

An unsupervised learning framework for detecting adaptive cruise control operated vehicles in a vehicle trajectory data

Mohammadreza Khajeh Hosseini^a, Alireza Talebpour^{a,*}, Saipraneeth Devunuri^a,
Samer H. Hamdar^b

^a Department of Civil and Environmental Engineering, University of Illinois at Urbana-Champaign, 205 North Mathews Ave., Urbana, IL 61801, USA

^b Department of Civil and Environmental Engineering, George Washington University, 800 22nd Street NW Washington, DC 20052, USA

ARTICLE INFO

Keywords:

Vehicle trajectory
Aerial data collection
Automated driving
Advanced driver assistance systems
Clustering

ABSTRACT

The traffic dynamics are expected to change with the widespread utilization of advanced driver assistance systems (ADAS). Currently, simulation tools are adopted to capture the impacts of ADAS technologies on traffic dynamics. Real-world data collection of different ADAS technologies is required to support realistic modeling of these technologies in simulation tools. Vehicle trajectories are one of the cornerstones of modern traffic flow theory with applications in driver behavior studies and automated vehicle (AV) research. Unfortunately, the current trajectory datasets fail to provide any information on the utilization of ADAS technologies. This study proposes collecting and using a new trajectory dataset that contains multiple instances of probe vehicles using adaptive cruise control (ACC) to identify ACC-type behavior across the entire trajectory dataset. Since the trajectory data is not labeled based on ACC utilization, clustering is an excellent approach to arrange similar trajectories in the dataset into the same group. Using this dataset combined with clustering, this study identifies the vehicle trajectories with similar dynamics to the vehicles using ACC.

1. Introduction and background

SAE International (2021) classifies vehicles into six automation levels ranging from level zero with no automation feature to level five with full automation of driving tasks in all driving conditions. The lower levels of automation (levels 0–2) include Advanced Driver Assistance Systems (ADAS), which are developed to support drivers and improve safety by automating some of the driving tasks. The driver-assistance features include adaptive cruise control (ACC), traction control, lane maintenance, and emergency braking, which have become more prevalent on roads. Most of these existing ADAS technologies support the local motion planning of the vehicles, which could result in different driving behavior compared to conventional driving behaviors. The underlying dynamics of Traffic flow are expected to change with the widespread adoption of Advanced Driver Assistant Systems (ADAS) and automated driving technologies (Rahmati, Khajeh Hosseini, Talebpour, Swain, & Nelson, 2019). Meanwhile, utilizing simulation tools is the only feasible approach to capturing such technologies' impacts on traffic flow dynamics at high market penetration rates (MPR). Unfortunately, there is a clear mismatch between how these technologies are being developed and how they are being modeled in traffic simulation

tools (Dosovitskiy, Ros, Codevilla, Lopez, & Koltun, 2017). Accordingly, bridging that gap is a critical step towards realistic modeling of the impacts of automation on traffic flow dynamics. Achieving such an objective is only possible through real-world data collection from various automated driving technologies. Several studies have collected data from various levels of vehicle automation (Caesar et al., 2020; Maddern, Pascoe, Linegar, & Newman, 2017). Unfortunately, these studies mainly focus on the vehicle and its vicinity and do not provide a comprehensive view of the entire traffic stream, essential information to analyze traffic flow dynamics. Accordingly, there is a critical need to collect such a dataset that contains information from all vehicles in the segment of interest. One meaningful format of such a dataset can be in the form of vehicle trajectories.

Vehicle trajectory is a concise yet comprehensive way to store data of an individual or collective group of vehicles for both micro- and macro-level traffic analyses. With the advancements in sensing and imaging technologies, the trajectories can be generated using cameras, infrared sensors, RADAR, and LiDAR. However, video-based imaging has been the most popular method of extracting vehicle trajectories. The early studies collected vehicle trajectories using pole-mounted

* Corresponding author.

E-mail addresses: mk49@illinois.edu (M. Khajeh Hosseini), ataleb@illinois.edu (A. Talebpour), sd37@illinois.edu (S. Devunuri), hamdar@gwu.edu (S.H. Hamdar).

<https://doi.org/10.1016/j.eswa.2022.118060>

Received 11 February 2022; Received in revised form 2 July 2022; Accepted 2 July 2022

Available online 9 July 2022

0957-4174/© 2022 Published by Elsevier Ltd.

cameras at intersections (Michalopoulos, 1991; Zhou, Gao, & Zhang, 2007). Aerial imagery for trajectory extraction overcomes issues related to occlusion and cluttering that are associated with using pole-mounted cameras. Satellites, helicopters, Unmanned Aerial Vehicles (UAVs), and airplanes are the primary means of obtaining aerial videos and images.

Some of the existing vehicle trajectory datasets are FHWA Next Generation Simulation Models (NGSIM) (FHWA, 2006), Strategic Highway Research Program (SHRP2) (Hankey, Perez, & McClafferty, 2016) and TrafficNet (Zhao, Guo, & Jia, 2017). NGSIM is a well-known open-source trajectory dataset collected in 2006 using digital cameras at different locations, including US Highway 101 and Interstate 80 freeway. The vehicle trajectories are extracted from the images of multiple cameras combined to create a single image that looks like an aerial shot. The NGSIM trajectory data contains the location of each vehicle at a frequency of 10 Hz over a 1600 to 3200 ft (488 to 975 m) stretch of roadway. However, the NGSIM data suffers from noise and inaccurate detection due to the low-resolution cameras at a considerable distance. Coifman and Li (2017) analyzed the NGSIM dataset and confirmed inaccuracies in the speed and positioning of vehicles. The SHRP2, in collaboration with Virginia Tech Transportation Institute (VTTI), had collected naturalistic driving data in 2012 (Hankey et al., 2016). The dataset includes more than 5 million trips that include sensory data such as speed, location, acceleration, and also vehicle and driver characteristics. The SHRP2 dataset is not freely available to public access and has a limited preview. This dataset is collected using probe vehicles, and the collected data is limited to the field of view of the onboard sensors and does not entirely define the surrounding vehicles and traffic dynamics. The TrafficNet provides processed naturalistic data with libraries for researchers to perform data analytics (Zhao et al., 2017). TrafficNet separated driving into six scenarios, such as free flow, car-following, cut-in, and the like, and classified the entire dataset into these scenarios curated to research. It is a web-based platform with MySQL database used to store the information. The HighD dataset (Krajewski, Bock, Kloecker, & Eckstein, 2018) is another dataset that offers vehicle trajectories on German Autobahn. HighD accounts for variability in traffic composition by collecting data at six different locations. It has a truck ratio varying from 0%–50% and trajectories collected at different times of the day. The trajectories are analyzed and classified into specific maneuver types, such as lane changes and critical maneuvers. More recently, the pNEUMA dataset (Barmounakis & Geroliminis, 2020) used a swarm of drones to collect arterial traffic data in sequential sessions with blind gaps in between sessions. Their objective was to study Origin-Destination information, travel time and congestion propagation, and lane-changing behavior.

While the datasets mentioned earlier provide the means to analyze driver behavior, they fail to provide any information on the utilization of the ADAS by drivers. This is unfortunate, as ADAS technologies are becoming an integral part of our roadways, and capturing their impacts on traffic flow dynamics and congestion is critical. There is another key challenge associated with utilizing the existing datasets: considering that ADAS technologies have a compound annual growth rate of 12% (Bhutani & Bhardwaj, 2019), there is a very high chance that some of the vehicles in the existing datasets already utilized ADAS technologies. Assuming that ADAS technologies can potentially change the interactions among drivers on the road and lead to new traffic flow dynamics and possibly new types of high-risk driving instances, utilizing these datasets with the assumption that human drivers control all vehicles can lead to unrealistic assessments and bias. One of the more recent studies by Makridis, Mattas, Anesiadou, and Ciuffo (2021) has also recognized these shortcomings and developed the OpenACC database. It is an open-access database with two public and two private test campaigns conducted in Sweden and Hungary. The data collection process involves using onboard sensors such as accelerometers and global navigation satellite systems (GNSS) to record the trajectories of the probe vehicles. This limits the microscopic and macroscopic analysis to only probe vehicles, and the interaction between the adaptive

cruise control (ACC) and conventional vehicles in the traffic is not captured.

Considering the challenges mentioned earlier and focusing on ACC as the ADAS technology of interest, the main contributions of this study are: (1) to develop a robust and scalable methodology to identify vehicles with ACC-type behavior in a vehicle trajectory dataset, and (2) to investigate the difference in the behavior of conventional vehicles and the vehicles using a full range ACC, or the ones with similar dynamics, in a real-world setting. In the process of achieving these objectives, this study also introduces a robust, scalable, and cost-effective methodology for vehicle trajectory data collection through aerial imagery using aerial videography. This data collection approach can be utilized for collecting vehicle trajectory data from different levels of automated driving. Furthermore, to test and validate the proposed algorithms, a vehicle trajectory dataset is collected on Interstate 35 near Austin, TX. This trajectory dataset contains information on five platoons of three ACC-operated vehicles mixed with human-driven vehicles on Interstate 35 in Austin, TX. Table 1 summarizes different aspects of the existing trajectory datasets discussed in this section as well as the new vehicle trajectory dataset collected in this study (Austin-ACC). In this table, the column for view specifies if the data collection is performed in the bird's eye view (BEV) or first-person view (FPV). BEV refers to the data collection using aerial videography (HighD, pNEUMA, and Austin-ACC) or videography from tall structures (NGSIM). FPV refers to the data collection using cameras and sensors mounted on probe vehicles. BEV data collection is usually performed over a fixed region and captures the trajectory of all the vehicles traveling through the fixed segment. FPV is performed by the probe vehicles traveling through the study area and is limited to the trajectory of the vehicles in the surrounding probe vehicles. Moreover, Table 1 also specifies if the dataset contains the naturalistic behavior of road users or is collected in a controlled testing environment. Moreover, in this table, the travel time and segment length are only reported for the datasets collected over a fixed segment of the roadway rather than the transportation network. According to Table 1, OpenACC and the vehicle trajectory dataset collected in this study (Austin-ACC) are the only datasets that include ACC-operated trajectories. In addition, the vehicle trajectory collected in this study is performed in bird's eye view and includes not only the trajectory of the ACC-operated vehicles but also all the vehicles traveling the study area. This comprehensive dataset provides the opportunity to further investigate the impacts of the ACC-operated vehicles on the traffic stream and identify the vehicles with ACC-type behavior in the traffic stream.

The remainder of this paper is organized as follows: the next section presents the details and steps of the proposed methodology for detecting ACC-type behavior. This section is followed by the details of the data collection methodology, as well as the location and procedure of data collection. The paper continues by presenting the clustering results of different distance measures and statistical and qualitative discussion on the results. Finally, the paper is concluded with a summary of findings and future research needs.

2. Methodology

Despite fundamental differences between ACC behavior and human drivers, it is pretty challenging to differentiate between the vehicles utilizing ACC and conventional vehicles in a vehicle trajectory dataset (not labeled based on the ACC utilization). This is due to several factors, including noise in data, limited length of the trajectory, multi-vehicle interactions, and the like.

Clustering is an unsupervised approach to identify and group similar data points. Accordingly, Clustering seems to be an excellent approach to arrange similar trajectories in the dataset into multiple groups. A recent study by Makridis et al. (2021) indicated that the ACC-driven behave much more homogeneously even among different makes and models compared to human-driven vehicles. In order to distinguish

Table 1

Comparison of the existing vehicle trajectory datasets and the trajectory dataset collected in this study on Interstate 35 near Austin, TX.

Dataset	View	Sensors	Traffic condition	Vehicle type	ADAS technology	Environment	Behavior	Travel time	Segment length	Year
NGSIM	BEV	Camera	Congested	Small, large	–	Highway	Naturalistic	74.4 s	1870 ft	2006
SHRP2	FPV	Speedometer, accelerometer, camera, radar, GPS	Variable	Small	–	Highway, rural	Naturalistic	N/A	N/A	2012
TrafficNet	FPV	Camera, radar, CAN	Variable	Small	–	Urban	Naturalistic	N/A	N/A	2017
HighD	BEV	Camera	Variable	Small, large	–	Highway	Naturalistic	14.6 s	1345 ft	2018
pNEUMA	BEV	Camera	Congested	Small, large	–	Urban	Naturalistic	N/A	N/A	2020
OpenACC	FPV	Accelerometer, GPS	Variable	Small	ACC	Highway, testing facility	Naturalistic, experimental	N/A	N/A	2021
Austin-ACC	BEV	Camera	Congested	Small, large	ACC	Highway	Naturalistic	14.2 s	500 ft	2019

ACC-driven vehicles from human-driven vehicles, this study proposes the use of probe vehicles equipped with ACC in the traffic stream during the data collection. The behavior of these probe vehicles provides the basis to identify the trajectories that have comparable dynamics to the vehicles using ACC. Clustering helps to categorize the vehicles with similar behavior to the probe ACC vehicles into the same group. It is assumed that the behavior of vehicles is captured in their trajectory. As a result, comparing the similarity between the trajectories is an essential step in grouping them into the same or different clusters.

It is important to note that several studies utilized different clustering algorithms on vehicle trajectory data. A good example of these studies is a study by Talebpour, Mahmassani, and Bustamante (2016). They utilized a clustering approach to minimize the broadcasting need for Vehicle-to-Vehicle (V2V) communications. In their approach, each cluster member can only communicate with its cluster head and cluster heads disseminate the information among all their cluster members as well as other cluster heads. This approach was shown to reduce signal interference and information loss in V2V communications network. The goal of their clustering, however, was fundamentally different from this study (i.e., they did not consider similarity in the behavior as the base for clustering). Their clustering aimed at increasing the life of a cluster considering the communications range and vehicles' speed. Note that they selected that objective based on the fact that re-clustering the vehicles is a computationally costly process and increases communications loss significantly in the network as all the vehicles had to broadcast their information. In another recent study, Zhong, Lee, Nejad, and Lee (2020) utilized clustering to group vehicles for more efficient Cooperative Adaptive Cruise Control (CACC) strategies that reduce the disturbance in the traffic stream. Another recent study by Wang, Lin, Wu, and Yu (2020) presents a methodology to group vehicles into a pre-defined number of classes based on their behavior. The outcome of the proposed clustering methodology can be utilized to improve motion planning for automated vehicles. There is, however, a key difference between the objectives of this study with the aforementioned studies: This study aims for a more strict objective, i.e., any mis-classification of the probe ACC vehicles is considered a failure.

2.1. Vehicle trajectory as a time series of data points

A full range ACC system, which is among the core features of automated vehicles, can adjust the vehicle's speed (i.e., longitudinal driving) in all ranges of traffic state from the stop and go to free flow. The ACC system automatically adjusts the vehicle's speed using the throttle and brake to maintain a desired distance or a desired time headway to the leading vehicle (He et al., 2019). Consequently, this study

considers the speed, acceleration, time headway, and space headway, as well as their changes over time as potential features in the clustering process. Each vehicle trajectory is a time series of data points with features, including time, location, speed, and acceleration. Based on the location and time, the vehicle's leader and follower are identified, and consequently, the time and space headways are estimated. Fig. 1 presents examples of time series of speed, acceleration, time headway, and space headway of three ACC-driven vehicles and their three immediate leading and following vehicles. According to this figure, the time series of the vehicles using ACC (3541, 3544, and 3548) are more similar compared to the other immediate surrounding vehicles. This similarity is more noticeable for the time headway (Fig. 1(c)) and space headway (Fig. 1(d)) series. The similarity between the trajectories can be quantified based on a distance measure between the trajectories.

2.2. Distance measure between trajectories

A vehicle trajectory is a time series of different features, including location and other features such as speed and acceleration. Comparing the similarity or dissimilarity between the trajectories is an essential step in grouping them into the same or different clusters (Yuan, Sun, Zhao, Li, & Wang, 2017). A typical distance measure used in the clustering approaches for static data points is the Euclidean distance. The Euclidean distance is also used to compare the distance between the data points of two trajectories referring to the same time step. The Euclidean distance can be used to measure the similarity of trajectories, T^i , and T^j with a similar length of n time steps and d dimensional data points, p :

$$D_{Euclidean}(T^i, T^j) = \frac{1}{n} \sum_{k=1}^n \sqrt{\sum_{m=1}^d (p_k^{i,m} - p_k^{j,m})^2} \quad (1)$$

where $p_k^{i,m}$ refers to the m th feature of k th point of trajectory T^i , and $p_k^{j,m}$ refers to the m th feature of k th point of trajectory T^j . One of the challenges with the Euclidean distance is that each data point in one trajectory is only compared to one data point in the second trajectory with the same time step. However, in most cases, including this study, the trajectories' length (time steps) are not equal, and the euclidean distance is not a suitable distance measure. To address this issue, Besse, Guillouet, Loubes, and Royer (2016) divides the distance measurements between trajectories with different lengths into two groups of warping-based distances and shape-based distances. The warping-based distances and shape-based distances allow measuring the distance between trajectories with different lengths.

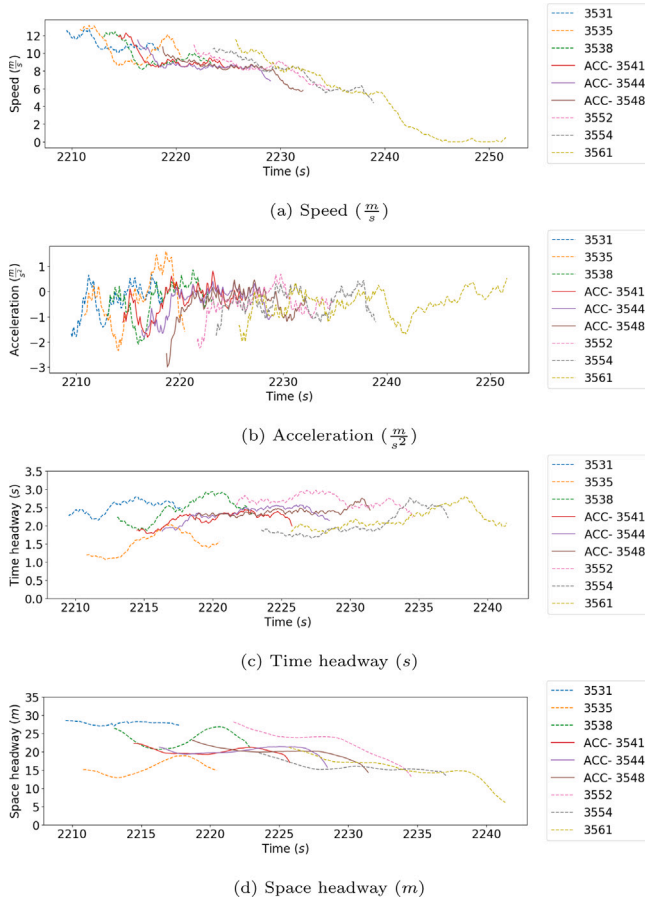


Fig. 1. Trajectory data example: first run.

2.2.1. Warping-based distance measures

Warping-based distance measures address the challenge of length difference by finding an optimal (i.e., matching) alignment between the two trajectories regardless of their length. The objective of these distance measures is to find the warping path, w , between two trajectories, T^i (with n^i points) and T^j with (with n^j points), with the optimal cost when arranging two trajectories in the form of a $n^i \times n^j$ grid. The minimum or maximum warping cost depends on the cost function to measure dissimilarity or similarity between points. The warping path allows us to match data points from one trajectory to those with different indexing in the second trajectory. Two of the common warping distances include the dynamic time warping (DTW) (Berndt & Clifford, 1994) and longest common subsequence (LCSS) (Vlachos, Kollis, & Gunopulos, 2002).

DTW distance identifies the minimum cost of the warping path between two trajectories T^i and T^j recursively:

$$D_{DTW}(T^i, T^j) = \begin{cases} 0 & \text{if } n^i = n^j = 0 \\ \infty & \text{if } n^i = 0 \text{ or } n^j = 0 \\ \begin{cases} cost_{DTW}(p_1^i, p_1^j) \\ + \min \begin{cases} D_{DTW}(rest(T^i), rest(T^j)), \\ D_{DTW}(rest(T^i), T^j), \\ D_{DTW}(T^i, rest(T^j)), \end{cases} \end{cases} & \text{otherwise} \end{cases} \quad (2)$$

where $rest(T^i)$ refers to the time series T^i without its first time step data point. The cost structure in the DTW is the dissimilarity between the

data points based on the Euclidean distance:

$$cost_{DTW}(p_1, p_2) = \|p_1 p_2\|_2 \quad (3)$$

LCSS distance finds the longest common subsequence between two trajectories, T^i and T^j , by counting the number of times the difference between pairs of data points is less than ϵ (a model parameter to be selected) recursively:

$$D_{LCSS, \delta, \epsilon}(T^i, T^j) = \begin{cases} 0 & \text{if } n^i = 0 \text{ or } n^j = 0 \\ 1 + D_{LCSS}(rest(T^i), rest(T^j)) & \text{if } cost_{LCSS}(p_1^i, p_1^j) = 1 \text{ and } |n - m| \leq \delta \\ \max \begin{cases} D_{LCSS}(rest(T^i), T^j) \\ D_{LCSS}(T^i, rest(T^j)) \end{cases} & \text{otherwise} \end{cases} \quad (4)$$

where δ controls how far in time the measurement can go to find a matching data point. The cost structure of the LCSS distance is as follows:

$$cost_{LCSS}(p_1, p_2) = \begin{cases} 1 & \text{if } \|p_1 p_2\|_2 < \epsilon \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

The final LCSS distance between two trajectories is divided by the minimum of the length of the two trajectories resulting in a value between 0 and 1. Unlike the DTW, the LCSS distance is a similarity measure between two trajectories. δ and ϵ are the two parameters of the LCSS distance that give some control over noise. If ϵ is too small, the longest common subsequence could become too small, resulting in a very low similarity, and if the ϵ is too large, the common subsequence could become too large, resulting in a high similarity.

2.2.2. Shape based distances

Shape-based distances capture the geometric similarity between two time series (Besse et al., 2016). Two of the common shape-based distance measures for time series are Fréchet distance (Fréchet, 1906) and Hausdorff distance (Hausdorff, 1914). Fréchet distance is a measure of similarity between two curves, A and B :

$$D_{Fréchet}(A, B) = \inf_{\alpha, \beta \in X} \max_{t \in [0, 1]} \{\|A(\alpha(t)), B(\beta(t))\|_2\} \quad (6)$$

One intuitive definition of the Fréchet distance is the minimum cord length sufficient to connect two travelers along two different curves, each traveling forward at a different speed. $\alpha(t)$ and $\beta(t)$ are continuous and increasing functions such that $\alpha(0) = 0$, $\alpha(1) = m$, $\beta(0) = 0$ and $\beta(1) = n$, and m and n are the last vertices of curve A and B respectively. Therefore, $A(\alpha(t))$ and $B(\beta(t))$ are the location of the two travelers at time t on curve A and curve B . The Fréchet distance between two trajectories can be estimated using the discrete Fréchet distance algorithm proposed by Eiter and Mannila (1994).

Hausdorff distance is another shape-based distance that can be used to measure the distance between two trajectories T^i and T^j . The maximum distance of the points in one trajectory to the nearest point in the other trajectory is the Hausdorff distance. Accordingly, for each point in trajectory T^i , the infimum of distances from this point to all the points in trajectory T^j are estimated, and the supremum of these infima is found for each trajectory. The Hausdorff distance is the maximum of the two suprema from the two trajectories:

$$D_{Hausdorff}(T^i, T^j) = \max \left\{ \sup_{p^i \in T^i} \inf_{p^j \in T^j} \|p^i p^j\|_2, \sup_{p^j \in T^j} \inf_{p^i \in T^i} \|p^j p^i\|_2 \right\} \quad (7)$$

2.3. Clustering algorithm for non-metric distance

A distance measure is considered a metric if it satisfies non-negativity, symmetry, reflexivity, triangle inequality, and indiscernible identity. Fréchet and Hausdorff distances meet metric requirements, while the warping distances, DTW and LCSS, do not satisfy

the triangle inequality (Besse et al., 2016). Hierarchical clustering and affinity propagation (AP) (Frey & Dueck, 2007) are among the clustering algorithms capable of handling non-metric distance measures in the clustering process by directly taking the distance matrix between the trajectories. Alternatively, some clustering methods, such as DBSCAN clustering, may perform better with metric distances. This study adopts affinity propagation to increase the robustness of the clustering process as it does not require metric distance matrix or specifying the number of clusters (unlike hierarchical clustering).

2.3.1. Affinity Propagation (AP)

Affinity propagation (AP) clustering considers the similarity between the trajectories and evaluates all the potential cluster heads (exemplars). Two types of messages, responsibility, and availability, are exchanged between trajectories in AP. The responsibility message, $r(i, j)$, is sent from the trajectory, T^i to a candidate exemplar trajectory, T^j , and quantifies the appropriateness of trajectory T^j to serve as the exemplar for trajectory T^i considering all the other candidate exemplars. The availability message, $a(i, j)$, is sent from exemplar candidate, trajectory T^j , to trajectory T^i , and indicates the fitness of T^j to serve as the exemplar of T^i considering the support from other trajectories that take T^j as their exemplar. AP starts by considering the similarity matrix between the trajectories and setting the availability between all pairs of trajectories to zero. AP updates the responsibility and availability messages between pairs of trajectories through an iterative process until they converge. For each trajectory, T^i , the trajectory T^j that maximizes sum of the $r(i, j)$ and $a(i, j)$ is its exemplar (cluster head).

The similarity between a pair of trajectories is defined based on the distance measures considered in the previous section. LCSS is the only distance measure in this study that directly provides the similarity between two trajectories. All the DTW, Hausdorff, and Fréchet distances are dissimilarity measures; thus, the negative of those distances are considered as the similarity measure between two trajectories.

3. Data collection

Recent years have seen a significant increase in using aerial vehicles for remote sensing applications such as photogrammetry imaging. Such an increase is mainly due to the lower operating costs than hiring an aircraft. This study also proposes the use of aerial videography of the traffic stream. The trajectory of the vehicles can be extracted from the video frames recorded in the bird's-eye view from a segment of the roadway (Fig. 2.a). In every video frame, the location of the vehicles can be estimated for a fixed coordinate system and reference point on the ground. Every video recording is converted to a sequence of images (i.e., frames) separated at a constant rate over time (e.g., 25 frames per second). Tracking the location of any vehicle over the sequence of images enables extracting the vehicle's trajectory over time. The vehicle trajectory extraction is performed in four steps: *image stabilization*, *vehicle detection* (Fig. 2.b), *vehicle tracking* (Fig. 2.c), and *trajectory construction*. All the images are transformed to match a reference field of view in the image stabilization step. Then the vehicles are detected in every frame and tracked over the sequence of images. Finally, the vehicle's location and trajectories are constructed by converting the image coordinates to the adopted reference coordinates on the ground.

The key contribution of this section is to present a pipeline to convert video files to vehicle trajectories. Although the utilized models and algorithms mostly exist outside of this pipeline, to the best of our knowledge, this paper is one of the very few research studies that outline the pipeline and steps needed to extract the trajectory of vehicles using aerial videography. Accordingly, while none of the models are developed by the authors (although carefully calibrated), this detailed information is included in the paper to help interested readers with replicating the process. It should be noted that the proposed trajectory extraction methodology is modular, and there exist different candidate models and algorithms that can be adopted for

each of the four steps. This study does not focus on comparing all the possible candidate models for each of the trajectory extraction steps. In return, this study focuses on keeping the methodology simple and introducing some practical and popular models that could be utilized in each step. These steps are further elaborated in the following sections.

3.1. Image stabilization

The location of every vehicle in an image frame is estimated by converting its position on the image map to the fixed coordinate system picked on the ground. Consequently, it is essential to find the mapping function between the image coordinate to the adopted ground coordinate. Image stabilization is the process of converting the field of view of all the image frames to a reference image for which the mapping function to the ground coordinate is known. Fig. 3 presents an example of a reference image and the input and output of the image stabilization. Ensuring that the input image covers all (or majority) of the study segment is vital. Otherwise, as depicted in Fig. 3, the missing areas will be black in the stabilized image.

The image stabilization is performed in three steps; first, detecting the key features in both reference and input images, second, finding the matching features between the two images, and third, estimating transformation between them. The process searches for specific unique patterns, such as corners, which are good features that can be tracked from one image to another image. There exist different algorithms for good features detection in images such as Harris corner detector (Derpanis, 2004), Scale-Invariant Feature Transform (SIFT) (Lowe, 2004), Speeded up Robust Feature (SURF) (Bay, Tuytelaars, & Van Gool, 2006), and Oriented FAST and Rotated BRIEF (ORB) (Rublee, Rabaud, Konolige, & Bradski, 2011). ORB and Harris corner detectors are among the fast feature detectors. However, SIFT and SURF are the top-performing feature detectors in terms of scale and transformation. The result of the comparison between the SIFT, SURF, and ORB by Karami, Prasad, and Shehata (2017) suggested that SIFT has the best performance in most image distortion scenarios. As a result, this study employs the SIFT algorithm to detect good features. The SIFT algorithm improves the image stabilization by identifying features in both reference and input images that are invariant to distortions such as rotation, scale, and point of view (see Fig. 3).

The second step in image stabilization is matching the features between the reference image and the input image. One naive approach is to compare every feature in the reference image with every feature in the input image to find the best matching pairs. This approach is known as brute-force matching. The deficiency with this approach is that it takes a lot of computation time that is impractical for a video data collection at a high frame rate (e.g., 25 frames per second). Instead, to improve the computation speed, the Fast Library for Approximate Nearest Neighbors (FLANN) matcher (Muja & Lowe, 2013) is utilized to match features between the images. The FLANN algorithm adopts a distance measure (e.g., L2 norm) to compare the level of matching between the descriptors of two features from two images. A lower distance between the descriptors indicates a better match between the two features. For every feature in the first image, the FLANN algorithm finds the approximate K best (i.e., nearest) matching features in the second image.

The algorithm may identify many matching features; however, the top matching pairs are picked considering Lowe's ratio test (Lowe, 2004). In this test, features m (in reference image) and n (in input image) are a top matching pair if n is the best match for m . Also, the distance between the descriptors of m and n should be less than a threshold multiplied by the distance between m and its second-best match. Accordingly, $K = 2$ for the FLANN algorithm and the threshold of 0.7, as recommended by Lowe (2004), are adopted to find the top pairs of matching points.

The final step of the image stabilization is finding the perspective transformation between the reference and input images considering

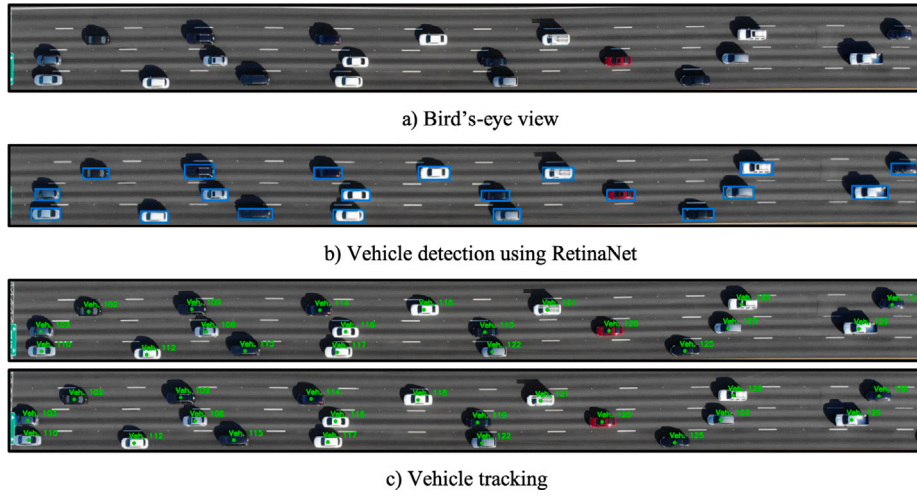


Fig. 2. Vehicle detection and tracking in aerial images.

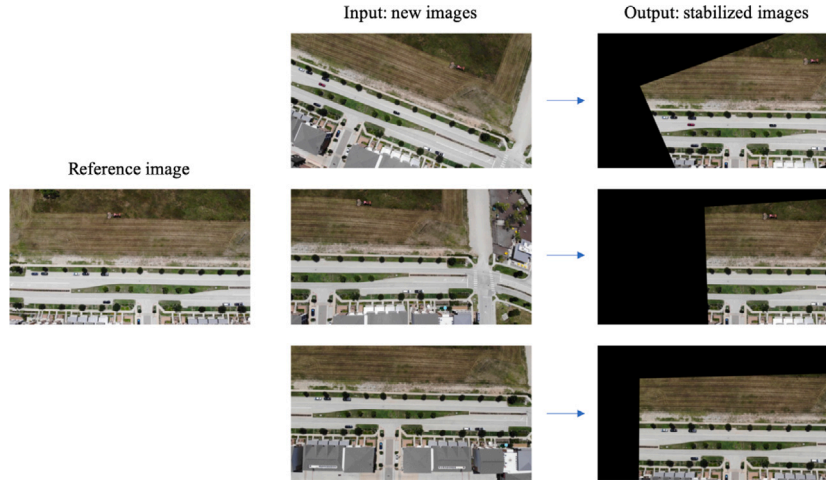


Fig. 3. Image stabilization.

the best matching features among them. Homography, H , is a 3×3 transformation matrix that maps the points (x_2, y_2) from one image to the points (x_1, y_1) in another image in accordance with the following equation:

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = H \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} \quad (8)$$

Four correct matching points are enough to estimate the homography transformation between the two images. However, there are chances that some of the matches are incorrect. The Random Sample Consensus (RANSAC) algorithm (Derpanis, 2010) is a technique to find the model parameters from a dataset with many outliers. The RANSAC is an iterative process that searches for the model parameters that agree with most data points. As a result, the RANSAC technique is utilized to find the homography parameters from the best matches between the images. Then, every pixel in the input image can be transformed to the perspective of the reference image using the homography matrix to create a stabilized image.

3.2. Vehicle detection

Object detection is the task of providing both the class and location of the objects in an input image. The classical approach for object detection is to identify the informative regions in the image that contain

objects of interest, then extract semantic and representative features from them, and finally, classify the objects in those regions. Deep neural networks (DNN) made a great performance breakthrough in the task of object detection due to the capacity of the convolutional neural networks (CNN) to learn more complex features compared to shallower models (Zhao, Zheng, Xu, & Wu, 2019). There are multiple popular CNN based object detectors such as R-CNN (Girshick, Donahue, Darrell, & Malik, 2014), Fast R-CNN (Girshick, 2015), Faster R-CNN (Ren, He, Girshick, & Sun, 2015), SSD (Liu et al., 2016), YOLO (Redmon, Divvala, Girshick, & Farhadi, 2016) and RetinaNet (Lin, Goyal, Girshick, He, & Dollár, 2017).

Some of the DNN object detectors, such as the R-CNN family (Girshick et al., 2014; Ren et al., 2015) are two-stage object detectors. In the first stage, regions of interest are identified, and these regions are classified in the second stage. Other networks such as the SSD (Liu et al., 2016), YOLO (Redmon et al., 2016) and RetinaNet (Lin et al., 2017) are one stage object detectors. One-stage approaches perform the detection in one shot by classifying fixed and dense regions of the image and adjusting the location and size of each region to enclose the object inside it. One-stage object detectors are faster than the two-stage detectors but usually have lower accuracy. The lower accuracy of the one-stage approaches is due to the extreme imbalance between the number of foregrounds (objects of interest) and background regions

evaluated in every image that affect the network's training. Conventionally, the cross-entropy loss function is used to train the classification networks. The following equation estimates the cross-entropy loss for an evaluated region for K different classes.

$$Loss_{(cross-entropy)} = - \sum_{i=1}^K Y_i \log(p_i) + (1 - Y_i) \log(1 - p_i) \quad (9)$$

where p_i is the probability of class i estimated by the network, and Y_i is equal to 1 for the true class and 0 for the rest of the classes. The background regions are considered easy examples (i.e., when p_i is relatively high) to be detected by the classification network. The high number of backgrounds in the training set dominates the loss function and prevents training. RetinaNet (Lin et al., 2017) proposes the use of focal loss for the classification loss function. The focal loss uses the focusing parameter γ that puts less emphasis on the easily classified examples (i.e., when p_i is relatively high) such as backgrounds and puts more emphasis on hard examples (i.e., when p_i is relatively low). In addition, the balancing parameter α_i is used to mitigate the effect of the imbalance between different classes. The following equation presents the focal loss estimation for one region and K classes of objects.

$$Loss_{(focal-loss)} = - \sum_{i=1}^K \alpha_i Y_i (1 - p_i)^\gamma \log(p_i) + (1 - \alpha_i)(1 - Y_i) p_i^\gamma \log(1 - p_i) \quad (10)$$

Adopting the focal loss for the classification significantly improves the accuracy of the RetinaNet compared to other one-stage networks and makes the RetinaNet an ideal object detector in terms of speed and accuracy.

3.2.1. RetinaNet

This study utilizes the RetinaNet to detect vehicles in the aerial images; however, it should be noted that any of the modern CNN-based object detection models have the potential to serve as the vehicle detection module in the proposed model. Fig. 4 presents a high level network architecture of RetinaNet. This network comprises four types of subnetworks, including ResNet, feature pyramid network, and multiple boxes and classification subnets. The resNet component of the network is the fully convolutional part of the ResNet, which enables the RetinaNet to take input images with any size (He, Zhang, Ren, & Sun, 2016). At every layer of the ResNet, the feature map from the previous layer is transformed into a new feature map with a stronger semantic level. The feature map transforms from the bottom (i.e., input layer) of the ResNet to the top of it, while the spatial resolution of the feature map decreases, which is concerning for small objects. The classification and box subnetworks evaluate regions with fixed sizes and aspect ratios centered at every location in the feature map for potential objects. Consequently, both a strong semantic level and proper spatial resolution are the requirements to detect objects in every feature map. The ResNet is complemented with the feature pyramid network to address these requirements. The feature maps are upsampled using the nearest neighbor upsampling, moving from the top of the feature pyramid network to its bottom. Each feature map in the feature pyramid network has a corresponding feature map with the same scale in the ResNet connected to it with lateral connections to improve its spatial resolution. The bottom-up and top-down transformation of the feature maps, as well as the lateral connections, are presented in Fig. 4. ResNet and feature pyramid network construct the backbone of the RetinaNet that provides five levels of semantically and spatially strong feature maps (only three levels shown in Fig. 4).

Each feature map level is used as the input to separate classification and box subnetworks. At every location on the feature map, A rectangular regions with different aspect ratios and sizes centered at that location are evaluated for the presence of any of the K object classes inside it. These rectangular regions are also known as anchor

Table 2

Training dataset for aerial vehicle detection.

Complete training dataset	
Small vehicles	11 379
Large vehicles	751
Total	12 130

boxes. These anchor boxes are the initial estimate of the areas enclosing a single object. The classification subnetworks are fully convolutional networks that output a tensor with size $(W, H, K \times A)$. W and H are the height and width of the feature map that is proportional to the input image size. The box subnetwork estimates four offset variables for each anchor box. These offsets are the differences between the center, height, and width of the anchor box and the actual bounding box enclosing the object. The box subnetwork is also a fully convolutional network that outputs a tensor with the size of $(W, H, 4 \times A)$. The fully convolutional structure of all the subnetworks of the RetinaNet enables it to take any image size as the input for inference.

3.2.2. Training RetinaNet

The open source Keras library for RetinaNet, developed by Gaiser et al. (2019), is used to build the network for vehicle detection in aerial images. The vehicles can be grouped into small and large vehicles based on their size and appearance. Accordingly, the classification subnet is adjusted to accommodate two classes of objects. In addition, the ResNet-50 (He et al., 2016) pretrained on Microsoft coco dataset (Lin et al., 2014) is adopted as the backbone for the RetinaNet. The resulting network contains more than 37.2 million parameters; however, the parameters of the backbone are fixed, which reduces the trainable parameters to nearly 13.7 million.

More than two hundred images are sampled from the collected aerial images to create a training dataset. The sampled images are manually annotated using the annotating software developed by Dutta and Zisserman (2019). This training dataset includes annotated aerial images of large and small vehicles. For this study, car, van, and pick-up truck instances are combined into the small vehicle class, and trucks, buses, and recreational vehicles (RVs) are combined into the large vehicle class. The training dataset consists of 220 aerial images with 4K (4096×2160 pixels) resolution, and the number of instances for each class is presented in Table 2.

The dataset is divided into training (training and validation) and testing sets with the proportions of 0.9 and 0.1, respectively. In addition, data augmentation in the form of random image transformation and scaling is used to further improve the training dataset's size and, ultimately, the generalization of the network. The loss function of the RetinaNet considers both the focal loss for classification task (Eq. (10)) and the standard smooth L_1 loss (Girshick, 2015) for box regression. Training is the iterative process of adjusting the trainable parameters of the model to gradually minimize the loss function. The focusing parameter, $\gamma = 2$, and balancing parameter, $\alpha = 0.25$ are adopted for the focal loss based on the findings of Lin et al. (2017). The model is trained with a batch size of four images due to the large image size (4096×2160 pixels). The batch size could be increased by reducing the image size by cropping out the study area's background or resizing the images. The model trained up to 45 epochs (45 complete iterations over the entire dataset) is selected as the vehicle detector in the aerial images. Table 3 presents the performance of the trained model on the test dataset in the form of average precision for each class and the mean Average Precision (mAP) for Intersection over Union threshold (IoU) of 0.5. The trained RetinaNet is used for the vehicle detection step of the trajectory extraction. The input to the vehicle detection is a stabilized image, and the output is the coordinates of the bounding boxes enclosing the vehicles in the image. Fig. 2.b presents the detected vehicles and their visualized bounding boxes.

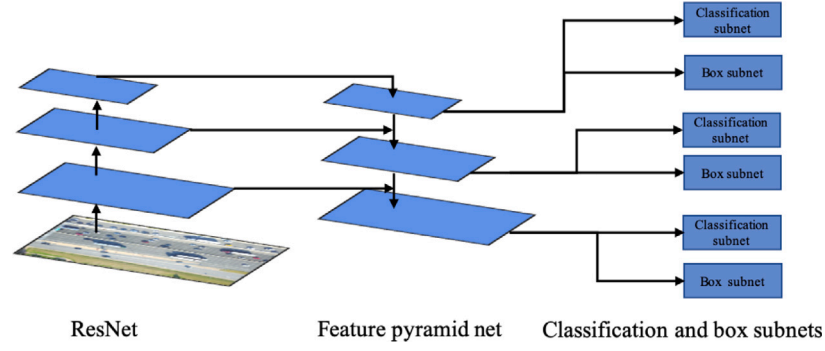


Fig. 4. RetinaNet architecture (Lin et al., 2017).

Table 3

Detection model performance on testing dataset.

	Instances	Average precision
Small vehicles	1316	0.9938
Large vehicles	98	0.9302
mAP using weighted average of precision		0.9894
mAP		0.9620

3.3. Vehicle tracking

Tracking is the process of linking the new detections to previous observations. The tracking methodology adopted for this study includes data association and track maintenance. Data association is the process of associating the detected vehicles in the current frame to the vehicles identified in the previous ones. The track maintenance is in charge of initiating new tracks, maintaining them, and deleting them.

The track maintenance initiates tracks with unique ids to all the vehicles detected in the first frame. After that, for every frame, all the newly detected vehicles are compared with the existing tracks using the data association. The tracks are updated as a new detection is associated with them. A new track is constructed for any new observation that is not associated with the current tracks. Moreover, if a track is not updated in the last n previous frames, the track maintenance deletes that track. A track object maintained by the track maintenance contains both the unique id of the track and the coordinates of the center of the bounding box of its last observation. $n = 5$ is found to be appropriate for the images collected at 25 frames per second when using the trained vehicle detector (i.e., RetinaNet) of this study.

The simple data association considers the euclidean distance between the center of the bounding boxes of the newly detected vehicles and the ones for the tracks using a greedy policy. Every new observation is associated with the closest track, considering the Euclidean distance between the center of the bounding box of the new detected vehicle and the ones for the other tracks. This nearest center association is based on the assumption that the movement of the center of a vehicle between frames is less than the distance between the center of different vehicles. This assumption can be validated by estimating the maximum movement of a vehicle at maximum speed between two frames compared with the center to center lateral and longitudinal spacing between the vehicles. Moreover, the tightness of the bounding boxes from the vehicle detection plays a key role here in reducing the noise in the measured euclidean distance between the bounding boxes. The input to the vehicle tracking component is the set of bounding boxes detected for each frame, and the output is in the same format but with a vehicle id added to each bounding box. The simple data association component of the tracking step works very well for this study due to: (1) images are stabilized and share the same field of view, (2) the high frequency of images ensures the movement of the vehicles is much less than the lateral and longitudinal spacing between the vehicles, and

(3) the data is collected in bird's eye view with no occlusion. For the cases that the trajectory data collection is performed using cameras mounted on a structure with the possibility of occlusion in images and also lower frame rate, more advanced association measures such as Mahalanobis distance with Kalman Filter, or even more advanced tracking algorithms such as Deep SORT (Wojke, Bewley, & Paulus, 2017) can be adopted. It should be noted that if the vehicle detection component fails to detect a vehicle in more than five frames, a new tracking id will be initiated, which is not desired. Fig. 2.c presents the vehicle ids tracked for two images, five frames apart for a video with 25 frames per second rate.

3.4. Trajectory construction

The aerial images are transformed and stabilized, considering a reference image before extracting the vehicle trajectories. The trained RetinaNet detects the vehicles in the stabilized images, and the resulting bounding boxes are used to track the vehicles from one frame to another. In the case that the vehicle detector (i.e., RetinaNet) fails to detect a previously seen vehicle in a frame, the track maintenance keeps its track active up to n frames. If the previously observed vehicle gets detected again within the n frames, its track is updated, and the same vehicle id is assigned to its bounding box. Also, the bounding boxes for the missing frames are interpolated between the frames before and after the missing frames.

The bounding boxes represent the location of vehicles in image coordinates (i.e., row and column of pixels). For trajectory extraction, these coordinates need to be converted to a fixed ground coordinate system (e.g., meters or feet). A digital image is a 3D tensor, and every pixel is located by its row and column number in the image tensor. The location of every vehicle in an image frame is estimated by converting its position on the image map to the fixed coordinate system picked on the ground. Consequently, it is essential to find the mapping function between the image coordinate ($row, column$) to the adopted ground coordinate (x, y):

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \begin{bmatrix} row \\ column \end{bmatrix} \quad (11)$$

The above transformation matrix has six parameters that can be calibrated using coordinates of three known points on the ground and image. In the presence of more known matching points with outliers, the RANSAC algorithm (Derpanis, 2010) introduced in the image stabilization section can be used to find the parameters of the transformation matrix that agree with most of the matching data points. In this study, the reference image is picked such that the roadway is parallel to the row axis in the image map. In this case, the X axis is picked along the direction of traffic and the Y axis perpendicular to the direction of travel. In other words, the X axis represents the vehicle's location on the roadway with respect to a reference starting point (i.e., center), and the Y axis estimates the current lane of the vehicle. The pixel

size, s , on the ground depends on the flight elevation and is the key to the mapping function between the two coordinate systems. Besides, keeping the axes of the two coordinate systems parallel makes the transformation between them simple. The front bumper is taken as the location of the vehicle on the roadway, and the trajectory of the vehicle is the list of its location over space and time (x, y, t) .

3.4.1. Kalman filter

The output of image stabilization and vehicle detection could be noisy. As a result, a Kalman filter is applied to reduce the noise in the state estimation of the vehicles. The vehicle state at each point, $x_i^{t_i}$, is characterized by its location and kinematic state. The state attributes include the position information, $p_i^{t_i}$, speed, $v_i^{t_i}$, and acceleration, $a_i^{t_i}$. The expected state of the vehicle after t seconds (i.e., rate of data generation), $\hat{x}_i^{t_i+t}$, can be estimated by multiplying the transition matrix A by the initial state vector.

$$\hat{x}_i^{t_i+t} = A x_i^{t_i} = \begin{bmatrix} 1 & t & \frac{t^2}{2} \\ 0 & 1 & t \\ 0 & 0 & 1 \end{bmatrix} [p_i^{t_i}, v_i^{t_i}, a_i^{t_i}]^T \quad (12)$$

In the state estimation process, the Kalman filter is usually applied to estimate the best guess on the current state of the vehicle considering the previous state and current measurements (i.e., from aerial images). Previous vehicle state, $x_i^{t_i-1}$ is transitioned to the expected current state, $\hat{x}_i^{t_i}$, based on the process model and applying the transition matrix A :

$$\text{Process model: } x_i^{t_i} = A \hat{x}_i^{t_i-1} + \omega \quad (13)$$

where ω is the process noise. In this study, the process model considers $\omega = [t^2, t, 1]^T \sigma_{ap}^2$, where σ_{ap}^2 is the acceleration variance. ω is assumed to be normally distributed with covariance matrices of Q :

$$Q = \begin{bmatrix} \frac{t^4}{4} & \frac{t^3}{2} & \frac{t^2}{2} \\ \frac{t^3}{2} & t^2 & t \\ \frac{t^2}{2} & t & 1 \end{bmatrix} \sigma_{ap}^4 \quad (14)$$

The expected current state is converted to the expected measurement, $\hat{z}_i^{t_i}$, through the following measurement model:

$$\text{Measurement model: } \hat{z}_i^{t_i} = H x_i^{t_i} + \nu \quad (15)$$

where ν is the measurement noise. Since only the position of the vehicles is directly measured from the aerial images, the resulting state to measurement conversion matrix, H , is $[1, 0, 0]$. ν is assumed to be normally distributed with covariance matrices of R :

$$R = \begin{bmatrix} \sigma_p^2 \end{bmatrix} \quad (16)$$

where σ_p^2 is the position variance. Note that the measurement covariance matrix considers the variance in position alone. A 2D Cartesian coordinate system is considered for the measurements, and the state of the vehicle is evaluated along the two axes, x and y , separately. Taking σ_{ap}^2 and σ_p^2 , equal to $0.5 (\frac{m}{s})^2$ and 0.5 m^2 , respectively, performed well in addressing the noise in the state estimates. These values are carefully selected using a grid search method and choosing the values that result in smooth trajectories and not in losing recent measurements.

3.5. Adaptive Cruise Control (ACC) operated vehicles

One of the primary motivations of this study was to observe how recent advancements in vehicle technology and ADAS impacts traffic flow dynamics. This study focuses on the impacts of ACC as a core feature amongst all automated vehicles. Unfortunately, it is not possible to determine if a vehicle is using ACC from the birds-eye view without additional information. A potential solution to deal with this problem is to use probe vehicles during the data collection. Accordingly, a platoon of three probe vehicles, including two Toyota Prius and one Toyota

Avalon, was used under ACC for data collection. In the rare events of temporary disengagement, the drivers activated the ACC shortly after the disengagement. However, the disengagement events are not recorded during the data collection. Therefore, this study assumes that short-term disengagement and reactivation of the ACC is an expected behavior among the drivers when using the ADAS technologies. The probe vehicle drivers were all male with an age between 30 to 35 years. Note that it is expected to see different mechanical performances among vehicles with different makes and models. However, according to Makridis et al. (2021), with enough data from other a few ACC systems, the proposed methodology can be generalized to capture all ACC vehicles. Makridis et al. (2021) studied 27 different models of vehicles and found that the ACC platoons behave much more homogeneously even among different manufacturers.

The leader of the platoon was following an arbitrary vehicle on the roadway in front of it using ACC. The other two vehicles were also following their leaders with ACC. A total of five runs were performed along the study roadway segment. Note that all vehicles had full-range ACC with stop-and-go capability. These vehicles, as will be seen in the next section, serve as the ground truth to identify other ACC-driven vehicles in the traffic stream. The collected vehicle trajectories include the trajectory of the probe vehicles using ACC and the trajectory of other vehicles, some of which could be using ACC technologies. Since using the ACC technology is not known for all the vehicles in the collected data, this study adopts an unsupervised approach (clustering) to group the vehicles with similar behavior into the same group. The vehicles grouped with the probe vehicles in the same cluster are considered to have similar behavior to the ones using the ACC technologies. Note that despite the possibility of false positives and false negatives, as long as the behavior of two vehicles are similar, their impacts on the traffic can be similar as well.

3.6. Location and procedure

The data is collected on the southbound of Interstate Highway 35 between Exit 237B and Exit 238A in Austin, Texas (see Fig. 5). A single stretch of nearly 160 meters roadway was recorded for 2 h between 07:30 AM and 09:30 AM on a Friday. The trajectory data is collected during the morning peak hour such that the ACC vehicles are exposed to changes in traffic dynamics and speed from uncongested to stop-and-go traffic. In a free-flow traffic state, the driving behavior of ACC vehicles and human-driven vehicles are not expected to be much different since the vehicles are traveling at their desired speeds. Moreover, Flying at a higher altitude could help collect data from a more extended study segment and longer trajectory data. This segment has four lanes in the southbound direction. Note that this highway has directional traffic with congested southbound in the morning and congested northbound in the afternoon. The traffic video is recorded with 4K resolution at 25 frames per second. Table 4 presents a summary of the collected trajectory dataset. The complete dataset includes the trajectory of 8927 small vehicles (i.e., cars, vans, and pick-up trucks) and the trajectory of 718 large vehicles (i.e., trucks, buses, and recreational vehicles). In this dataset, each trajectory is represented with a unique identification number. For each trajectory, the driving lane, lateral and longitudinal location (front bumper), speed, acceleration of the vehicle over time, and type of the vehicle are reported.

3.7. Complete vehicle trajectory data

One of the main features of this dataset is the continuity in data recording for over two hours during the morning traffic peak hours. The continuity in data collection ensures that no information or interaction between the vehicles is lost. A vehicle trajectory is recorded from the first time the vehicle was seen on the study segments. For most of the vehicles, the starting point of the trajectory would be at the beginning of the study segment. However, at the start of the recording, some of

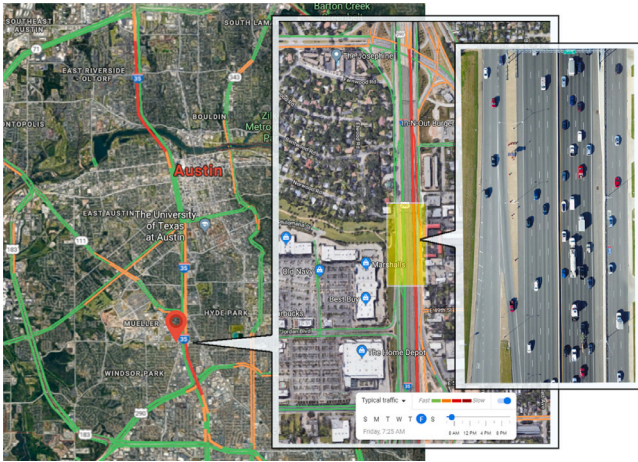


Fig. 5. Data collection location.

Table 4

Vehicle trajectory dataset collected on the Southbound of I-35 between Exit 237B and 238A, Austin, TX.

Trajectories	9645
Small vehicles	8927
Large vehicles	718
Labeled ACC trajectories	15
Number of ACC runs	5
Probe ACC vehicles	3
Data collection period	2 h
Study area length	160 m

the vehicles were further downstream of the study segment, and the trajectory of those vehicles starts from where they were seen first. Fig. 6 presents the examples of the time-space diagram of the trajectories extracted using the methodology of this study on a roadway segment of nearly 160 meters over 10 min. Note that some of the trajectories are not continuous in this figure due to lane-changing maneuvers in most cases. However, there are few cases that the vehicle detector (RetinaNet) has failed to detect a vehicle in the image, causing a discontinuity in its trajectory. In this study, two actions are applied to address these types of errors. First, the false-negative error in detection is mitigated in the tracking process by combining a high frame rate (i.e., 25 frames per second) and maintaining track of the vehicle for five consecutive frames after the last time it was seen. Second, the false-positive error is reduced by eliminating the trajectories with less than three data points. The trajectories with less than three data points account for less than 0.02 percent of the total data points. In future trajectory data collection studies, it would be a good practice to also record vehicles' trajectories using onboard sensors in the probe vehicles (e.g., GPS and IMU sensors) to compare with the collected trajectories using the proposed methodology.

Travel-time, flow, and density can be directly extracted from the vehicle trajectory data. Fig. 7 presents the distribution of the travel time of all the vehicles over the study segment and for the whole study duration. The travel time is estimated for the middle 120 meters part of the study segment (locations 15 meters to 135 m). This segment selection is due to the partial observation of the large vehicles at the edges of the images, which results in inaccurate bumper location estimation at those edges. Basic statistics on the collected travel-time are presented in Table 5. Flow and density plots with an aggregation level of 30 s for individual lanes as well as their average are presented in Fig. 8. According to these plots, the traffic dynamics of the leftmost lanes (Figs. 8(a), 8(b)) are different from the traffic dynamics of the rightmost lanes (Figs. 8(c), 8(d)). The flow and density data points for

Table 5

Basic statistics on travel-time.

Travel-time	
Segment length (middle part of the study segment)	120 m
Minimum	5.21 s
Maximum	53.90 s
Median	13.21 s
Mean	14.19 s

lanes one and two (leftmost lanes) are more in the congested region compared to the data points for lanes three and four (rightmost lanes).

3.8. ACC platoon data

This study introduces one of the first collected comprehensive trajectory datasets from both ACC and human-driven vehicles. Five runs of the platoon of probe vehicles using ACC are recorded during the data collection. The first three runs are conducted in the rightmost lane (lane 4), and the last two runs are performed in the second rightmost lane (lane 3). Figs. 9 and 10 illustrate the overview of the platoons and the traffic dynamics for each of the five runs. The platoon overview, Fig. 9, presents the identification number of the ACC vehicles in the platoon, as well as the leader of the first ACC vehicle and three human-driven vehicles behind the last ACC vehicle. The identification numbers are arbitrary and unique numbers assigned to each of the vehicle trajectories in the dataset. The time-space diagrams of Fig. 10 are generated for the period of 30 s before the ACC platoons enter the segment and up to 30 s after exiting the study segment. The trajectories of the ACC vehicles are depicted with blue lines in the time-space diagrams.

4. Results and discussion

The key objectives of this study are to (1) to develop a robust and scalable methodology to identify vehicles with ACC-type behavior in a vehicle trajectory dataset, and (2) to investigate the difference in the behavior of conventional vehicles and the vehicles using a full range ACC, or the ones with similar dynamics, in a real-world setting. Accordingly, a clustering approach is adopted to identify the trajectories that have comparable traffic dynamics to the three probe vehicles using ACC during the data collection.

The trajectory dataset collected in this study includes five runs of data collection for probe vehicles using ACC. Each vehicle trajectory (i.e., time series) is unique due to different vehicle dynamics, driver behavior, and the time of data collection (and the existence of different traffic states). This study investigates the vehicle trajectories of each run separately to control for the difference in traffic states. The period of each run is considered ten seconds before the first probe vehicle entrance onto the study segment to ten seconds after the last probe vehicle exiting the segment. For each run, all the trajectories of the vehicles observed during that period are considered in the analysis.

4.1. Feature normalization and feature selection

Acceleration, speed, time headway, and space headway, as well as their changes from the previous time step, are the eight features that are considered for each data point in the trajectory dataset. For the instances that a vehicle did not have a leader, a space headway of 100 meters is considered to avoid missing features for any data point. Each feature has a different scale, and it could contribute differently to the measurement based on the similarity/dissimilarity measure adopted in the clustering process. For example, in the case of using Euclidean distance as the dissimilarity measure, the feature with a larger scale and dispersion could dominate the measurement. Accordingly, all the features are normalized using the mean and variance of the data points

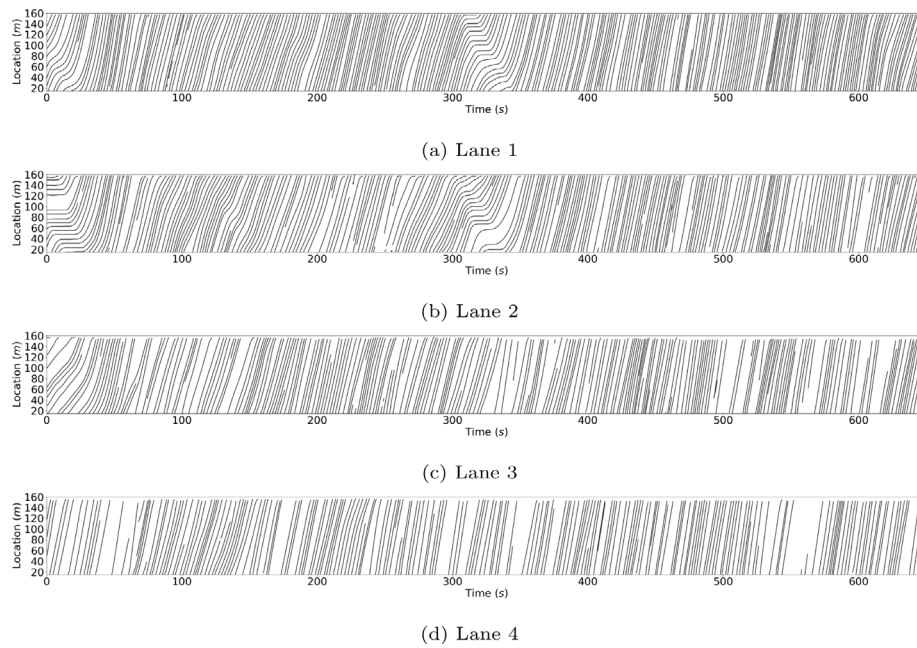


Fig. 6. Time-space diagram of the vehicles for ten minutes on the Southbound of I-35 between Exit 237B and 238A, Austin, TX, during the morning peak time.

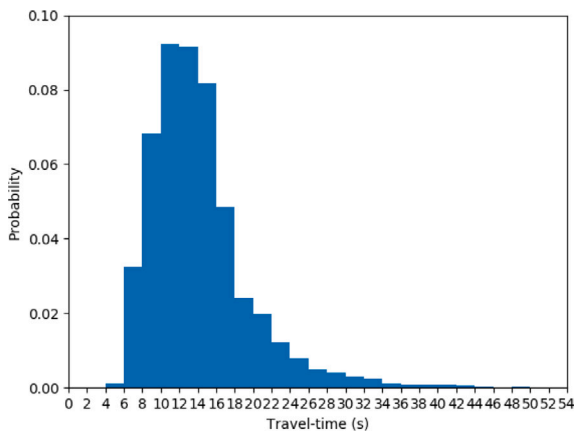


Fig. 7. Travel time distribution.

Table 6
Feature statistics.

Feature	Mean	Standard deviation	Unit
Speed	7.36	2.98	$\frac{m}{s}$
Acceleration	-0.01	0.76	$\frac{m}{s^2}$
Time headway	9.26	26.30	s
Space headway	34.78	30.91	m
Speed change	-0.02	0.14	$\frac{m}{s}$
Acceleration change	-0.02	0.18	$\frac{m}{s^2}$
Time headway change	0.10	4.73	s
Space headway change	0.31	5.21	m

in all five runs (as shown in Table 6) to maintain a similar scale and dispersion. Besides, normalizing features before principal component analysis helps prevent the domination of the first component with the feature with the highest variance.

The eight normalized features of acceleration, speed, time, and space headway and their changes have high correlations. In the cases that the features are highly correlated, the same information contributes higher in the measurements (Sambandam, 2003). Principal

component analysis (PCA) is applied to transform the eight correlated features to construct new uncorrelated features and potentially reduce the number of features. The principal components are estimated considering all the normalized data points for all trajectories of the five runs. The first seven principal components (presented in Table 7) are kept to maintain a minimum of 95 percent of the variance to be retained. Each principal component is a weighted combination of the eight original features. Also, a whitening transformation is applied by multiplying the components by the square root of the number of samples divided by the singular values to keep the variance of all features as unit components.

4.2. Clustering results based on different distance measures

Four different distance measures, including DTW, LCSS, Fréchet, and Hausdorff, along with affinity propagation (AP) clustering, are considered to group similar trajectories for each data collection run. LCSS distance requires two hyperparameters of ϵ and δ that control the similarity margin and how far in time the measurement can go to find a matching data point, respectively. Three values for $\epsilon \in [0.01, 0.05, 0.1]$ and three values for $\delta \in [10, 50, 100]$ are examined in the clustering process when using LCSS.

The purpose of clustering is to identify trajectories with the traffic dynamics similar to the three probe vehicles using ACC during the data collection. The performance of the different distance measures in the clustering process is compared based on their effectiveness in identifying a similar behavior between the three probe vehicles or assigning them into the same cluster. Clustering helps group the vehicles with similar traffic dynamics and behavior into the same group. It should be noted that such an approach will result in both false positives and false negatives. False negatives are easier to evaluate and have been the focus of this study. In other words, the performance of the proposed clustering process is evaluated based on its effectiveness in identifying a similar behavior between the three probe vehicles or assigning them to the same cluster. The false positives cannot be captured based on this dataset. However, from the behavior prediction perspective, as long as the behavior of a vehicle is similar to ACC, its impact on the traffic can be similar and its behavior can be accurately predicted under the ACC models. Accordingly, false positives, unlike false negatives, will not introduce a major challenge.

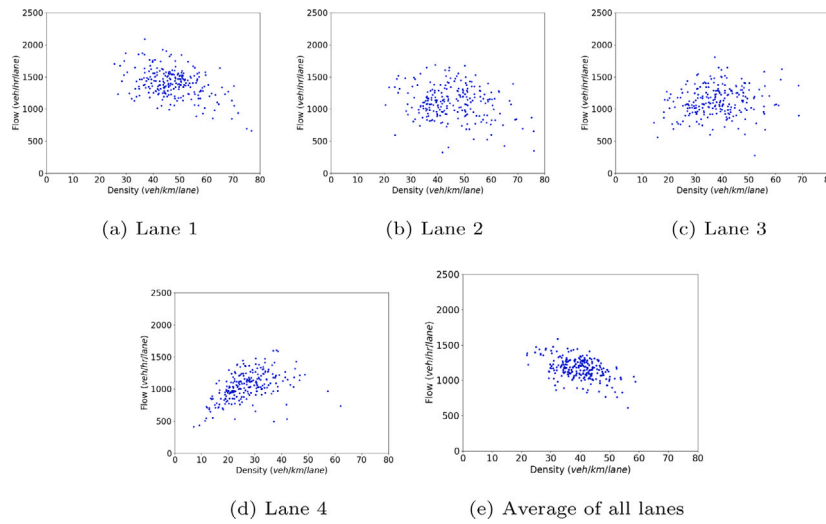


Fig. 8. Flow and density plots for each lane with aggregation resolution of 30 s.

Table 7
Principal component analysis of the normalized features.

Feature	PC1	PC2	PC3	PC4	PC5	PC6	PC7
Speed	0.128	-0.602	-0.111	-0.437	-0.033	-0.096	0.638
Acceleration	-0.159	-0.412	-0.197	0.335	-0.680	0.300	-0.151
Time headway	-0.019	0.639	-0.101	0.063	-0.432	0.067	0.621
Space headway	0.229	0.199	-0.148	-0.761	-0.356	0.127	-0.405
Speed change	-0.684	-0.069	0.008	-0.152	-0.110	0.104	-0.023
Acceleration change	-0.650	0.099	0.093	-0.291	0.175	-0.021	0.049
Time headway change	0.123	-0.038	0.681	-0.072	0.048	0.706	0.118
Space headway change	-0.005	-0.055	0.667	-0.017	-0.422	-0.608	-0.061
Eigenvalue	2.006	1.550	1.170	1.042	0.932	0.819	0.447
Eigenvalue ratio	0.252	0.194	0.147	0.131	0.117	0.103	0.056

Table 8 presents the number of clusters discovered for the three probe vehicles using ACC. In this table, a value of one is the most favorable and indicates that the three probe vehicles are grouped into the same vehicle cluster. A value of three is the least desirable value indicating the probe vehicles are grouped into three different clusters. According to Table 8, using DTW distance in the clustering process results in a better performance compared to the other distance measures. When using DTW distance combined with the AP, the clustering process grouped the three probe vehicles into the same clusters in all runs except for the fourth run of data collection. In the fourth run, the last vehicle in the probe vehicles platoon is clustered in a single group. Fig. 10(d) presents the time-space diagram of the three probe vehicles (8806, 8808, and 8813) using ACC during the fourth run. According to this figure, the first two probe vehicles leave the study segment before it becomes congested, and the last probe vehicle is left on the study segment. Since the leader of the last probe vehicle is not on the study segment, the time series of this trajectory contains a large number of data points with high space headway and very low speed (as indicated previously, when a leader is not available, a large value is utilized for the space headway), which make this trajectory an outlier. As a result, the last probe vehicle using ACC is clustered separately from the other two. In the rest of the analysis, the clusters with less than three members are considered outliers.

4.3. Statistical analysis of clustered trajectories

In the clustering process, DTW distance performed much better than the other distance measures in grouping the three probe vehicles into the same cluster. The remainder of this study evaluates the clustering results when using DTW distance and only for clusters with more than three vehicles. Moreover, the statistical comparison between the

Table 8

Distance measures comparison based on the number of clusters identified for the three probe vehicles with active ACC.

Distance measure	Run 1	Run 2	Run 3	Run 4	Run 5
LCSS ($\epsilon = 0.01, \delta = 10$)	3	3	3	3	3
LCSS ($\epsilon = 0.01, \delta = 50$)	3	3	3	2	3
LCSS ($\epsilon = 0.01, \delta = 100$)	3	3	3	2	3
LCSS ($\epsilon = 0.05, \delta = 10$)	2	2	3	2	3
LCSS ($\epsilon = 0.05, \delta = 50$)	1	1	2	3	1
LCSS ($\epsilon = 0.05, \delta = 100$)	1	1	2	3	2
LCSS ($\epsilon = 0.10, \delta = 10$)	1	2	3	1	2
LCSS ($\epsilon = 0.10, \delta = 50$)	1	2	1	2	2
LCSS ($\epsilon = 0.10, \delta = 100$)	2	2	1	2	2
DTW	1	1	1	2	1
Fréchet	2	3	1	3	2
Hausdorff	1	3	2	3	2

clusters is performed separately for each run and also separately for time headway and space headway.

Table 9 presents the average and standard deviation of the time headway, space headway and speed for each cluster in every run. In this table, the id of clusters that contain the trajectory of the probe vehicles using ACC is complemented by “-ACC”. Table 9 also presents the number of vehicles in each cluster, and according to this table, the probe vehicles are grouped with multiple other vehicles into the same cluster, indicating that those vehicles have similar trajectories compared to the probe vehicles using ACC.

A normality test based on D’Agostino and Pearson’s (D’Agostino & Pearson, 1973) is applied on the time headway and space headway of the clusters separately. As expected, the normality tests concluded that none of the clusters follow a normal distribution at a significance level of 0.05. The headway distribution is skewed to the right for both

Table 9

Statistics of time headway (s), space headway (m), and speed (m/s) for different clusters of the five runs.

	Cluster Id	# Vehicles	Avg. T-Headway	Std. Dev. T-Headway	Avg. S-Headway	Std. Dev. S-Headway	Avg. Speed	Std. Dev. Speed
Run 1	0	7	2.43	0.58	19.70	4.94	8.21	1.77
	1	16	3.24	1.33	21.12	11.10	6.81	2.65
	2	8	2.29	0.97	26.71	9.98	11.89	1.54
	3	4	2.75	0.79	15.64	3.42	5.95	1.41
	4-ACC	7	2.22	0.65	20.60	4.20	9.7	2.16
	5	7	3.38	1.28	19.66	7.93	6.05	2.16
	6	4	2.81	1.03	15.26	2.70	5.94	1.72
Run 2	7	5	2.32	0.68	16.11	4.56	7.41	2.75
	1	8	2.26	0.83	24.72	7.66	11.16	1.27
	2	4	4.66	1.12	32.15	9.28	6.9	1.1
	3-ACC	16	2.74	1.11	32.82	10.31	12.44	1.99
	4	18	2.33	0.80	16.47	4.73	7.2	0.9
Run 3	5	7	4.15	1.07	29.05	6.07	7.15	1.04
	6	7	3.41	1.09	27.67	9.27	8.48	2.33
	7	7	2.44	1.27	20.64	8.55	9.05	1.97
	8-ACC	15	2.58	1.05	24.60	7.91	9.96	1.83
	11	9	2.71	1.25	23.38	6.26	9.44	2.09
Run 4	12	6	2.96	1.13	34.67	14.28	11.91	2.7
	0	7	3.50	1.27	18.6	6.26	5.78	1.96
	1	8	1.98	0.59	14.99	2.80	7.83	1.19
	2	15	3.29	1.07	26.81	7.59	8.37	1.52
	3	6	3.81	2.02	40.49	20.16	10.84	0.58
	4	4	4.57	0.76	27.57	3.35	10.84	0.58
	8	3	3.44	1.09	22.93	4.36	5.34	1.56
Run 5	10-ACC	4	3.45	0.91	28.8	5.28	8.62	1.4
	0	11	4.18	1.51	33.82	11.61	8.28	1.53
	2	11	2.89	0.91	16.54	3.84	5.99	1.43
	3	11	2.86	1.07	31.52	10.83	11.19	1.68
	4	5	3.39	1.46	18.85	5.20	6.14	1.73
	5-ACC	16	2.82	1.40	18.87	8.00	7.21	1.98
Run 5	7	3	4.44	1.38	24.81	8.72	5.87	2.12

time headway and space headway due to headway values being positive and the existence of some significant headways when the vehicle's speed is low or when there is a large distance to the leading vehicle. Following the normality test, Bartlett's test (Snedecor & Cochran, 1989) is conducted to evaluate the homogeneity of variances of headways between the clusters of each run. Bartlett's test does not require close to normality distribution, unlike Levene's test (Levene, 1961). The result of Bartlett's tests at a significance level of 0.05 suggested that for each run, at least two of the clusters have different variances for both time headway and space headway. Besides, the Kruskal–Wallis test (Daniel, 1990) is conducted to compare the similarity between the headway distribution of clusters in each run. Kruskal–Wallis is a non-parametric method and does not require the normal distribution of the samples. For each run, and for both time headway and space headway, at a significance level of 0.05, it is concluded that at least two clusters have a different distribution.

For both time headway and space headway, the results of Bartlett's test and the Kruskal–Wallis test suggest that at least two of the clusters have different variances and distributions in each run. Following these two tests, each run's clusters are compared pairwise for the similarity in their variances and means at a significance level of 0.05. Bartlett's test is applied for the pairwise comparison of the variances, and Welch's t-test is adopted to evaluate the similarity of means. The advantage of Welch's t-test over the student t-test is that it does not require equal variances and the number of samples. Most of the pairwise comparisons concluded that there is a statistical difference between the means and variances of the clusters in each run for both time headway and space headway. The few pairwise tests that failed to reject the null hypotheses are presented in Table 10. According to this table, the pairwise tests were unable to reject equal means and variances between the time headways of clusters 0 and 10 in run 4; however, the means and variances of space headways of these two clusters were statistically different at a significance level of 0.05. From the pairwise comparison between the clusters, it can be concluded that detected clusters have

Table 10

Hypotheses that are failed in the pairwise comparisons of the clusters for each run.

	Between	Null hypothesis	P-value
Run 1	Clusters 0 and 5	Equal space headway means	0.37
	Clusters 1 and 5	Equal time headway variances	0.33
	Clusters 2 and 3	Equal time headway variances	0.53
	Clusters 4-ACC and 7	Equal time headway variances	0.12
Run 2	Clusters 1 and 4	Equal time headway variances	0.79
	Clusters 2 and 5	Equal time headway variances	0.35
Run 3	Clusters 7 and 8-ACC	Equal time headway means	0.54
Run 4	Clusters 0 and 2	Equal time headway variances	0.56
	Clusters 0 and 10-ACC	Equal time headway variances	0.48
	Clusters 0 and 10-ACC	Equal time headway means	0.37
	Clusters 2 and 10-ACC	Equal time headway variances	0.73
	Clusters 4 and 8	Equal space headway variances	0.44
	Clusters 8 and 10-ACC	Equal time headway means	0.10
Run 5	Clusters 0 and 3	Equal space headway variances	0.05
	Clusters 0 and 4	Equal time headway variances	0.92
	Clusters 2 and 3	Equal time headway means	0.58
	Clusters 2 and 5-ACC	Equal time headway means	0.63
	Clusters 3 and 5-ACC	Equal time headway means	0.40
	Clusters 4 and 5-ACC	Equal space headway means	0.87
	Clusters 5-ACC and 7	Equal space headway variances	0.55

a different distribution of time headway and space headway, and the clustering approach is capable of grouping the trajectories into different traffic flow dynamics.

4.4. Traffic flow dynamics within each cluster

This section discusses the macroscopic traffic flow dynamics within each cluster and compares them with the dynamics in the entire segment. In particular, speed-flow diagrams are created for each cluster and the entire segment. In order to create these diagrams for each cluster, the average value of time headway and speed for all the data

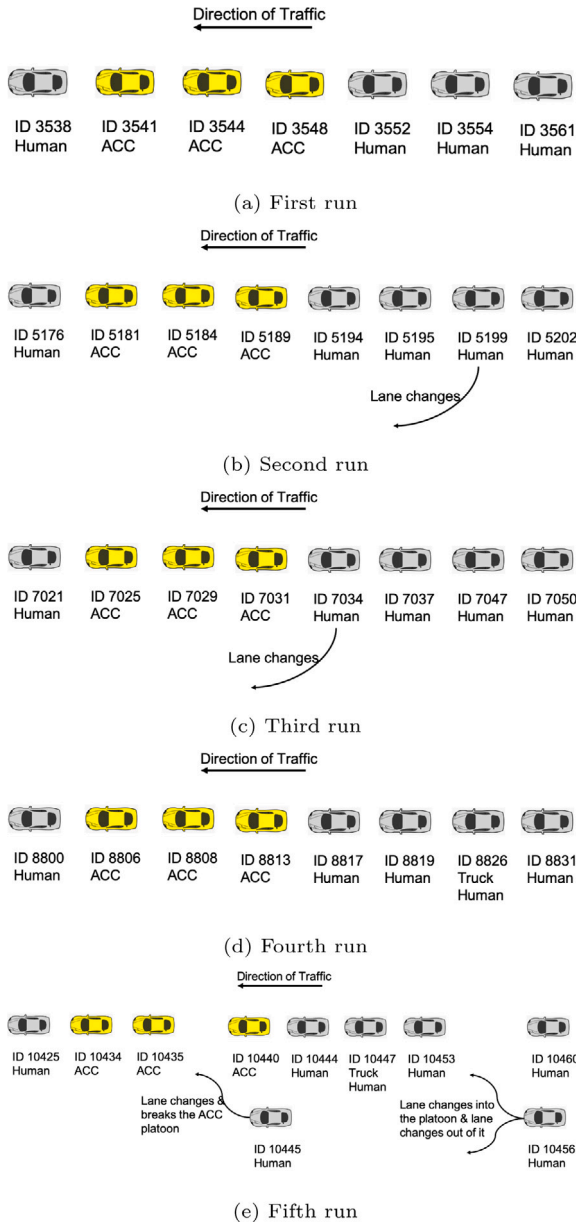


Fig. 9. Overview of the platoon of the probe vehicles over five runs of data collection.

points of trajectories that fall within a cluster are estimated for each time step during each run. Besides, the trajectory data points that did not have a leader or have a speed value of less than $0.1 \frac{m}{s}$ are ignored in the calculation to ensure meaningful time headways. Each cluster's flow rate at each time step is approximated by the inverse of the average time headway of that cluster at that time.

Fig. 11 shows the speed-flow graphs for each cluster in each run. This figure indicates a clear distinction between the macroscopic behavior of clusters that contain ACC vehicles and other clusters in runs 1 and 2. In the remaining runs, the macroscopic behavior of clusters that contain ACC vehicles is fundamentally different from the majority of the clusters. In fact, in runs 3, 4, and 5, only two, one, and two other clusters show similar behavior, respectively. While the ACC behavior is significantly different from other human-driven vehicles in some cases, there are cases where the difference is not very clear. Those cases mostly happen when the average speed is low, and traffic contains several instances of complete stops for a considerable amount of time. Unfortunately, when vehicles are at full stop or travel at very

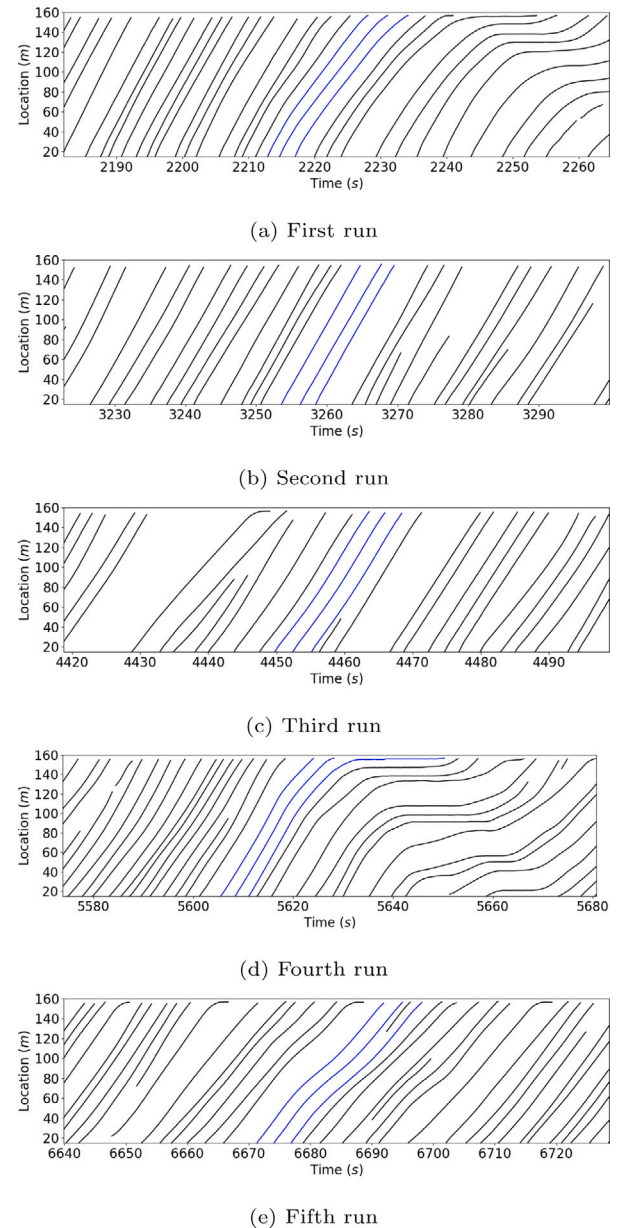


Fig. 10. Time-space diagram of the probe vehicles over five runs of data collection.

low speeds, distinguishing ACC vehicles from human-driven vehicles is impossible, and this can be true regardless of the methodology. Regardless, these observations suggest that traffic regime plays an important role in the detection accuracy, and detecting ACC-driven vehicles might be easier in some traffic regimes compared to others. Further investigation, however, is necessary to draw a meaningful conclusion.

From the perspective of scattering in the speed-flow diagram, interestingly enough, clusters that contain ACC vehicles have the least amount of scattering in all five runs. Lower scattering can potentially delay breakdown formation. Note that Makridis et al. (2021) reported a similar finding, i.e., a more homogeneous distribution of headways for vehicles using ACC technologies compared to human-driven vehicles. This shows that the behavior of these vehicles is more predictable, and they are the least likely group of vehicles to contribute to traffic flow breakdown.

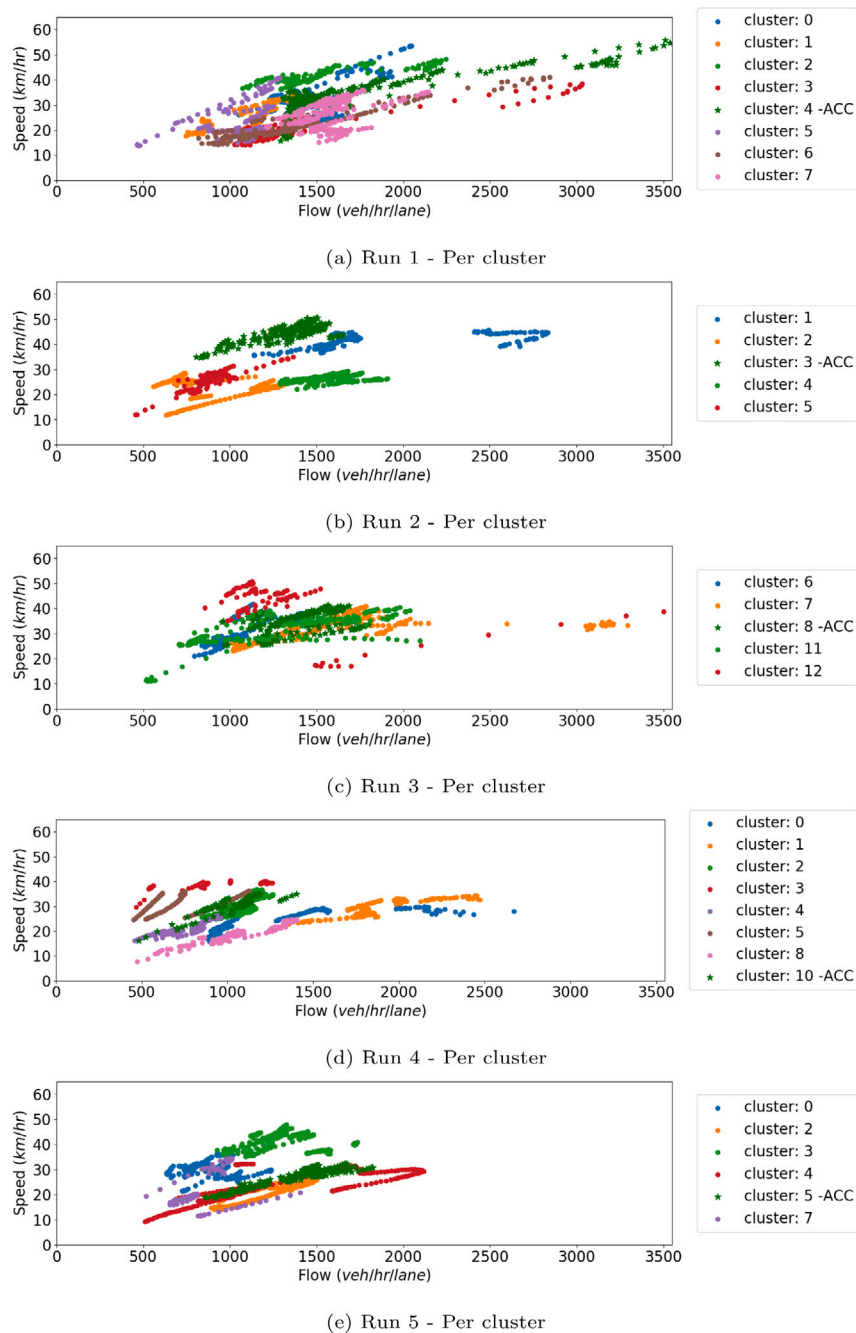


Fig. 11. Speed and flow for each cluster for each run.

Fig. 12 shows the average speed-flow graphs for the entire segment. Comparing this figure with Fig. 11 reveals interesting observations. First, while some clusters have very large flow rates (e.g., the flow rate in cluster 4 in run 1 reaches 3500 veh/h/lane), the overall segment has a fairly average flow rate (about 1500 veh/h/lane). This difference suggests that while some runs show potential to significantly increase the flow rate through platooning, the impact of platooning in a mixed driving environment might not be as significant until the high penetration rates of ACC vehicles. Second, the amount of scatter in Fig. 12 is significantly less than Fig. 11. This observation suggests that while different clusters behave differently at different time steps, their average behavior stays the same. In other words, the impact of ACC-type behavior on the entire system (if any) remains fairly constant throughout the data.

5. Conclusion

The new vehicles equipped with Advanced Driver Assistance Systems (ADAS), such as adaptive cruise control (ACC), can potentially change the interaction among drivers on the road. The existing trajectory datasets fail to provide any information on the utilization of ADAS technologies. Accordingly, this study introduces a clustering approach to identify ACC-driven vehicles in vehicle trajectory datasets. In the process of developing this clustering approach, an accurate, scalable, and cost-effective data collection methodology is introduced to collect vehicle trajectory data based on aerial images. The methodology for trajectory extraction is modular, and every component has the potential for improvement.

The trajectory data is collected for over two hours from a nearly 160 meters long segment of I-35 near Austin, TX. The collected trajectories

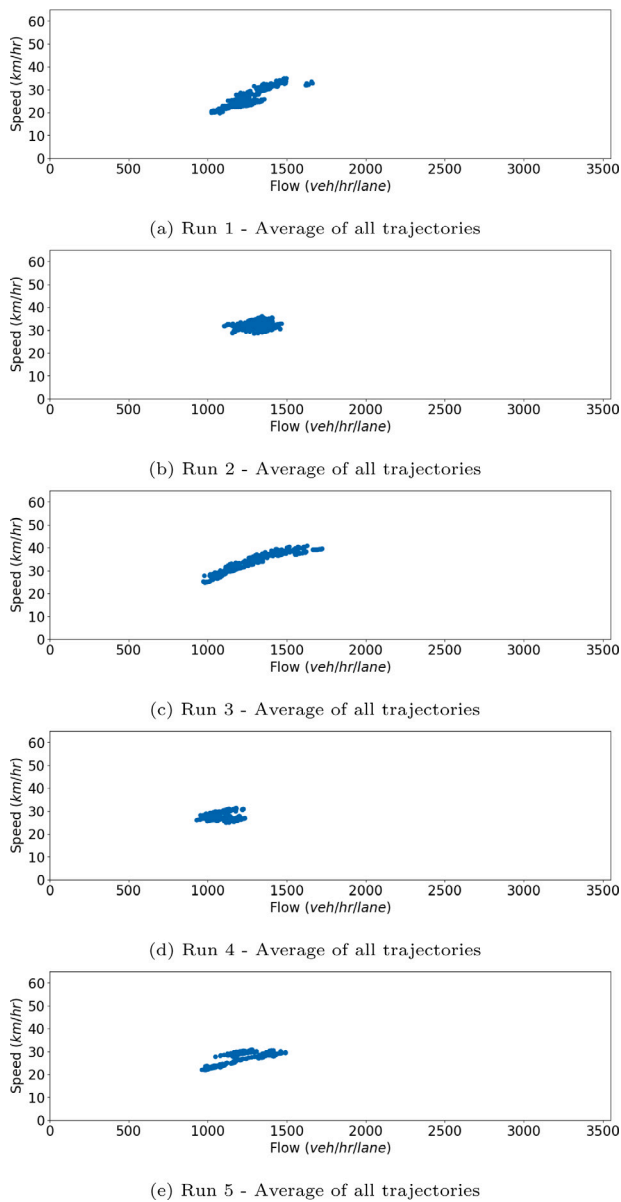


Fig. 12. Speed and flow for all trajectories for each run.

contain five runs of data collection from one platoon consisting of three vehicles operated based on an ACC system. The vehicle trajectory is a time series of different features, including location and other features such as speed and acceleration. Since the trajectory data is not labeled based on ACC utilization, clustering is an excellent approach to arrange similar trajectories in the dataset into the same group. Comparing the similarity between the trajectories is an essential step in grouping them into the same or different clusters. One of the challenges with vehicle trajectory data is that the trajectories do not have equal lengths (i.e., number of time steps), and the typical Euclidean distance is not a suitable distance measure between trajectories. The distance measures used to compare time series with different lengths include warping-based distances such as dynamic time warping (DTW) and longest common subsequence (LCSS), and shape-based distance such as Fréchet and Hausdorff. Besides, some of the distance measures between the trajectories do not satisfy the triangle inequality, limiting the clustering method to algorithms such as affinity propagation (AP), which can work directly with the distance matrix.

The clustering results with different distance measures indicated that the DTW distance between the trajectories has a better performance in keeping the probe vehicles using ACC in the same group. The statistical analysis of the time headway and space headway indicated a statistical difference between the traffic dynamics of different clusters. The unique trajectory dataset of this study combined with the clustering provides the opportunity to identify vehicle trajectories with comparable traffic dynamics to the vehicles using ACC.

This study proposes a methodology to identify clusters of trajectories with similar traffic dynamics to the vehicles using ADAS systems. The clustering results could be used to calibrate different car following models to gain further information on the behavior of different clusters. This step is left for future studies. Moreover, one of the key contributions of this study is to develop and test the ACC detection algorithm using real-world data with all the noise and unknown elements. While utilizing a simulation tool can help test more diverse cases, including higher market penetration rates and various traffic states, the scope of this study, as a proof-of-concept study, is not to evaluate all possible cases and variations in ACC vehicles and human behavior. Accordingly, this approach has also been left for future research. Furthermore, this study focuses on the ACC system that only helps with longitudinal motion planning of the vehicle. Other ADAS systems that help with lateral motion planning, such as lane-keeping or lane changing, are left for future studies. A similar data collection methodology proposed in this study can be adopted to collect trajectories for vehicles using other ADAS technologies. However, it is expected that additional features that capture lateral maneuvers are needed to be considered in the clustering process.

CRedit authorship contribution statement

Mohammadreza Khajeh Hosseini: Data curation, Data analysis, Model development, Writing. **Alireza Talebpour:** Data curation, Writing – review & editing, Supervision. **Saipraneeth Devunuri:** Data curation, Data analysis. **Samer H. Hamdar:** Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Alireza Talebpour reports financial support was provided by National Science Foundation. Alireza Talebpour reports a relationship with National Science Foundation that includes: funding grants.

Data availability

Data will be made available on request.

Acknowledgment

This material is based upon work supported by the [National Science Foundation[<http://dx.doi.org/10.13039/100000001>] under Grant No. 1826410.

References

- Barmounakis, E., & Geroliminis, N. (2020). On the new era of urban traffic monitoring with massive drone data: The pNEUMA large-scale field experiment. *Transportation Research Part C (Emerging Technologies)*, 111, 50–71.
- Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Surf: Speeded up robust features. In *European conference on computer vision* (pp. 404–417). Springer.
- Berndt, D. J., & Clifford, J. (1994). Using dynamic time warping to find patterns in time series. In *KDD workshop*, vol. 10 (16), (pp. 359–370). Seattle, WA, USA.
- Besse, P. C., Guillouet, B., Loubes, J.-M., & Royer, F. (2016). Review and perspective for distance-based clustering of vehicle trajectories. *IEEE Transactions on Intelligent Transportation Systems*, 17(11), 3306–3317.

- Bhutani, A., & Bhardwaj, P. (2019). Automotive camera market share 2019–2025: Global industry report. <https://www.gminsights.com/industry-analysis/automotive-camera-market>.
- Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., et al. (2020). nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11621–11631).
- Coifman, B., & Li, L. (2017). A critical evaluation of the next generation simulation (NGSIM) vehicle trajectory dataset. *Transportation Research, Part B (Methodological)*, 105, 362–377.
- D'Agostino, R., & Pearson, E. S. (1973). Tests for departure from normality. Empirical results for the distributions of b_2 and b_6 . *Biometrika*, 60(3), 613–622.
- Daniel, W. W. (1990). Kruskal–Wallis one-way analysis of variance by ranks. *Applied Nonparametric Statistics*, 226–234.
- Derpanis, K. G. (2004). *The harris corner detector* (pp. 2–3). York University.
- Derpanis, K. G. (2010). Overview of the RANSAC algorithm. *Image Rochester NY*, 4(1), 2–3.
- Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., & Koltun, V. (2017). CARLA: An open urban driving simulator. In *Proceedings of the 1st annual conference on robot learning* (pp. 1–16).
- Dutta, A., & Zisserman, A. (2019). The VIA annotation software for images, audio and video. In *MM '19, Proceedings of the 27th ACM international conference on multimedia*. New York, NY, USA: ACM, <http://dx.doi.org/10.1145/3343031.3350535>.
- Eiter, T., & Mannila, H. (1994). *Computing discrete Fréchet distance: Technical Report*, Citeseer.
- FHWA (2006). U.S. federal highway administration. Next generation simulation (NGSIM). <https://ops.fhwa.dot.gov/trafficanalysis/tools/ngsim.htm>.
- Fréchet, M. M. (1906). Sur quelques points du calcul fonctionnel. *Rendiconti Del Circolo Matematico Di Palermo (1884-1940)*, 22(1), 1–72.
- Frey, B. J., & Dueck, D. (2007). Clustering by passing messages between data points. *Science*, 315(5814), 972–976.
- Gaiser, H., et al. (2019). Fyzyr/keras-retinanet 0.5.1. <http://dx.doi.org/10.5281/zenodo.3250670>.
- Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440–1448).
- Girshick, R., Donahue, J., Darrell, T., & 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).
- Hankey, J. M., Perez, M. A., & McClafferty, J. A. (2016). *Description of the SHRP 2 naturalistic database and the crash, near-crash, and baseline data sets: Technical Report*, Virginia Tech Transportation Institute.
- Hausdorff, F. (1914). *Grundzüge der mengenlehre*, vol. 7. von Veit.
- He, Y., Ciuffo, B., Zhou, Q., Makridis, M., Mattas, K., Li, J., et al. (2019). Adaptive cruise control strategies implemented on experimental vehicles: A review. *IFAC-PapersOnLine*, 52(5), 21–27.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).
- Karami, E., Prasad, S., & Shehata, M. (2017). Image matching using SIFT, SURF, BRIEF and ORB: performance comparison for distorted images. arXiv preprint [arXiv:1710.02726](https://arxiv.org/abs/1710.02726).
- Krajewski, R., Bock, J., Kloeker, L., & Eckstein, L. (2018). The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems. In *2018 IEEE 21st international conference on intelligent transportation systems (ITSC)*.
- Levene, H. (1961). Robust tests for equality of variances. In *Contributions to probability and statistics. essays in honor of harold hotelling* (pp. 279–292). Stanford University Press.
- Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision* (pp. 2980–2988).
- Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., et al. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision* (pp. 740–755). Springer.
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., et al. (2016). Ssd: Single shot multibox detector. In *European conference on computer vision* (pp. 21–37). Springer.
- Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2), 91–110.
- Maddern, W., Pascoe, G., Linegar, C., & Newman, P. (2017). 1 year, 1000 km: The oxford RobotCar dataset. *International Journal of Robotics Research*, 36(1), 3–15.
- Makridis, M., Mattas, K., Anesiadou, A., & Ciuffo, B. (2021). OpenACC: An open database of car-following experiments to study the properties of commercial ACC systems. *Transportation Research Part C (Emerging Technologies)*, 125, Article 103047.
- Michalopoulos, P. G. (1991). Vehicle detection video through image processing: the autoscope system. *IEEE Transactions on Vehicular Technology*, 40(1), 21–29.
- Muja, M., & Lowe, D. (2013). Fast library for approximate nearest neighbors (FLANN), [git://github.com/mariusmuja/flann.git](https://github.com/mariusmuja/flann.git). url: <http://www.cs.ubc.ca/research/flann>.
- Rahmati, Y., Khajeh Hosseini, M., Talebpour, A., Swain, B., & Nelson, C. (2019). Influence of autonomous vehicles on car-following behavior of human drivers. *Transportation Research Record*, 2673(12), 367–379.
- Redmon, J., Divvala, S., Girshick, R., & 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).
- Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (pp. 91–99).
- Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. R. (2011). ORB: An efficient alternative to SIFT or surf. In *ICCV*, vol. 11 (p. 2). Citeseer.
- SAE International (2021). Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. *SAE International*, J3016.
- Sambandam, R. (2003). Cluster analysis gets complicated. *Marketing Research*, 15(1), 16–21.
- Snedecor, G. W., & Cochran, W. G. (1989). *Statistical methods* (eight edition). Ames, Iowa: Iowa State University Press.
- Talebpour, A., Mahmassani, H. S., & Bustamante, F. E. (2016). Modeling driver behavior in a connected environment: Integrated microscopic simulation of traffic and mobile wireless telecommunication systems. *Transportation Research Record*, 2560(1), 75–86.
- Vlachos, M., Kollios, G., & Gunopulos, D. (2002). Discovering similar multidimensional trajectories. In *Proceedings 18th international conference on data engineering* (pp. 673–684). IEEE.
- Wang, L., Lin, Q.-F., Wu, Z.-Y., & Yu, B. (2020). A data-driven estimation of driving style using deep clustering. In *CICTP 2020* (pp. 4183–4194).
- Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)* (pp. 3645–3649). IEEE.
- Yuan, G., Sun, P., Zhao, J., Li, D., & Wang, C. (2017). A review of moving object trajectory clustering algorithms. *Artificial Intelligence Review*, 47(1), 123–144.
- Zhao, D., Guo, Y., & Jia, Y. J. (2017). Trafficnet: An open naturalistic driving scenario library. In *2017 IEEE 20th international conference on intelligent transportation systems (ITSC)* (pp. 1–8). IEEE.
- Zhao, Z.-Q., Zheng, P., Xu, S.-t., & Wu, X. (2019). Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*.
- Zhong, Z., Lee, E. E., Nejad, M., & Lee, J. (2020). Influence of CAV clustering strategies on mixed traffic flow characteristics: An analysis of vehicle trajectory data. *Transportation Research Part C (Emerging Technologies)*, 115, Article 102611.
- Zhou, J., Gao, D., & Zhang, D. (2007). Moving vehicle detection for automatic traffic monitoring. *IEEE Transactions on Vehicular Technology*, 56(1), 51–59.