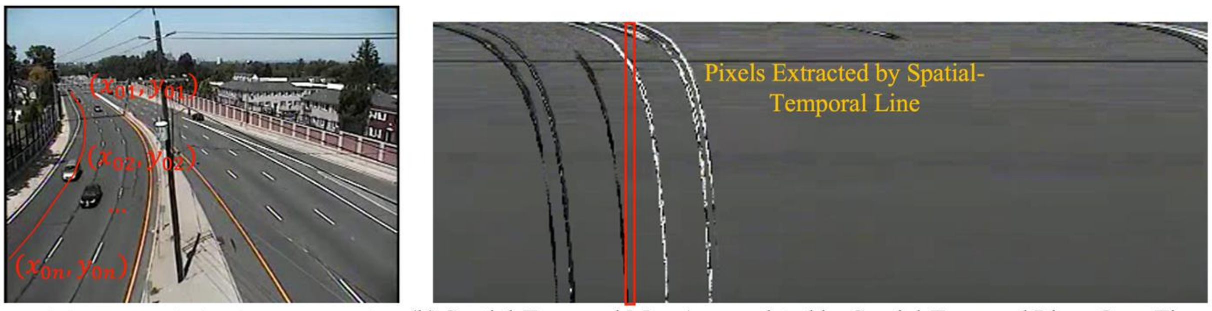
More recently, deep-learning-based computer vision algorithms emerge that use pre-trained object detection models to recognize and track vehicles. Typical examples include the R-CNN (Region Proposal Convolutional Neural Network, Girshick et al. (2014)), Faster-CNN (Ren et al. (2015)), YOLO (You Only Look Once, Redmon et al. (2016)). The key limitation of those models for video analytics with CCTV traffic cameras lies in its training data requirement. Often those models are trained with high-resolution, ground-level datasets and cannot be easily applied to low-resolution roadside video. Furthermore, with the use of GPUs, the computational cost can be significant for large-scale deployment.

2.1.5. Video Analytics with the Scanline-based Spatial-Temporal Diagrams

In this thesis, the proposed algorithm improves an existing longitudinal scanline-based vehicle detection method. The longitudinal scanline is a scanline in a video that goes along with the centerlines of travel lanes. Such a scanline still only scans a limited number of pixels in an entire video frame to save computational cost. However, it retains the motion information of vehicle trajectories, unlike localized intensity changes in scanline or scanning box based methods. Prior development includes Zhu et al. (2000)'s VISTRAM systems, Taniguchi et al. (1999)'s directional temporal plane transformation (DTT) method, Cho and Rice (2006)'s longitudinal mask-based method, and Malinovskiy et al. (2009)'s ST (Spatial-Temporal) maps based method for highway CCTV traffic video. Ardestani et al. (2016) developed a longitudinal-scanline-based method for traffic signal timing detection by tracking vehicle stoppings on the scanlines. More recently, Zhang and Jin (2019) developed a High-angle Spatial-Temporal Diagram Analysis (HASDA) model to reconstruct high-resolution vehicle trajectories from high-angle NGSIM-like traffic video. An adaptation of the model over the arterial traffic scene is proposed by the research group in 2020 (Zhang et al. (2020)). In this thesis, the proposed method improves the HASDA algorithm for low/medium-angle roadside CCTV traffic video scenes.

3. Methodologies

3.1. Spatial-Temporal Line (STLine) and Spatial-Temporal Map (STMap)



(a) Spatial-Temporal Line in Camera View (b) Spatial-Temporal Map Accumulated by Spatial-Temporal Lines Over Time

Figure 6 STLine-the Connection between Camera View and STMap

As Figure 6 (a) shows, STLine is defined as a polyline marked in camera view along the center of a lane.

$$\boldsymbol{ST_{il}} = \{(\boldsymbol{x_{lm}}, \boldsymbol{y_{lm}}) : \boldsymbol{m} = \mathbf{1}, \dots, \boldsymbol{M_l}\} \qquad (1)$$

where ST_{il} indicates the STLine defined for the lane l at camera i, (x_{lm}, y_{lm}) is the pixel coordinate of the mth point on the STLine ST_{il}, which indicates the physical location of STLine, $m = 1, \dots, M_l$ where M_l is the total number of points that define the STLine ST_{il}. $l = 1, \dots, L$ where L is the total number of lanes at the camera location i. As shown in Figure 1 (b) shows, STMap is a stacking of the pixels of STLines along the direction of time (video frame). The vertical axis of a STMap is the pixel index on the STLine, and the horizontal axis is video time or frame number. Every vehicle that passes through a STLine will leave a trace of its body when covering the STLine. Such foreground connected areas left by a vehicle on STMAP is called a strand. The color of strands may vary based on the vehicles' color.

3.2. The Integrated STMap-Based Traffic Detection Framework

The proposed STLine-based video analytic algorithms include the following key processing steps, as shown in Table 1. Some example processing results are provided to illustrate the outputs for each step. The algorithm is based on the HASDA model (Zhang et al., 2019) developed for high-angle traffic video. The thesis made a significant improvement to the shaded steps that achieved significant performance improvement for large-scale deployment with low-resolution roadside traffic video.

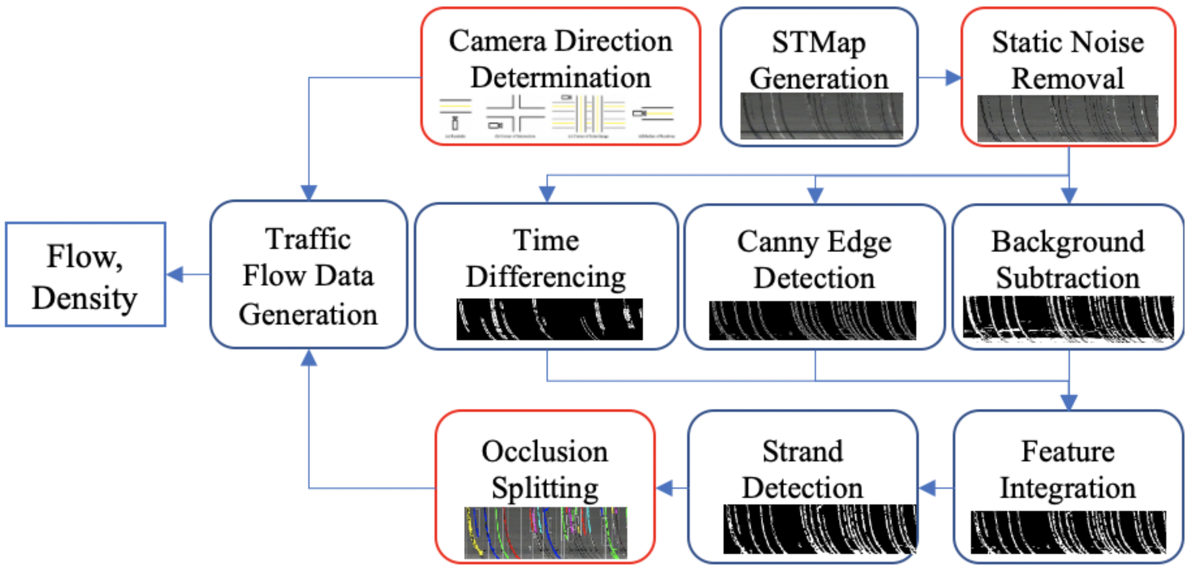
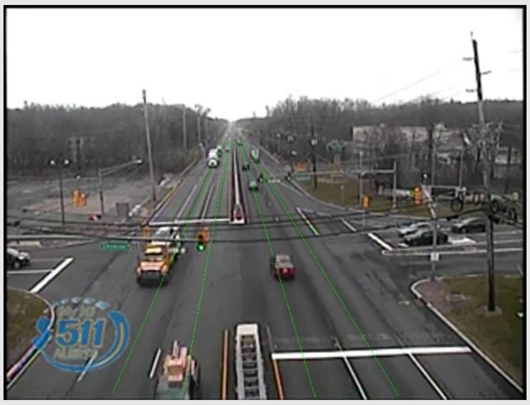
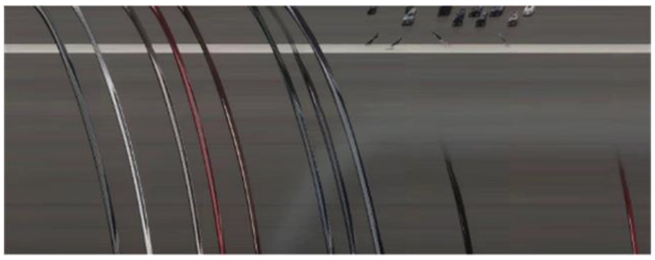
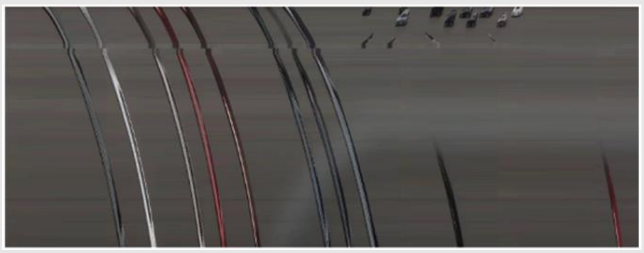
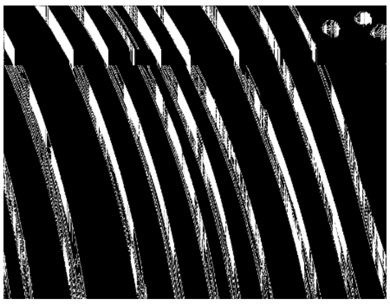
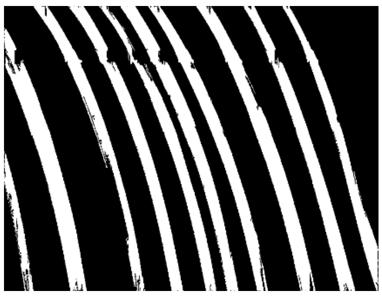
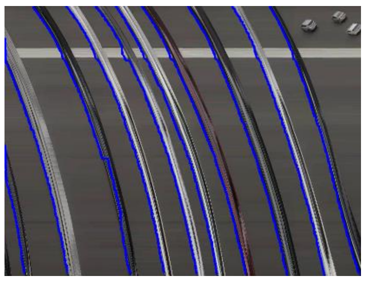
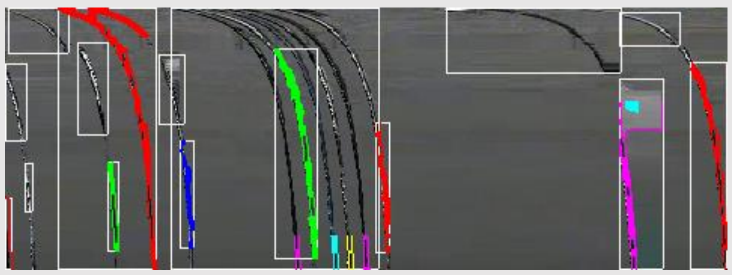


Figure 7 Key Modules and Sample Outputs

Table 1 Key modules and sample outputs of the proposed video analytic algorithms

Processing Methods/Modules	Sample Output
a. Scanline based Camera and Lane Direction Determination	

b. Spatial-Temporal Diagram	
c. ST Diagram Denoising and Preprocessing	
d. Vehicle Strands Detection from ST Diagram	
e. Crossing Vehicle Removal from ST Diagram	
f. Strand Edge (Front bumper location) Detection and Pixel Trajectory Output	
g. Occlusion Separation and Lane Change Tracking	

A STMap is first generated for each lane (Zhang et al., 2019) from the CCTV traffic video stream. Then a novel static noise removal method is conducted, which detects static noise rows from the STMap and generates clean rows by detecting and removing vehicle strands to replace the static noise rows. The static-noise-free STMap is then processed with time differencing module, the Canny edge detection module, and the background subtraction module as in HASDA. Vehicle strands are detected and extracted from the STMap by integrating the features from modules b and c. An occlusion splitting module is then proposed to separate the occlusion caused by the CCTV camera angle. The lane-changing detection and tracking module is introduced to reduce errors caused by vehicle crossing lanes. At last, combined with the camera direction determination method, traffic flow data is generated.

3.3. STLine-direction-based Camera Direction Detection

The raw 511 traffic video streams from NJDOT do not contain any directional code indicating the direction of traffic flow. The traffic management software platform often puts a directional code and time code onto the video based on preconfigured positions within the platform. However, the raw video streams published do not contain such information. Furthermore, these traffic cameras have pan, tilt, and zoom capabilities and can be moved to focus on different segments or incident sites. Such directional information is vital to ensure the detected traffic flow data is assigned to the correct roadway links. The camera direction determination algorithm is proposed as follows.

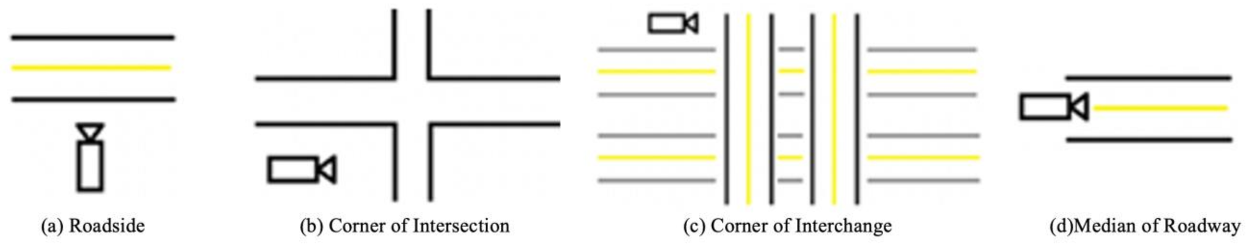


Figure 8 Camera Direction Determination Scenarios

Figure 8 Camera Direction Determination Scenariosillustrates four different types of camera locations with respect to the roadway types covered based on the reviewing of NJDOT 511 camera locations. An Excel spreadsheet is obtained from NJDOT to provide the relative position of each camera with respect to the roadway geometry around the camera. Such information includes which side of the roadway that camera is located, such as EB, SB, WB, NB for roadside cameras and SE, SW, NE, NW for corner cameras. Second, each STLine l will be manually drawn so that its starting point (x_{l1}, y_{l1}) and its endpoint (x_{l2}, y_{l2}) match with the direction of traffic on each lane in the actual 511 video footage. The STLine can also be generated from inspecting the optical flows of vehicle trajectories in the image, which will be explored in future works.

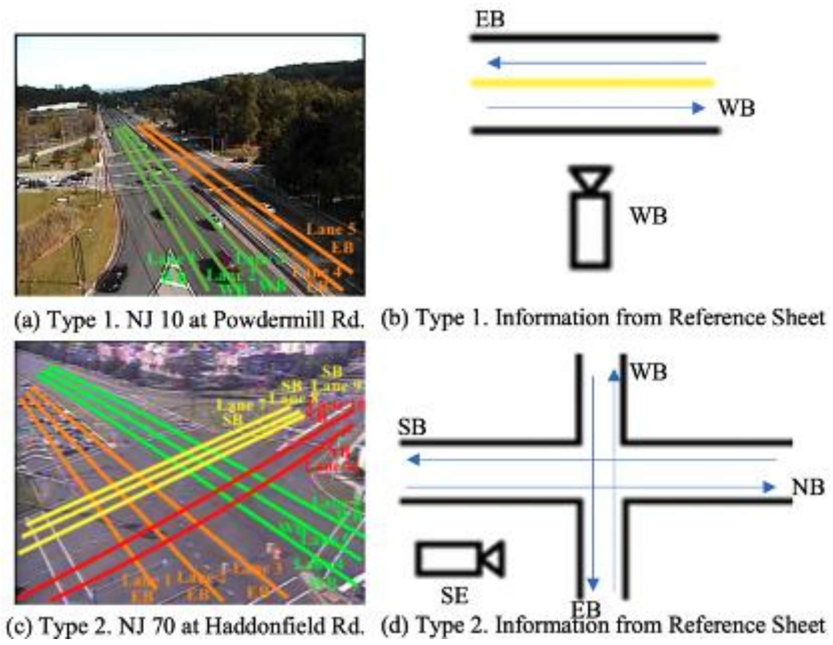

Lane 5
EB
EB
WB
WB
SB
NB
SB
EB
SE
(a) Type 1. NJ 10 at Powdermill Rd.
(b) Type 1. Information from Reference Sheet
(c) Type 2. NJ 70 at Haddonfield Rd.
(d) Type 2. Information from Reference Sheet

Figure 9 Camera Direction Determination Example

By comparing the direction of traffic in the STLine and the relative position table of each camera, the position of the camera and the direction of the STLine links can be determined. The main idea is as follows. Since the US follows the right-side driving rule, if one found that the vehicle moves from left to right in the video footage, then that traffic stream will be on the same side of the road with a camera or close to the corner of the camera at an intersection or interchange. Such logic cannot be easily extended to where the camera is located at a median without other object detection algorithms implemented, such as intersection and landmark detection. Table 2 provides a lookup table for three of the scenarios shown in Figure 9.

Table 2 Relations between Camera Directions and STLine Pixel Coordinate Characteristics

Roadside Camera Position	Northbound Roadside		Southbound Roadside	
Scanline direction	$x_{l1} > x_{lM_l}$	$x_{l1} < x_{lM_l}$	$x_{l1} > x_{lM_l}$	$x_{l1} < x_{lM_l}$
Direction of lane	SB	NB	NB	SB
Roadside Camera Position	Eastbound Roadside		Westbound Roadside	
Scanline direction	$x_{l1} > x_{lM_l}$	$x_{l1} < x_{lM_l}$	$x_{l1} > x_{lM_l}$	$x_{l1} < x_{lM_l}$
Direction of lane	WB	EB	EB	WB
Corner Camera Position	Northeast Corner of an Intersection			
Scanline direction	$x_{l1} > x_{lM_l}$ $y_{l1} > y_{lM_l}$	$x_{l1} > x_{lM_l}$ $y_{l1} < y_{lM_l}$	$x_{l1} < x_{lM_l}$ $y_{l1} > y_{lM_l}$	$x_{l1} < x_{lM_l}$ $y_{l1} < y_{lM_l}$
Direction of lane	SB	EB	WB	NB
Corner Camera Position	Southeast Corner of an Intersection			
Scanline direction	$x_{l1} > x_{lM_l}$ $y_{l1} > y_{lM_l}$	$x_{l1} > x_{lM_l}$ $y_{l1} < y_{lM_l}$	$x_{l1} < x_{lM_l}$ $y_{l1} > y_{lM_l}$	$x_{l1} < x_{lM_l}$ $y_{l1} < y_{lM_l}$
Direction of lane	WB	SB	NB	EB
Corner Camera Position	Northwest Corner of an Intersection			
Scanline direction	$x_{l1} > x_{lM_l}$ $y_{l1} > y_{lM_l}$	$x_{l1} > x_{lM_l}$ $y_{l1} < y_{lM_l}$	$x_{l1} < x_{lM_l}$ $y_{l1} > y_{lM_l}$	$x_{l1} < x_{lM_l}$ $y_{l1} < y_{lM_l}$
Direction of lane	EB	NB	SB	WB
Corner Camera Position	Southwest Corner of an Intersection			
Scanline direction	$x_{l1} > x_{lM_l}$ $y_{l1} > y_{lM_l}$	$x_{l1} > x_{lM_l}$ $y_{l1} < y_{lM_l}$	$x_{l1} < x_{lM_l}$ $y_{l1} > y_{lM_l}$	$x_{l1} < x_{lM_l}$ $y_{l1} < y_{lM_l}$
Direction of lane	NB	WB	EB	SB

3.4. Static Noise Removal on STMap

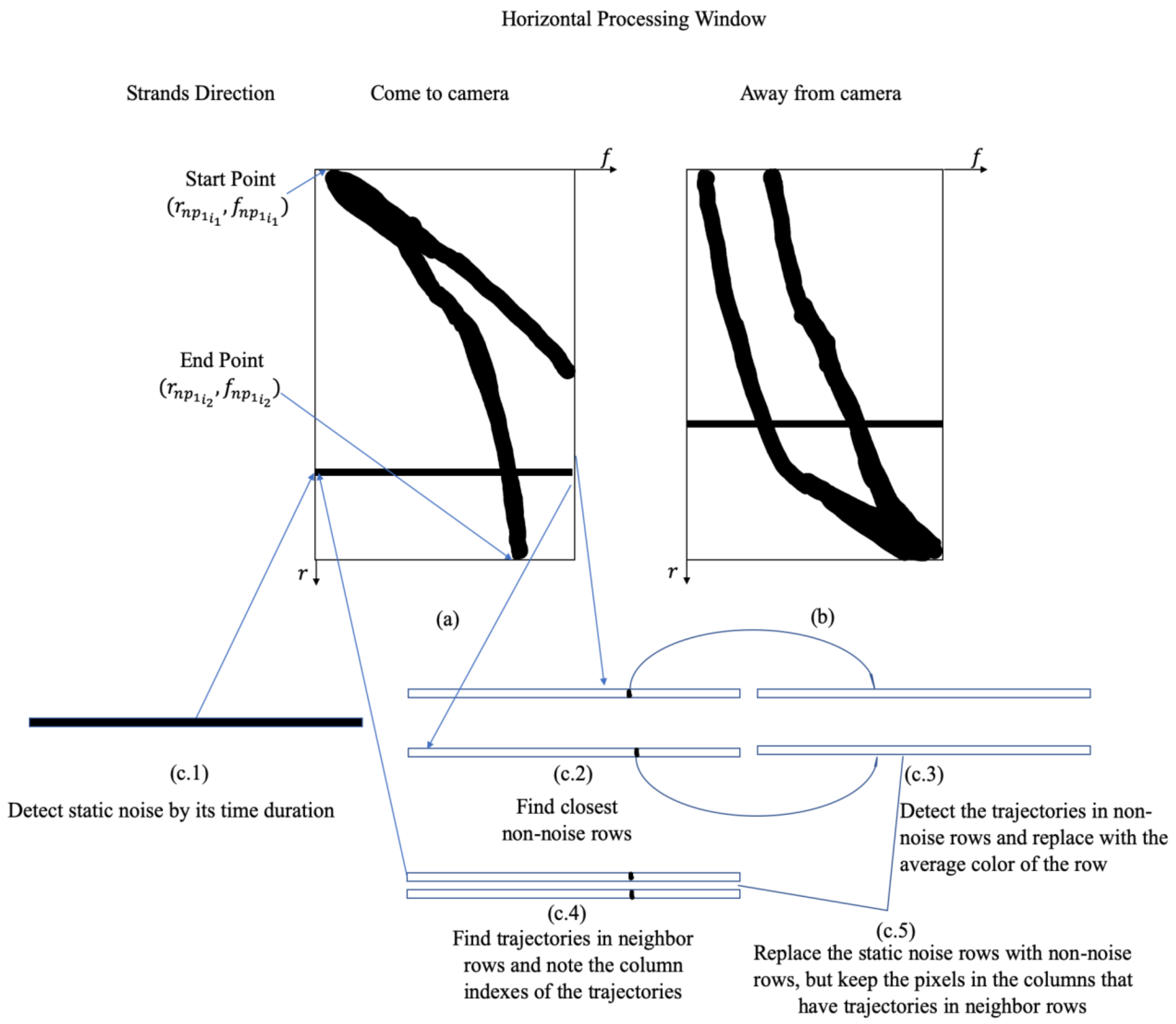

Horizontal Processing Window
Strands Direction
Come to camera
Away from camera
Start Point
$(r_{np_{1i_1}}, f_{np_{1i_1}})$
End Point
$(r_{np_{1i_2}}, f_{np_{1i_2}})$
f
r
(a)
(b)
(c.1)
Detect static noise by its time duration
(c.2)
Find closest non-noise rows
(c.3)
Detect the trajectories in non-noise rows and replace with the average color of the row
(c.4)
Find trajectories in neighbor rows and note the column indexes of the trajectories
(c.5)
Replace the static noise rows with non-noise rows, but keep the pixels in the columns that have trajectories in neighbor rows

Figure 10 Illustration of the Static Noise (Light poles, lane markings, etc.) Removal Algorithms

As Figure 10(a)(b) shows, the static noise on STMap results from static objects usually stays longer in the same row, which has a significant difference with the strands caused by moving vehicles and can be detected by its duration and color difference. For example, in the camera view of US1 at NJ18, there was a pole covering all three westbound lanes, and there were horizontal patterns in its STMaps of the three westbound lanes which did not have vertical movement at all and occupied the same rows for a long time.

Detection: For a processing window consists of $T_{processing\ window}$ of video frames,

$$TD_p = max(f_{p_i}) - min(f_{p_i})$$

if $TD_p > thrld_{ub} \times T_{processing\ window}$ or ($TD_p > thrld_{lb} \times T_{processing\ window}$ and $|RGB_p - RGB_{nr}| > thrld_{row\_diff}$), then p is static noise.

Where TD_p indicates the time duration of the strand p, (r_{p_i}, f_{p_i}) is the coordinate of the ith point in strand p, r_{p_i} for the vertical axis and f_{p_i} for the horizontal axis, $T_{processing\ window}$ indicates the total time of the processing circle, RGB_p indicates the mean RGB value of the strand, RGB_{nr} indicates the mean RGB value of a neighbor row, $thrld_{row\_diff}$ is a set value for RGB difference threshold. $thrld_{ub}$ and $thrld_{lb}$ are two preset values for time duration percentage, 80% and 40% were used for $thrld_{ub}$ and $thrld_{lb}$ in this thesis, respectively.

Removal: To remove the static noise after the detection, a search for nearest non-noise-rows will be activated based on the detected static rows, and the nearest non-noise-rows will be used to replace the static noise rows after filling in the potential trajectory columns on the nearest rows without noises. Sobel vertical edge detector is used to detect the trajectory in nearest rows without noises, and the detected trajectories will be replaced with the average color of nearest rows without noises from above and below the noisy rows. As Figure 10(c) shows, during the filling, if the noisy rows have vehicle trajectories on them, the blocks of pixels of vehicles will not be replaced to avoid removing critical trajectory information. If the filling rows, that is rows above or below the noisy rows without noises, have vehicle trajectories, the blocks of pixels of vehicles on those rows will be replaced by pixels before or after those blocks to avoid creating ghost vehicles.

```
For $r_{noise}$ in $R_{noise}$,
For row $r$ from detected noise rows $r_{noise}$ to both up and down boundaries,
if $r \notin R_{noise}$,
```

use Sobel edge detector to extract the column indexes of vertical edges $f_1, f_2, \ldots, f_n$ in row $r, r-1 \ and \ r-2$,

$$C_{traj} = \{r_{f_1}, r_{f_2}, \ldots, r_{f_n}\}$$

$$r_{C_{traj}} = AverageRGB(\complement_R C_{traj})$$

$$r_{noise} = r$$

Where R_{noise} is the aggregation of the detected noise rows r_{noise}, $r_{f_1}, r_{f_2}, \ldots, r_{f_n}$ are the columns of detected trajectory in row r, C_{traj} is the aggregation of detected trajectory columns $r_{f_1}, r_{f_2}, \ldots, r_{f_n}$, $r_{C_{traj}}$ are the corresponding columns of detected trajectory in row r, R is the aggregation of all the columns on row r, $\complement_R C_{traj}$ is the complementary set of C_{traj}.

3.5. STMap-based Vehicle Crossing Removal

Similar to Static Noise Removal, vehicle crossing can be detected by its time duration and height/width ratio too. The following is the proposed algorithm.

$$Size_p = \left(r_{p_i}\right) \ - min(r_{p_i})$$

If $TD_p < Thrld_{TD} \ and \ \frac{Size_p}{TD_p} > Thrld_{\frac{S}{T}}$, or $Size_p < Thrld_{size}$,

then strand p is vehicle crossing,

$$P = \{p_1, p_2, \ldots, p_i\}$$

$$P = Color_{background}$$

Where $Size_p$ indicates the size of the object, $Thrld_{TD}$, $Thrld_{\frac{S}{T}}$ and $Thrld_{size}$ are threshold values set manually, $Color_{background}$ is the color of the background.

3.6. STMap-based Vehicle Occlusion Detection and Separation

From the roadside angle, vehicle occlusions can create significant issues for computer vision algorithms. In the previous HASDA model, due to the use of aerial video, the algorithm does not need to deal with severe vehicle occlusions, which lead to significant undercounting of vehicles. In the proposed model, a new occlusion treatment method is proposed. The method is based on the observations that when vehicles are getting close to the camera locations, the separation among vehicles can be large enough for their strands to split on the STMap. The bisection method is used for occlusion separation. The detailed algorithm is as follows.

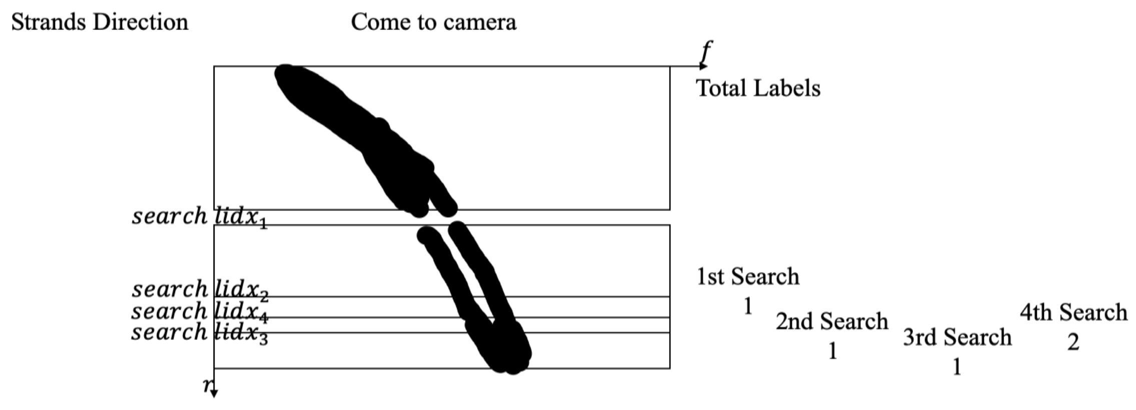


Figure 11 Illustration of the Proposed Occlusion Detection and Removal Algorithm

Average Strand Duration based Occlusion Detection: First, the black and white STMap $bwM_{R_l \times F}$ is labeled by the connected components inside, where R_l is the total number of pixels of the STLine on Lane l, F is the total number of video frames to be analyzed. The average strand duration in the number of video frames is calculated with the labeled components to determine if occlusion occurs.

$$ASD = \sum_{r_{min}}^{r_{max}} \frac{len(fids_r)}{(fids_r)\ - \min(fids_r)} \div (r_{max} - r_{min})$$

Where ASD stands for mean frames, r_{max} is the maximum row index that label p has, r_{min} is the minimum row index that label p has, $fids_r$ is the frame indexes that label p occupied on row r, $len(fids_r)$ is the total number of $fids_r$.

If $MF < thrld_{MF}$, there exists at least one occlusion, where $thrld_{MF}$ is a threshold for the average strand duration. Then the bisection-based occlusion separation will be activated.

Bisection-based Occlusion Separation: The bisection-based occlusion separation will keep searching for splits within the search area chosen by the bisection method. The first search area is the half STMap near the camera, and it will keep being divided into two parts, and the one close to the camera will be searched until the splits occlude each other again or the search area is too narrow to supply valid information.

For search area n between $lidx_n$ and $lidx_{n-1}$, where $lidx_0$ is the boundary, e.g., the bottom of Figure 11)

Connect the components inside n and label the components, total labels number is p_n,

Group the p_ns together if the distance between any two of them is less than the separation threshold in case of over-separations,

Else,

Use Mean Frames based Occlusion Detection to check if there is occlusion inside the search area n,

If there is occlusion inside n,

$$\boldsymbol{lidx_{n+1} = \frac{lidx_n + lidx_{reference}}{2}}$$

Else,

$$\boldsymbol{lidx_{reference} = lidx_{n-1}}$$

$$\boldsymbol{lidx_{n+1} = \frac{lidx_n + lidx_{reference}}{2}}$$

If $p_n < P_n - 1$ or search area n is too narrow,

Stop and Return separated labels,

Else,

Search $n + 1$.

3.7. STMap-based Vehicle Counting Combined with Lane-changing Detection

The proposed system effectively reduces the impact on vehicle counting caused by lane-changing. In the proposed system, the strands are only counted on the camera side for three reasons:

a) The camera side has a better resolution of the vehicle.
b) The camera side has a better angle, at which the vehicles usually separate from each other.
c) Only counting one side reduces the possibility of overcounting caused by lane-changing.

For strands that come to camera,

If $max(r_{np_i}) < thrld_{mint} \times L$,

then the strand p belongs to a vehicle that merges out of the current lane, and it should not be counted.

Else,

$count += 1$,

For strands that are away from the camera,

If $min(r_{np_i}) > thrld_{maxt} \times L$, then the strand p belongs to a vehicle that merges in the current lane, and it should not be counted.

Else,

$count += 1$.

Where $max(r_{np_i})$ is the vertical index of the bottommost point in strand p, $min(r_{np_i})$ is the vertical index of the topmost point in strand p, L is the length of the STLine, $thrld_{mint}$ and $thrld_{maxt}$ are two thresholds to determine whether the vehicle belongs to the current lane. In this thesis, 0.3 and 0.7 were used as $thrld_{mint}$ and $thrld_{maxt}$.

4. Experiment Design

4.1. Experiment Scope and Data Source

New Jersey's 511NJ system is an Advanced Traveler Information System that is available by phone or web 24 hours a day, 365 days a year. The information posted to this system is gathered by multiple public agencies, including the New Jersey Department of Transportation, and consists of information including but not limited to travel times, incident and construction information, as well as live traffic video. The 511NJ traffic video streams provide over 450 real-time traffic feeds to the motoring public and include video streams from NJDOT as well as the New Jersey Turnpike Authority. These video streams are generated from permanent traffic cameras installed by both organizations to help monitor traffic conditions and monitor New Jersey's roadways for incidents. These cameras installed along interstates, highways, and arterials are located at key strategic locations determined by Traffic Operations from each agency.

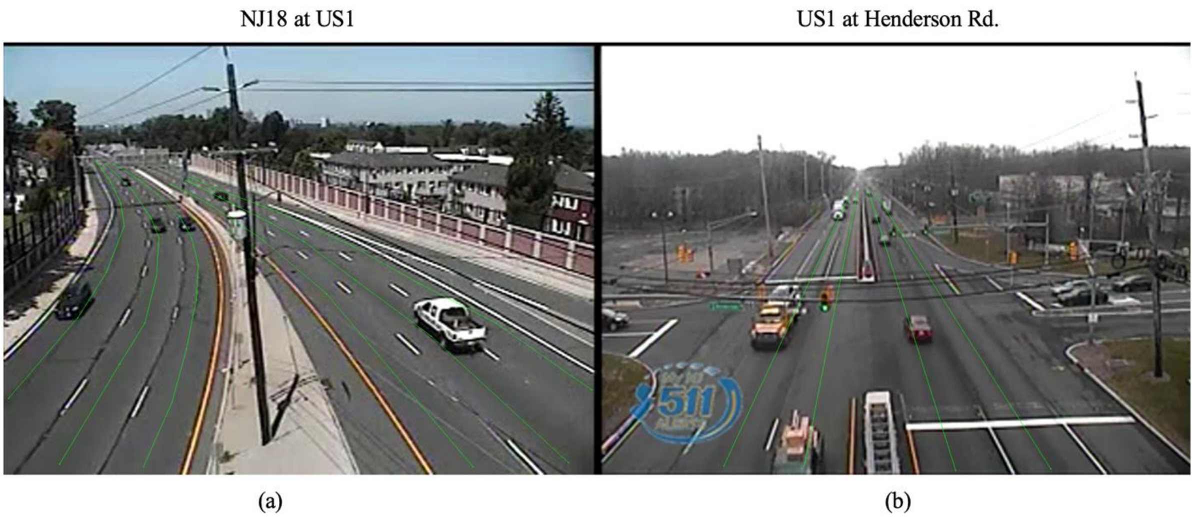


Figure 12 Camera View and STLines

The recorded videos used in this thesis are from the 511NJ system, which has a resolution of 320*240. The 12-min-long video tested in this thesis was captured around 12:05 P.M. on 21st July 2019 from a roadside camera of NJ18 at US1. It was sunny with good lighting.

The other video tested in this thesis was captured around 10:40 A.M. 10th April 2018 from an intersection camera of US1 at Henderson Road.

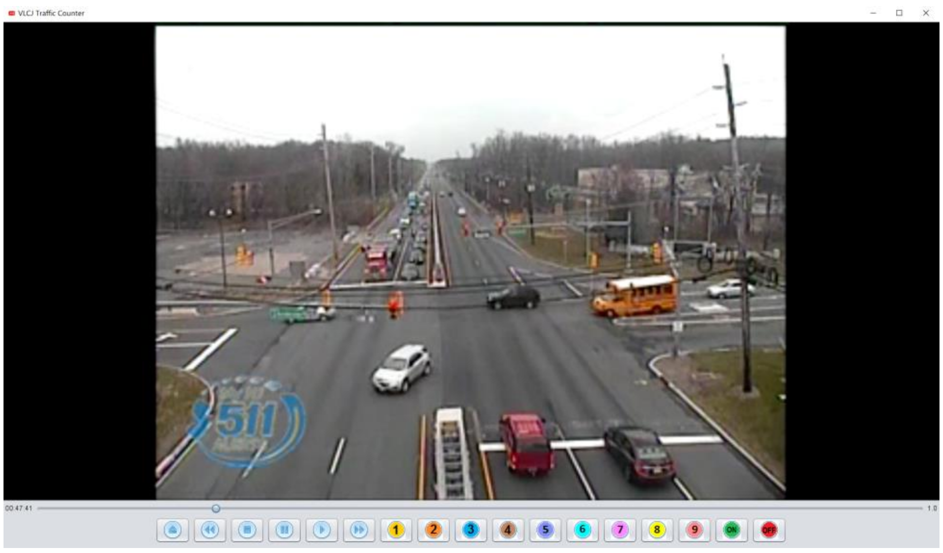


Figure 13 Snapshot of the VLC Traffic Counter for Generating Ground Truth Data

The ground truth data used was obtained through manual counting on traffic videos. The raw ground truth consisted of the time and the lane number of each vehicle passing and was grouped into 1-min-interval or 5-min-interval traffic counts according to demand.

The local test devices include a 15-inch MacBook Pro which has an Intel 8850H@2.6GHz (Max Turbo Frequency 4.3GHz) CPU with 32G RAM, and the system is macOS Mojave 10.14.5, and a Dell rack server which has 2 Xeon E5-2470 v2@2.4GHz (Max Turbo Frequency 3.2GHz) processors and 64GB RAM, with Windows Server 2012 R2 Standard installed. The cloud test server is AWS EC2 t2.micro instance, which has a 2.5GHz vCPU with 1GiB memory, and the system is Amazon Linux AMI release 2018.03. The evaluation was based on 1 min count.

4.2. Evaluation Framework

The evaluation framework used in this thesis includes both accuracy evaluation and power consumption evaluation. The entire evaluation procedure contains three parts, data preparation, traffic video processing module, and evaluation module.

- Data Preparation
  1) Record videos from 511nj.org.
  2) Use VLC Traffic Counter to count vehicles manually.
  3) Generate GoodVision results.
- Traffic Video Processing Module
  4) Mark the STLines for each video location manually using STLine Marker.
  5) Input the recorded videos and the STLine files for processing.
- Evaluation Module
  6) Calculate the cost of cloud deployment, local deployment, and compare with GoodVision cost.
  7) Evaluate the accuracy of each pre-processing module's output, such as direction determination module, static noise removal module, time differencing module, crossing vehicle removal module.
  8) Check the bounding boxes of the generated figures and calculate the detection rate and misdetection rate.
  9) Compare the counting result with ground truth and other algorithms' results.

4.3. Model Evaluation

$$E = \frac{1}{M_{Total}} \sum_{1}^{M_{Total}} (C_{GT} - C_D)$$

$$MAE = \frac{1}{M_{Total}} \sum_{1}^{M_{Total}} |C_{GT} - C_D|$$

$$MAPE = \frac{1}{M_{Total}} \sum_{1}^{M_{Total}} \frac{|C_{GT} - C_D|}{C_{GT}}$$

Where $M_{Total}$is the total minutes, $C_{GT}$is the ground truth count, $C_D$is the detected count. Mean Error is calculated to evaluate the undercounts and overcounts of the proposed algorithm through the whole video. The more it is close to 0, the better the result is. Mean Absolute Error and Mean Absolute Percentage Error is used to evaluate the average performance and stability of the proposed algorithm in every minute. The lower the better. Ground truth vehicle counts are manually counting results that have been grouped by a time interval to reduce the impact of human reaction time.

4.4. Key Parameters

The noise ratio threshold is the threshold parameter for static noise detection. For each row of a STMap, if the average color is different from neighbor rows and the column occupancy of the pixels with different colors is bigger than the noise ratio threshold, then it will be considered a row with static noise. Time frame duration is the parameter to filter small blocks. Patterns that are either longer than this parameter or wider than this parameter will not be filtered. The time difference threshold is the parameter for the time difference detection algorithm. Traditional motion detection algorithms compare different frames of a video, in STMap-based video analytic methods, different columns of a STMap are compared to segment out moving vehicles on STMaps. The lane change threshold is the threshold parameter to avoid overcounting caused by lane changes. The lane changes happen before the threshold will be removed, but the incomplete strands that end before the threshold because of other issues will also be removed.

Table 3 Key parameters

Parameter List	**Descriptions**
Noise ratio threshold	The recommended range is 0.2-0.6 for all lanes.
Time frame duration	The preset value is 15, and the recommended range is 10-20.
Time difference threshold	The preset value is 15, and the recommended range is 10-20.
Lane change threshold	The recommended range for this threshold is between 0.3 and 0.7.

4.5. System Design and Architecture

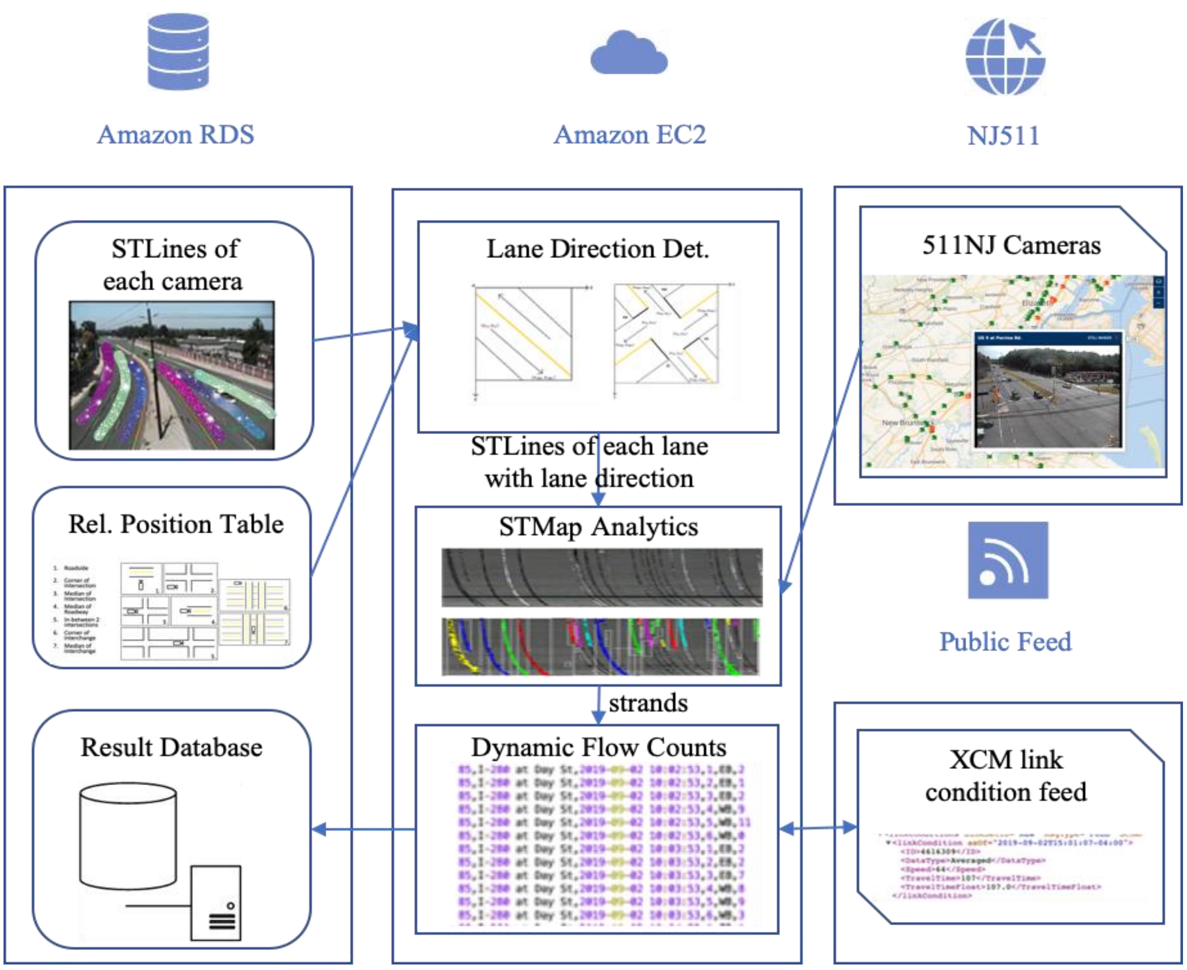


Figure 14 The Proposed System Framework for Cloud-based Video Traffic Counter

Figure 14 shows the design of the proposed cloud-based traffic counter system based on CCTV traffic cameras. The system will be built on two cloud platforms, including a cloud database system (Amazon RDS (Relational Database System)) and a cloud computing system (EC2 (Elastic Compute Cloud)). The system takes the traffic video feed from 511 NJ cameras, generate dynamic traffic flow data that can be added to TRANSCOM (XCM) link condition data feed. Several key system components are summarized as follows.

- **STLine Generation**: The STLines are longitudinal scanlines with which the video analytic algorithms detect and track vehicles. The STLines will be manually processed for all CCTV traffic camera feeds used and need to be periodically updated to reflect changes due to major PTZ operations. The automated algorithm will be developed to readjust the STLines due to minor PTZ operations. The output of the STLine generation modules is detailed lane-by-lane ST scanline geometries with all pixel coordinates of the turning points recorded. The results will be fed into the lane direction determination module and the STMap analytic module.
- **Lane Direction Determination**: The relative positions of cameras are critical in the proposed cloud counter since it is the key information for lane direction detection, which can match the generated flow count with its corresponding direction of the traffic flow. A table containing the camera installation information can be acquired from NJDOT. The output of this module will be used in the lane direction determination module.
- **STMap Analytics**: This is the core video analytic module that generates and analyzes the STMap (Spatial-Temporal Map) from traffic video streams. Every STMap consists of accumulated pixels from the scanline of each lane from continuous video frames. STMap turns the 3-Dimension frame*time into 2-Dimension line*time. Some denoising techniques are introduced to remove the impact of static objects, occlusions, and lane changes. Traffic counts are conducted by tracking the strands generated by vehicles on the STMap. The output of the STMap analytics is vehicle counts for the Dynamic Flow Counting module.

- **Dynamic Flow Counting**: The vehicles can be counted based on the extracted strands from the STMap Analytics module. The detected strands are vehicle parts extracted by the corresponding STLine, one strand for one vehicle. The counts of this module will be stored in Amazon RDS and integrated with the XCM feed.
- **Result Archiving, Exporting, and Integration with XCM Fee**d: The result generated by dynamic flow counting will be stored in the database for long-term storage and other analysis in the future. Meanwhile, the result will also be integrated with the XCM feed and published to the public.

5. Result Analysis

5.1. Computing Evaluation

5.1.1. Computing Time Comparison

HASDA Model took only 91.33s on Intel 8850H@2.6GHz (Max Turbo Frequency 4.3GHz) and 111.54s on AWS t2.micro instance while the proposed algorithm took 151.71s to process the 12-min-long video with 5760 frames using one thread on Intel 8850H under Mac OS and 205.68s to process the video on AWS t2.micro instance. The longer processing time was mostly caused by complex occlusion separation, and new static noise removal methods might also have a slight influence. Although the processing time was longer than the HASDA Model, 205.68s for processing a 12-min-long video was still good enough for large-scale cloud deployment.

5.1.2. Computing Cost Comparison

Considering that the proposed algorithm was designed for daytime processing, the assumed working time was from 5:30 a.m. to 8:00 p.m., 14.5 hours per day, 30 days per month.

AWS Computing Cost Estimation Results: The three-day live test was conducted on AWS over workable cameras on the US-1 corridor between Sep 6th-9th, 2019. 7, 15, and 12 cameras worked for over 90% of the total time, respectively, for those days. The results cost was $5.50, $6.62, and $6.02, respectively, for each day. On average, the cost for each camera is around $0.5 per day.

Local Server Cost Estimation: The tested local server has 2 Xeon E5-2470 v2 processors with 64GB RAM, a CPU occupancy test was conducted, and the result was 2%-5% CPU occupancy and 95MB-115MB RAM utilization for each camera's processing. Based on this result, a server with 2 Xeon E5-2470 v2 processors can process 20-30 cameras, depending on the lane numbers, and the RAM required will be around 32GB. The proposed algorithm can be deployed on rack servers with 2 Xeon Gold 5215 processors (10C20T, Base Frequency at 2.5GHz and Max Turbo Frequency at 3.4GHz) and 32 GB RAM. The configuration is better than the tested server with 2 Xeon E5-2470 v2 processors in order to process at least 25 cameras. Sixteen servers will be needed to handle 400 camera video streams. The price of this rack server on Dell.com is $3885.41 ([83]), which makes the total hardware cost 3812.34×16=$60,997.44.

Different from cloud deployment, local server deployment needs to consider the network, server maintenance, power supply, etc. The bandwidth is 5.9Gbps for 400 video streams at 320*240 resolution and eight frames per second. The power consumption of 16 servers will be 495×16=7920Watts. These will add extra expenses to the total cost. The local server hardware cost plus power fee was calculated by assuming the $0.09/kWh ([84]) power cost.

Based on the above estimation, the estimated cost of AWS cloud computing versus local server for a 400-camera network was calculated and plotted in Figure 15.

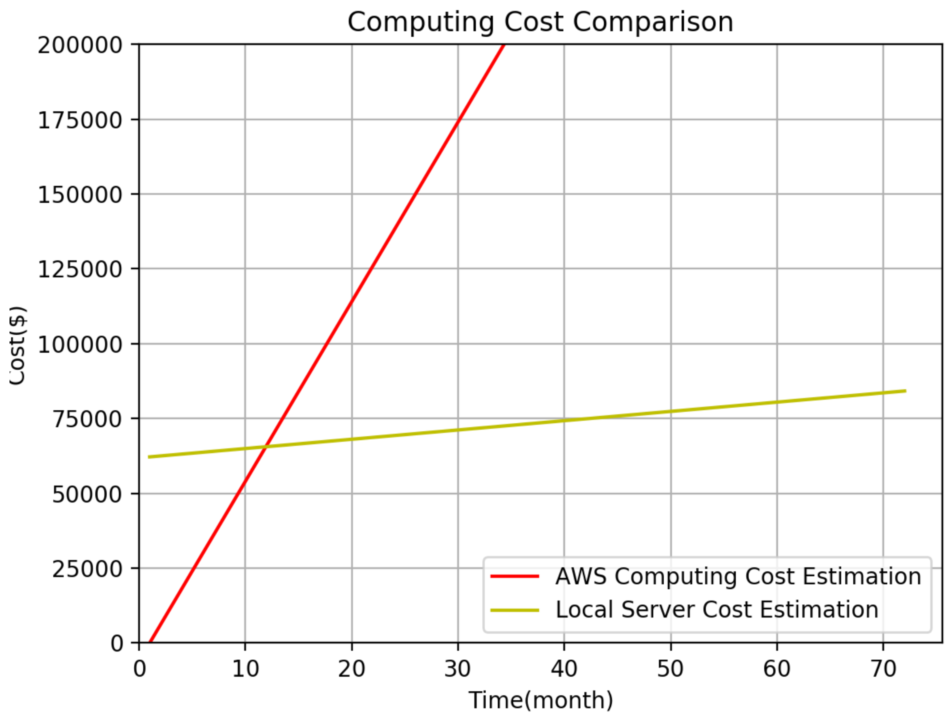


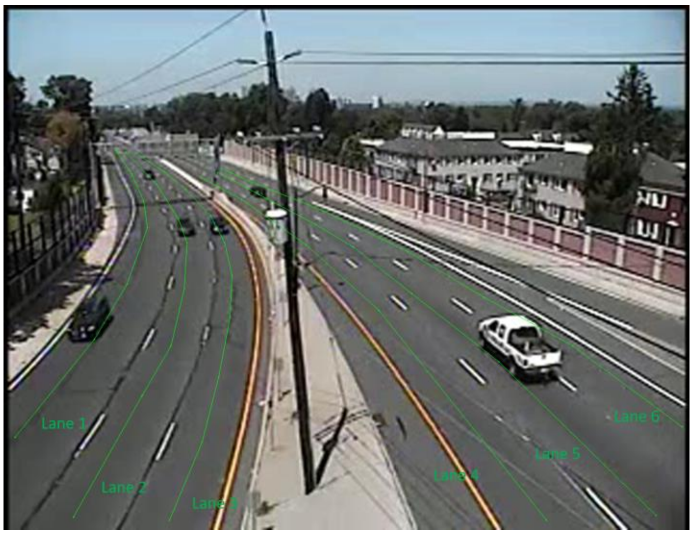
Figure 15 Local and Cloud Cost Comparison (Interpolated for 400 Camera Deployment)

Figure 15 indicates that with the current fee structure for long-term traffic flow monitoring, it is still more cost-effective to deploy dedicated servers for long-term deployment over a large-scale network. However, for on-demand services, especially within a year, the cloud computing platform will have significant savings on top of not worrying about Information Technology procurement, maintenance, and support effort.

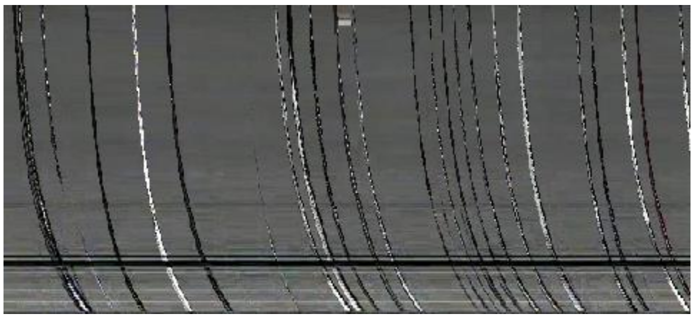
In addition, the charges of Amazon EC2 GPU instances vary from \$0.526/h to \$24.48/h. Comparing to the charges of Amazon EC2 CPU instances, which vary from \$0.0042 to \$6.528, GPU-based deep-learning methods are much more expensive than the proposed CPU-based CCTV traffic video processing method.

5.2. Video Analytic Model Validation Results

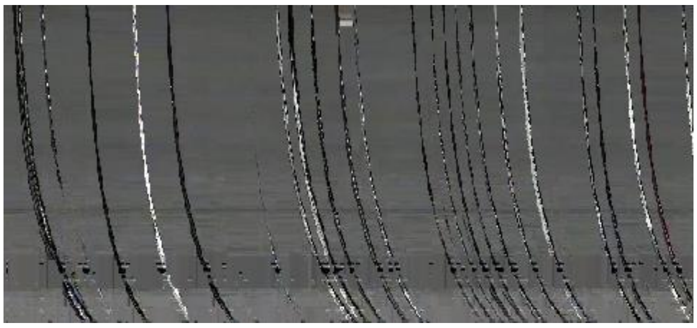
5.2.1. STLine Creation and STMap Generation and Denoising



(a)NJ18 at US1 with STLines



(b)Raw STMap



(c)Static Noise Free STMap

Figure 16 STMap Generation from Traffic Video with Pre-marked STLines

The STLines marked in Figure 16(a) were used to generate the STMap. Figure 16(b) was the STMap of lane 4 between the 4801st frame and 5280th frame from a 12-min-long video recorded from NJ18 at US1. The horizontal axis of a STMap consists of frames, and the vertical axis consists of STLine points. The raw STMap cannot be directly used for vehicle detection before some appropriate denoise processing. It was easy to find that the static noise in Figure 16(b) was caused by the black pole in Figure 16(a). The system does not process the whole camera view but detects static noise in STMap by its color difference and time duration. Figure 16(c) shows the Static-Noise-Free STMap, in which the static noise rows in Figure 16(b) have been detected and filled with the searched clean rows. Most of the trajectory columns in the filled row have been detected and replaced with the row's average color. By comparing Figure 16(a) and Figure 16(b), it is obvious that the static noise has been cleaned effectively, and some of the missing parts of strands has been fixed, which allows the following steps to extract the strands and count vehicles. There are some pixels kept in static noise rows because they have trajectory neighbors.

5.2.2. Strand Clustering Processing Results

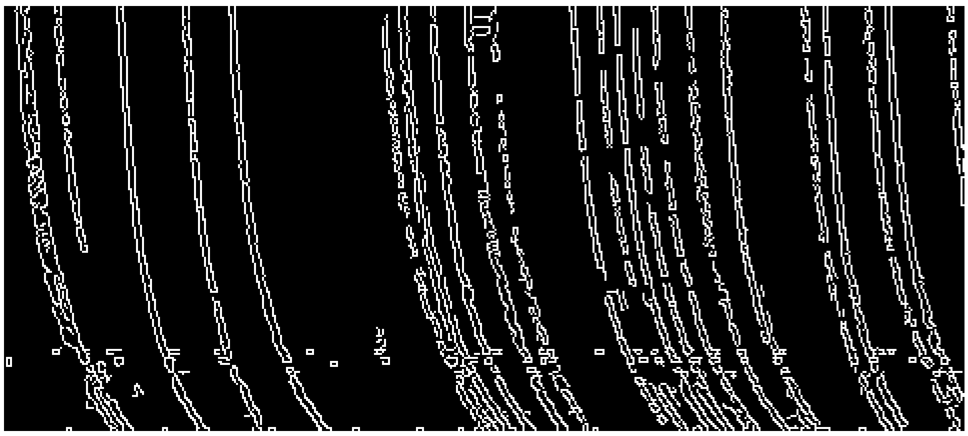
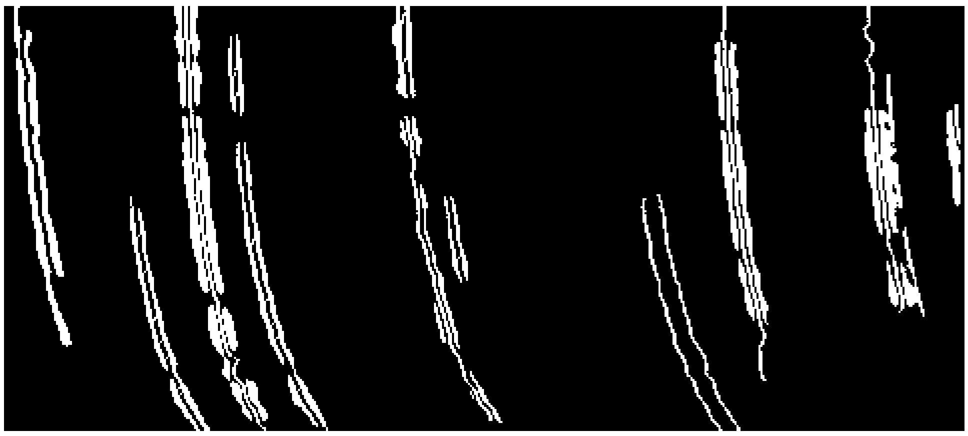
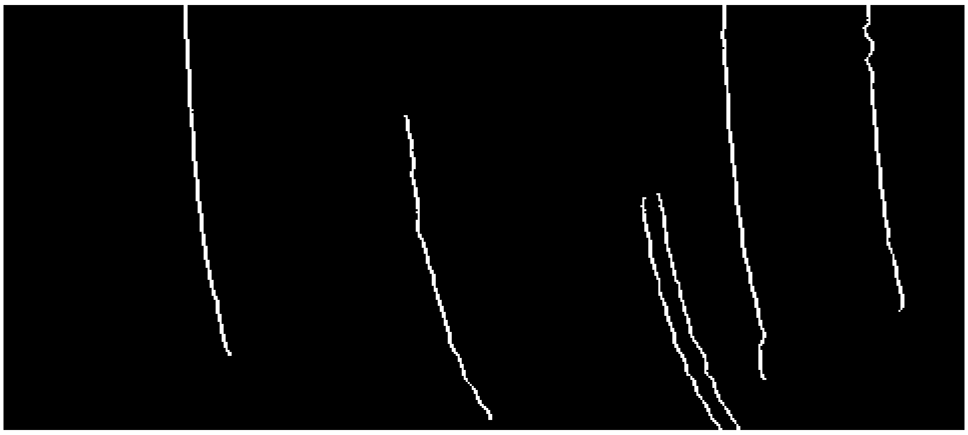


Figure 17 Sample output of Canny Edge Processing Algorithms for the STMap

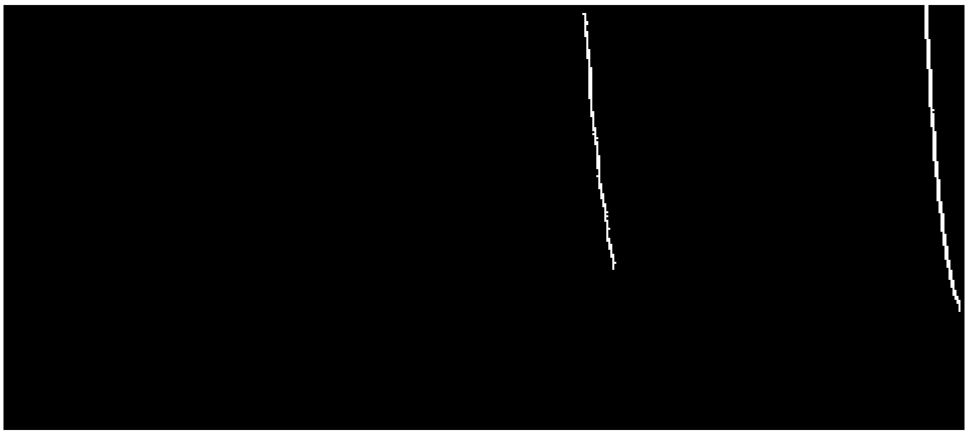
Canny edge detector was used for edge detection to extract strand from denoised STMap (Figure 16(c)). From Figure 17, it’s easy to find that there are ungregarious edges around the detected static noise rows. Those were caused by the pixels kept with potential trajectories determined by their neighbor rows, among which 51/63 were in real strands. Generally speaking, the canny edge detector has decently completed its job and filtered the irrelative background, but further steps are required to separate the connections between strands and extract the strands correctly.



(a)Time difference: Time difference threshold=10



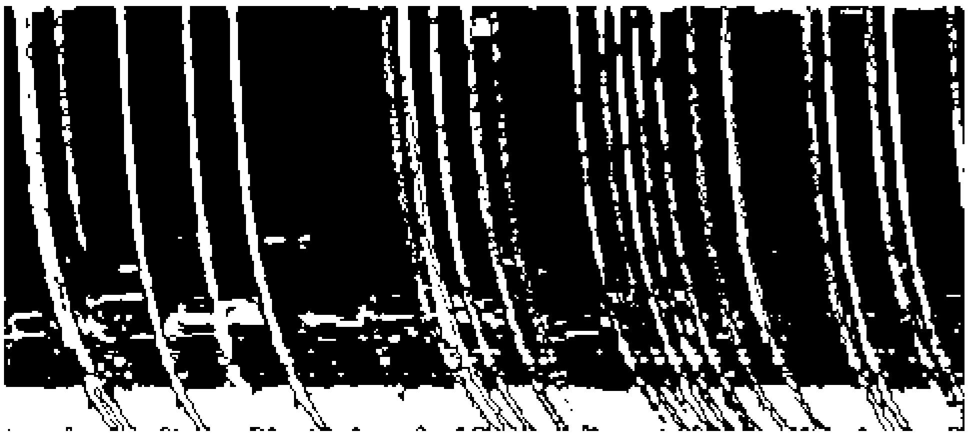
(b)Time difference: Time difference threshold=15



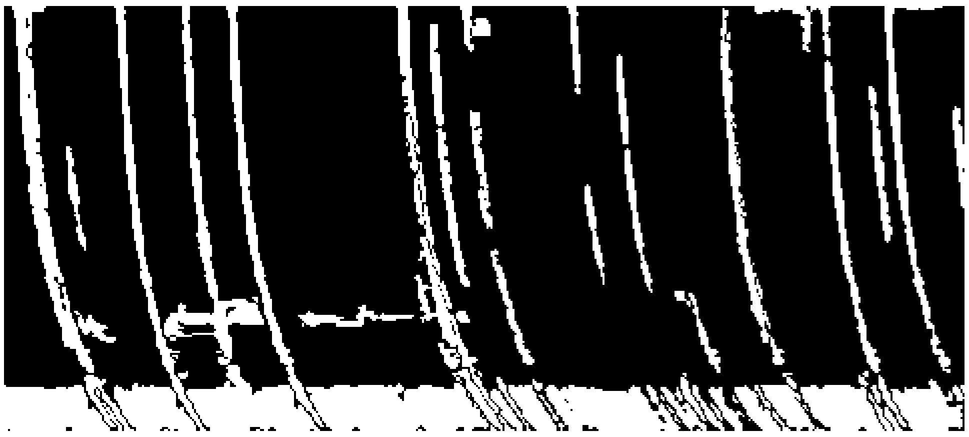
(c)Time difference: Time difference threshold=20

Figure 18 Sample Output of Time Differencing Processing of the STMap

The time-difference-based motion detector was used as a reference for strand extraction. The time difference algorithm is based on the assumption that the change of background is slower than the change between background and the vehicle strands within a time interval. From Figure 18(a)(b)(c), 10 was too small for a threshold, which failed to filter all the noise. Although the one with 15 as threshold only detected 6 strands, the detected strands were all correct, which was acceptable for a supplement. 20 was too big that kept only two strands.



(a) Sample Output of Thresholding



(b) Sample Output of Thresholding After Filtering the Smaller Components

Figure 19 Sample Output of Thresholding

As another source of reference, thresholding was used to separate the roadway background and vehicle strands based on the assumption that they occupy different ranges of the intensity value. It used the denoised STMap as input and used the thresholding method to separate different parts in the STMap. The threshold values were generated automatically using the triangle method, which resulted in the misdetection at the bottom of Figure 19(a). The removal of static noise cut the background apart, which made the triangle method separate two parts of the background too. As Figure 19 shows, there was a lot of small noise detected because their intensity values exceeded the threshold value. Therefore, a size filter was applied to filter the small noise, as Figure 19(b) shows.

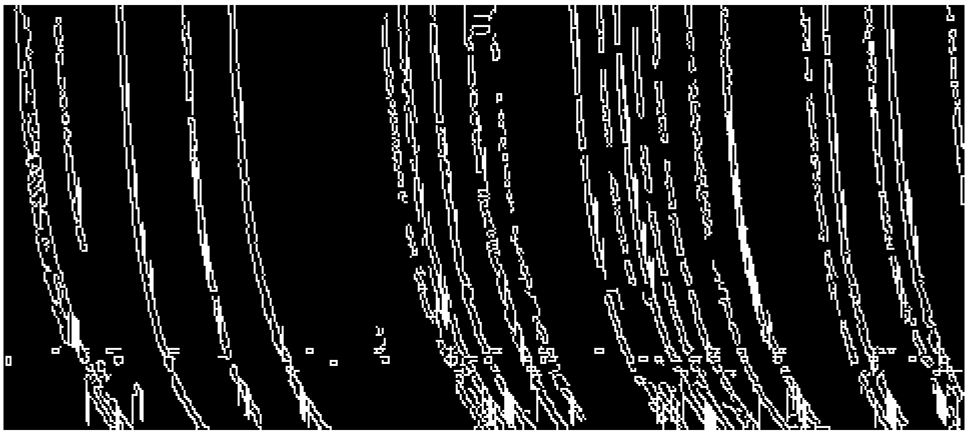


Figure 20 Sample Output of Canny Combined with Time Difference and Threshold

The output of the time difference module and threshold module were then combined together as a mask to process the output of canny edge detection, as Figure 20 shows.

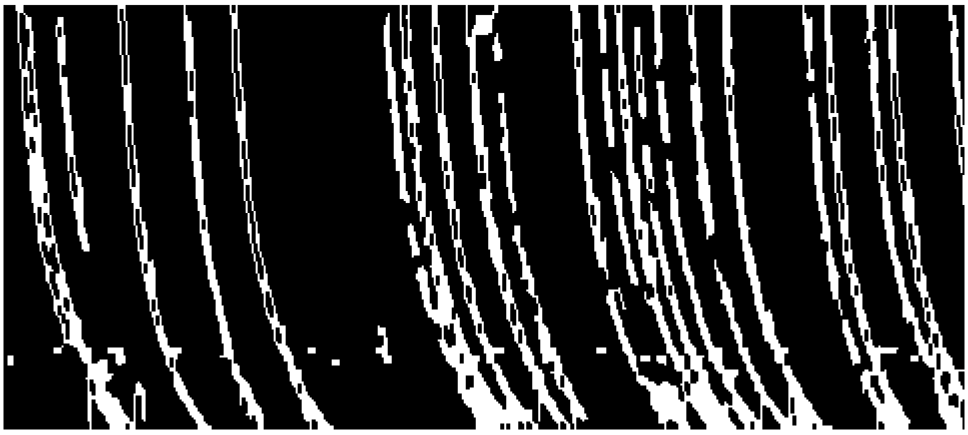


Figure 21 Sample Output of Filled Canny

The edges were not enough for strand extraction. Before strand extraction, the edges have to be filled, as Figure 21 shows. After filling, there was still small noise around the static noise rows, which should be removed and could be removed based on their duration.

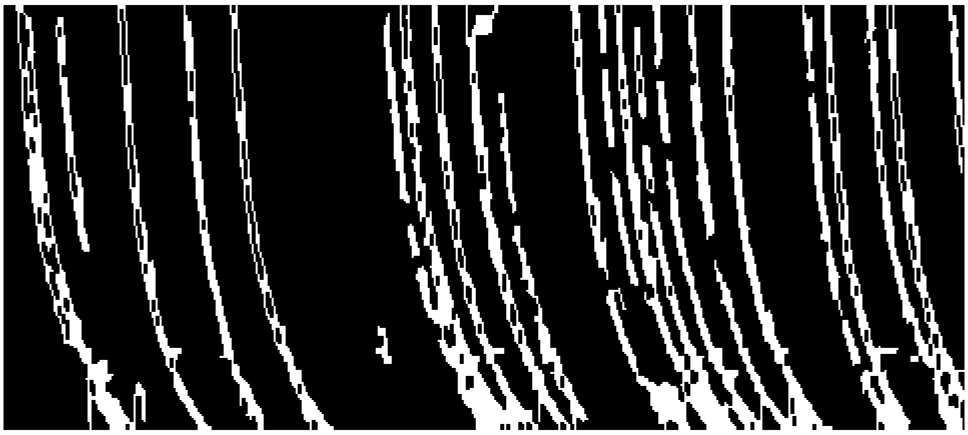


Figure 22 Sample Output after Removing Noise by Duration

A duration-based filter was used to remove the small noise in Figure 22 left by the conservative static noise removal.

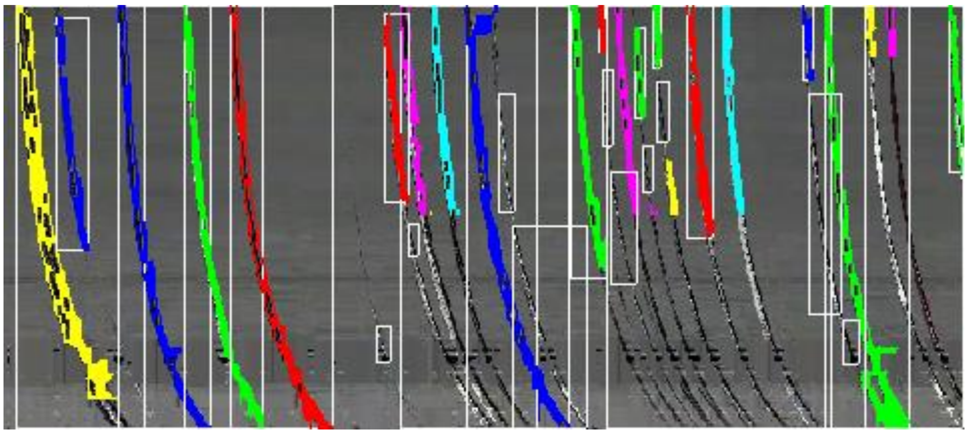
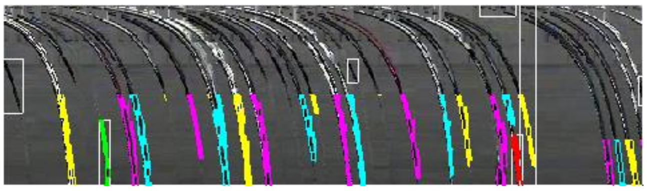
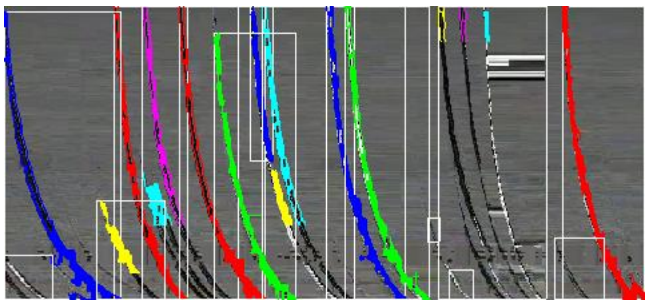


Figure 23 Sample Output of Counted Strands

Figure 23 shows the result of strand detection. Most strands, including the smaller ones caused by lane-changing, were detected successfully. However, there was still over-counting caused by inconsistent strands. Combining with the raw STMap, the separation results from the big change in color, which might be the effect of irregular vehicle movement. Generally, the strand extraction and vehicle detection had decent performances.



(a) Come to camera



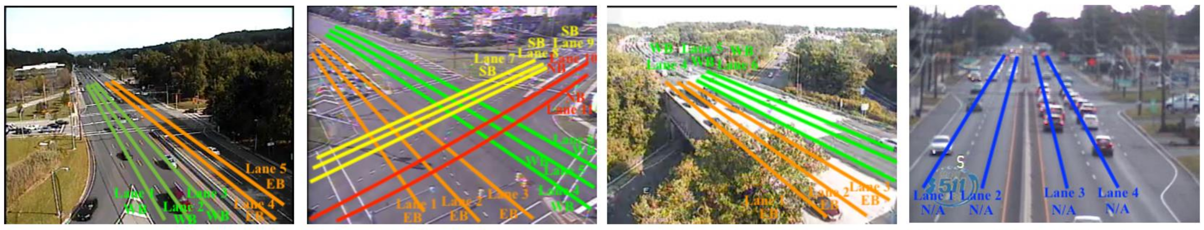
(b) Away from camera

Figure 24 Strand Detection Sample

As the figure above shows, Figure 24(a) is the final STMap of a "come to camera" lane, and the strands' endpoints have been used to count vehicles because the occlusion usually appears at the start. In a "come to camera" lane, lane changes are dealt with by only counting the ones ends in the lane. As can be seen in Figure 24(a), although a lane change threshold of 0.5 was set, there were still lane changes that were overcounted. Figure 24(b) was the final STMap of an "away from camera" lane where the strands' start points were used to count vehicles. Only the vehicles that started from "away from camera" lane should be counted to avoid double counting caused by lane changes.

5.3. Lane Direction Results

Using the Table 2 combined with the camera relative position table, the directions of the lanes in 220 of 444 NJDOT cameras of type 1, 2, 3 could be determined automatically, and the directions of the lanes in 11 cameras of type 4 could be determined easily by checking whether the camera was facing the intersection or not. The rest cameras of type 4 might require other solutions to match lanes manually, such as comparing the features in the camera view with the features in Google Street Map.



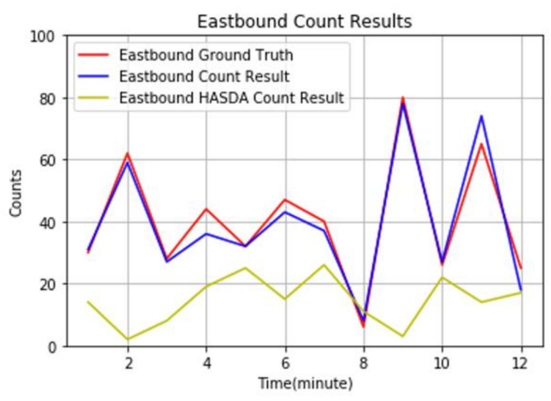
(a) Type 1. NJ 10 at Powdermill Rd. (b) Type 2. NJ 70 at Haddonfield Rd. (c) Type 3. US 30 at I-295 (d) Type 4. US 1 at Bakers Basin Rd.

Figure 25 Lane Direction Determination Sample

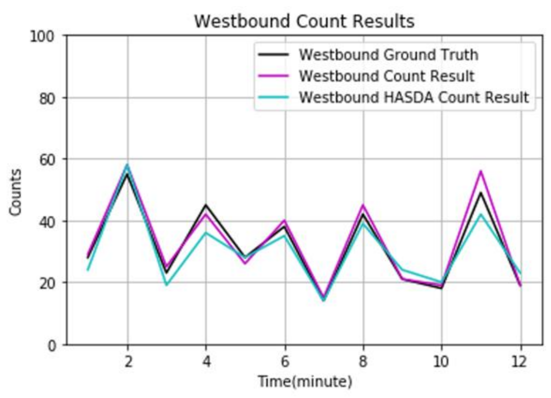
Figure 25 shows some results of Lane Direction Determination. Figure 25(a) is a sample of type 1 cameras that are installed at the roadside. Type 1 cameras that only had two directions of lanes were easy to deal with, and the lanes of different directions were marked in different colors. Figure 25(b) is a sample of type 2 cameras that are installed at the corner of the intersection. The direction of the lanes in type 2 cameras were also determined appropriately. Figure 25(c) is a sample of type 3 cameras, which are installed at the corner of interchanges. Usually, type 3 cameras are similar to type 2 cameras. The only difference is that type 3 cameras may not be able to see the lanes beneath. Figure 25(d) shows a sample of type 4 cameras, which are installed in the median of a roadway. The proposed lane direction determination method was not applicable to symmetry views, which meant that type 4 cameras could not use the proposed method directly, and more features were required for the determination. For example, Figure 25(d) was facing the intersection of US 1 at Bakers Basin Rd., combined with this extra information, the directions of the lanes in Figure 25(d) could be determined manually. By default, the camera should face the intersections, but it still requires the manual checking result of whether it’s facing the intersection or not.

5.4. Traffic Flow Detection Results

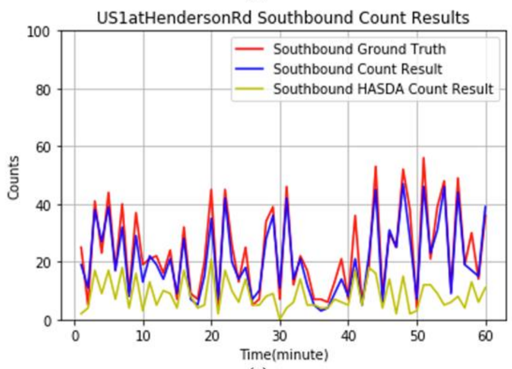
The ground truth and auto count results from both the proposed algorithm and the HASDA model are as follows.



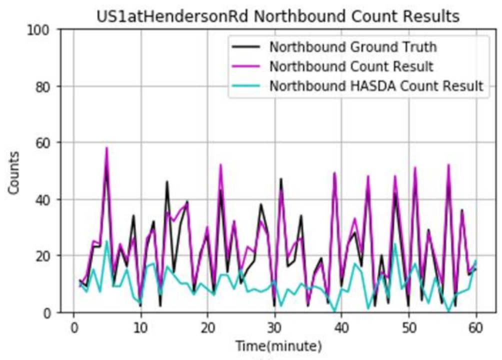
(a)



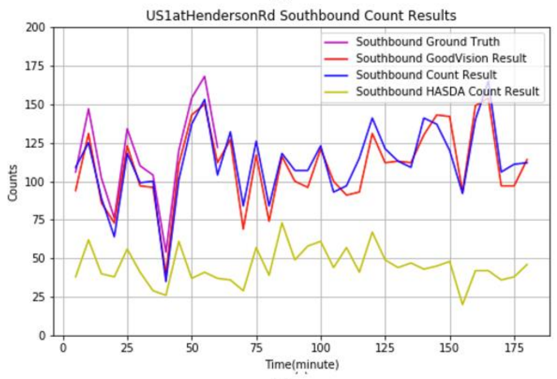
(b)



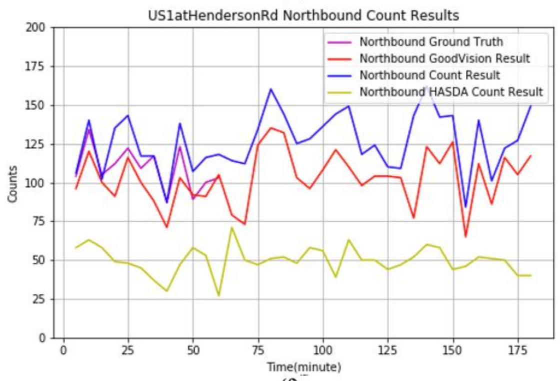
(c)



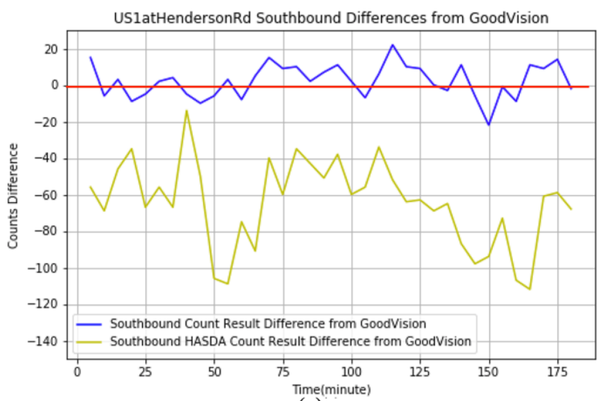
(d)



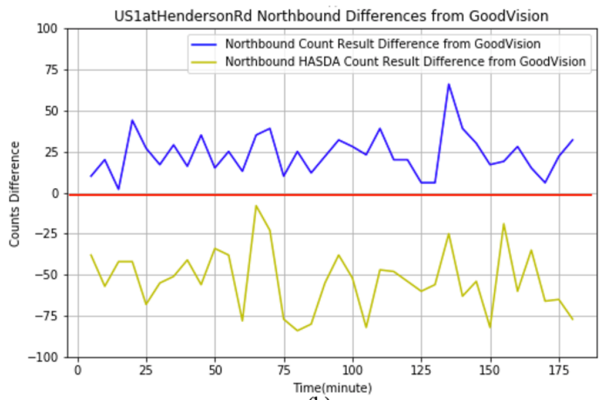
(e)



(f)



(g)



(h)

Figure 26 Count Results

Figure 26(a)(b) show the comparison of Ground Truth, Count Result, and HASDA Count Result of a 12-min-long video captured from NJ 18 at US 1. Ground Truth was generated from a manual count result completed and checked by experienced humans.

From Figure 26(a), which was the comparison of count results in eastbound lanes, HASDA's performance was unsatisfying, especially during those minutes with a large number of vehicles. HASDA is a model designed for an aerial view, in which there is no occlusion. However, occlusions are quite common in traffic camera views.

With occlusion detection and removal modules added, the count result of the proposed system had a better correlation with ground truth. The count results of 1st, 2nd, 3rd, 5th, 8th, 9th, and 10th minute were almost the same, and for the results of other minutes, the biggest error was in the 4th minute, when the manual count result was 44 while the auto count result was 36. The reason for undercounts might be the glitch, which influenced the strand extraction module of the proposed system and would be discussed in the next section. The ME of the eastbound result generated by the proposed system was 1.25, which meant that it had 1.25 counts/min on average in ground truth data. The MAE and MAPE of the eastbound result were 3.416 and 10.62%, which was acceptable considering the motorbikes and trucks that might influence the vehicle detection performance and was much better than 26.58 and 59.06% of HASDA.

From Figure 26(b), HASDA had better performance on westbound than eastbound. As has been mentioned before, HASDA was designed for an aerial view, in which most things were symmetry. However, in the traffic camera view, things were no more symmetry. The ME of the westbound result generated by HASDA was 1.5, which meant that the eastbound

result was 1.5 counts/min in the ground truth. The MAE was 3.5, which reflected that on average, the HASDA's result in eastbound had 3.5 counts/min difference with ground truth. The MAPE was 11.07%, which meant that the vehicle detection has around 89% accuracy. The direction determination module enabled the proposed system to adjust vehicle detection strategies based on the direction of lanes. In a traffic camera view, vehicles were divided into two types based on their movements: Come to Camera and Away from the Camera. The size and resolution of the vehicles in the camera view varied with the distance, which was quite different from the situation that HASDA had been dealing with and led to the performance difference of HASDA in processing Come to Camera and Away from Camera STMaps. The ME of the westbound result generated by the proposed system was -1.25, which meant that it had 1.25 counts/min on average over ground truth. The MAE and MAPE of the westbound result were 2.083 and 5.91%, which meant that it had 2.08 counts difference on average with ground truth and 94% accuracy of vehicle detection at 1-min-level.
Another test was conducted to evaluate the performance of vehicle counting at intersections.

Figure 26(c)(d) show the result of a 1-hour-long video at the intersection of US 1 and Henderson Rd. This time the proposed system was influenced by the frequent appearance of vans, trucks, and crossing vehicles, and the ME rise to 6.7 on southbound and -5.2 on northbound, which meant that every 5 minutes, there were 6.7 undercounts on southbound and 5.2 overcounts on northbound totally. The undercounts in southbound were caused by the vehicles turned-in, which did not get into lanes in time and missed both STLines in neighboring lanes, which will be discussed in the next section. The overcounts in northbound was because the manual count result only counted the vehicles moving out of the intersection, while the proposed system also detected the vehicles joined in from the intersection. There were obvious differences between the manual count result and the auto count result of southbound during the 41st and 58th minute and northbound during the 33rd and 46th minute, which was caused by the red light. In both minutes, there were big waves of vehicles waiting for the green light, which made the occlusion more severe and made lane-changing happen more frequently. The MAE of the proposed system was around 6.79 for southbound and 6 for northbound, which indicated that every 5 minutes, the number of misdetected vehicles was around 6. The MAPE was 11.97% for southbound and 11.92% for northbound, which meant that the accuracy of vehicle detection was around 88% for the intersection. The result of HASDA had 61% MAPE on southbound and 54% MAPE on northbound, which was much worse than the proposed system.

Figure 26(e)(f) show the 5-min-interval result of a 3-hour-long video at the intersection of US 1 and Henderson Rd. The count results of the proposed system and GoodVision had similar counts in the southbound direction, which were 4028 and 3947. Using GoodVision results as ground truth data, the MPE for the southbound results of the proposed system and HASDA were 5.68% and 53.45%, which indicated that the proposed system could generate vehicle counts much more accurately than the HASDA model. For northbound direction, the proposed system had an MPE of 25.2% compared to GoodVision. But comparing both the proposed system’s first-hour result and GoodVision first hour result with manual counting data, the proposed system’s first-hour result was closer to the manual counts.

Figure 26(g)(h) show the difference between the proposed system's result and the GoodVision result, and the difference between HASDA result and GoodVision result, which indicated that the proposed system's result was closer to GoodVision than HASDA.

The proposed algorithm had better performance and stability on videos shot from the roadside camera compared with the HASDA model. HASDA model had trouble dealing with the occlusion caused by the camera angle, which led to the severe undercounts and had been solved in the proposed algorithm. It also had a limitation on removing static noise because of its color-based static noise detection, which led to severe misdetection of vehicles because a pole covered the view of WB lanes. In the proposed algorithm, static noise was detected by horizontal edges rather than the mean value of color and replaced with clean rows nearby.

5.5. Limitations of the Proposed Models

The proposed models still have some limitations that need to be addressed in future work. The main limitations include the limitations with lane direction determination for mid-block cameras, vehicle shadows caused by glitches in video streams, low-angle lane or vehicle occlusions, and scanline readjustment for PTZ operations. The detailed descriptions are as follows.

5.5.1. Lane direction determination for Type 4

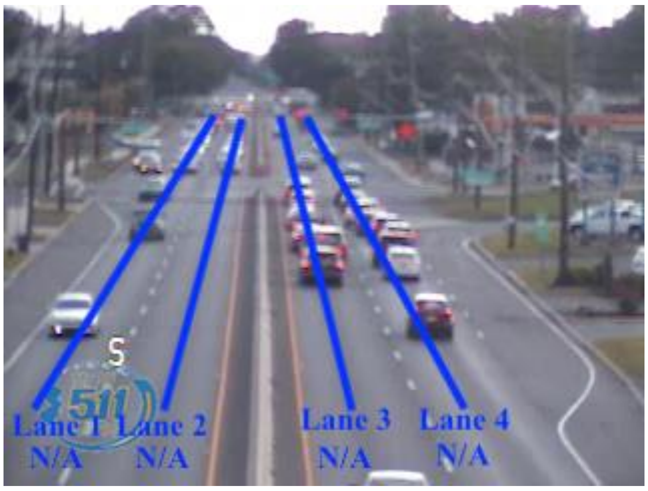
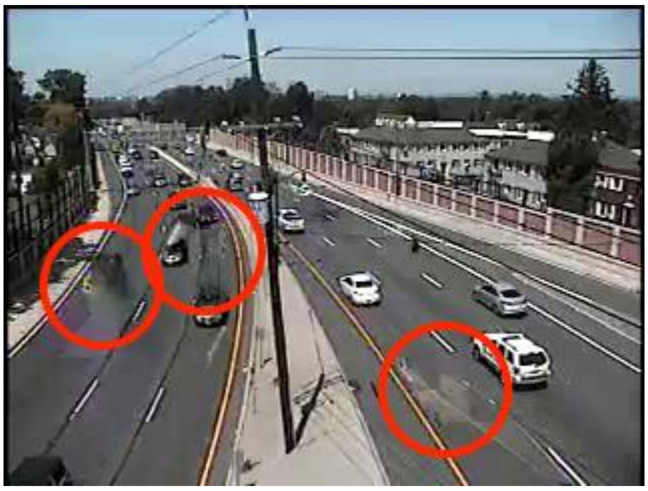


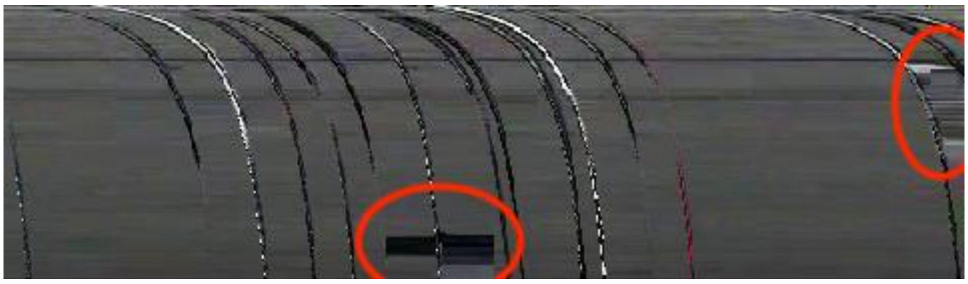
Figure 27 Lane Direction Determination Sample at US 1 at Bakers Basin Rd.

The proposed lane direction determination model was based on the asymmetry of installation and traffic view. However, there were 217 of 482 NJDOT cameras installed symmetrically or did not have the installation information which could not be determined. For example, the camera installed at US 1 at Bakers Basin Rd. was one of the lane-direction-undetermined type 4 cameras. The only solution to match the lanes was to check and compare the features such as intersection, buildings, forests, billboards, etc.

5.5.2. Glitch caused missing information in STMaps



(a)



(b)

Figure 28 Glitch Samples at 4th min in the tested 12-min video

As Figure 28 shows, the glitches resulted from poor bandwidth connected different vehicles together, which resulted in the severe occlusion and could not be separated by analyzing STMaps with missing information.

5.5.3. Vehicle occlusion caused by low angle

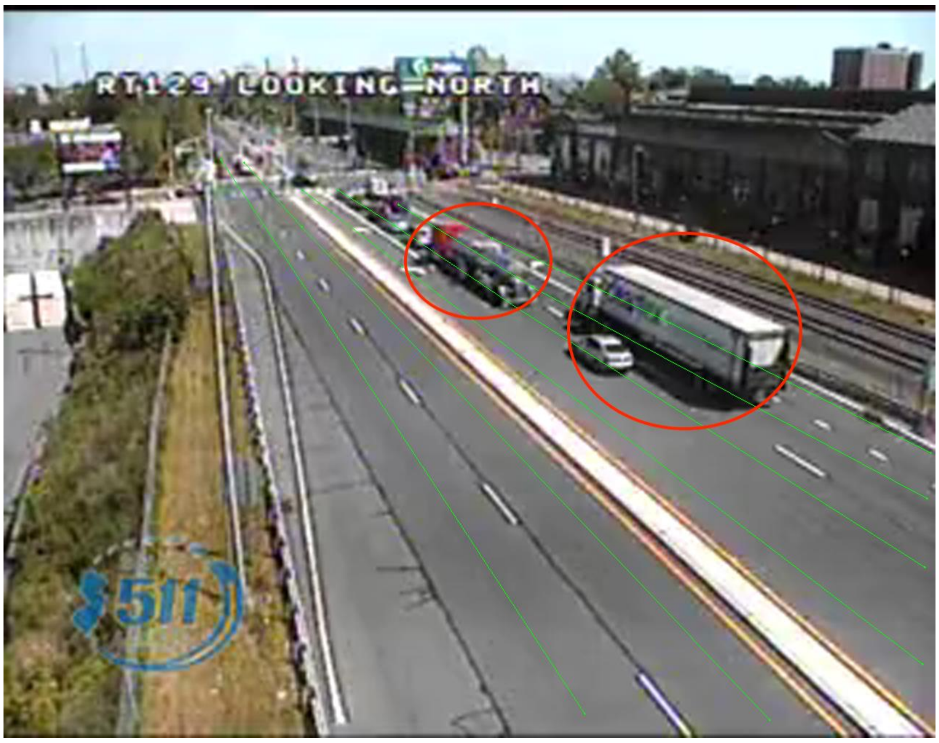


Figure 29 Sample of Occlusion by Large Vehicles in Neighboring Lanes

As has been emphasized, the proposed STLine based vehicle detection method saved a lot of computing resources by only extracting the information from STLines, which also resulted in the loss of information and disabled the ability to look outside the STMaps. In

Figure 29, limited by the loss of information, what the proposed system could process was only the STMap generated from the information that STLines extracted, which prevented it from detecting the vehicles covered by large vehicles in neighboring lanes even if they have shown parts for a while that humans could see and recognize them.

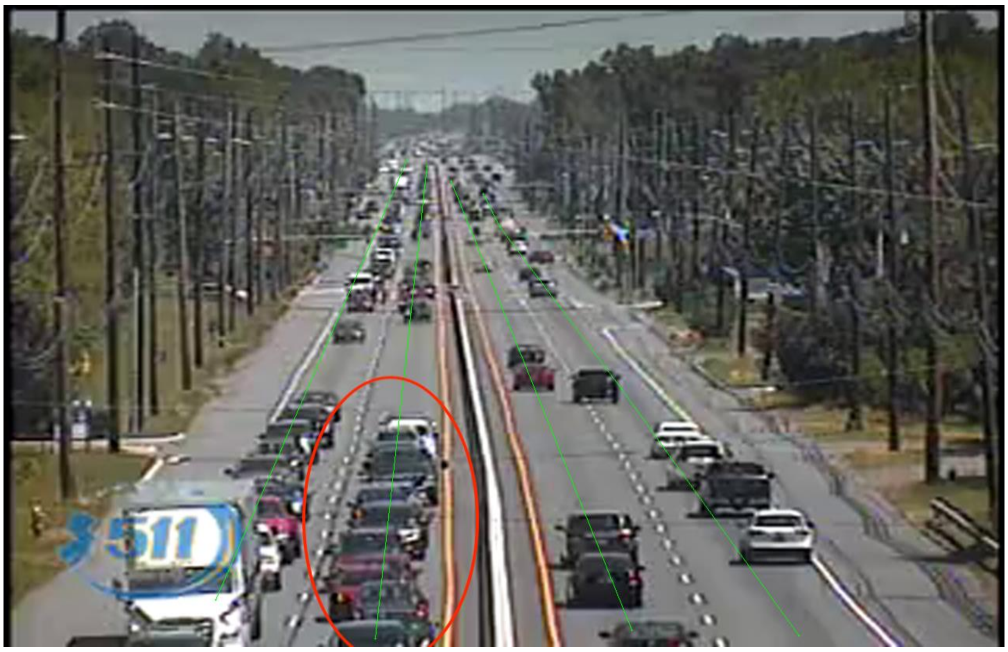


Figure 30 Sample of Occlusion by Vehicles in Same Lane

The proposed occlusion detection model relied on the separation that usually happened when the vehicles came close to the camera. However, if the camera angle was too low, there would not be any separation, which directly made it impossible to deal with occlusions.

5.5.4. PTZ operations caused STLine matching issue

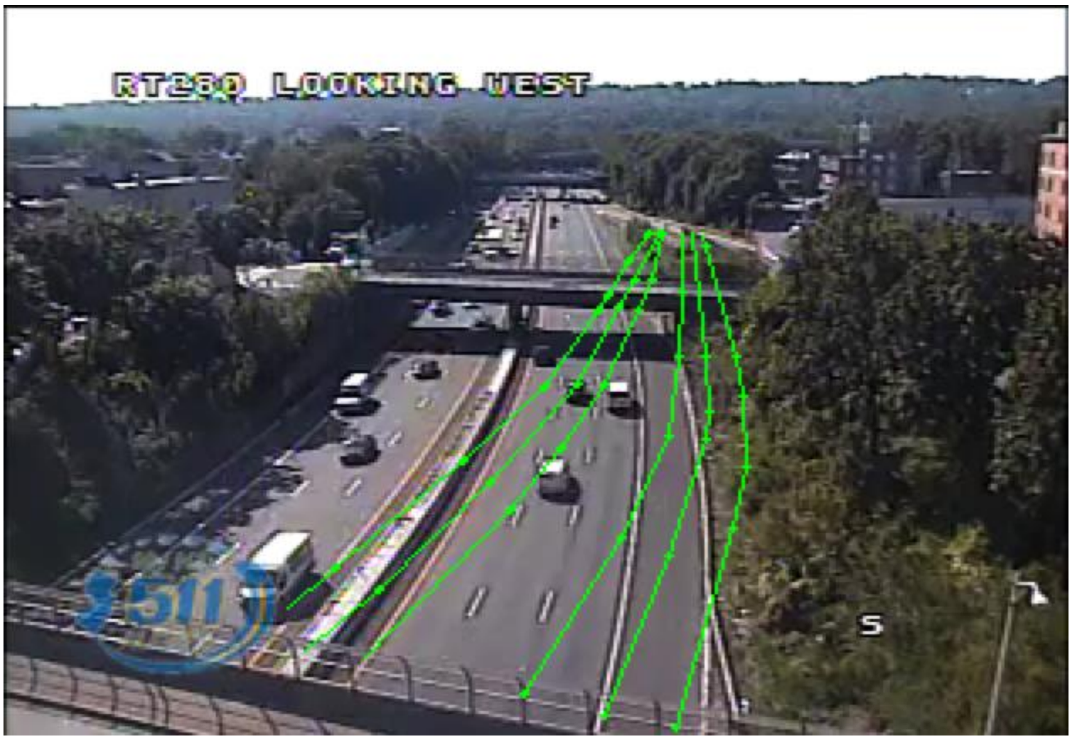


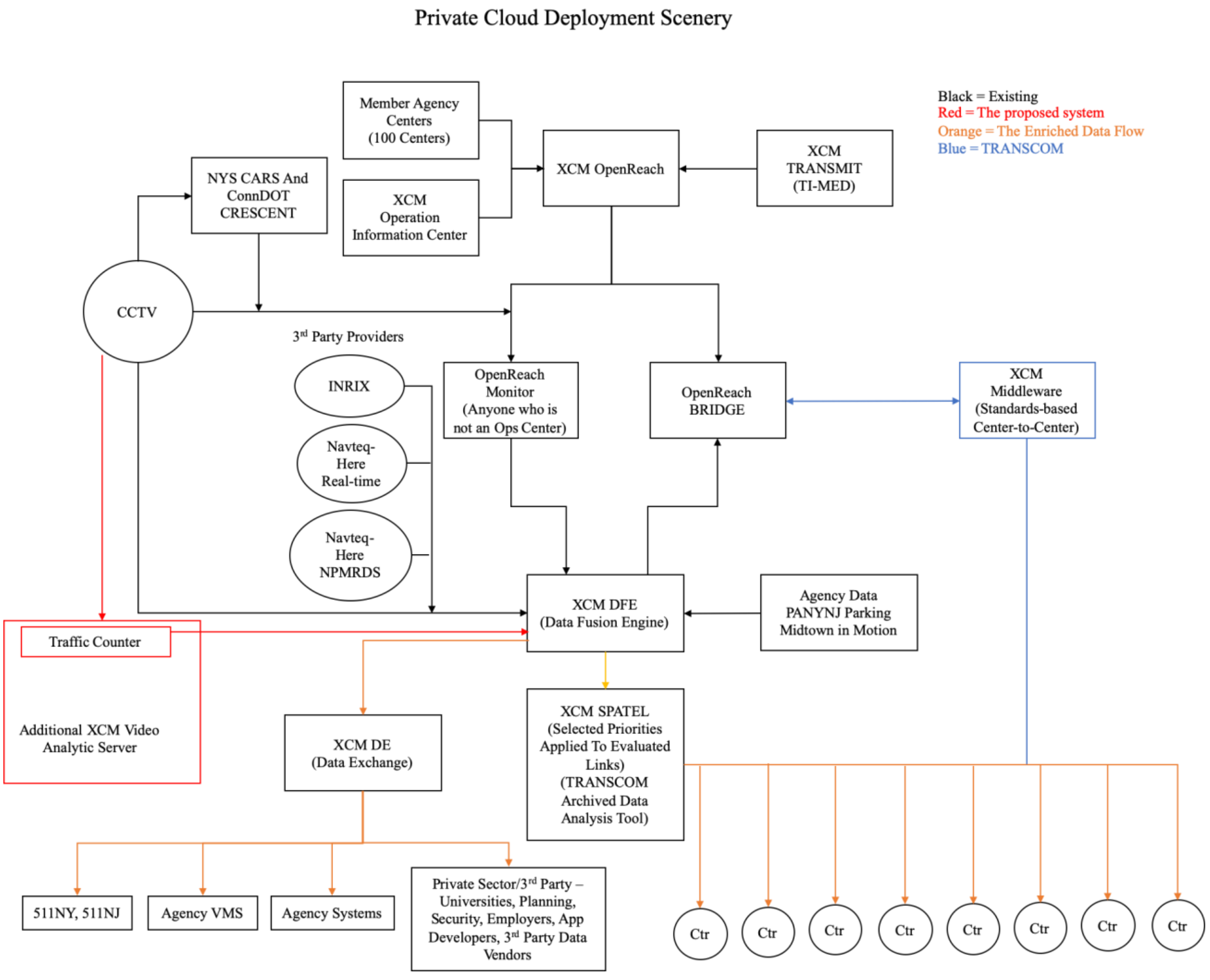
Figure 31 Sample of STLine Shifting Caused by PTZ Operations

Most of the NJDOT cameras supported PTZ operations, which would result in the shift of STLines, as Figure 31 shows. Then the STLines would have to be updated, or they would not be able for vehicle counting.

6. Deploy Recommendation

6.1. Proposed System Deployment Schematics with Existing TRANSCOM Systems

The following figures show the deployment schematics of the proposed cloud-based traffic counter over the existing TRANSCOM (XCM) system (Consensus Systems Technologies, 2015). The traffic counter takes the real-time CCTV video feed either from through 511NJ or the video servers by Traffic Management Center or TRANSCOM. The traffic count will generate traffic flow data with the same roadway link system used by TRANSCOM to be integrated into their data fusion engine (XCM DFE). Then the flow data will be incorporated into the XCM Data Exchange (XCM DE) and XCM SPATEL and archiving systems for agency data archiving and sharing.



Private Cloud Deployment Scenery
Black = Existing
Red = The proposed system
Orange = The Enriched Data Flow
Blue = TRANSCOM
Member Agency Centers (100 Centers)
XCM Operation Information Center
XCM OpenReach
XCM TRANSMIT (TI-MED)
NYS CARS And ConnDOT CRESCENT
CCTV
3rd Party Providers
INRIX
Navteq-Here Real-time
Navteq-Here NPMRDS
OpenReach Monitor (Anyone who is not an Ops Center)
OpenReach BRIDGE
XCM Middleware (Standards-based Center-to-Center)
XCM DFE (Data Fusion Engine)
Agency Data PANYNJ Parking Midtown in Motion
Traffic Counter
Additional XCM Video Analytic Server
XCM DE (Data Exchange)
XCM SPATEL (Selected Priorities Applied To Evaluated Links) (TRANSCOM Archived Data Analysis Tool)
511NY, 511NJ
Agency VMS
Agency Systems
Private Sector/3rd Party – Universities, Planning, Security, Employers, App Developers, 3rd Party Data Vendors
Ctr
Ctr
Ctr
Ctr
Ctr
Ctr
Ctr
Ctr

Figure 32 Private Cloud Deployment Schematics with Existing TRANSCOM Systems

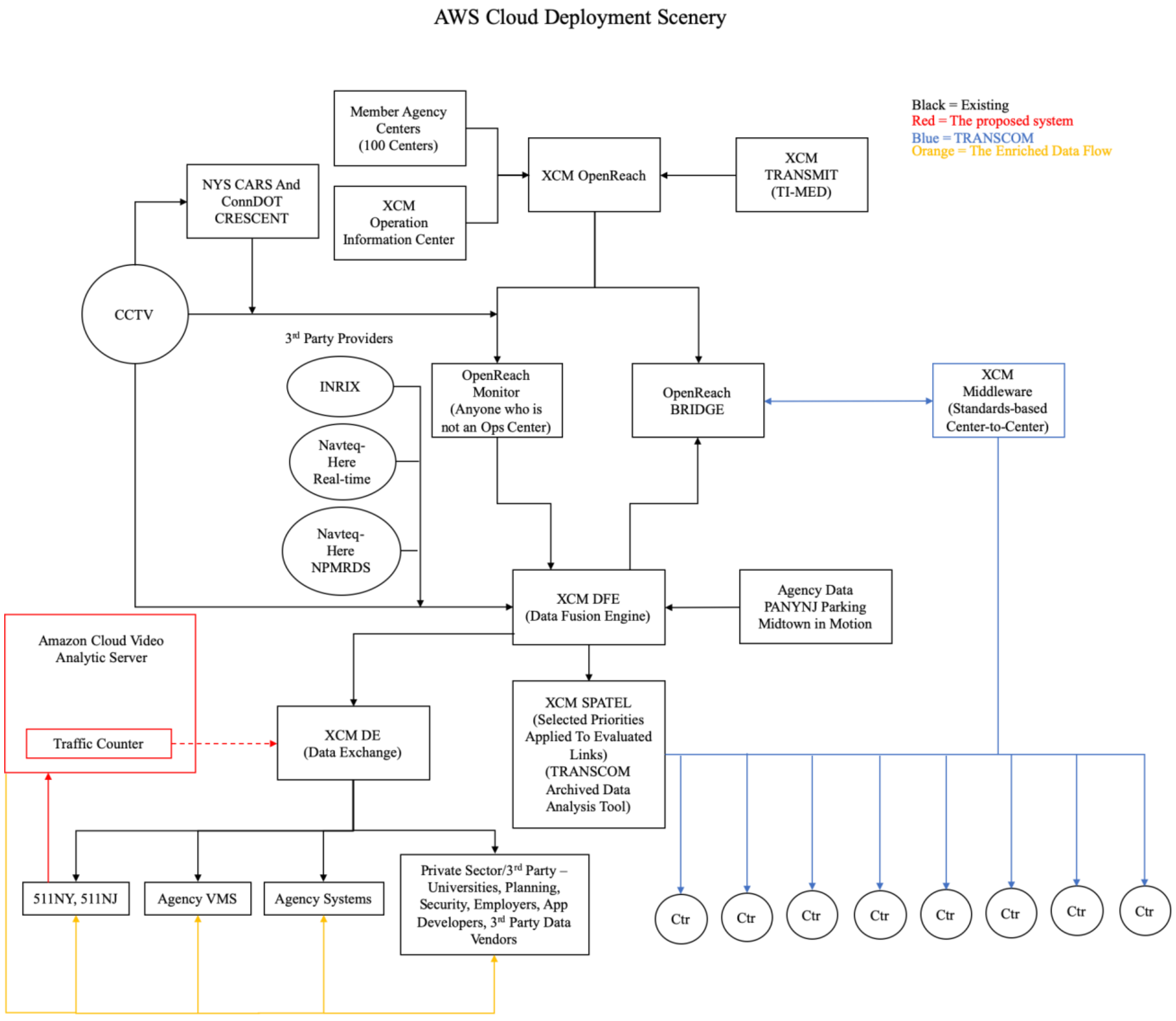

AWS Cloud Deployment Scenery
Black = Existing
Red = The proposed system
Blue = TRANSCOM
Orange = The Enriched Data Flow
Member Agency Centers (100 Centers)
XCM Operation Information Center
XCM OpenReach
XCM TRANSMIT (TI-MED)
NYS CARS And ConnDOT CRESCENT
CCTV
3rd Party Providers
INRIX
Navteq-Here Real-time
Navteq-Here NPMRDS
OpenReach Monitor (Anyone who is not an Ops Center)
OpenReach BRIDGE
XCM Middleware (Standards-based Center-to-Center)
XCM DFE (Data Fusion Engine)
Agency Data PANYNJ Parking Midtown in Motion
Amazon Cloud Video Analytic Server
Traffic Counter
XCM DE (Data Exchange)
XCM SPATEL (Selected Priorities Applied To Evaluated Links) (TRANSCOM Archived Data Analysis Tool)
511NY, 511NJ
Agency VMS
Agency Systems
Private Sector/3rd Party – Universities, Planning, Security, Employers, App Developers, 3rd Party Data Vendors
Ctr

Figure 33 Amazon Cloud Deployment Schematics with Existing TRANSCOM Systems

Traffic counter module can be deployed either on an outside third-party cloud like Amazon Web Services (AWS) or be deployed at a dedicated server system within TRANSCOM or NJ Traffic Management Center. If it is deployed at AWS, there are some additional communication and network configuration to access the video stream and transfer the result.

6.2. Private versus Commercial Cloud Deployment

Deploying the proposed platform can be deployed both in a private cloud in dedicated server clusters within transportation agencies and public cloud services like AWS, Microsoft Azure, or other cloud platforms. The following is a comparison of the pros and cons of using different deployment strategies.

Table 4 Comparison between Commercial Cloud Deployment and Personal Cloud Deployment

	Commercial Cloud	Personal Cloud
Server Types	Cloud Server Instances	Dedicated Servers
Initial Cost	Free	High
Computing Cost	High	Low
Operations/Maintenance	Easy	Complicated
Communication Cost	Free (Usually High)	Low
Storage Cost	Low (Usually High)	Low
Upgrades	Easy and Cheap	Difficult and Expensive
Security	AWS Security	Private
Data Dissemination	Easy and Free (Usually Expensive)	Low

$$C_{Total} = C_{int} + C_{cp} * T_{cp} + C_{tsf} * D_{stg} + C_{hd} * D_{stg}$$

Cloud server instances do not require any initial cost, while the private cloud on dedicated servers do cost a lot to purchase hardware.

- **Initial Cost:** Cloud server instances do not require any initial cost because they charge for the usage, while dedicated servers require an initial cost for server purchasing and deployment.
- **Computing Cost:** The proposed system requires many computational resources, which results in high computing costs for cloud server instances. In the long term,

dedicated servers cost less for computing because only power consumption and appropriate maintenance fee will cost after purchase.

- **Operations/Maintenance:** The operation/maintenance of cloud server instances is much easier than dedicated servers as all it needs is to sign in to the console and click the actions. However, the dedicated servers require manual management, security check, periodically backup, etc.
- **Communication Cost:** Benefit from the high-in-low-out network demand of the proposed video analytic system and the output-charge-only policy of AWS, the communication cost of cloud server instances is almost free, while the dedicated servers require high bandwidth to access real-time traffic videos. The low-resolution video streams take up around 800kbps/camera, which means it requires around 320Mbps of download bandwidth to access 400 cameras.
- **Storage Cost:** The Get-Process-Drop design of the traffic counter module reduces the need for storage. Only the output result will be recorded and stored in the database for long term storage, which makes the storage cost very little.
- **Upgrades:** The upgrades of cloud server instances are quite easy without extra hardware purchase fee. All it needs is just to open new instances with better/newer CPUs. However, dedicated servers require a complete upgrade instead, which will be close to the initial cost.
- **Security:** The data of cloud server instances are protected by AWS security, which is reliable enough. The data of dedicated servers are private.
- **Data Dissemination:** As has been mentioned before, the high-in-low-out network demand of the proposed system fits into the free 1GB/month output data policy, which does not cost extra money. Meanwhile, the low bandwidth demand for uploading data does not cost much either.

7. Conclusions and Future Work

In this research, a cloud-based traffic counting system based on CCTV traffic video streams was proposed, built, deployed, tested, and evaluated. The proposed system improved some of the existing traffic counting algorithms for low-angle CCTV cameras by novel methods to use a fraction of the video frames for analytics (the STLine), efficient processing of static noises caused by roadway infrastructure and signs, and occlusion among vehicles. The streamlined workflow of the proposed platform alleviated the limitation, and the instability of storage and the modularized system design allows for further improvement to be easily deployed in the future. The proposed system is able to completely support the automatic detection of camera directions with three types of roadside and intersection camera location scenarios and the manual processing of the camera directions with other cameras. Compared with traditional video traffic monitoring systems, the proposed high-efficiency STMap-based system can process real-time video with low consumption of computing and publish the result data feed with a slight delay. The detailed contributions are as follows.

- **Video Analytic Models**: To solve the ubiquitous lane matching problem, an installation-asymmetry-based lane direction determination method is proposed to match the lanes in camera view to real lanes. Combining the limited information that STLine extracted with the asymmetry in CCTV traffic camera view, the proposed system simplifies the occlusion problem in finding the separate parts of the occlusion in STMap. An occlusion detection method is proposed based on the assumption that most of the occlusions happen in CCTV camera views are far from the camera and will separate from each other when they are close to the cameras.

An existing-time-based static noise removal method is proposed to help with background removal in STMap.

- **Adapting for NJ CCTV Traffic and Video Data Sources:** The proposed system combined 511NJ traffic video stream, pre-marked STLine coordinates, camera installation details table, and TRANSCOM link condition data feed together, which enabled the whole Get-Process-Drop process of reading real-time CCTV traffic video streams, generating and processing STMaps, sending the counting results to the database, publishing the counting results as feed. The pre-marked STLine coordinates were used to generate STMaps from the video streams. A camera installation detailed table from NJDOT was used to match the cameras and the lanes in camera views with TRANSCOM links. In case that the pre-marked STLines might be ineffective due to PTZ operations, the proposed system also has a local version that can display the video stream and show the pre-marked STLines, which enables the operator to determine whether the STLines are effective and re-mark the STLines if needed.
- **Cloud-based Deployment:** The proposed system, including a video processing module, a database module, and a feed publishing module, was deployed on the cloud using AWS RDS DB instances and AWS EC2 instances. An Amazon Machine Image (AMI) sample containing Amazon Linux with the required software and dependency packages was created for easy duplication of the proposed system and can be shared with Amazon Account ID easily. Daily reboot and auto-re-initiation were set to further reduce the cost and improve the reliability.
- **Computational Cost Reduction:** The proposed real-time traffic counter cost much less than traditional systems in the market as it can be deployed on a normal computer with a CPU and network access, unlike the traditional systems such as

Autoscope, let alone the deep-learning-based traffic monitoring system such as GoodVision. In terms of effectiveness, it also processes existing video much faster than traditional systems.

The proposed system took 151.71s to process the 12-min-long video with 5760 frames using one thread on Intel 8850H under Mac OS and 205.68s to process the video on AWS t2.micro instance. The MAPEs were 5%-10%, which were fine considering the low cost it took and could be improved in the future. When dealing with real-time video, the proposed system took up 2%-5% CPU and 100MB RAM per camera on a server with 2 E52470 v2 Xeon CPUs, which proved that the proposed system was quite suitable for large scale deployment.

- **Promising Detection Performance and Efficiency:** Comparing the generated results with ground truth, the proposed system has better performance dealing with cameras installed on the highway at a medium angle, in which the vehicles get separated from each other. For the occlusion happens at intersections because of red lights, the occlusion detection may not work as expected.

Future work on the proposed platform will be conducted from the following key directions.

- **Automated Detection of Camera Direction:** The proposed methods for camera detection still relies on some knowledge from transportation agencies regarding the relative locations of cameras with respect to the highway or intersections. Future research will focus on the use of existing Google satellite and google street view images to graphically match the video images to determine the precise location of the cameras and traffic directions.
- **Adaptive STLine detection with PTZ operations**: For now, the STLines still need to be marked manually, and every time a Pan-Tilt-Zoom (PTZ) operation is

performed, the STLines have to be re-marked, which is not a smart choice. There are two potential solutions to solve this problem. 1. Since that the STLines are already marked, what needs to be done is to realize the PTZ operation detection, and once a PTZ operation is performed, do the feature-based auto recalibration to adjust the STLines. 2. Since the STLines should be close to the most frequent trajectory of each lane, the STLines can be marked using traditional trajectory detection. However, these two solutions both require video processing for the whole video view, which is not compatible with the purpose of STLine methods: make it compute easily.

- **Occlusion Removal for CCTV Traffic Cameras**: The proposed separation-detection-based occlusion detection is limited by the assumption that the vehicles will separate from each other when they come close to the camera, which requires the camera installed at the medium or high angle. To solve the occlusions that happen near the camera because of large vehicle coverages at low angle cameras or side view angle cameras, the color and size of the strands may need to be considered as reference data to separate occlusion.
- **Lane-change Tracking and Processing**: The proposed system has a simple lane change detection which can only correct the vehicle counts rather than track the vehicles in neighboring lanes. To track the whole process of vehicle lane change behavior, it’s necessary to create the relationship between neighboring lanes and calibrate the coordinates of neighbor STLines.
- **Deep-learning based Video Analytic Models:** Besides, deep-learning-based image analysis may also be added to analyze the STMap to extract the strands, which is different from a deep-learning-based traffic monitoring system because it will only focus on the extracted STMaps rather than the whole video.

- **Potential in Processing Other Data Sources:** The proposed model has potential in significantly reducing the computing resources in processing dynamic 3D point cloud data.

8. Reference

[1] Ardestani, S.M., Jin, P.J. and Feeley, C., 2016. Signal Timing Detection Based on Spatial–Temporal Map Generated from CCTV Surveillance Video. *Transportation Research Record*, *2594*(1), pp.138-147.

[2] Bas, E., Tekalp, A. M., & Salman, F. S. (2007, 06). Automatic Vehicle Counting from Video for Traffic Flow Analysis. 2007 IEEE Intelligent Vehicles Symposium. doi:10.1109/ivs.2007.4290146

[3] Beymer, D., Mclauchlan, P., Coifman, B., & Malik, J. (n.d.). A real-time computer vision system for measuring traffic parameters. Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. doi:10.1109/cvpr.1997.609371

[4] Bharati, P., & Pramanik, A. (2019, 08). Deep Learning Techniques—R-CNN to Mask R-CNN: A Survey. Computational Intelligence in Pattern Recognition Advances in Intelligent Systems and Computing, 657-668. doi:10.1007/978-981-13-9042-5_56

[5] Birchfield, S. T., Sarasua, W. A., & Kanhere, N. K. (2010). Computer vision traffic sensor for fixed and pan-tilt-zoom cameras (No. Highway IDEA Project 140).

[6] Canny, J. (1987). A Computational Approach to Edge Detection. Readings in Computer Vision, 184-203. doi:10.1016/b978-0-08-051581-6.50024-6

[7] Chen, Y., Wu, B., Huang, H., & Fan, C. (2011, 05). A Real-Time Vision System for Nighttime Vehicle Detection and Traffic Surveillance. IEEE Transactions on Industrial Electronics, 58(5), 2030-2044. doi:10.1109/tie.2010.2055771

[8] Cheng, H., Du, H., Hu, L., & Glazier, C. (2005, 01). Vehicle Detection and Classification Using Model-Based and Fuzzy Logic Approaches. Transportation Research Record: Journal of the Transportation Research Board, 1935, 154-162. doi:10.3141/1935-18

[9] Cho, Y., & Rice, J. (2006, 12). Estimating Velocity Fields on a Freeway From Low-Resolution Videos. IEEE Transactions on Intelligent Transportation Systems, 7(4), 463-469. doi:10.1109/tits.2006.883934

[10] Coifman, B., Beymer, D., Mclauchlan, P., & Malik, J. (1998, 08). A real-time computer vision system for vehicle tracking and traffic surveillance. Transportation Research Part C: Emerging Technologies, 6(4), 271-288. doi:10.1016/s0968-090x(98)00019-9

[11] Corovic, A., Ilic, V., Duric, S., Marijan, M., & Pavkovic, B. (2018, 11). The Real-Time Detection of Traffic Participants Using YOLO Algorithm. 2018 26th Telecommunications Forum (TELFOR). doi:10.1109/telfor.2018.8611986

[12] Cucchiara, R., Piccardi, M. and Mello, P., 2000. Image analysis and rule-based reasoning for a traffic monitoring system. *IEEE Transactions on Intelligent Transportation Systems*, *1*(2), pp.119-130.

[13] Dailey, D., Cathey, F., & Pumrin, S. (2000, 06). An algorithm to estimate mean traffic speed using uncalibrated cameras. IEEE Transactions on Intelligent Transportation Systems, 1(2), 98-107. doi:10.1109/6979.880967

[14] Dalal, N., & Triggs, B. (n.d.). Histograms of Oriented Gradients for Human Detection. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). doi:10.1109/cvpr.2005.177

[15] Dong, Z., Wu, Y., Pei, M., & Jia, Y. (2015, 08). Vehicle Type Classification Using a Semisupervised Convolutional Neural Network. IEEE Transactions on Intelligent Transportation Systems, 16(4), 2247-2256. doi:10.1109/tits.2015.2402438

[16] Fischler, M. A., & Bolles, R. C. (1981, 06). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM, 24(6), 381-395. doi:10.1145/358669.358692

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

[18] Gupte, S., Masoud, O., Martin, R., & Papanikolopoulos, N. (2002, 03). Detection and classification of vehicles. IEEE Transactions on Intelligent Transportation Systems, 3(1), 37-47. doi:10.1109/6979.994794

[19] Home. (n.d.). Retrieved from http://www.autoscope.com/

[20] Hsieh, J., Yu, S., Chen, Y., & Hu, W. (2006, 06). Automatic Traffic Surveillance System for Vehicle Tracking and Classification. IEEE Transactions on Intelligent Transportation Systems, 7(2), 175-187. doi:10.1109/tits.2006.874722

[21] Ishak, S. S., Codjoe, J., Mousa, S., Jenkins, S., & Bonnette, J. (2016). Traffic counting using existing video detection cameras. Louisiana Transportation Research Center.

[22] Iwasaki, Y., Takehara, H., Miyata, T., Kuramoto, T., Kitajima, T., & Setoguchi, M. (2018, 12). Automatic Measurement of Road Traffic Volumes and Vehicle Trajectories Using an Object Detection Algorithm YOLO. Proceedings of The 6th Virtual Multidisciplinary Conference. doi:10.18638/quaesti.2018.6.1.387

[23] Jordan, B., Lennon, W., & Holm, B. (1973, 12). An Improved Algorithm for the Generation of Nonparametric Curves. IEEE Transactions on Computers, C-22(12), 1052-1060. doi:10.1109/t-c.1973.223650

[24] Kaewtrakulpong, P., & Bowden, R. (2002). An Improved Adaptive Background Mixture Model for Real-time Tracking with Shadow Detection. Video-Based Surveillance Systems, 135-144. doi:10.1007/978-1-4615-0913-4_11

[25] Kamijo, S., Matsushita, Y., Ikeuchi, K., & Sakauchi, M. (2000, 06). Traffic monitoring and accident detection at intersections. IEEE Transactions on Intelligent Transportation Systems, 1(2), 108-118. doi:10.1109/6979.880968

[26] Kanhere, N. K., Birchfield, S. T., & Sarasua, W. A. (2008, 01). Automatic Camera Calibration Using Pattern Detection for Vision-Based Speed Sensing. Transportation Research Record: Journal of the Transportation Research Board, 2086(1), 30-39. doi:10.3141/2086-04

[27] Kanhere, N. K., Birchfield, S. T., Sarasua, W. A., & Whitney, T. C. (2007, 01). Real-Time Detection and Tracking of Vehicle Base Fronts for Measuring Traffic Counts and Speeds on Highways. Transportation Research Record: Journal of the Transportation Research Board, 1993(1), 155-164. doi:10.3141/1993-21

[28] Kanhere, N., & Birchfield, S. (2008, 03). Real-Time Incremental Segmentation and Tracking of Vehicles at Low Camera Angles Using Stable Features. IEEE Transactions on Intelligent Transportation Systems, 9(1), 148-160. doi:10.1109/tits.2007.911357

[29] Kilger, M. (n.d.). A shadow handler in a video-based real-time traffic monitoring system. [1992] Proceedings IEEE Workshop on Applications of Computer Vision. doi:10.1109/acv.1992.240332

[30] Koller, D., Weber, J., Huang, T., Malik, J., Ogasawara, G., Rao, B., & Russell, S. (n.d.). Towards robust automatic traffic scene analysis in real-time. Proceedings of 1994 33rd IEEE Conference on Decision and Control. doi:10.1109/cdc.1994.411746

[31] Koller, D., Weber, J., & Malik, J. (1994). Robust multiple car tracking with occlusion reasoning. Computer Vision — ECCV '94 Lecture Notes in Computer Science, 189-196. doi:10.1007/3-540-57956-7_22

[32] Laureshyn, A., & Nilsson, M. (2018, 06). How Accurately Can We Measure from Video? Practical Considerations and Enhancements of the Camera Calibration Procedure. Transportation Research Record: Journal of the Transportation Research Board, 2672(43), 24-33. doi:10.1177/0361198118774194

[33] Li, Y., Zhu, F., Ai, Y., & Wang, F. (2007, 06). On Automatic and Dynamic Camera Calibration based on Traffic Visual Surveillance. 2007 IEEE Intelligent Vehicles Symposium. doi:10.1109/ivs.2007.4290140

[34] Lin, J., & Sun, M. (2018, 11). A YOLO-Based Traffic Counting System. 2018 Conference on Technologies and Applications of Artificial Intelligence (TAAI). doi:10.1109/taai.2018.00027

[35] Magee, D. R. (2004, 02). Tracking multiple vehicles using foreground, background and motion models. Image and Vision Computing, 22(2), 143-155. doi:10.1016/s0262-8856(03)00145-8

[36] Malinovskiy, Y., Wu, Y., & Wang, Y. (2009, 01). Video-Based Vehicle Detection and Tracking Using Spatiotemporal Maps. Transportation Research Record: Journal of the Transportation Research Board, 2121(1), 81-89. doi:10.3141/2121-09

[37] Masoud, O., Papanikolopoulos, N., & Kwon, E. (2001, 03). The use of computer vision in monitoring weaving sections. IEEE Transactions on Intelligent Transportation Systems, 2(1), 18-25. doi:10.1109/6979.911082

[38] Michalopoulos, P. (1991, 02). Vehicle detection video through image processing: The Autoscope system. IEEE Transactions on Vehicular Technology, 40(1), 21-29. doi:10.1109/25.69968

[39] Morris, B., & Trivedi, M. (2008, 09). Learning, Modeling, and Classification of Vehicle Track Patterns from Live Video. IEEE Transactions on Intelligent Transportation Systems, 9(3), 425-437. doi:10.1109/tits.2008.922970

[40] Oh, J., Min, J., Kim, M., & Cho, H. (2009, 01). Development of an Automatic Traffic Conflict Detection System Based on Image Tracking Technology. Transportation Research Record: Journal of the Transportation Research Board, 2129(1), 45-54. doi:10.3141/2129-06

[41] Omeroglu, A. N., Kumbasar, N., Oral, E. A., & Ozbek, I. Y. (2019, 04). Mask R-CNN Algoritması ile Hangar Tespiti Hangar Detection with Mask R-CNN Algorithm. 2019 27th Signal Processing and Communications Applications Conference (SIU). doi:10.1109/siu.2019.8806552

[42] Pang, C., Lam, W., & Yung, N. (2007, 09). A Method for Vehicle Count in the Presence of Multiple-Vehicle Occlusions in Traffic Images. IEEE Transactions on Intelligent Transportation Systems, 8(3), 441-459. doi:10.1109/tits.2007.902647

[43] Pang, C., Lam, W., & Yung, N. (2004, 09). A Novel Method for Resolving Vehicle Occlusion in a Monocular Traffic-Image Sequence. IEEE Transactions on Intelligent Transportation Systems, 5(3), 129-141. doi:10.1109/tits.2004.833769

[44] Rad, R., & Jamzad, M. (2005, 07). Real time classification and tracking of multiple vehicles in highways. Pattern Recognition Letters, 26(10), 1597-1607. doi:10.1016/j.patrec.2005.01.010

[45] Rahman, M., Islam, M., Calhoun, J., & Chowdhury, M. (2019, 05). Real-Time Pedestrian Detection Approach with an Efficient Data Communication Bandwidth Strategy. Transportation Research Record: Journal of the Transportation Research Board, 2673(6), 129-139. doi:10.1177/0361198119843255

[46] Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).

[47] Ren, S., He, K., Girshick, R., & Sun, J. (2017, 06). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6), 1137-1149. doi:10.1109/tpami.2016.2577031

[48] Schoepflin, T., & Dailey, D. (2003, 06). Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation. IEEE Transactions on Intelligent Transportation Systems, 4(2), 90-98. doi:10.1109/tits.2003.821213

[49] Schoepflin, T. N., & Dailey, D. J. (2003, 01). Correlation Technique for Estimating Traffic Speed from Cameras. Transportation Research Record: Journal of the Transportation Research Board, 1855(1), 66-73. doi:10.3141/1855-08

[50] Schoepflin, T. N., & Dailey, D. J. (2004, 01). Cross-Correlation Tracking Technique for Extracting Speed from Cameras Under Adverse Conditions. Transportation Research Record: Journal of the Transportation Research Board, 1867(1), 36-45. doi:10.3141/1867-05

[51] Song, K., & Tai, J. (2006, 10). Dynamic Calibration of Pan–Tilt–Zoom Cameras for Traffic Monitoring. IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), 36(5), 1091-1103. doi:10.1109/tsmcb.2006.872271

[52] Stauffer, C., & Grimson, W. (n.d.). Adaptive background mixture models for real-time tracking. Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149). doi:10.1109/cvpr.1999.784637

[53] Stauffer, C., & Grimson, W. (2000). Learning patterns of activity using real-time tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(8), 747-757. doi:10.1109/34.868677

[54] Tai, J., Tseng, S., Lin, C., & Song, K. (2004, 06). Real-time image tracking for automatic traffic monitoring and enforcement applications. Image and Vision Computing, 22(6), 485-501. doi:10.1016/j.imavis.2003.12.001

[55] Taniguchi, H., Nakamura, T. and Furusawa, H., 1999, October. Methods of traffic flow measurement using spatio-temporal image. In *Proceedings 1999 International Conference on Image Processing (Cat. 99CH36348)* (Vol. 4, pp. 16-20). IEEE.

[56] Unzueta, L., Nieto, M., Cortes, A., Barandiaran, J., Otaegui, O., & Sanchez, P. (2012, 06). Adaptive Multicue Background Subtraction for Robust Vehicle Counting and Classification. IEEE Transactions on Intelligent Transportation Systems, 13(2), 527-540. doi:10.1109/tits.2011.2174358
[57] Veeraraghavan, H., Masoud, O., & Papanikolopoulos, N. (2003, 06). Computer vision algorithms for intersection monitoring. IEEE Transactions on Intelligent Transportation Systems, 4(2), 78-89. doi:10.1109/tits.2003.821212
[58] Zhang, B. (2013, 03). Reliable Classification of Vehicle Types Based on Cascade Classifier Ensembles. IEEE Transactions on Intelligent Transportation Systems, 14(1), 322-332. doi:10.1109/tits.2012.2213814
[59] Zhang, G., Avery, R. P., & Wang, Y. (2007, 01). Video-Based Vehicle Detection and Classification System for Real-Time Traffic Data Collection Using Uncalibrated Video Cameras. Transportation Research Record: Journal of the Transportation Research Board, 1993(1), 138-147. doi:10.3141/1993-19
[60] Zhang T, Guo M, Jin PJ, Ge Y, Gong J. Longitudinal-Scanline-based Arterial Traffic Video Analytics with Coordinate Transformation Assisted by 3D Infrastructure Data, Presented at 99th Annual Meeting of the Transportation Research Board, Washington, D.C., 2020
[61] Zhang, T., & Jin, P. J. (2019, 06). A longitudinal scanline based vehicle trajectory reconstruction method for high-angle traffic video. Transportation Research Part C: Emerging Technologies, 103, 104-128. doi:10.1016/j.trc.2019.03.015
[62] Zhang, W., Wu, Q., Yang, X., & Fang, X. (2008, 03). Multilevel Framework to Detect and Handle Vehicle Occlusion. IEEE Transactions on Intelligent Transportation Systems, 9(1), 161-174. doi:10.1109/tits.2008.915647
[63] Zhang, Z., Tan, T., Huang, K., & Wang, Y. (2013, 03). Practical Camera Calibration From Moving Objects for Traffic Scene Surveillance. IEEE Transactions on Circuits and Systems for Video Technology, 23(3), 518-533. doi:10.1109/tcsvt.2012.2210670
[64] Zhao, R., & Wang, X. (2013, 06). Counting Vehicles from Semantic Regions. IEEE Transactions on Intelligent Transportation Systems, 14(2), 1016-1022. doi:10.1109/tits.2013.2248001
[65] Zheng, Y., & Peng, S. (2014, 04). A Practical Roadside Camera Calibration Method Based on Least Squares Optimization. IEEE Transactions on Intelligent Transportation Systems, 15(2), 831-843. doi:10.1109/tits.2013.2288353
[66] Zhou, J., Gao, D., & Zhang, D. (2007, 01). Moving Vehicle Detection for Automatic Traffic Monitoring. IEEE Transactions on Vehicular Technology, 56(1), 51-59. doi:10.1109/tvt.2006.883735
[67] Zhu, Z., Xu, G., Yang, B., Shi, D., & Lin, X. (2000, 07). VISATRAM: A real-time vision system for automatic traffic monitoring. Image and Vision Computing, 18(10), 781-794. doi:10.1016/s0262-8856(99)00046-3
[68] Autoscope. Image Sensing Systems, Inc. www.autoscope.com
[69] Citilog, www.citilog.com
[70] TrafficVision, www.trafficvision.com
[71] GRIDSMART, www.gridsmart.com
[72] AgentVi, www.agentvi.com
[73] intruVision, www.intuvisiontech.com
[74] SVS, www.smartcctvltd.com
[75] Placemeter, www.placemeter.com
[76] Traficon, www.traficon.com
[77] MetroTech, https://metrotech-net.com
[78] GoodVision, https://goodvisionlive.com
[79] Miovision, https://miovision.com
[80] Genetec, https://www.genetec.com/solutions/all-products/omnicast/kiwivision-video-analytics
[81] Aventura, https://www.aventurasecurity.com
[82] TRANSCOM System Concept of Operations, http://www.infosenseglobal.com/wp-content/uploads/2018/04/transcom-systems.pdf

[83] Dell. *PowerEdge R440 Rack Server Price*. https://www.dell.com/en-us/work/shop/cty/pdp/spd/poweredge-r440/pe_r440_12423_vi_vp?configurationid=b8c2d6d0-847d-4e6d-a52b-bd2d0a4d2ba0. Accessed July 31, 2020
[84] ElectricRate. *New Jersey Residential Electric Rates.* https://www.electricrate.com/residential-rates/new-jersey. Accessed July 31, 2020