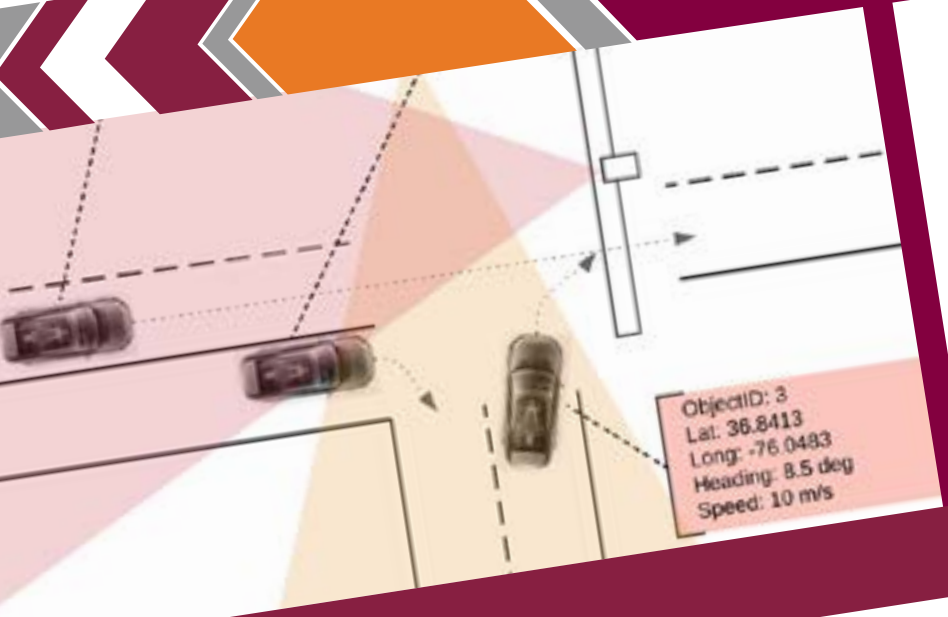
Development of an Infrastructure-Based Data Acquisition System to Naturalistically Collect the Roadway Environment

December 2023 | Final Report



SAFE-D
SAFETY THROUGH DISRUPTION

Texas A&M Transportation Institute

San Diego State University
Leadership Starts Here

VIRGINIA TECH TRANSPORTATION INSTITUTE
VIRGINIA TECH

Disclaimer

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated in the interest of information exchange. The report is funded, partially or entirely, by a grant from the U.S. Department of Transportation's University Transportation Centers Program. However, the U.S. Government assumes no liability for the contents or use thereof.

TECHNICAL REPORT DOCUMENTATION PAGE

1. Report No. 04-121	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Development of an Infrastructure Based Data Acquisition System to Naturalistically Collect the Roadway Environment		5. Report Date December 2023 6. Performing Organization Code:
7. Author(s) Abhijit Sarkar, Ioannis Papakis, Eileen Herbers, Reginald Viray		8. Performing Organization Report No.
9. Performing Organization Name and Address: Safe-D National UTC Virginia Tech Transportation Institute 3500 Transportation Research Plaza Blacksburg, VA 24061		10. Work Unit No. 11. Contract or Grant No. 69A3551747115/[Project 04-121]
12. Sponsoring Agency Name and Address Office of the Secretary of Transportation (OST) U.S. Department of Transportation (US DOT)		13. Type of Report and Period Final Research Report 01/2019-12/2023 14. Sponsoring Agency Code
15. Supplementary Notes This project was funded by the Safety through Disruption (Safe-D) National University Transportation Center, a grant from the U.S. Department of Transportation – Office of the Assistant Secretary for Research and Technology, University Transportation Centers Program.		
16. Abstract: Automatic traffic monitoring is becoming an important investment for transportation specialists, especially as the overall volume of traffic continues to increase, as do crashes at intersections. Infrastructure cameras can be a good source of information for automatic monitoring of traffic situations at intersections. However, efficient computer vision methods that can process the video data effectively are required for this endeavor. Intersection cameras often record video data that are of low quality and low frame rate, making them challenging to use. In this project, we have demonstrated how traffic cameras can be used to automatically track roadway agents, find their kinematic behavior, and devise a safety measurement strategy, leveraging recent advancements in computer vision and deep learning. In this process, we have specifically focused on the Virginia Beach area and used publicly available traffic data to demonstrate our results. We developed a full computer vision pipeline that trains a custom object detector specifically using traffic data. We also used an optical flow method and a graph neural network to improve the accuracy of object tracking. The tracked objects from the image frames were further used as a point source and mapped to their GPS locations. Finally, the speed of each object was calculated to understand the traffic dynamics and determine possible crash predictors. This information can be used to quickly alert traffic control operators to a specific intersection that likely needs their attention so that crashes can be mitigated.		
17. Key Words Computer Vision, Road safety, Traffic monitoring, Object detection, Object tracking	18. Distribution Statement No restrictions. This document is available to the public through the Safe-D National UTC website, as well as the following repositories: VTechWorks, The National Transportation Library, The Transportation Library, Volpe National Transportation Systems Center, Federal Highway Administration Research Library, and the National Technical Reports Library.	

19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 36	22. Price $0

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

Abstract

Automatic traffic monitoring is becoming an important investment for transportation specialists, especially as the overall volume of traffic continues to increase, as do crashes at intersections. Infrastructure cameras can be a good source of information for automatic monitoring of traffic situations at intersections. However, efficient computer vision methods that can process the video data effectively are required for this endeavor. Intersection cameras often record video data that are of low quality and low frame rate, making them challenging to use. In this project, we have demonstrated how traffic cameras can be used to automatically track roadway agents, find their kinematic behavior, and devise a safety measurement strategy, leveraging recent advancements in computer vision and deep learning. In this process, we have specifically focused on the Virginia Beach area and used publicly available traffic data to demonstrate our results. We developed a full computer vision pipeline that trains a custom object detector specifically using traffic data. We also used an optical flow method and a graph neural network to improve the accuracy of object tracking. The tracked objects from the image frames were further used as a point source and mapped to their GPS locations. Finally, the speed of each object was calculated to understand the traffic dynamics and determine possible crash predictors. This information can be used to quickly alert traffic control operators to a specific intersection that likely needs their attention so that crashes can be mitigated.

Acknowledgements

This project was funded by the Safety through Disruption (Safe-D) National University Transportation Center, a grant from the U.S. Department of Transportation – Office of the Assistant Secretary for Research and Technology, University Transportation Centers Program. The team wants to thank the Virginia Beach authorities for their help and collaboration. The team would also like to thank Thomas Gorman for his suggestions and help in building the image to GPS transformation and calculation of vehicle kinematics. The team would also like to thank the subject matter experts for their reviews and suggestions.

Table of Contents

List of Figures

List of Tables

Introduction

Year after year, roadways in the City of Virginia Beach have consistently ranked in the top 10 crash cluster locations in the state of Virginia (TREDS (Traffic Records Electronic Data System), n.d.). Many of these crashes occur at signal-controlled intersections along Virginia Beach Boulevard, which runs parallel to I-264, the only major interstate in the city. This corridor controls in- and out-flow from the interstate to major residential suburbs. Intersections often involve junctions of more than three lanes, representing major arterials that serve a diverse range of roadway users such as military personnel, tourists, and local citizens. In addition to vehicular safety issues, the Virginia Department of Transportation's (VDOT's) Pedestrian Action Plan [1] has identified Pacific Avenue, a major oceanfront resort corridor, as a priority pedestrian crash cluster location (VDOT, 2018). This resort area represents a number of different types of vulnerable roadway users, notably during summer months, where pedestrians, bicyclists, and emerging e-scooters present a high risk of safety conflicts with vehicles.

The city, like many municipalities, has invested a significant amount of resources in digitizing their transportation system infrastructure. Such is evident at signal-controlled intersections where myriad sensors, such as vision-based cameras, thermal-based cameras, radars, loop detectors, Wi-Fi detectors, Bluetooth detectors and, in some cases, wireless dedicated short-range communication road side units, are integrated. However, such assets are used primarily by cities to adapt light controller Signal Phase and Timing based on traffic flow rather than on safety countermeasures. Further, although the city has invested in deploying such systems, they are limited in the amount of resources and technical capabilities available to disseminate the breadth of data collected to address safety and mobility problems.

This project is specifically concerned with the traffic camera recordings and understanding how they can be leveraged to provide the city with insight and suggested countermeasures to address safety issues on these roadways. Traffic cameras are often used for traffic monitoring, but the scope of the review and the monitoring process is largely restricted to manual mode, where an operator needs to review the videos of interest. The project explores the opportunity of automatic traffic monitoring systems using artificial intelligence.

Interest in automatic traffic monitoring has grown due to the revolution in computer vision and deep neural networks over the last decade. Vehicle behavior at intersections and on highway ramps and vehicle interaction with vulnerable road users have always been a major concern for the U.S. Department of Transportation. With increasing traffic volume (including ride sharing and delivery services), and impending deployment of autonomous vehicles in the coming years, the need for automatic monitoring is greater than ever.

Traffic Surveillance Using Cameras and Associated Challenges

Monocular RGB cameras are often useful for monitoring vehicles and pedestrians. In recent years, a number of efforts have tried to address challenges in automated surveillance technology [2, 3]. Figure 1 shows an ideal surveillance method situation: the camera would detect the position and dynamics of each roadway object, including cars, trucks, and pedestrians. In general, it is difficult to execute proper object detection and tracking within complex scenarios. With an unconstrained interaction between roadway agents, most computer vision algorithms struggle to associate several objects from one frame to the next frame. The severity of this problem increases with increasing traffic volume. Additionally, a study by Ananthanarayanan and Zhang found that tracking algorithms are very costly in terms of their processing time [2, 4].

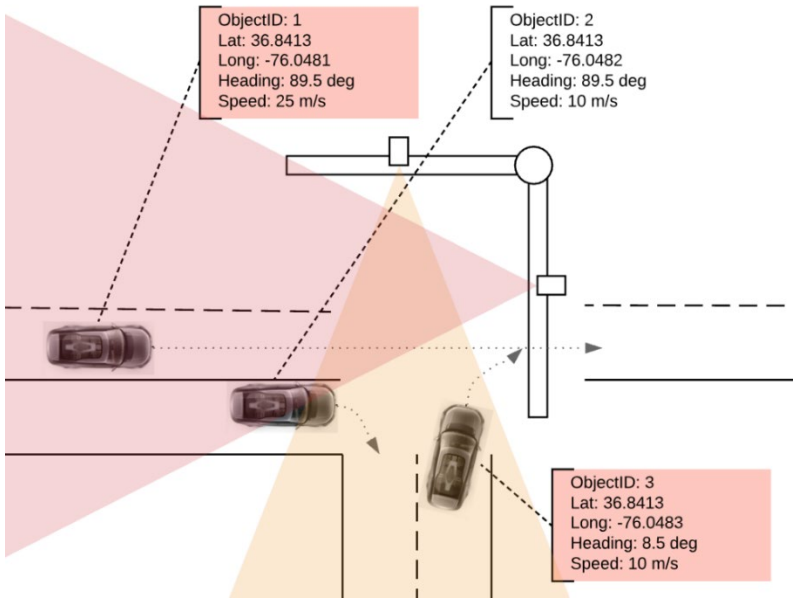


Figure 1. Schematic of an ideal traffic monitoring system. In Figure 1, the system should identify all objects individually and preserve their identity for the duration of their appearance while computing their exact GPS location with speed and heading in relation to the roadway scene.

Most previous work in this area has used high quality videos from private vendors with high spatial and temporal resolution. However, most public traffic cameras have low quality (lower than VGA quality), low frame rate (5 fps–15 fps), high compression artifacts, and low signal to noise ratio. Figure 2 shows some examples of these camera outputs. The video quality specifically deteriorates during nighttime with high spatial noise. Also, due to the impending infrastructural cost, it is not expected that these camera networks will be updated in the near future. But we cannot deny the potential of these cameras in capturing useful information. Therefore, we need to explore the effectiveness of the current infrastructure and consider how modern technology may allow us to better take advantage of it. Additionally, recent research shows that using transfer learning, deep neural networks can be tuned to process low resolution videos [5-12] including those of the vehicle, driver's face, work zone objects, and roadway agents.

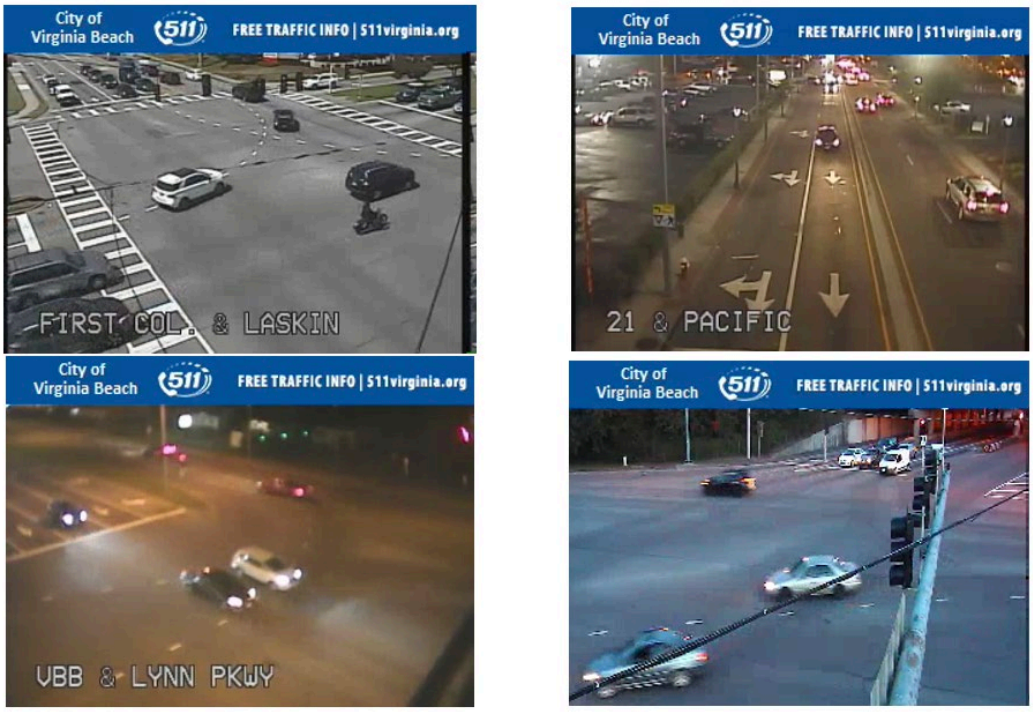


Figure 2. Example images from the 511 traffic cameras.

The cameras used to take the videos shown in Figure 2 often vary by distance and camera angle. The appearance of the videos taken during the night versus the day are also quite different.

In this work, we have investigated how to use the already established traffic camera system to inform traffic operation centers of potential conflicts. This work includes a simple mapping algorithm to estimate the kinematic behavior of traffic to identify crashes or areas that that may pose a traffic safety risk.

Scope of the Project

This project aims to use current intersection traffic camera video feeds to detect, track, and calculate the kinematics of each vehicle that goes through the intersection. These kinematic calculations are then used, along with crash precursor surrogate measures, to detect potential conflicts in real-time. This process is broken down into the following five procedures:

1. Object detection
    a. We specifically focused on pedestrian tracking.
2. Multi-object tracking
3. Image pixel coordinate to GPS transformation
4. Kinematic calculation
5. Conflict identification

The object detection and tracking algorithms are validated by comparing them to many different algorithms currently in use, as well as comparing calculated pixel values to manually tracked pixel locations. The GPS and kinematic calculations were compared observationally to the given traffic

video feed, but needed further validation to compare with a test-vehicle traveling in the specified intersection equipped with an accurate GPS and speed recording device. The conflict identification is a proposed application that could be used to predict traffic conflicts in real time to notify traffic control operators of potential crashes and conflicts. Known surrogate measures were used to validate any conflict identification, though this requires some crash and safety analysis of each intersection to be used effectively.[1] We specifically focused on the nine intersections shown in Figure 3. These are the traffic intersections at the aforementioned Pacific Avenue corridor.

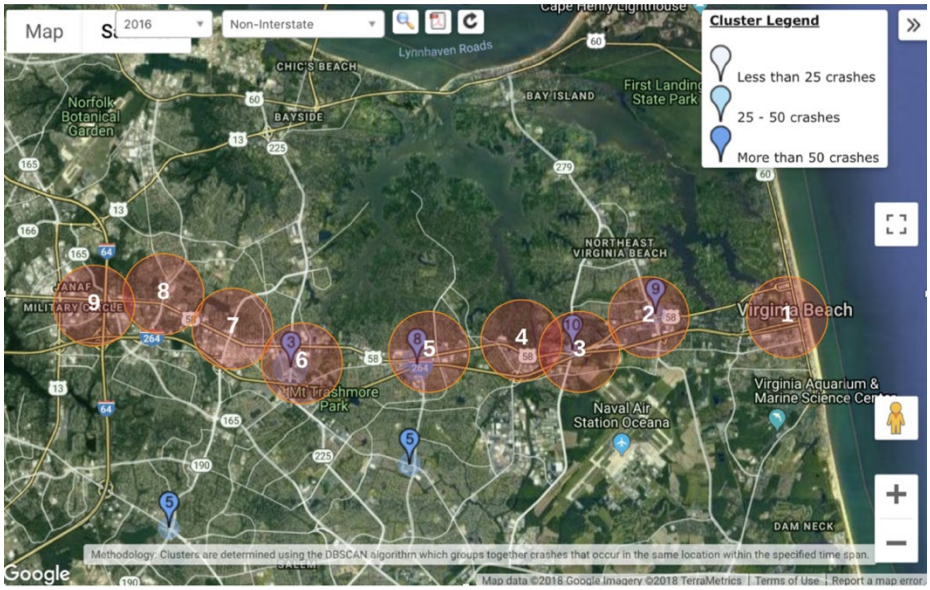


Figure 3. Google Earth view of the Pacific Avenue crash corridor. The circles indicate the traffic intersections.

Computer Vision Based Object Detection

This chapter describes the steps taken to demonstrate our process in developing object detection and tracking algorithms using deep neural network models. Deep learning models require training data with proper annotations. We started with a collection of data from nine intersections on Virginia Beach Boulevard to train the algorithms. In order to develop a more focused approach towards the pedestrian detection and tracking, we leveraged a public dataset (the MOT Dataset) from motchallenge.net. This dataset is a benchmark dataset used in the computer vision community to develop multi-object tracking.

Data Collection and Annotations

To validate the output of the traffic surveillance method described in the previous section, a naturalistic data collection experiment was performed. The collection of intersection video data was done by utilizing the FFMPEG application (a set of open-source libraries for recording and

[1] This project originally ended in 2021. Hence the methods and results presented in this report are from 2021 and all the algorithms presented are referenced until 2021.

converting digital audio and video recordings in various formats), and publicly available live intersection streaming video URLs extracted from the Virginia511 website (https://www.511virginia.org/). To collect video data for processing by the computer vision traffic surveillance method, a script was developed to automatically download and save target streaming intersection videos to an MP4 file. Video was simply recorded throughout several days in varying lighting and weather conditions. Video data was then used to evaluate the performance of the computer vision methods, as highlighted in the proceeding sections.

We selected eight videos from four different intersections for annotations. The videos were chosen from both day and nighttime recordings. The annotations were done at pixel level using CVAT software. The CVAT annotation tool provides an interface for annotating objects in image frames and draws bounding boxes. We annotated a total of 9,158 frames and a total of 170 k instances of objects. We used four object categories: vehicle, person, bicycle, and motorcycle. It should be noted that the object categories of SUV, car, pickup truck, bus, and truck were joined in a common category of vehicle." More details on the annotations can be found in Appendix A: Data Annotation.

Object Detection

Object detection refers to identifying and locating roadway objects in the image or video frame. Recent developments in computer vision and convolutional neural networks (CNNs) have substantially accelerated the accuracy of object detection. In this project, we first used standard and state-of-the-art object detectors. We used both single step (YOLO) and two step detectors (Mask-RCNN, Cascade-Net, Retina-Net, Faster-RCNN). Due to the lower quality of the traffic camera videos, these methods often failed to identify the objects. In many cases, the objects were not detected for a few consecutive frames, and then were redetected at a later frame. This created sparsity in the object detection and tracking framework and often interfered in the association task of tracking (see the Multi Object Tracking (MOT) section for details). Apart from the low quality of the videos, other major reasons for such failures include low illumination, high contrast shadow, high spatial noise, and the fact that most of the networks were trained with a separate dataset. In recent work, it has been shown that most of these CNNs are sensitive to spatial noise and texture [13], and we believe that the special noise and video compression creates distortions and often confuses the detection ability. In this work, we conducted a number of experiments to understand how noise may affect detection. Eight different filters ranging from median and gaussian filters to edge preserving filters were used.

Finetuning Object Detectors

In this section we describe the testing of different deep-learning-based object detectors. We specifically chose 11 object detectors from MMDetection library. These models were pretrained on a MSCOCO dataset. MSCOCO is a benchmark dataset that contains a total of 80 object classes, including car, pedestrians, bicycle, and motorcycle. Our initial testing and validation using MSCOCO-trained models show very poor results. The best result was achieved from a Faster-

RCNN and Cascade-RCNN framework with a mean average precision of 4.1% and 4.5%. This result clearly indicates that standard models trained on high quality data cannot generalize over the low-quality traffic camera data. Hence, it was necessary to finetune these algorithms to be used for our traffic domain. This finetuning is referred as transfer learning. We used the annotated data from traffic videos (as described in the last section) and retrained the models, finding that Faster RCNN provided the best result. The mean average precision (mAP) for cars increased to 25.6% from 4.1% and the mean average recall (mAR) increased to 31.4% from 5.8%. Figure 4 shows a comparative result highlighting the gain in performance due to the domain adaptation.

Figure 5 shows some of the quantitative results. After manual observations of the failure cases, we observed the following: (1) The detector often missed dark colored vehicles, (2) the detector performed really well at nighttime, which may be due to vehicle headlights, and (3) the detector often confused the shadow of the vehicle as part of the car. Additional results with noise tests are presented in Appendix B: Object Detection Results.

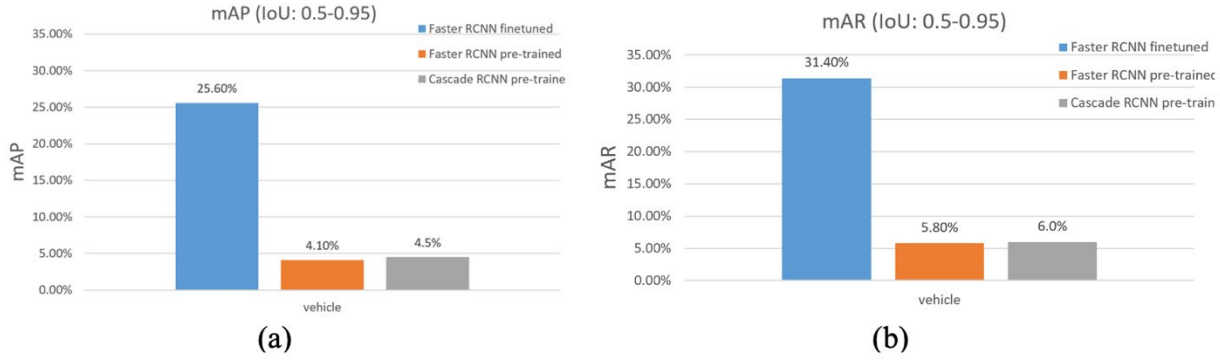


Figure 4. The result for the finetuned model trained with annotated image data from traffic cameras.

(a) (b)

Figure 5. Qualitative output of the object detector in day and night. The vehicles are detected with very high confidence.

Object Detection Using Temporal Information

The finetuning of object detectors significantly improved the results. However, in many cases, an object was detected in one frame, but was missing in the next frame. For those scenarios, we proposed a temporal aware object detector. In this work, we used temporal information efficiently to increase both detection and tracking performance. In other words, we added a "detection by tracking" framework along with the existing "tracking by detection" method. We used optical flow to study the temporal changes in the static image frame and used this information to predict the position of an object. Optical flow measures the displacement of every pixel between two consecutive frames [15,16]. As shown in Figure 6, most of the sparse detections show evidence of the presence of a certain car, but missing points pose a problem. In Figure 6, the car is detected in frame 1 and frame 4, but the object is not detected for frame 2 and frame 3. Therefore, we used the known two detections (A1 and A2) and the temporal information to connect the intermediate points and improve the detection. A1 and A2 are called the "anchors" of the specific object. This proposed method increased the detection accuracy and removed some of the false negatives.

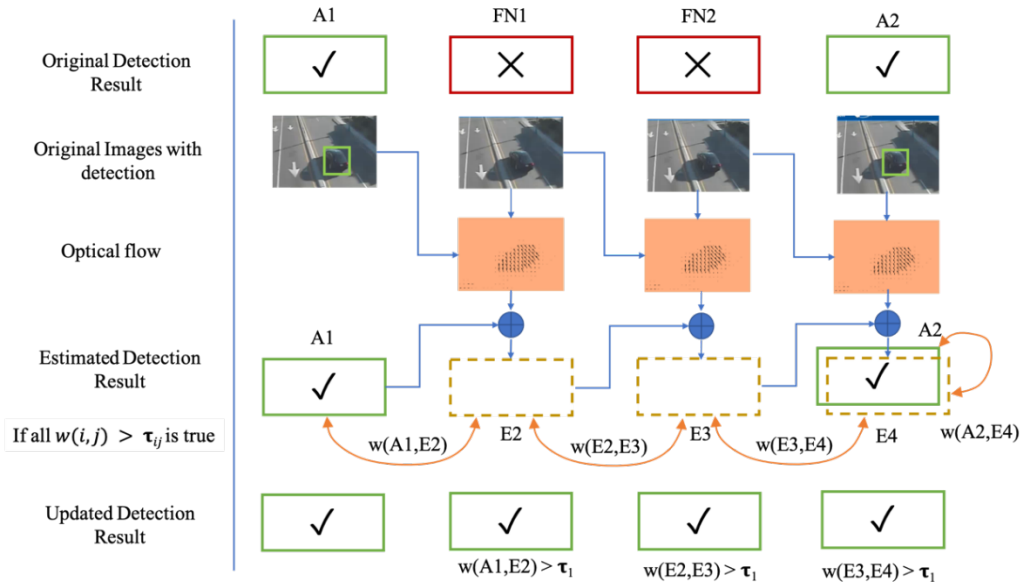


Figure 6. Improving object tracking using optical flow-based methods.

The object detection pipeline consists of the Region Proposal Network (RPN), Region of Interest Pooling (RoI), and the object classifier (RCNN). As seen in Figure 7, after the CNN extracts features, a number of proposals are obtained in frame *t*. These are processed to become final object detections after classifying them into classes and obtaining a detection confidence through RCNN. The $t + 1$ frame again computes object proposals, to which the detections of the previous frame are added after using one of the processing methods in the proposed framework.

Two methods for processing the previous detections were tested. The first method uses optical flow to propagate detections from previous frames to a new frame. In more detail, a detection of the previous frame is propagated to the new frame as an object proposal. The same detection confidence is transferred as confidence for the proposed object score. The method of FlowNet (Brox) was applied to the problem. An example graph is shown in Figure 7 to illustrate how the approach is applied. Using optical flow, we receive a vector that predicts the displacement of the previous box (red) to the new frame (light blue). The second tested method simply places the proposed object at the same position as in the first frame. To increase the likelihood of detection, more than one proposal is added around the center of the optical-flow-based or same-position proposal. More specifically, 20 proposals were added to increase detection likelihood and these were perturbed around a center location. Experimental results using the proposed methodology are shown in Table 1, which shows an improvement of 0.4% in mAR and 3.2% in mAP (Intersection over Union [IoU] = 0.5) while mAP (IoU = 0.5–0.95) remained almost the same.

Table 1. Quantitative results on improving detection using past frame information

Vehicles	mAP (.5-.95)	mAP (.5)	mAR (.5-.95)
Traditional	**25.6**	58.3	31.4
Same position	25.4	61.3	31.7

Opt. flow based	25.5	**61.5**	**31.8**

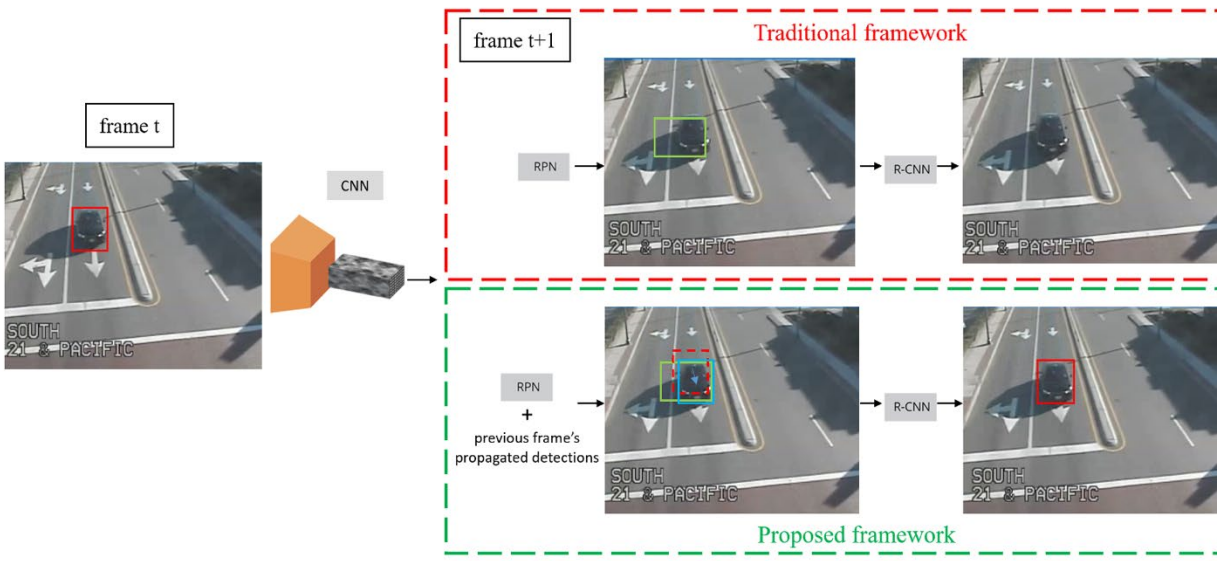


Figure 7. Proposed framework for improving detection using past frames. (a) Simplified illustration of object proposal that fails to be identified as object and one that succeeds. Dashed red rectangle shows previous frame's detection and cyan the propagated one with optical flow.

Multi Object Tracking (MOT)

Multi object tracking (MOT) helps to track multiple objects over a series of frames. A majority of previous work in MOT is based on the paradigm of tracking-by-detection [23, 24], which is comprised of three basic stages. In the first stage of detection, objects are identified at every frame using bounding boxes. In the next stage, feature extraction methods are applied to the detected objects to extract appearance, motion, and other interaction features of the objects, which are then used to compute similarity or affinity scores among object pairs. In the final stage of association, an assignment problem is solved to match objects at previous frames with objects at the current frame. In this project, we experimented with two different algorithms. We tested Deep-Sort, a benchmark algorithm in the community. We also introduced a new graph neural-network-based algorithm: GCNNMatch.

Object Tracking: Deep-SORT

Each tracker needs to associate objects by using some similarity metrics across frames. These metrics are a factor of geometric characteristics such as size, position, and appearance characteristics. The online tracker Deep-SORT [19] is a fast and accurate tracking method that utilizes these two metrics. Deep-SORT uses a Kalman filter to predict object positions using a constant velocity motion and linear observation model. For the geometric similarity metric computation, the Mahalanobis distance is computed between the Kalman predicted states and the

new detections in each frame. For the appearance metric, a reidentification model is used, which is trained based on the Cosine Metric learning method [20]. The CNN architecture uses a wide residual network that is composed of two convolutional layers, followed by six residual blocks. A feature map of 128 dimensions is computed and used to derive the cosine distance. Finally, the two metrics are combined using a weighted sum.

The method described above was applied to our VDOT data via the following steps. First, a reidentification model was trained using the Cosine Metric learning method. Two separate reidentification models were trained, one targeting human reidentification and the other one vehicle reidentification. The two datasets used were Market1501 [21] and VeRi [22] for humans and vehicles, respectively. Then Deep-SORT was applied to the dataset.

MOT uses tracklets to connect between past and future instances. A "tracklet" refers to a short track segment that corresponds to a continuous trajectory of an object over a brief period of time. To determine tracking-related parameters, such as minimum detection confidence, initial object instances before tracklet generation, maximum IoU for matching, maximum frames to keep each tracklet active, and maximum cosine distance for matching, multiple tests were carried out on one of the videos. Following that, we chose the best sets of parameters and ran them on our test data and reported tracking performance using standard parameters including Multi-Object Tracking Accuracy (MOTA), Identity F1 score (IDF1), Mostly Tracked objects (MT, the ratio of ground-truth trajectories that are correctly predicted by at least 80%), Mostly Lost objects (ML, the ratio of ground-truth objects that are correctly predicted at most 20%), Partially Tracked (PT, the ratio of ground-truth objects that are correctly predicted between 80% and 20%), ID Switches (ID Sw.), and Fragmentations (Fragm.). More details of the metrics and the parameter selection can be found in Appendix C: Object Tracking Results. Table 2 and Table 3 show the results. The pretrained category reflects cases for object detection using Faster RCNN without any domain adaptation. Finetuned and Finetuned +Opt. Flow refer to a Deep-SORT algorithm that uses output from the previous section: Computer Vision Based Object Detection.

Table 2. Tracking-related Evaluation Metrics Derived From Tracking Output (Bold Numbers = Best Results)

	MOTA	IDF1	MT	PT	ML	ID Switch.	Fragm.
Pretrained	8.2%	8.4%	5	73	126	533	666
Finetuned	34%	43%	77	84	43	**319**	**443**
Finetuned + Opt. Flow	**34.6%**	**43.8%**	**79**	**85**	**40**	326	465

Table 3. Detection-related Evaluation Metrics Derived From Tracking Output (Bold Numbers = Best Results)

	IDP	IDR	Recall	Precision	Unique Obj.	MOTP
Pretrained	34.1%	4.8%	11.6%	82.9%	204	24.8%
Fine-tuned	**73.6%**	30.4%	37.9%	**91.8%**	204	26.7%
Fine-tuned + Opt. Flow	68.8%	**32.1%**	**40.9%**	87.7%	204	**27.8%**

Figure 8. Example of object tracking from DeepSort.

Graph Neural Network Based MOT

In this section, we propose a novel method for online MOT using GCNN-based feature extraction and end-to-end feature matching for object association. The GCNN-based approach incorporates both appearance and geometry of objects at past frames as well as the current frame into the task of feature learning. This new paradigm enables the network to leverage the "context" information of the geometry of objects and allows us to model the interactions among the features of multiple objects. Another central innovation of our proposed framework is the use of the Sinkhorn algorithm for end-to-end learning of the associations among objects during model training. The network is trained to predict object associations by considering constraints specific to the MOT task. Experimental results demonstrate the efficacy of the proposed approach in achieving top performance on the MOT '15, '16, '17 and '20 Challenges[2] (datasets designed for the task of multiple object tracking) among state-of-the-art online approaches.

One of the major paradigms in MOT is the tracking-by-detection paradigm, where an object detector is first used to extract object locations at each frame separately, followed by a tracker, which associates detected objects across frames. The goal of the tracker is to solve the bipartite

[2] https://motchallenge.net/faq/

graph matching problem, where every object instance in a past frame is associated to at most one object instance in the current frame, using pair-wise object affinities. There are two variants of the matching problem considered in MOT: online matching, where objects are associated only using past frames, and offline matching, where information from both past and future frames are used to track a given object. In this work, we only focus on the MOT problem involving online matching.

GCNNMatch

Figure 9 provides an overview of our proposed approach. The method consists of a feature extraction component and an association component. The feature extraction component takes as input the bounding boxes of tracklets and detections, available through an object detector. In MOT, the goal is to associate and track objects across multiple frames in a video sequence, maintaining the identity of each object as it moves through the scene. The cropped image for each bounding box is fed into a CNN architecture which extracts h_{app}, represented as flat high-dimensional vectors. The inputs to the GCNN then consist of both node and edge features. The appearance features h_{app} constitute the node features h^v at any node \$v\$. Then, by concatenating the appearance features h_{app} and geometric features h_{geom} at the pair of nodes and sending them to a fully connected neural network (FC-NN) f_{edge}, the edge features $z^e \in R$ are computed for a pair of tracklet and detection nodes.

The GCNN architecture transforms non-linearly the node and edge features of each hidden layer $k-1$ using graph structure to update node H_k and edge Z_k features at layer k. Further, an adjacency matrix is created including self-edges, $\widetilde{Z_k} = Z_k + I$, where I is an identity matrix. The degree of the adjacency matrix $\widetilde{Z_k}$ is denoted by τ_k. The node features are then updated at layer k using GNN training. The overall system uses a combination of FC-NN and GNN. The node features produced at the final layer are the interaction features, H_{inter}.

Interaction features H_{inter} are combined with geometric features h_{geom} to produce an affinity metric between a tracklet T_m and a detection D_n. Initially, an appearance affinity is computed using cosine similarity between the H_{inter} features at T_m and D_n. Then, we compute the IoU of the bounding boxes using the geometric features h_{geom} of each bounding box, which are the pixel coordinates. Both the cosine similarity score and IoU score are fed into another FC-NN, $f_{affinity}$, that produces the affinity score, $s_{m,n}$ in S.

Conventionally, in order to satisfy MOT constraints, loss function combinations such as binary cross entropy loss are used for each element with the addition of a softmax loss to satisfy constraints. In contrast to that, the Sinkhorn algorithm is leveraged in this work to automatically satisfy MOT constraints during training. It iteratively normalizes the rows and columns of S and directly approximates the constraints. During testing, the Hungarian method is applied over S^* to perform hard assignments of 0 or 1.

More details of the process can be found in the Conference on Intelligent Transportation Systems Conference on Intelligent Transportation Systems paper and the ArXiV manuscript by the authors in [17,18].

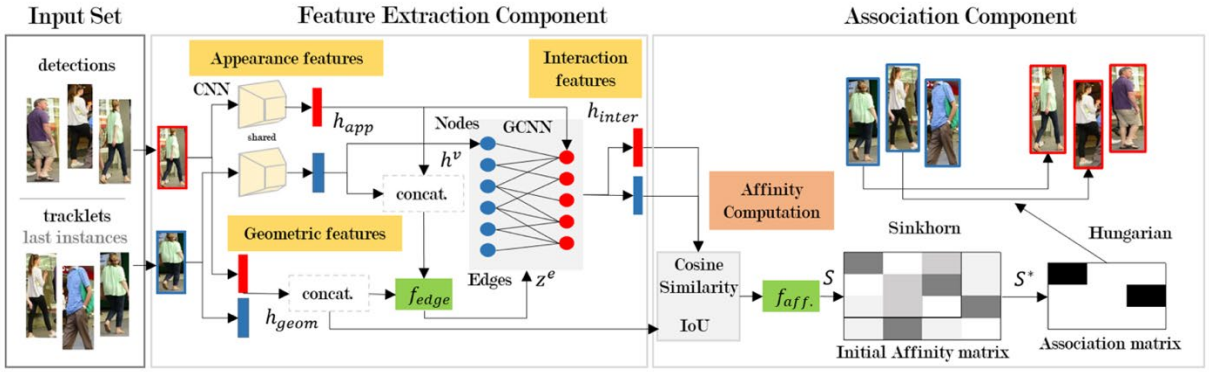


Figure 9. Overview of our proposed approach.

In Figure 9, Two input sets are given, consisting of bounding box images of the last instances of tracklets and the new detections (nodes). A CNN extracts appearance features h_{app}, which are used as node features h^v. They are then concatenated with geometric features h_{geom} and fed to a function f_{edge} to compute edge features z^e. Using cosine similarity and IoU, $f_{affinity}$ computes the similarity scores S among pairs of tracklets and detections. The Sinkhorn algorithm normalizes S to satisfy MOT constraints and produces the association matrix S^*. The Hungarian algorithm is produced during testing to convert values of S^* to binary.

Tracking Results

For this dataset, we provided two baseline object tracking approaches. Both Deep-SORT and the proposed GCNNMatch were finetuned using traffic-related data and were applied on the test set. The Deep-SORT re-identification model was trained using the Vehicle Re-identification Dataset, since it requires pairs of images. Table 4 shows quantitative results for different cases. First, the object detector pretrained on COCO was applied and then used with Deep-SORT, achieving very low scores of 8.2% MOTA and 8.4% IDF1. Next, the finetuned object detector yielded a MOTA of 34% and IDF1 of 43%, a significant increase using the same tracker, depicting the importance of accurate input detections. Then, the detections were enhanced using the proposed optical flow propagated detections. Results showed an increase of 0.6% MOTA and 0.8% IDF1. These results also increased recall by 3% at the cost of 4.1% in precision. Recall was increased to 40.9% from 37.9% by adding the flow information. The results are promising since the involvement of optical-flow-based extra object proposals results in increased overall MOTA and IDF1, the two important metrics in tracking performance. GCNNMatch was applied for an increase of 0.1 MOTA and 4.2 IDF1 over Deep-SORT, again using the optical flow method for detections. Moreover, this significantly minimized the number of ID switches. Qualitative results of the proposed approach are also shown in Figure 10. More results are available in Appendix C: Object Tracking Results.

Table 4. Quantitative Results of Selected Baseline Approaches for Object Tracking on VA Beach Traffic Dataset.

	MOTA	IDF1	MT	ML	Precision	Recall	ID Switch.
DeepSort + Finetuned + Opt. Flow	34.6	43.8	**79**	40	**87.7**	40.9	326
GCNNMatch + FineTuned + OptFlow	**34.7**	**48.4**	71	**31**	83.6	**43.8**	**245**

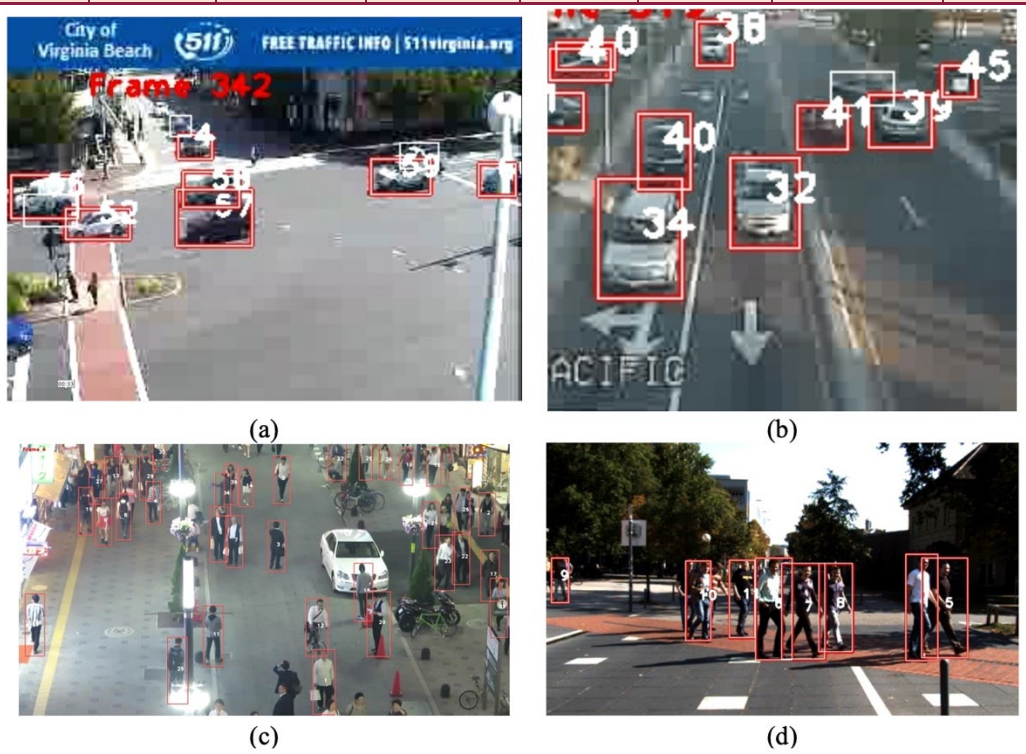


Figure 10. Results of the MOT algorithms from GCNNMatch. (a, b) VA beach traffic data tracking vehicles, (c, d) MOTChallenge data set tracking pedestrians.

Kinematic Characteristics of Traffic Actors

In this chapter, we discuss calculation of the kinematic behavior of actors in the traffic scene from the output of object detection and tracking, as described in the previous sections. Both the detection and tracking provide us output in the pixel space. However, the pixel space does not provide actual information about the kinematics of the actors, and we need to transform this output to the road coordinate system. Hence, we applied a Pixel to GPS coordinate transformation first. This helps track each object in road coordinates and computes their relative position and velocities, which in turn helps identify conflicting situations.

Pixel to GPS

The GPS coordinates of the traffic camera were given; thus, the corresponding Google Map image was used to illustrate the relative position of each object during the video collection period. Reference points were then chosen manually to match pixel positions to corresponding points on the map. Two-dimensional interpolation takes a series of point pairs with their associated output and generates an estimated output value for a new pair of points. For this project, the inputs were pairs of pixel coordinates with the known latitude or longitude associated with each pixel pair. This then created two interpolation objects, LAT and LON, corresponding to latitude and longitude.

The 2D interpolation was performed twice for each pixel coordinate: once for the known latitude and once for the known longitude. We selected key points both from the camera image and the Google Map image. These key points were mainly visible landmarks, corner points (edges) that were visible in both the images. The natural neighbor method of interpolation was used because it was shown to be the most accurate of the methods available, most likely due to the geometry of the road.

The data (pixel locations and object ID) given from MOT was combined with the Google Image to display a video with the tracked object next to its corresponding map with the objects simultaneously superimposed. Each object's color was determined by the origin location of the object. An example of the video output is given in Figure 11.

Figure 11 shows the intersection used for the analysis in this report at 21st Street and Pacific Avenue in Virginia Beach, Virginia. The traffic camera is pointed southbound along Pacific Avenue, which is the mainly vertical street shown on the map. The colors correspond to the starting position of each object: yellow represents northbound traffic on Pacific Avenue, red represents southbound traffic on Pacific Avenue, and the blue dots represent all other objects that did not start on Pacific Avenue. Calibration is needed for each intersection and each camera view.

Estimation of Vehicle Kinematics

From the GPS coordinates calculated above, along with the traffic feed framerate, the speed of each object was easily calculated using the distance each object moved between frames. The acceleration was also calculated using differentiation. However, using this real-time framerate, the second order differentiation showed to be unstable. To combat this instability, we used a moving average filter for the speed every 5 frames as shown in Figure 12.

To validate the kinematic calculations, 16 test runs were performed through the intersection of interest. The real-time speed and GPS coordinates of the test vehicle were recorded with a smart phone app while performing multiple turns and speed changes in the intersection. The procedures explained above were then performed to calculate the GPS coordinates and kinematics of the test vehicle from the recorded traffic camera footage. Unfortunately, the low spatial resolution of each

GPS ping prevented an accurate baseline to compare the vehicle's location at each timestep on the road. If the test vehicle stayed at a relatively constant speed, the calculated speed had a variance of ±1.435 mph. Otherwise, we encountered higher error due to GPS resolution.

Figure 11. Example output of the Pixel to GPS algorithm. (a) Left side shows traffic camera feed with each detected object. (b) Right side shows map corresponding to the GPS location of each object.[3]

In addition to the average speed of any vehicle, Figure 12 also shows the lateral and longitudinal velocities of each object. These were calculated by projecting the vehicle's velocity onto the road in question. In this example, the longitudinal velocity is the directional speed along Pacific Avenue, while the lateral velocity is the directional speed perpendicular to Pacific Avenue. The lateral velocity then indicates any lane drifting, lane-change, or turning behavior.

Conflict Prediction

The original scope of this project was to detect and track objects from a traffic camera feed, and then calculate the basic kinematics of each object to be used for future analysis. A strong application of this project could be to predict conflicts that may occur at these intersections by using the calculated kinematics and known crash surrogate measures. Thus, we performed an initial analysis into the possibility of predicting conflicts at this example intersection.

In our example, we identified three potential conflicts throughout the duration of the available video feed. These potential conflicts represent what could be deemed a near-crash event. Figure 13a shows an example of the vehicle trajectories of one of these potential conflicts. The blue line represents a vehicle that was turning right out of the parking lot to travel south on Pacific Avenue. The red line represents a vehicle that was already traveling south on Pacific Avenue. In the video, the red vehicle had begun to change lanes into the right lane (into which the blue vehicle turned), which we identified as a potential conflict.

The vehicle behavior can also be seen in Figure 13b, which shows the relative velocities of each vehicle. Object ID 168 is the blue vehicle that turned right out of the parking lot to travel

[3] Google Maps. 21st St. and Pacific Ave., Virginia Beach, VA

southbound on Pacific Avenue. Object ID 181 is the red vehicle that was already traveling southbound on Pacific Avenue.

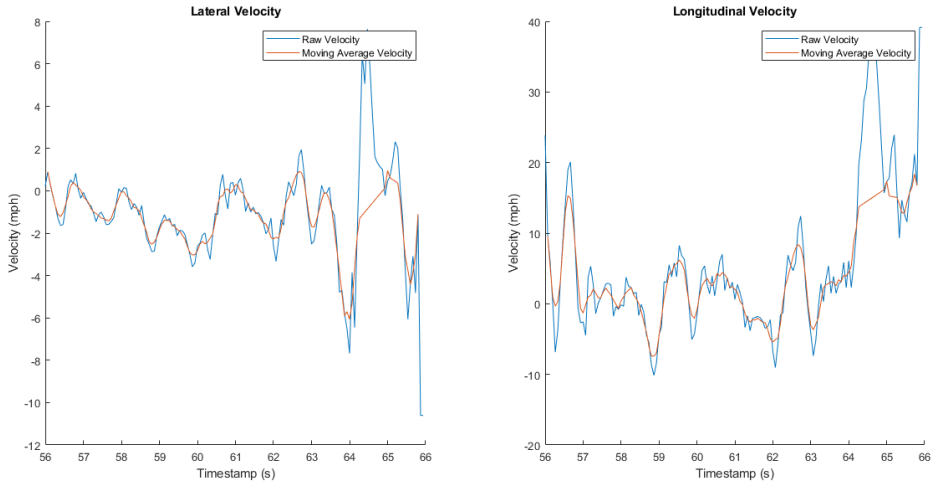


Figure 12. The lateral and longitudinal velocity of one object identified in the traffic video feed (longitudinal velocity is along Pacific Avenue, while lateral velocity is perpendicular to Pacific Avenue).

In Figure 12, The blue represents the first derivative of the distance as compared to the video framerate. The red line represents the moving average of the velocity

In this initial analysis, a near crash event was identified if two vehicles were within 4 meters of each other, and there was a sudden change in the lateral or longitudinal velocity of either vehicle. This "spike" in velocity was determined if the velocity exceeded 50% of its previous average value during the time in which the vehicles were within 4 meters of each other. Therefore, if a potential conflict was identified, a video clip including that timestamp was created so that we could evaluate if it was indeed a crash or near-crash event. This allowed for some initial validation in the crash surrogate measures we were using.

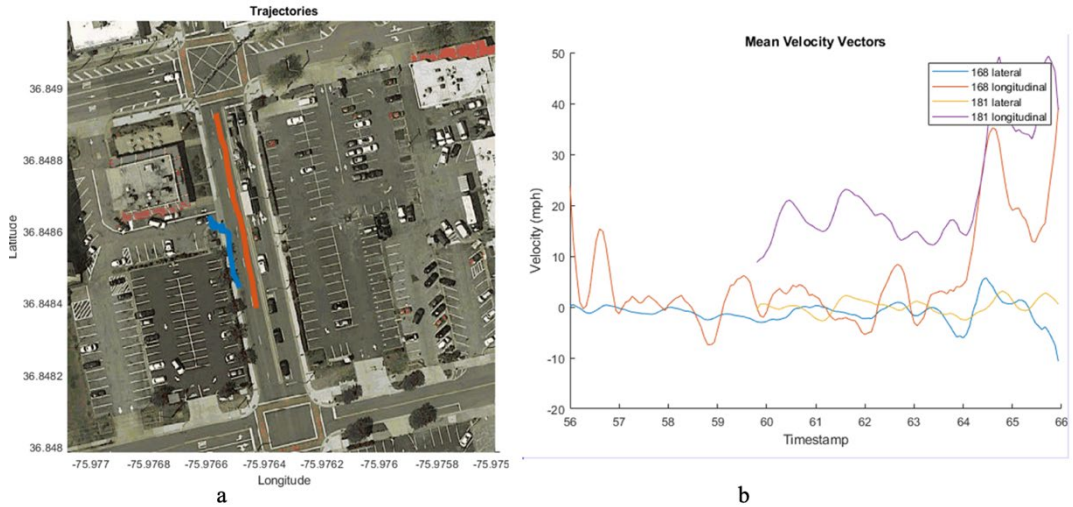


Figure 13. (a) The trajectories of two vehicles that had a "near crash" event.

In Figure 13, the blue line represents one vehicle turning right out of the parking lot to travel south on Pacific Avenue. The red line represents one vehicle traveling southbound on Pacific Avenue. (b) is the lateral and longitudinal velocities of the vehicles in the near-crash event example. Vehicle 168 is the blue vehicle in the previous figure (turning right from the parking lot to travel south on Pacific Avenue).

This initial analysis presented is subjective (as to how a near-crash is determined from the video feed as well as the crash surrogate measures used) and requires much more analysis and expert data reduction. Additionally, the near-crash examples were specific to this intersection; the turn out of the parking lot onto Pacific Avenue seemed to be a critical area of conflict. The most common conflict configuration (and location) may be different for each intersection, and the conflict calculations may differ per intersection. Thus, in order to more accurately predict potential conflict, each intersection should be analysed separately. Each interpolation object used to calculate the GPS coordinates is already specific to each intersection, and thus should not require too much additional analysis.

Conclusions and Recommendations

Traffic cameras across the country capture valuable information about traffic. By tracking the behavior of each agent, we can benefit in addressing key traffic management tasks. From real time traffic surveillance and congestion detection to traffic violation monitoring and crash surrogate measurements, traffic cameras can be used in diverse applications to help traffic engineers. However, their true potential is still unexplored. While many researchers have used high quality cameras and sensors, large scale deployment will require a considerable amount of funding. Existent traffic cameras, on the other hand, are already installed and provide visual information.

In this project, we have shown that with the advent of deep neural networks, automatic processing of large-scale traffic camera data is possible. We have specifically designed a new object detector that is fine tuned for applications to traffic cameras. We have also proposed a new tracking algorithm, GCNNMatch, which is a highly ranked multi object tracker in a MOT benchmark. We specifically showed how pedestrian tracking can be done in challenging situations. We further extended the capabilities of GCNNMatch to track vehicles and other objects through traffic scenes. Finally, we demonstrated a process from the detection and tracking on how to calculate the kinematics of the objects in the scene. Through demonstration, we have shown how the overall process can help us identify potential conflicts and near crash scenarios.

Recommendations and Future Directions

In this section, we present a series of recommendation, including limitations to the current state-of-the-art to provide a future direction for traffic camera based automatic surveillance.

- Current infrastructure camera data is transmitted with **low resolution, low frame rate, and high compression**. It turns out that the resolution of the images may be adequate to adapt for most object detection algorithms. The major reason for the failure to adapt computer vision algorithms is a high amount of noise in the image. This noise can be attributed to multiple sources. First of all, the cameras are often not suitable for nighttime capture. The nighttime data includes large amount of spatial noise. The second source, and often the major source, of noise is the high compression of the videos. Due to live streaming and low network bandwidth, these camera videos are highly compressed before being transmitted. The compression generates irreversible artifacts in the video. As most deep-learning-based computer vision methods use frame by frame processing, the artifacts in each frame are large enough to fail the detectors. As a **future direction**, camera compression can be changed to facilitate better performance. Most MOT algorithms need a high frame rate to guarantee continuity. In this project, we used data at 15 frames per second for most cases. With a roadway posted speed of 25 mph to 40 mph, this frame rate is adequate. However, we have seen degradation of the frame rate for some periods. This may hamper performance.
- The **field of view** of most of the cameras are narrow. They are often between 75 degrees to 110 degrees. This covers a small subsection of the intersection. Unless we access all directions of the traffic, it is difficult to understand the overall traffic scenario at any intersection.
- Deep neural network is a data driven method. In this project, we have shown that annotating a small dataset and using transfer learning, we can significantly increase the performance of the object detectors. We currently need benchmark datasets that can be used by researchers to facilitate innovation and enhance accuracy for traffic camera data processing. We need more annotated data to develop more advanced algorithms.
- The near-crash examples shown in the report were specific to one intersection; the outbound turn at the parking lot onto Pacific Avenue seemed to be a critical area of conflict. The most common conflict configuration (and location) may be different for each intersection, and the conflict calculations may differ per intersection. Thus, in order to more accurately predict potential conflict, each intersection should be modelled separately. Also, the kinematics can help in studying the severity of the crashes, and thus aid in identifying crash severity.
- Each intersection is different in terms of structure; however, some are more similiar to other others in terms of attributes, including N-way traffic, total number of lanes, pedestrian facilities, etc. A possible future step could be to identify a group of intersections that are distinctly different and to then classify each individual intersection to one of those categories. This way, modeling a small number intersections may cover all of the intersections and facilitate better modeling of conflicts.

- There is a scarcity of annotated data for crash and near crash data at intersections. Recent work by Bareiss [27] has partially addressed this, but more structured efforts are required. In order to learn the safety critical characteristics of different actors, more annotated data with associated metadata are required. With renewed interest in automated vehicles, a series of safety metrics have been developed, including RSS and Pegasus [28-30]. However, more dedicated efforts are required to apply these metrics and test their effectiveness in traffic scenario.

Please note that a follow up project from this work was also completed at the time of this report. This project is 06-012: *Real time risk prediction at signalized intersection using graph neural network*. This project mainly focuses on modeling the safety from the kinematic interactions between different actors.

Additional Products

Education and Workforce Development Products

Student Engagement

Two graduate students were funded and worked on this project: Eileen Herbers, and Ioannis Papakis. Ioannis's master's thesis was inspired by this project, as he made fundamental contributions to the field of pedestrian tracking.

- Papakis, I. (2021). *A Graph Convolutional Neural Network Based Approach for Object Tracking Using Augmented Detections with Optical Flow* (master's thesis, Virginia Tech).

Award

Ioannis Papakis won the best master's thesis award in the 2021 Torgersen annual student award.

Publication

This work has resulted in two publications, which, in combination, have received 65 citations from researchers across the world.

- Papakis, I., Sarkar, A., & Karpatne, A. (2021, September). A graph convolutional neural network-based approach for traffic monitoring using augmented detections with optical flow. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)* (pp. 2980-2986). IEEE.
- Papakis, I., Sarkar, A., & Karpatne, A. (2020). Gcnnmatch: Graph convolutional neural networks for multi-object tracking via sinkhorn normalization. *arXiv preprint arXiv:2010.00067*.

Presentation and outreach

The work was presented in multiple venues.

- Dr. Sarkar attended IEEE ITSC in Indianapolis to present this work in the conference.
- This work was selected as an entry for Virginia Tech's entry for DOT

- This research was showcased as part of USDOT RD & T video forum in February 2022. Please find more details in the following links: https://www.youtube.com/watch?v=Vt1fqfh18zM, https://www.transportation.gov/utc/usdot-rdt-utc-video-forums .
- In coming December, Dr. Sarkar will visit Indian Institute of Technology, Bombay to deliver an invited talk in the IndoML conference. He will discuss some of the outcome of this project to the students attending the conference.

Educational material

- The work from this project has helped in developing class material as part of the lecture that Dr. Sarkar provides to the class of Dr. Zachary Doerzaph's graduate course at Virginia Tech.

Technology Transfer Products

This project has resulted in online codebase as part of the project. One of the code bases are the outcome of GCNNMatch. The code is available at https://github.com/IPapakis/GCNNMatch. This codebase is targeted for pedestrian tracking. This codebase has received 54 stars to date and 10 forks in Github.

Additionally, we have released the object detection code in the following Github repository: https://github.com/VTTI/object-detection.

Data Products

We have also released the annotation of all the images that were produced as part of the object detection pipeline. The data consists of images and pixel level annotations of images.

References

1. Cole, M., & Read, S. (2018). Pedestrian Safety Action Plan.

2. Ananthanarayanan, G., Bahl, P., Bodík, P., Chintalapudi, K., Philipose, M., Ravindranath, L., & Sinha, S. (2017). Real-time video analytics: The killer app for edge computing. computer, 50(10), 58-67.

3. Naphade, M., Chang, M. C., Sharma, A., Anastasiu, D. C., Jagarlamudi, V., Chakraborty, P., ... & Lyu, S. (2018). The 2018 nvidia ai city challenge. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 53-60).

4. Zhang, H., Ananthanarayanan, G., Bodik, P., Philipose, M., Bahl, P., & Freedman, M. J. (2017). Live video analytics at scale with approximation and delay-tolerance. In 14th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 17) (pp. 377-392).

5. Winkowski, C., Sarkar, A., Hickman, J., Abbott, L., "Residual Network-Based Driver Gaze Classification in Naturalistic Driving Studies", Transportation Research Record (Accepted)

6. V. Sundharam, Sarkar, A., Hickman, J., Abbott, L., " Characterization, Detection, And Segmentation of Work Zone scenes From Naturalistic Driving Data", Transportation Research Record

7. Papakis, I., Sarkar, A., Svetovidov, A., J. Hichman, L. Abbott, "A CNN-Based In-Vehicle Occupant Detection and Classification Method Using SHRP 2 Cabin Images", Transportation Research Record: 0361198121998698.

8. Sonth, A., Sarkar, A., Bhagat, H., & Abbott, L. (2023, June). Explainable Driver Activity Recognition Using Video Transformer in Highly Automated Vehicle. In 2023 IEEE Intelligent Vehicles Symposium (IV) (pp. 1-8). IEEE.

9. Thapa, S., Sarkar, A (2022). GAN-Based Deidentification of Drivers' Face Videos: An Assessment of Human Factors Implications in NDS Data. In Proceedings 2023 IEEE Intelligent Vehicles Symposium (Accepted).

10. Thapa, S., & Sarkar, A. (2023). GAN-based Deidentification of Drivers' Face Videos: An Assessment of Human Factors Implications in NDS Data. *arXiv preprint arXiv:2306.02374*.

11. Bhagat, H., Jain, S., Abbott, L., Sonth, A., & Sarkar, A. (2023, June). Driver gaze fixation and pattern analysis in safety critical events. In 2023 IEEE Intelligent Vehicles Symposium (IV) (pp. 1-8). IEEE.

12. Xu, H., Sarkar, A., & Abbott, A. L. (2022). Color Invariant Skin Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 2906-2915).

13. Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., & Brendel, W. (2018). ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. arXiv preprint arXiv:1811.12231.

14. Xu, Y., & Wang, J. (2019). A unified neural network for object detection, multiple object tracking and vehicle re-identification. arXiv preprint arXiv:1907.03465.

15. Sun, D., Roth, S., & Black, M. J. (2010, June). Secrets of optical flow estimation and their principles. In 2010 IEEE computer society conference on computer vision and pattern recognition (pp. 2432-2439). IEEE.

16. Sun, D., Roth, S., & Black, M. J. (2014). A quantitative analysis of current practices in optical flow estimation and the principles behind them. International Journal of Computer Vision, 106(2), 115-137.

17. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., ... & Lin, D. (2019). MMDetection: Open mmlab detection toolbox and benchmark. arXiv preprint arXiv:1906.07155.

18. Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28, 91-99.

19. Wojke, N., Bewley, A., & Paulus, D. (2017, September). Simple online and real time tracking with a deep association metric. In 2017 IEEE international conference on image processing (ICIP) (pp. 3645-3649). IEEE.

20. Wojke, N., & Bewley, A. (2018, March). Deep cosine metric learning for person re-identification. In 2018 IEEE winter conference on applications of computer vision (WACV) (pp. 748-756). IEEE.

21. Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., & Tian, Q. (2015). Scalable person re-identification: A benchmark. In Proceedings of the IEEE international conference on computer vision (pp. 1116-1124).

22. Lou, Y., Bai, Y., Liu, J., Wang, S., & Duan, L. (2019). Veri-wild: A large dataset and a new method for vehicle re-identification in the wild. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 3235-3243).

23. Milan, A., Leal-Taixé, L., Reid, I., Roth, S., & Schindler, K. (2016). MOT16: A benchmark for multi-object tracking. arXiv preprint arXiv:1603.00831.

24. Ciaparrone, G., Sánchez, F. L., Tabik, S., Troiano, L., Tagliaferri, R., & Herrera, F. (2020). Deep learning in video multi-object tracking: A survey. Neurocomputing, 381, 61-88.

25. Papakis, I., Sarkar, A., & Karpatne, A. (2020). Gcnnmatch: Graph convolutional neural networks for multi-object tracking via sinkhorn normalization. arXiv preprint arXiv:2010.00067.

26. Papakis, I., Sarkar, A., & Karpatne, A. (2021, September). A graph convolutional neural network based approach for traffic monitoring using augmented detections with optical flow. In 2021 IEEE International Intelligent Transportation Systems Conference (ITSC) (pp. 2980-2986). IEEE.

27. Bareiss, M. G. (2023). A Dataset of Vehicle and Pedestrian Trajectories from Normal Driving and Crash Events in One Year of Virginia Traffic Camera Data (Doctoral dissertation, Virginia Tech).

28. Shalev-Shwartz, S., Shammah, S., & Shashua, A. (2017). On a formal model of safe and scalable self-driving cars. arXiv preprint arXiv:1708.06374.

29. Sarkar, A., Krum, A., Hanowski, R., & Hickman, J. (2021, June). Responsibility Sensitive Safety Analysis of Truck Following in US Highway. In International Conference on Applied Human Factors and Ergonomics (pp. 119-126). Cham: Springer International Publishing.

30. Sarkar, A., Engström, J., & Hanowski, R. J. (2022). Analysis of Car Cut-ins Between Trucks Based on Existing Naturalistic Driving Data.

Appendix/Appendices

Appendix A: Data Annotation

For the dataset, an estimate of 9158 frames were obtained.

The dataset was then split into:

- Training set: 4939 frames, 95960 annotations
- Test set: 4219 frames, 74517 annotations
- Object categories: vehicle, person, bicycle, motorcycle

It should be noted that the object categories of SUV, car, pickup truck, bus and truck were joined in a common category vehicle for further analysis. The next step is that of the annotation which can provide ground truth of objects in the dataset. That is necessary and is used both in training the models but also for evaluating them. An example of annotation is shown in Figure 14. Here the CVAT annotation tool is used which provides an interface for annotating objects in frames. Through that tool, objects can be annotated using bounding boxes and labels for each object. These labels represent the category of the object such as vehicle or person.

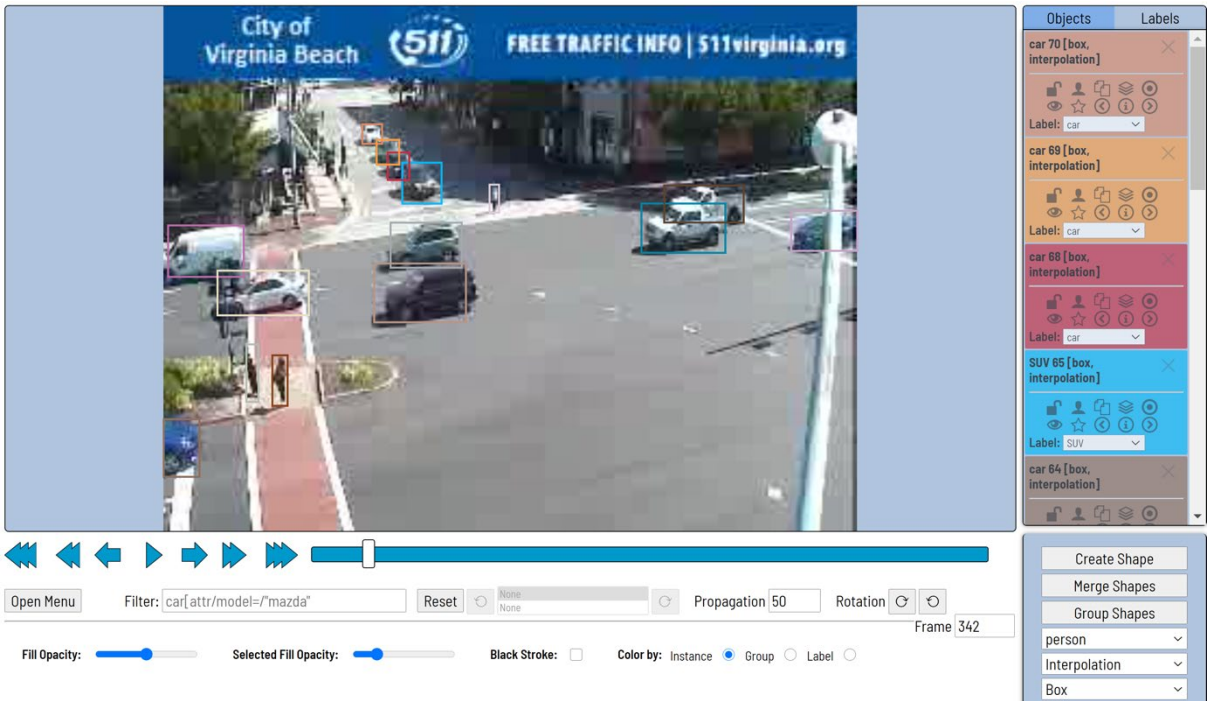


Figure 14. Annotation example in CVAT.

A complete list of the object instances, meaning how many times an object of a specific category was annotated, is shown in Table 5. the table illustrates the persistence of each object in the videos

while the second one the variety of objects of each category. This shows the diversity and size of the dataset used.

Table 5. Object Instances

Time	Night	Day	Night	Day	Subset Total
video ID	5627	4428	2126	3228	--
person	1018	1327	0	0	2345
bicycle	0	267	0	0	267
motorcycle	0	0	33	0	33
vehicle	12931	6741	4638	69005	93315
➢ **car**	10516	5235	4638	62674	83063
➢ **SUV**	2323	753	0	5195	8271
➢ **pickup truck**	60	753	0	239	1052
➢ **truck**	32	0	0	897	929
➢ **bus**	0	0	0	0	0

Appendix B: Object Detection Results

In order to analyze the performance of the selected object detectors, common evaluation metrics were used. These include the Average Precision (AP) and Average Recall (AR).

Following the COCO Challenge evaluation method, first the AP with an Intersection Over Union (IoU) of 50% is given and then the AP and AR with IoU, averaged over ten thresholds from 50% to 95%. Also, the small size refers to pixel number of bounding box being less than 32^2, whereas medium is between 32^2 and 96^2.

Figure 15 and Figure 16 illustrate the average Precision and Recall for different detection thresholds and detectors. For each detector shown with the same color, five confidence thresholds are chosen (namely 30%, 35%, 40%, 45%, 50%). The confidence values are not depicted in the graphs for clarity, however smaller confidence always resulted in higher recall, therefore it can be inferred.

It is evident that for medium size objects, both precision and recall achieve numbers higher than the maximum AP=17% and AR=30% of the small objects. The performance for the medium size is ranging from AP = 28% to 47% and AR=30% to 49.8%.

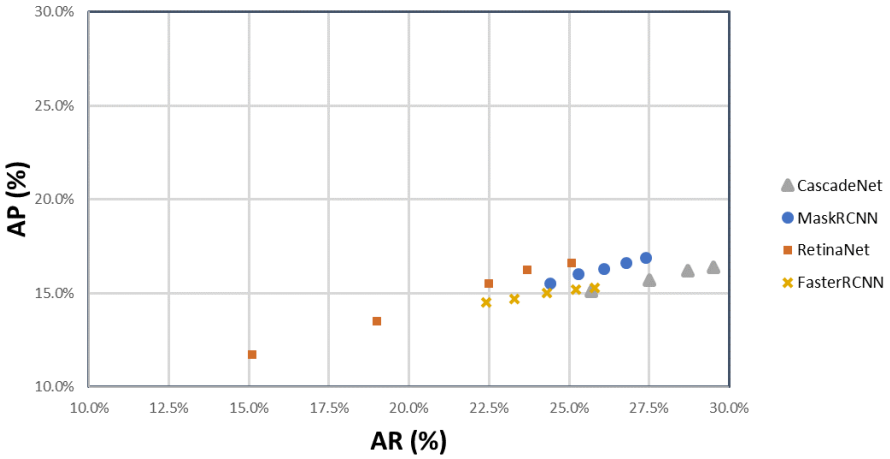


Figure 15. AP-AR Curve for small sized objects.

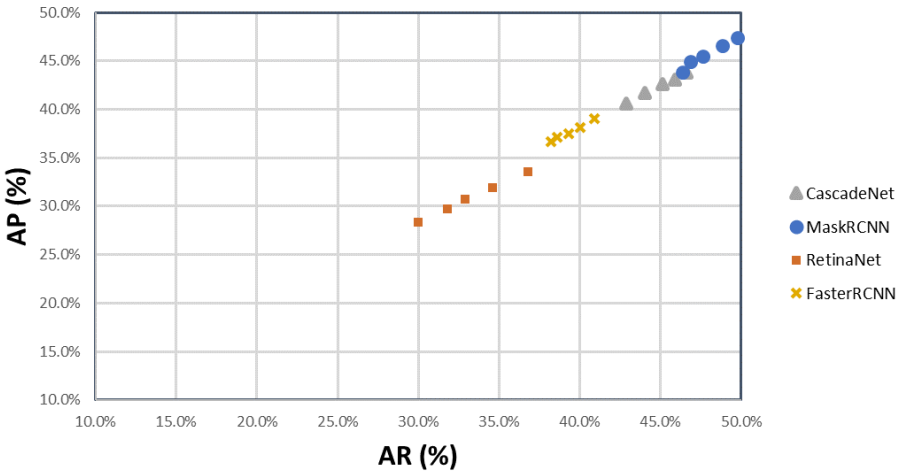


Figure 16. AP-AR Curve for medium size objects.

Considering those detectors performance in the popular COCO Challenge dataset, there is notably a performance drop using the specified traffic videos. A comparison is shown in table 1. The performance of the pre-trained models on COCO train set with an IoU= 50% and confidence threshold 50% is shown as reported in the original papers. In contrast, the performance drop can be seen in the case of traffic videos for cars specifically.

Table 6. Pretrained Models Comparison With Car Detection. (IoU = 50% and Confidence Threshold = 0.5)

Detector	COCO train (mAP %)	Phase I (car AP %)
Cascade RCNN [9]	62.1	43.3%
MaskRCNN [10]	62.3	44.5%
RetinaNet [11]	61.1	32.8%
FasterRCNN [12]	61.2	42.7%

The detectors performance is further investigated using filters in this dataset specifically, since it includes videos overlaid with noise and having low quality. Different filters were applied including Gaussian filter with standard deviation 1, 3 and 5, median filtering with neighborhood size 3 and 5 and finally isotropic Gaussian smoothing kernels with standard deviation 1, 2 and 3. An increased performance in Average Precision is observed in table 2 specifically for the Gaussian filter with a standard deviation of 5. The other detectors followed the same trend.

Table 7. Application of Filters to CascadeRCNN

Filter type	AP (50% IOU)
Median filter with 3X3 kernel	34.3%
Median filter with 5X5 kernel	18.6%
Gaussian filter with Sigma = 1	43.3%
Gaussian filter with Sigma = 0.5	**48.5%**
Gaussian filter with Sigma = 0.3	46.8%
Edge preserving filter	45.0%

After selecting the aforementioned filter and confidence threshold to be 30%, object detection, a comparative analysis is shown for Mask-RCNN and Cascade RCNN. Their performance after the improvements is shown in table 3. It can be observed that Mask RCNN achieves higher precision and recall for almost all cases and applying the filter improves it by 2-3% in AP and AR.

Therefore, Mask-RCNN, using the preferred filter, is chosen for the specific dataset using pre-trained models for optimization.

Table 8. Detectors Performance After Applying the Gaussian Filter and Optimizing Confidence Threshold for 30%.

Detector	Filter	AP (.5 IOU)	AP (IOU .5-.95)		AR (IOU .5-.95)	
			S	M	S	M
Cascade RCNN	None	48.80%	16.60%	43.80%	30.20%	46.60%
Cascade RCNN	gaussfilt Sigma5	50.30%	17.10%	45.10%	31.10%	48.40%
Mask RCNN	None	49.20%	16.90%	47.30%	27.40%	49.80%
Mask RCNN	gaussfilt Sigma5	50.90%	17.40%	49%	28%	52%

Appendix C: Object Tracking Results

Parameter tuning for Deep-SORT

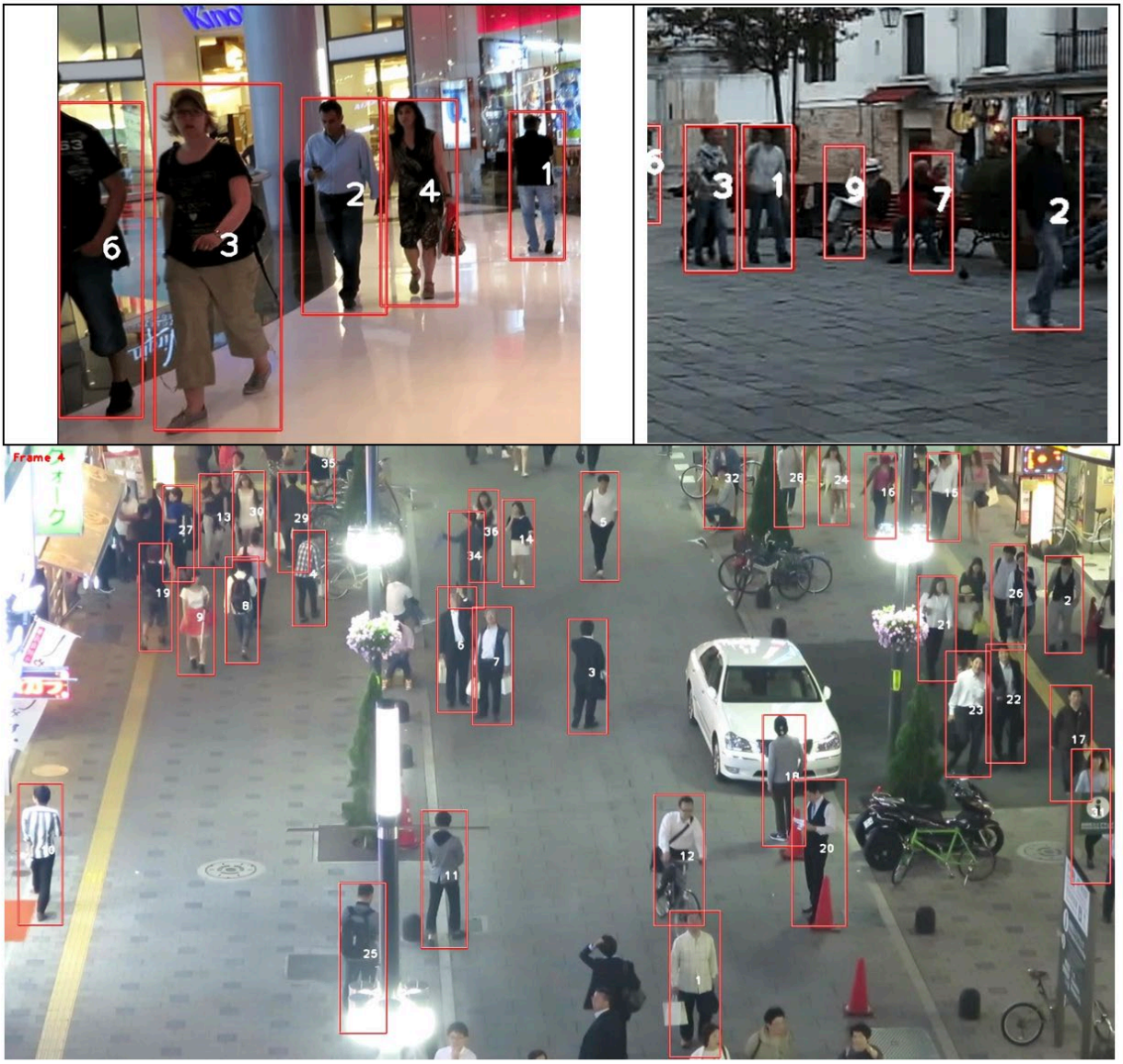
MOT methods are evaluated based on multiple parameters. For the evaluation, the most common in the literature MOT metrics were used. These are the Multi-Object Tracking Accuracy (MOTA), Identity F1 score (IDF1), Mostly Tracked objects (MT, the ratio of ground-truth trajectories that are correctly predicted by at least 80%), Mostly Lost objects (ML, the ratio of ground-truth objects that are correctly predicted at most 20%), Partially Tracked (PT, the ratio of ground-truth objects that are correctly predicted between 80% and 20%), ID Switches (ID Sw.), and Fragmentations (Fragm.). MOTA is defined as: $1 - \frac{\sum_t(FN+FP+IDSW)}{\sum_t(GT)}$, where GT is the ground truth objects. Some metrics, which are also part of the MOT but reflect the influence of the detector mostly, are the Identity Precision (IDP), Identity Recall (IDR), the Recall, the Precision, and Multi-Object Tracking Precision (MOTP). MOTP is defined as: $1 - \frac{\sum_{t,i} d_{t,i}}{\sum_t c_t}$, where c_t denotes the number of matches in frame t and $d_{t,i}$is the bounding box overlap of target i with its assigned ground truth object. More details on these metrics can be found in MOTChallenge [15].

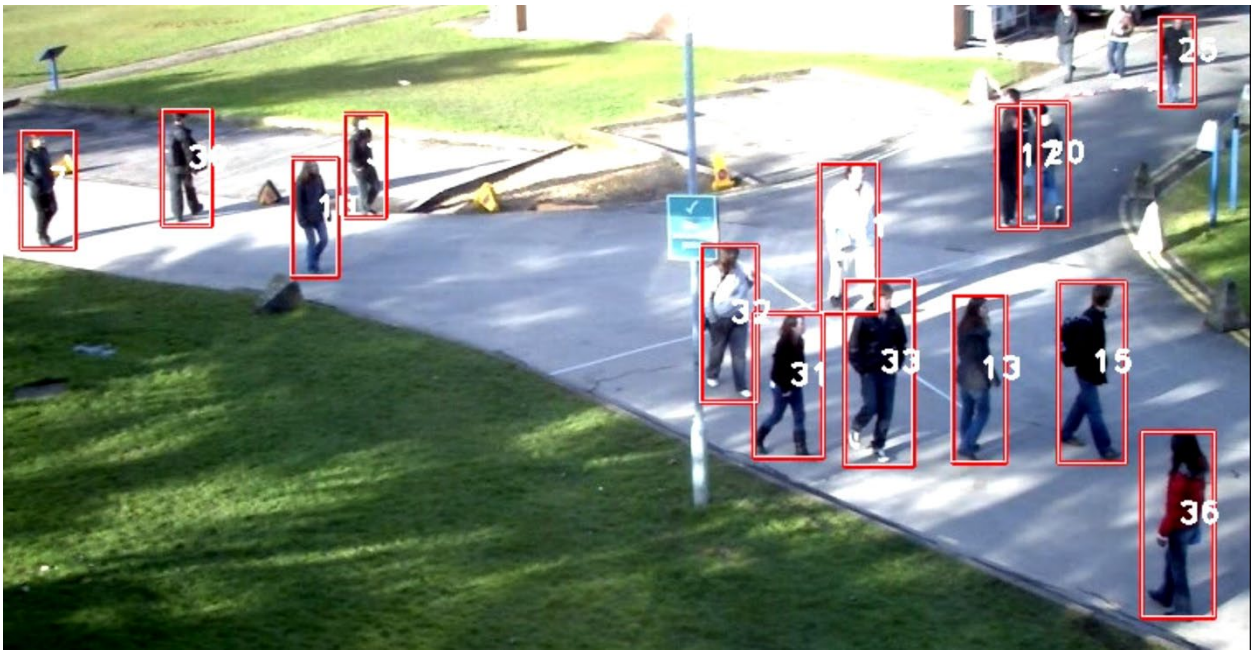
We ran multiple combinations of parameters as seen in Table 9. Test numbers 8, 10, and 11 gave an MOTA of 85.4%, which was the highest. The settings of test no. 10 were chosen for further analysis since it contains intermediate values between the other two. The parameters at test no. 10 were chosen for further analysis. A comparison was then performed on all test set videos using different confidence thresholds in detection. A confidence threshold of 0.9 showed the best performance with a 34% overall MOTA score. Results are also presented for the confidence threshold of 0.9 and for all the different tracking evaluation metrics in Table 2 and Table 3. Clearly, the fine-tuned object detector used with optical flow boosting performs best for Deep-SORT. Table 9 shows some qualitative results.

Table 9. Comparative Results of Finetuning Deep SORT.

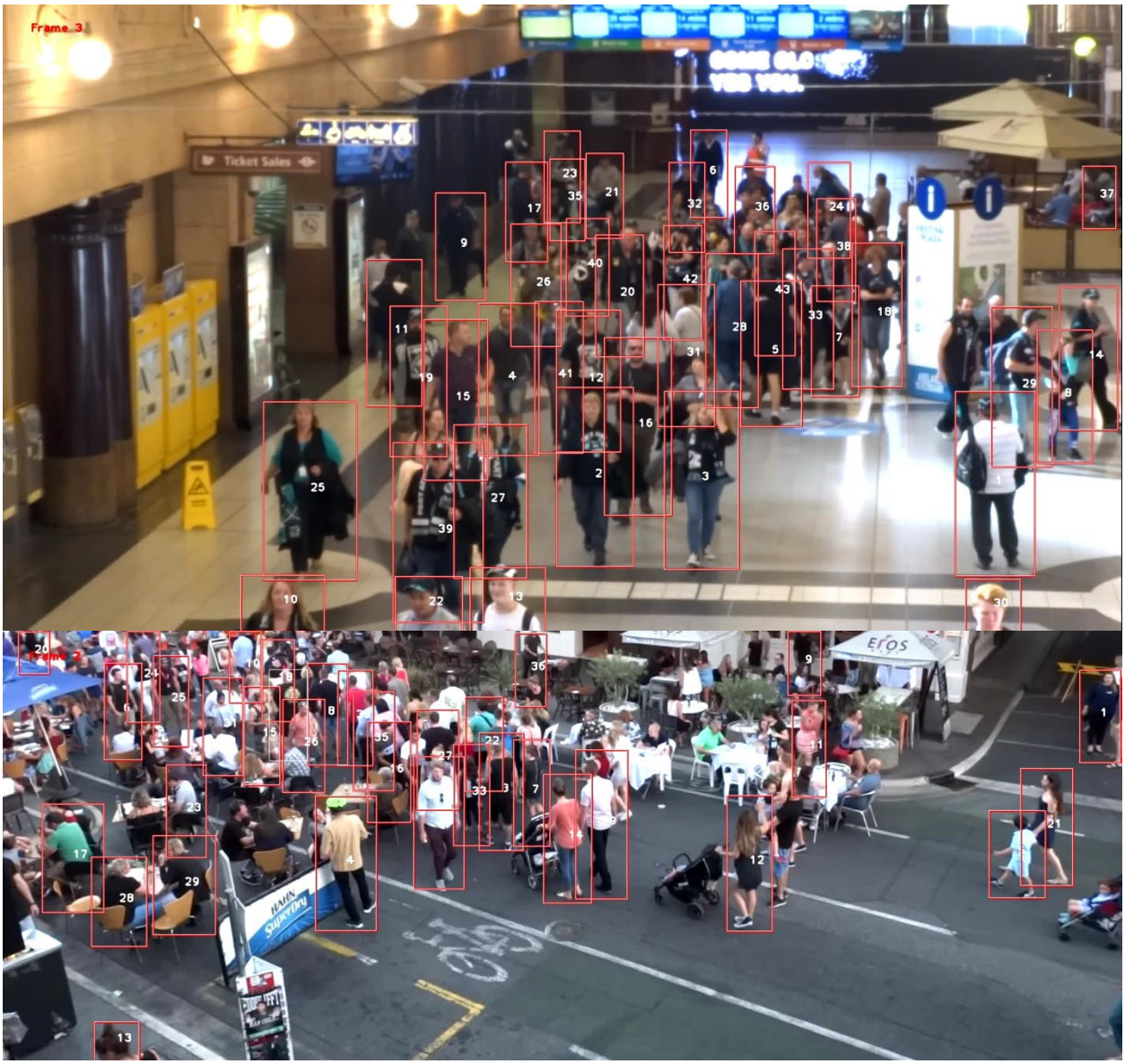
Test no.	min_det_conf	n init	Max IOU dist	Max age	max_cos._distance	MOTA
1	0.5	10	0.3	15	0.2	38.00%
2	0.5	10	0.4	15	0.2	56.90%
3	0.5	10	0.5	15	0.2	72.30%
4	0.5	10	0.6	15	0.2	73.70%
5	0.5	10	0.7	15	0.2	70.10%
6	0.5	10	0.6	10	0.2	73.70%
7	0.5	10	0.6	20	0.2	73.70%
8	0.5	5	0.6	15	0.2	**85.40%**
9	0.5	15	0.6	15	0.2	59.10%
10	0.5	5	0.6	15	0.3	**85.40%**
11	0.5	5	0.6	15	0.4	**85.40%**

GCNNMatch Results for pedestrian detection

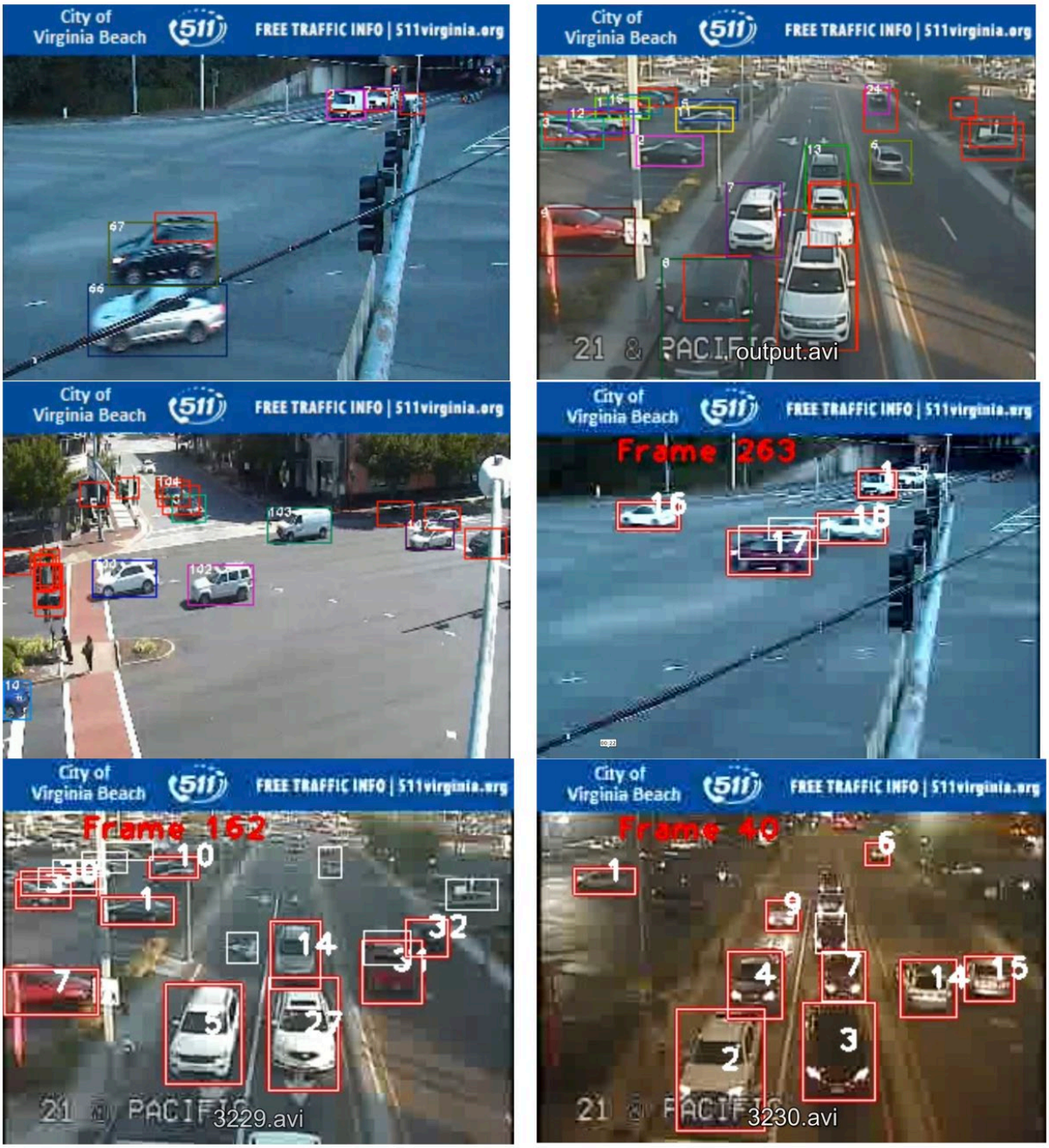



31
33
13
15
36


Frame: 397
PRIMARK
72
67
58
62
60
59
55
71
74
57
53
51
50
69
47


Frame_3
Ticket Sales
Frame_2
Eros

GCNNMatch Results for VA-Beach Vehicles



Appendix C: Additional Results for Kinematics

We did notice that the vehicle's distance from the traffic camera also contributed to an instability in the speed calculation. By plotting the object's speed versus the object's pixel distance from the camera, a threshold distance is determined in which to remove kinematic calculations outside of this threshold; an example is seen in Figure 17. We can see that the speed of each vehicle (represented by each individual line) stays under about 50mph when the vehicle is close to the traffic camera (left side of the vertical). Once each vehicle gets too far from the traffic camera, the speed becomes erratic (right side of the vertical line). Thus, a threshold distance of 132 pixels from the camera was chosen for the example. Removing the calculations of any vehicles beyond this distance reduces the possibility of erratic or incorrect data being used in future analysis. This analysis highlights the limitations of the camera system and defines their reliability.

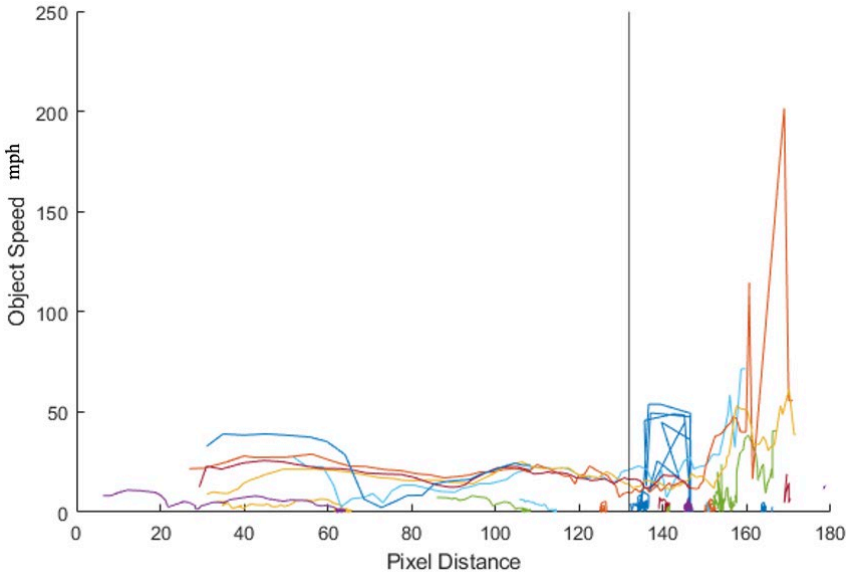


Figure 17. An example of the calculated speeds vs. distance from the camera.

In Figure 17, each separate line represents the speed of that individual object. The vertical line represents the threshold distance. This is the pixel distance in which any object speed calculation becomes unstable.