Automated Construction of Time-Space Diagrams for Traffic Analysis Using Street-View Video Sequences

Tanay Rastogi and Mårten Björkman

Abstract

Time-space diagrams are essential tools for analyzing traffic patterns and optimizing transportation infrastructure and traffic management strategies. Traditional data collection methods for these diagrams have limitations in terms of temporal and spatial coverage. Recent advancements in camera technology have overcome these limitations and provided extensive urban data. In this study, we propose an innovative approach to constructing time-space diagrams by utilizing street-view video sequences captured by cameras mounted on moving vehicles. Using the state-of-the-art YOLOv5, StrongSORT, and photogrammetry techniques for distance calculation, we can infer vehicle trajectories from the video data and generate time-space diagrams. The evaluation results demonstrate that our approach can generate trajectories from video data, although there are some errors that can be mitigated by improving the performance of the detector, tracker, and distance calculation components. In conclusion, the utilization of street-view video sequences captured by cameras mounted on moving vehicles, combined with state-of-the-art computer vision techniques, has immense potential for constructing comprehensive time-space diagrams. These diagrams offer valuable insights into traffic patterns and contribute to the design of transportation infrastructure and traffic management strategies.

Keywords: time-space diagram; moving camera; object detection; multi-object tracking; monocular depth estimation.

1 Introduction

Time-space diagrams are graphical representations that illustrate the trajectories of all vehicles that move through a specific road section over time. They provide insights into microscopic traffic characteristics such as time headways and space headways as well as macroscopic traffic characteristics such as density, flow, and mean speed (Treiber and Kesting (2013)). A complete time-space diagram is crucial to help design and plan transport infrastructure, help implement strategies to manage traffic and reduce congestion, help identify hazardous areas to improve safety, or used to identify areas where there are high levels of vehicle emissions that can be targeted to reduce the environmental impact of transportation.

To obtain data for time-space diagrams, various data collection methods can be used, including traditional traffic counting techniques, vehicle-level data from connected vehicles, and traffic sensor data. Traditional methods of data collection can be categorized into two distinct subgroups: stationary and mobile sensors. Stationary sensors, such as loop detectors, surveillance cameras and radar-based sensors, are commonly used to collect traffic data for a specific fixed location with high temporal resolution but limited spatial coverage, represented as a red-shaded region in the ***Fig.*** 1. On the other hand, mobile sensors consist of vehicles equipped with on-board sensors such as the Global Positioning System (GPS) or On-board Diagnostics Systems (OBD), which acquire measurements throughout their trips. These vehicles are commonly referred to as probe vehicles, and the data collected from them is known as Floating Car Data (FCD). Mobile sensors have the capability to collect traffic states over a broader spatial-temporal domain compared to stationary sensors. However, it should be noted that mobile sensors do not provide an exact representation of traffic state on the road due to the limited number of vehicles equipped with sensing equipment and have a sparse distribution of data, as illustrated by blue trajectories in ***Fig.*** 1. Furthermore, Rahmani et al. (2010) showed that for conducting any traffic analysis requires prior knowledge of the penetration rate of the probe vehicles, which refers to the number of probe vehicles present on the road in a specific space and time, and acquiring such information can be challenging.

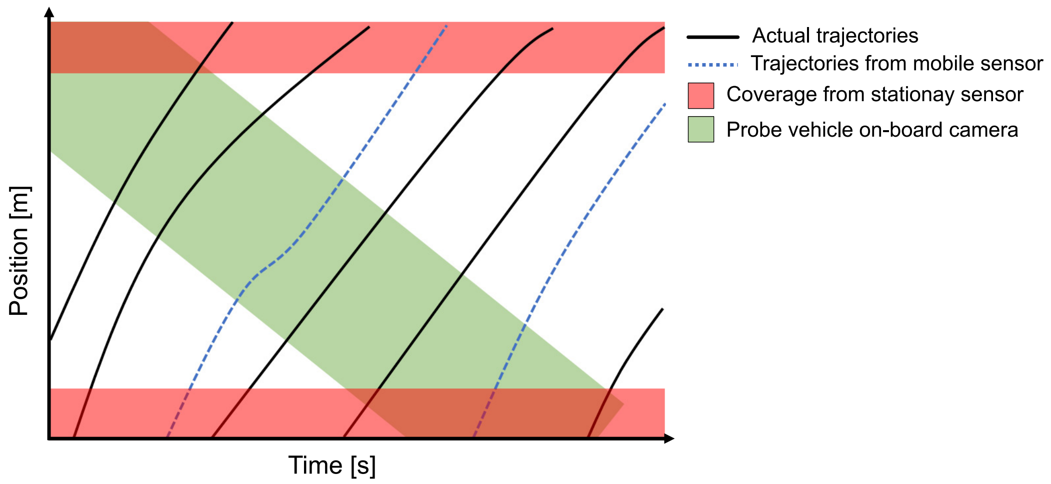

Actual trajectories
Trajectories from mobile sensor
Coverage from stationay sensor
Probe vehicle on-board camera
Position [m]
Time [s]

Figure 1: Example of time-space diagram highlighting different sections of the diagram measured by various types of sensors.

In recent times, there has been a widespread adoption of lightweight portable sensors, such as smartphones, dash cams, and embedded devices, which are installed in vehicles to capture traffic incidents. As shown by Kumar et al. (2021), these devices have become popular due to their ability to continuously interact with the surrounding environment, providing a valuable source of urban data . With the advancements in in-vehicle camera technology, there is significant potential to collect traffic data using street-view video sequences. The use of portable cameras for traffic data collection overcomes the limitations associated with stationary sensors. By equipping a single probe vehicle with a camera, it becomes possible to obtain measurements with a higher spatial resolution, similar to that of mobile sensors. Additionally, since the data is collected from images or video sequences, the proposed method for traffic analysis does not require prior knowledge of the penetration rate. In ***Fig.*** 1, the green-shaded region illustrates the sensor coverage of the proposed data collection methodology using cameras, which demonstrates a broader coverage compared to both stationary and mobile sensors.

This paper aims to introduce a novel approach to creating time-space diagrams for capturing traffic flow characteristics using an in-vehicle camera mounted on a moving vehicle. The proposed method employs state-of-the-art computer vision algorithms based on deep neural networks (DNN), photogrammetry, and geodesy to construct time-space diagrams from a video sequence. Unlike stationary sensors, the proposed method overcomes spatial limitations, and unlike mobile sensors, it does not rely on prior knowledge of the number of probe vehicles on the road.

The remainder of the article is structured as follows. ***Sec.*** 2 provides a brief summary of the literature on the topic. ***Sec.*** 3 describes the proposed traffic data collection approach, as well as the computer vision algorithms employed in the proposed algorithm and the dataset used to validate the proposed methodology. ***Sec.*** 4 presents the experimental results of the test on the proposed methodology, as well as a discussion of the results. Finally, ***Sec.*** 5 summarizes the experiment findings and concludes the article with suggestions for future work.

2 Related Work

Time-space trajectory diagrams play a crucial role in the field of traffic engineering and transportation planning. These diagrams serve as valuable visual tools to illustrate the movement of vehicles in both time and space. Using time-space diagrams, traffic engineers can effectively analyze and comprehend traffic flow patterns, pinpoint areas of congestion, and devise strategies to improve overall traffic operations. Numerous studies have leveraged time-space diagrams to derive optimal approaches to traffic control (Wang et al. (2020); Zheng et al. (2014); Peng and Wang (2023); Essa and Sayed (2018)). In addition, various methods have been proposed to predict traffic congestion through the application of deep learning techniques, training networks using information derived from space-time trajectories (Xing and Liu (2022); Khajeh Hosseini and Talebpour (2023)). Furthermore, time-space diagrams have been used to estimate traffic states (Nantes et al. (2016); Thodi et al. (2021); Allström et al. (2014)) and offer valuable insights to formulate policy recommendations (Zang et al. (2019); Li et al. (2017)).

The primary data source utilized for generating space-time trajectories is commonly derived from Floating Car Data (FCD). FCD involves employing mobile sensors installed within vehicles, such as Global Positioning System (GPS) devices, On-board

Diagnostics Systems (OBD), or embedded cameras, to capture measurements during their respective trips. Numerous research studies have extensively used FCD data to estimate crucial traffic parameters such as flow, speed, and density (Herrera and Bayen (2010); Sunderrajan et al. (2016); Yuan et al. (2021)). However, as emphasized in Rahmani et al. (2010), any analysis based on FCD data requires prior knowledge of the penetration rate, which can be challenging to obtain. To address this limitation, researchers have increasingly focused on integrating data from stationary and mobile sensors. In particular, Kashinath et al. (2021); Qing-Jie Kong et al. (2009) employ various data fusion techniques to combine information from loop detectors and GPS probe vehicles, allowing the estimation of mean speed and travel time along a specific link. By leveraging both data sources, researchers aim to mitigate the drawbacks associated with relying solely on one type of sensor.

The evolution of camera technology has revolutionized the field of traffic data collection and analysis, making cameras an indispensable component. Cameras serve as sensors capable of capturing real-time data, offering valuable insights into traffic flow, potential incidents, and road irregularities. Extensive research has been conducted on traffic flow analysis using high-mount surveillance or highway cameras highlighted in Ua-areemitr et al. (2019); Li et al. (2013); Pletzer et al. (2012). These stationary cameras are used to obtain data to estimate the mean speed, density, and level of service for specific sections of the road network within their field of view. Recently, there has been a growing focus on data collection using moving cameras, particularly for Traffic State Estimation (TSE). For example, Seo et al. (2015) used an on-board vehicle camera to measure the headway distance between vehicles in the same lane, enabling the calculation of flow and speed on the network. Additionally, Guerrieri and Parla (2021); Kumar et al. (2021) used on-board cameras to estimate TSE, with a specific emphasis on traffic in the opposite lane. Furthermore, Cao et al. (2011) developed a solution that utilizes a moving camera to collect traffic data and estimate traffic flow. These advances highlight the expanding utility of cameras and their application in traffic research and analysis.

Most of the previous studies mentioned centered around the collection of traffic data through the utilization of fixed sensors, including cameras. However, research addressing the utilization of on-board cameras as a primary data source for collecting traffic flow data remains limited and scarce. This study aims to advance research in the field of data collection using on-board vehicle cameras. In contrast to the previously mentioned studies utilizing moving cameras, our focus is specifically on capturing microscopic traffic flow characteristics, distinguishing it from the aforementioned studies. Therefore, the contributions of this article is providing a novel method for to extract time-space diagram using on-board vehicle cameras that can be used for estimating microscopic traffic states on a specific link of the road network.

3 Methodology

The proposed methodology aims to generate time-space diagrams utilizing trajectories inferred from street-view videos captured by an on-board camera mounted on a moving vehicle, along a specific link on a road network. This methodology involves three key steps: multi-object detection, multi-object tracking, and estimation of lane distance, as depicted in Fig. 2. For the detection and identification of vehicles in each frame of the video sequence, we employed the YOLOv5 multi-object detector. The structure and hyperparameters of the proposed YOLOv5 network are described in ***Sec.*** 3.2. Subsequently, each detected vehicle in the image is assigned a unique ID and tracked across consecutive frames using a multi-object tracking method known as StrongSORT, which employs tracking-by-detection. The details of the StrongSORT method for assigning IDs and tracking vehicles are explained in ***Sec.*** 3.3. Once vehicles have been detected and labeled, the distance of each detected vehicle link on a road network is computed using time-stamped GPS information, photogrammetry, and geodesy. The process of calculating the distance for each detected vehicle is elaborated upon in ***Sec.*** 3.4. Finally, utilizing the distance and timestamp information, the time-space diagram is generated with the spatial dimensions corresponding to the link length and the time dimension representing the travel time of the camera-mounted vehicle.

The methodology presented in this study is based on certain assumptions regarding the data collection process. Specifically, the methodology focuses on the *car-like* object class and is tested accordingly. While the proposed method has the capability to detect and track multiple objects, such as pedestrians, it is important to note that these objects are deemed irrelevant for the intended application of time-space trajectories. The position and length of the link in the road network are considered predetermined information. When extracting vehicle trajectories from the street-view videos, only vehicles visible in the video and traveling in the opposite lane are considered. This ensures that distinct trajectories are obtained for all vehicles traversing the link. Vehicles in the same lane as the moving camera are not included, as their speeds are usually similar, resulting in less informative samples. This approach aligns with previous research by Guerrieri and Parla (2021); Kumar et al. (2021), that aimed to analyze traffic flow and speed by considering vehicles in the opposite lane.

The suggested approach has solely been evaluated on real-world videos present within the dataset. To extend the evaluation of this approach concerning the impacts of traffic demand and congestion levels, simulated data is required. This simulated data would enable the creation of various traffic scenarios for testing, similar to the methodology employed in Kumar et al. (2022).

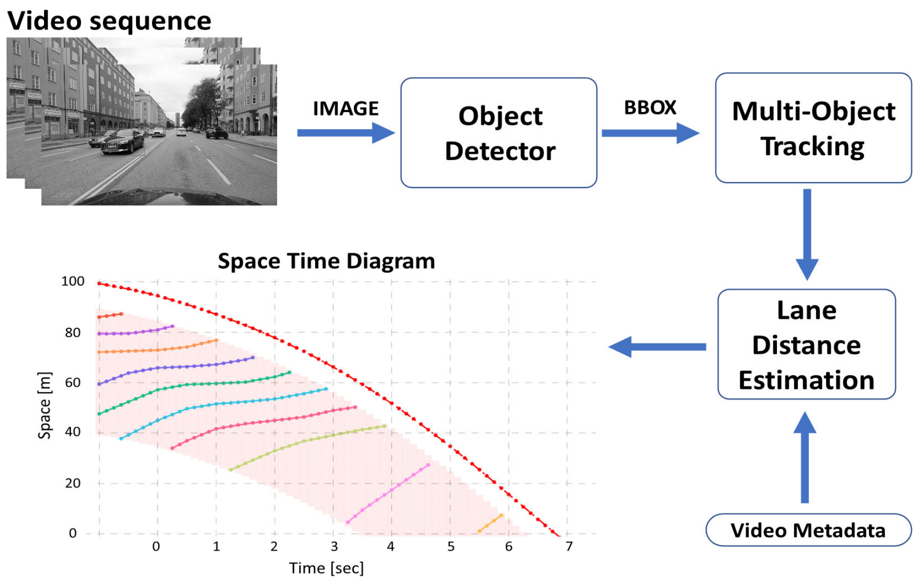


Figure 2: Flow chart that illustrates the proposed methodology of extracting the vehicle trajectory from a street-view video sequence using different computer vision algorithms.

3.1 Dataset

The analysis in the article relies on an open-source computer vision benchmark suite, called KITTI (Geiger et al. (2012)). KITTI comprises a comprehensive collection of high-resolution images, videos, and 3D point clouds, along with accurately calibrated camera and LiDAR data. The dataset was gathered by driving a car through urban environments, ensuring its relevance to real-world scenarios. The KITTI dataset covers a diverse range of tasks, such as object detection, tracking, 3D object detection, semantic segmentation, and optical flow. Its extensive content enables researchers to explore and evaluate various computer vision algorithms and methodologies across different aspects of urban visual perception.

We used annotated images from the *KITTI 2D object detection* dataset to train and validate the proposed object detector. The 7,481 images in the dataset are labeled with bounding boxes and classification for a total of eight categories, namely car, cyclist, misc, pedestrian, person-sitting, tram, truck, and van. The dataset is split into 70%-20%-10% ratios for training, validation, and testing sets, respectively, in such a way that each set has the same proportions of labels in them. The ***Fig.*** 3 presents the distribution of different labels across each split of *KITTI 2D object detection*. Notably, the dataset exhibits a significantly higher number of objects labeled "car" compared to other categories. This skewed distribution is appropriate for the purposes of this article, since the focus is primarily on the detection of cars.

To evaluate the performance of the multi-object tracker within the proposed methodology, we utilized annotated video sequences from the *KITTI 2D box tracking* dataset. This dataset comprises 21 videos, each annotated with information such as tracking ID, object type, 2D/3D bounding box dimensions, and object location in camera coordinates. To assess the effectiveness of the proposed multi-object tracker, we employed the evaluation suite provided by KITTI, known as TrackEval (Luiten and Hoffhues (2020)), which allowed us to evaluate the tracker's output on the annotated videos and benchmark against other well-known trackers.

Furthermore, object location information from the same set of videos was utilized to analyze the performance of the proposed lane distance estimation method. This information provided valuable insights into estimating the distance of objects. Lastly,

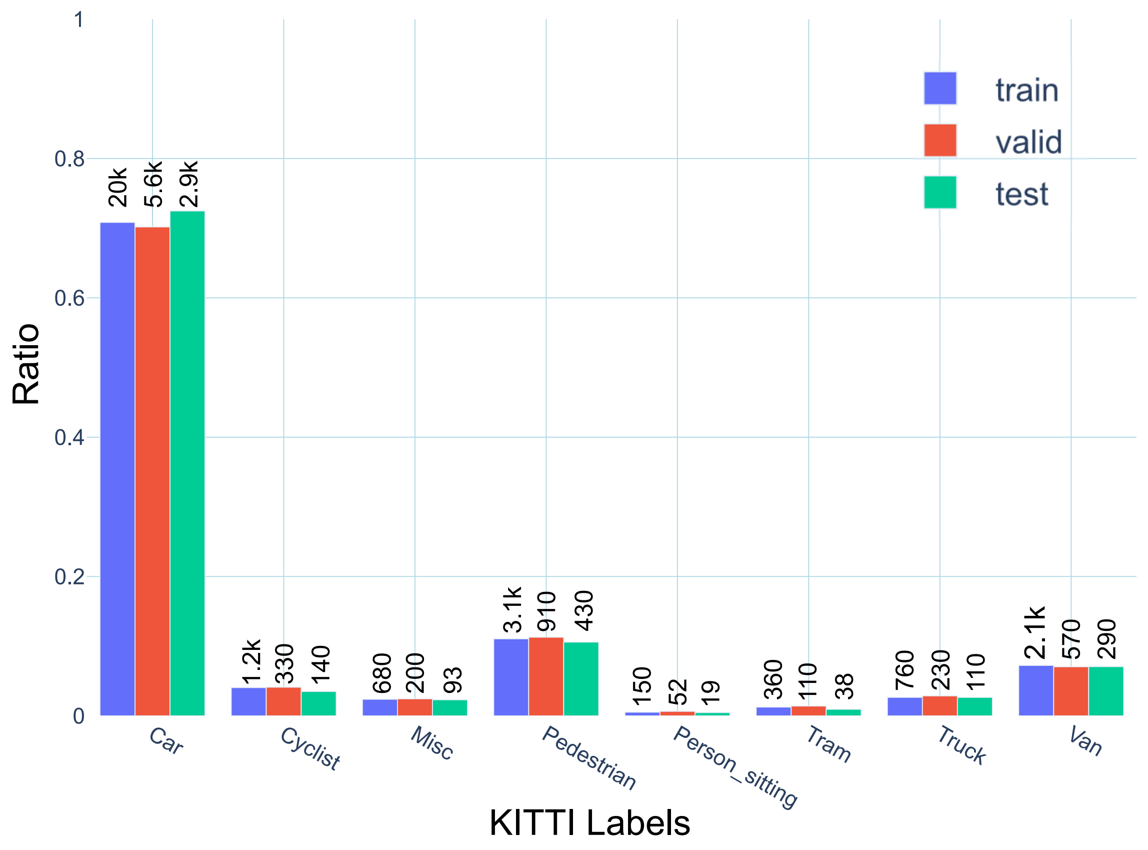


Figure 3: Number of annotations per categories in KITTI across train, validation and test set.

we used the raw GPS data for the videos in the *KITTI 2D box tracking* dataset to generate time-space diagrams. Subsequently, these diagrams were compared to the time-space diagrams generated from the proposed methodology, providing a means to evaluate the accuracy and effectiveness of the proposed approach.

3.2 Object Detection

Within the proposed methodology, we employed the YOLOv5 object detector network, which is part of the You Only Look Once (YOLO) family of single-shot detectors. As showcased by Nepal and Eslamiat (2022); Ge et al. (2021), YOLO networks are known for their exceptional speed and accuracy. These networks make predictions based on predefined anchors, representing potential object locations in the image, with varying box and aspect ratios centered at each pixel.

In the proposed methodology, we utilized the YOLOv5 network, developed by Glenn Jocher (2022) and maintained by an open source community called Ultralytics . This network was trained on the MS-COCO dataset (Lin et al. (2014)) and can detect a wide range of objects across eighty general categories, including classes such as car, bus, truck, and pedestrian. YOLOv5 has demonstrated state-of-the-art performance in object detection on the MS-COCO dataset.

Specifically, we re-trained the YOLOv5m model from Ultralytics on the KITTI 2D object detection dataset, tailoring it to our proposed methodology. To generate new anchor boxes for the KITTI dataset, we utilized the Auto Anchor developed by Ultralytics (Glenn Jocher (2022)). Auto Anchor applies k-means clustering on the KITTI dataset to obtain initial centroids for the Genetic Evolution (GE) algorithm. By running GE for 1000 iterations, the Auto Anchor identifies the optimal anchors that yield the best Intersection over Union (IoU) combined with the Best Possible Recall (BPR) value.

During the training process, we froze the first 15 layers of the YOLOv5m model, utilizing weights from the pre-trained network. The remaining layers were trained using the KITTI 2D object detection training set, which consisted of 5233 annotated images, while 1495 images were reserved for the validation set. Training was carried out with specific hyperparameters listed in ***Table*** 1. We conducted the re-training of the YOLOv5m network on a GRID T4-8C GPU with 8GB memory, running for 200 iterations. Subsequently, we selected the model that achieved the highest mean Average Precision (mAP) on the validation set for further analysis and evaluation.

3.3 Multi-Object Tracking

To generate trajectories for the detected vehicles, it is necessary to track their movements in multiple frames in the video sequence. In the proposed methodology, we utilize a tracking-by-detection multi-object tracker called StrongSORT, developed by Du et al. (2023). This method builds on the classical DeepSORT tracker by introducing improvements in detection, embedding, and association techniques.

Table 1: Hyperparameters used for training YOLOv5 on KITTI dataset

Hyperparmaters	Value
Input shape	(1280, 1280)
Batch size	16
Epochs	200
Learning rate	0.01
Activation function	SiLU (Elfwing et al., 2018)
Optimizer	SGD with momentum (0.937)
IoU threshold for training	0.2

StrongSORT enhances the appearance branch of the tracker by replacing the vanilla CNN network with an OSNet trained on the ImageNet dataset. This allows for the extraction of more discriminative features. Additionally, it incorporates exponential moving average (EMA) in the matching strategy. In the motion branch, StrongSORT implements camera motion compensation (ECC) and replaces the vanilla Kalman filter with the NSA Kalman filter. When evaluated on MOT17 and MOT20 datasets, StrongSORT demonstrates a significant improvement over DeepSORT and state-of-the-art performance (Du et al. (2023)).

In the proposed methodology, we utilize the PyTorch implementation of StrongSORT from Broström (2022). We replace the object detector in Mikel's implementation with the YOLOv5m network trained on the KITTI dataset, as described in ***Sec.*** 3.2. The remaining hyperparameters for the StrongSORT tracker are kept as default for the analysis, and they are listed in ***Table*** 2.

Table 2: Hyperparameters used for StrongSORT tracker.(Broström, 2022)

Hyperparmaters	Explanation	Value
NN_metric	Nearest neighbour distance metric	cosine
Max_dist	Nearest neighbour matching threshold	0.2
Max_IoU_dist	Max association threshold	0.7
Max_age	Maximum number of missed misses before a track is deleted	30
Max_init	Number of frames before object gets tracked	2

3.4 Lane Distance Estimation

Once we have frame-by-frame detection for vehicles with an unique ID, we can then calculate the distance travelled by each unique vehicle on the network link. We use the timestamps for each frame, as well as the GPS information of the camera-mounted vehicle, called the *probe vehicle* from now on. Using frame-by-frame timestamps and the distances for all detected vehicles, we can generate a time-space diagram for the link.

We calculate the distance travelled by detected vehicle by summing the distance traveled by the *probe vehicle* on the link and the distance of the detected vehicle from the *probe vehicle*. For each timestamp t, the distance from the i^{th} detected vehicle can be calculated as:

$$d_i^t = d_{probe}^t + d_{i,probe}^t \tag{1}$$

where d_{probe}^t is the distance traveled *probe vehicle* on the link at time t and $d_{i,probe}^t$ is the distance of the i detected vehicle from *probe vehicle* at time t.

The value of d_{probe}^t is derived from the metadata of the *probe vehicle*. The camera used in the project is equipped with a GNSS sensor to measure the latitude and longitude of the *probe vehicle* for each frame. Given the predetermined GPS location of the start of the link, we can find d_{probe}^t by calculating the geodesic distance between the two GPS coordinates at each timestamp t. To calculate the distance, we use the always convergent method for geodesics proposed by Karney (2011). This geodesic computation algorithm has advantages such as an always converging solution, full double precision accuracy over classical algorithms for the solution of geodesics.

The value of $d_{i,probe}^t$ is calculated using a photogrammetry algorithm based on the similarity triangle. Similar to other research Kumar et al. (2021); Kendal (2007); Diamantas et al. (2016); Nienaber et al. (2015), we use the similarity triangle algorithm to calculate the distance of i^{th} detected vehicle at each frame of the video sequence. The similarity triangle approach states that the ratio of height of the object i in the real world (in meters), $H_m^{i,Real}$ and height of object in image dimensions (in mm), $H_{mm}^{i,Img}$ can be described as,

$$\frac{H_m^{i,Real}}{H_{mm}^{i,Img}} = \frac{D_m^i}{F_{mm}} \tag{2}$$

where D_m^i is the distance of the object from the camera (in meters) and F_{mm} is the focal length of the camera (in mm). The value of $H_{mm}^{i,Img}$ can be calculated using the image height, I_{px}, the image sensor height, S_{mm}, and the height of the object in the image (in pixels), $H_{i,px}$. The relationship is given by,

$$H_{mm}^{i,Img} = \frac{S_{mm} * H_{px}^i}{I_{px}} \tag{3}$$

Using ***Eq.***2 and ***Eq.***3 and reshuffling them, we can calculate the distance of object from the camera (in meters), presented in ***Eq.***4

$$D_m^i = \frac{F_{mm} * H_m^{i,Real} * I_{px}}{H_{px}^i * S_{mm}} \tag{4}$$

For the purpose of analysis, we derive the values of parameters $H_m^{i,Real}$, I_{px}, F_{px} and S_{px} from the KITTI dataset. The values of these parameters used for analysis are presented in Table 3. The average height of the categories is calculated using the annotated 3D bounding box labels in the *KITTI 2D box tracking* dataset. The rest of the parameters refer to the intrinsic properties of the camera used in the KITTI data collection.

Table 3: Parameters used for lane distance estimation from KITTI dataset. (Geiger et al., 2013, 2012)

Parameter	Explanation	Value
$H^{Car,Real}$	Avg. height of car in KITTI	1.50 m
I_{px}	Image sensor height	376 px
F_{px}	Focal length of camera	721 px
S_{px}	Image sensor height	362 px

4 Results

We present the results of the training of YOLOv5 object detector on *KITTI 2D object detection* as well as the evaluation of the StrongSORT tracking and the lane distance estimation method on *KITTI 2D box tracking*. Finally, we conclude the section with a comparison of the time-space diagram generated using the ground-truth data and the proposed methodology.

4.1 Object Detection Results

The evaluation of the YOLOv5 model trained on *KITTI 2D object detection* dataset is carried out on a test set comprising 746 images. The model's performance was assessed using mean Average Precision (mAP), a standard metric for object detection benchmarks Padilla et al. (2020). mAP measures the accuracy of object detectors across all categories in the dataset. It is calculated as the average of the Average Precision (AP) values over all categories for a given Intersection over Union (IoU) threshold that quantifies the overlap of ground truth and prediction bounding boxes. The mAP is given as:

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{5a}$$

$$AP_i = \int\limits_{r=0}^{1} P(R)dR \tag{5b}$$

with AP_i being the AP in the i^{th} category and N is the total number of categories being evaluated.

In ***Table*** 4 we present the mAP at IoU of 0.5 (mAP[.5]) and the average mAP value for IoU ranging from 0.5 to 0.95 (mAP[.5 - .95]) for each category, as well as the combined results. We also present the confusion matrix in ***Fig.*** 4a and the PR curve in ***Fig.*** 4b for the same. On the basis of the evaluation results, we observed that the trained network performs exceptionally well in detecting "car"-like objects such as "car", "van", and "truck" compared to other categories. This finding can be attributed to the relatively higher number of instances in those categories and their rigid structure. On the contrary, categories like "pedestrian" and "cyclist" exhibit more structural variations, resulting in lower prediction capabilities of the model due to the limited variation in the training dataset. In the context of the proposed methodology, it is advantageous to have high accuracy specifically in detecting the categories "car" and "van", since the focus is on identifying motorized vehicles on the road in the subsequent steps.

Table 4: Results of evaluation of re-trained YOLOv5 on test set from KITTI.

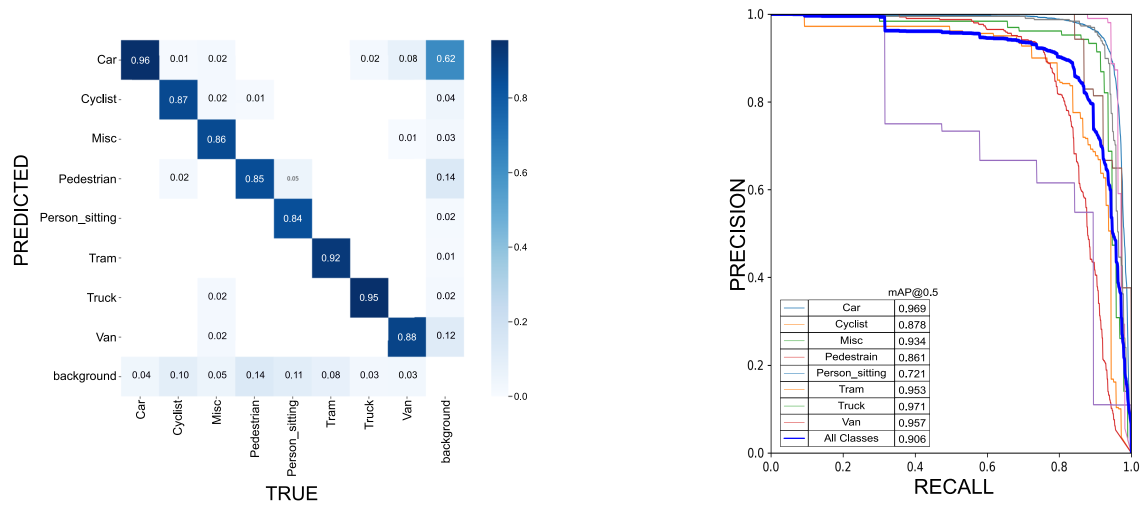
Class	Instances	P	R	mAP[.5]	mAP[.5-.95]
Car	2923	0.926	0.933	0.969	0.789
Van	285	0.898	0.93	0.957	0.785
Truck	107	0.943	0.944	0.971	0.784
Tram	38	0.933	0.868	0.953	0.729
Misc	93	0.929	0.903	0.934	0.66
Cyclist	141	0.873	0.794	0.878	0.556
Pedestrian	427	0.917	0.756	0.861	0.498
Person_sitting	19	0.602	0.797	0.721	0.456
all	**4033**	**0.878**	**0.866**	**0.906**	**0.657**

4.2 Multi-Object Tracking Results

The performance of the StrongSORT multi-object tracker was assessed using 21 annotated videos from the *KITTI 2D box tracking* dataset. The tracker's evaluation was conducted using the Higher Order Tracking Accuracy (HOTA) metric, which measures performance in terms of detection, association, and localization Luiten et al. (2021). HOTA represents the average of the metric function based on detection accuracy (DetA) and association accuracy (AssA) for a given IoU threshold value (α). The HOTA is given as:

$$HOTA = \int_{0<\alpha<1} HOTA(\alpha) \tag{6a}$$

$$HOTA(\alpha) = \sqrt{DetA(\alpha) * AssA(\alpha)} \tag{6b}$$



(a) Confusion Matrix **(b) PR curve**

Figure 4: Results of evaluation of re-trained YOLOv5 on test set from KITTI.

To further assess its performance, we conducted a comparative analysis between the StrongSORT tracker and two other state-of-the-art trackers, namely the Combined Image- and World-Space Tracking (CIWT) Osep et al. (2017) and ByteTrack Zhang et al. (2022b) trackers. Both CIWT and ByteTrack are widely recognized trackers within the tracking-by-detection paradigm. CIWT serves as a standard tracker for evaluating performance on the KITTI dataset and utilizes the Regionlets detector in combination with a Kalman Filter-based tracking method. On the other hand, ByteTrack utilizes a detector and performs association through a two-step process to identify bounding boxes with high and low confidence values. For the detection model, we employed the same YOLOv5 model trained on the *KITTI 2D object detection* dataset for both ByteTrack and StrongSORT. The evaluation focused specifically on the "car" category in the KITTI dataset, and we present the results of the HOTA metric for all three trackers in ***Table*** 5.

The analysis of the HOTA results clearly indicates that StrongSORT exhibits excellent performance on the KITTI dataset, surpassing the other two trackers in all aspects, including detection accuracy (DetA), association (AssA) and localization

(LocA). Furthermore, StrongSORT demonstrates consistent performance across a range of alpha values for the IoU threshold, as illustrated in ***Fig.*** 5.

Table 5: Multi-Object Tracking Results for category "car".

Tracker	DetA	AssA	LocA	HOTA
ByteTrack	44.311	62.176	80.916	51.933
CIWT	63.111	70.562	81.609	66.306
StrongSORT	76.904	72.719	89.221	74.639

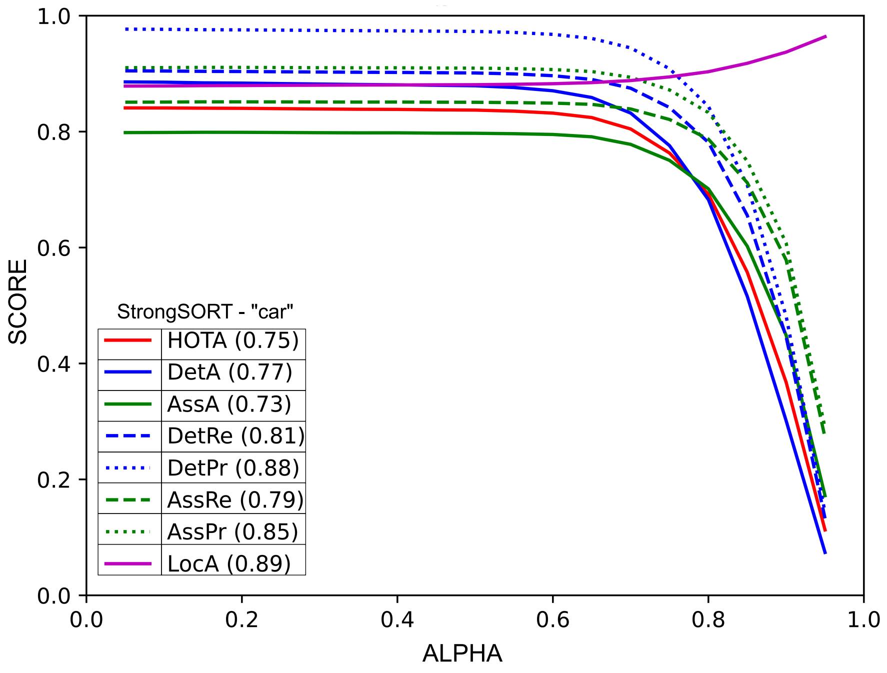


Figure 5: Performance of StrongSORT highlighted by HOTA metric curves for category "car" in KITTI dataset for varying alpha values.

Fig. 6 shows an image frame from the KITTI dataset that shows the bounding boxes and tracking IDs for each "car" found only on the opposite lane using a retrained YOLOv5m object detector and StrongSORT tracker.

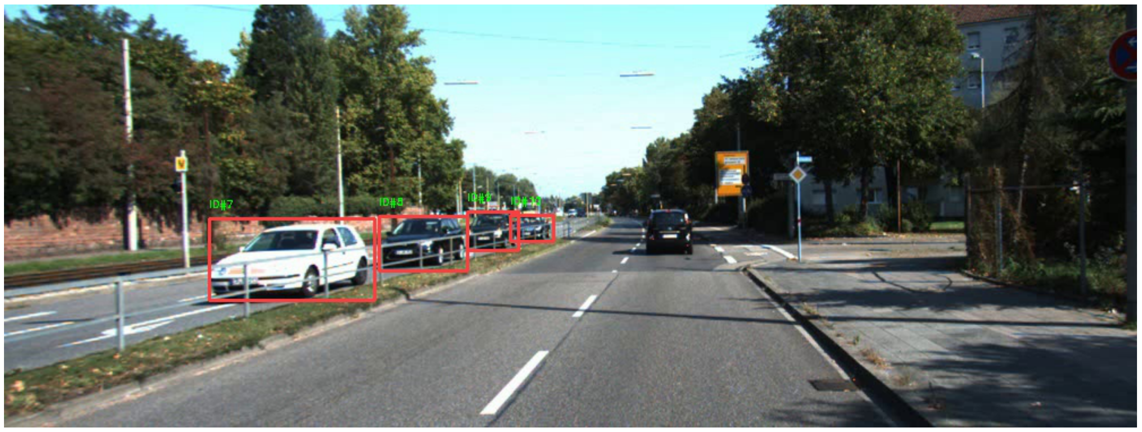


Figure 6: KITTI image frame showcasing the bounding boxes (in RED) and tracking IDs (in GREEN) for vehicles in the opposite lane derived using YOLOv5m and StrongSORT.

4.3 Lane Distance Estimation Results

The accuracy of the generated time-space diagram is contingent upon the precise calculation of the distance of each vehicle detected from the camera using the similarity triangle approach. To evaluate the performance of this approach, we employ the root mean square error (RMSE) as a metric. The evaluation was carried out on 21 videos from the *KITTI 2D box tracking* dataset.

During the evaluation, we compared the camera distance calculated using the bounding boxes for each true positive detection for "car" from the re-trained YOLOv5m model. The RMSE value was computed between the camera distance derived from the similarity triangle approach and the true depth values for each frame in each video. Additionally, we aimed to determine the extent of error caused by the bounding boxes generated by the YOLOv5m model. Therefore, we also calculated the distance from the camera using the ground truth bounding boxes.

Fig. 7 presents the RMSE values for a total of 24939 instances of a "car" across all videos. In the figure, *"RMSE (Calc, GT)"* shows the probability distribution of RMSE calculated between the ground truth depth and the camera distance based on known bounding box values, with a mean of 1.40 [m] and a standard deviation of 0.29 [m]. In the same way, *"RMSE (Pred)"* shows the probability distribution of RMSE calculated between the ground truth depth and the camera distance calculated using the bounding boxes from the re-trained YOLOv5m model, with a mean of 4.41 [m] and a standard deviation of 1.42 [m].

The results revealed that there is a considerable amount of error while calculating the distance using the proposed method. The error in the distance calculation can come from two sources. First, there is some error attributed to the limitations of the similarity triangle approach to accurately calculate the distance. It is evident from the *"RMSE (Calc, GT)"* values with high mean, pointing to a bias in the calculation. This can be because of incorrect camera intrinsic values and assumptions made about the height of the "car" in the real world. Second, inaccuracies in the retrained YOLOv5 model in determining precise bounding box values also contribute to the overall error in the distance calculation.

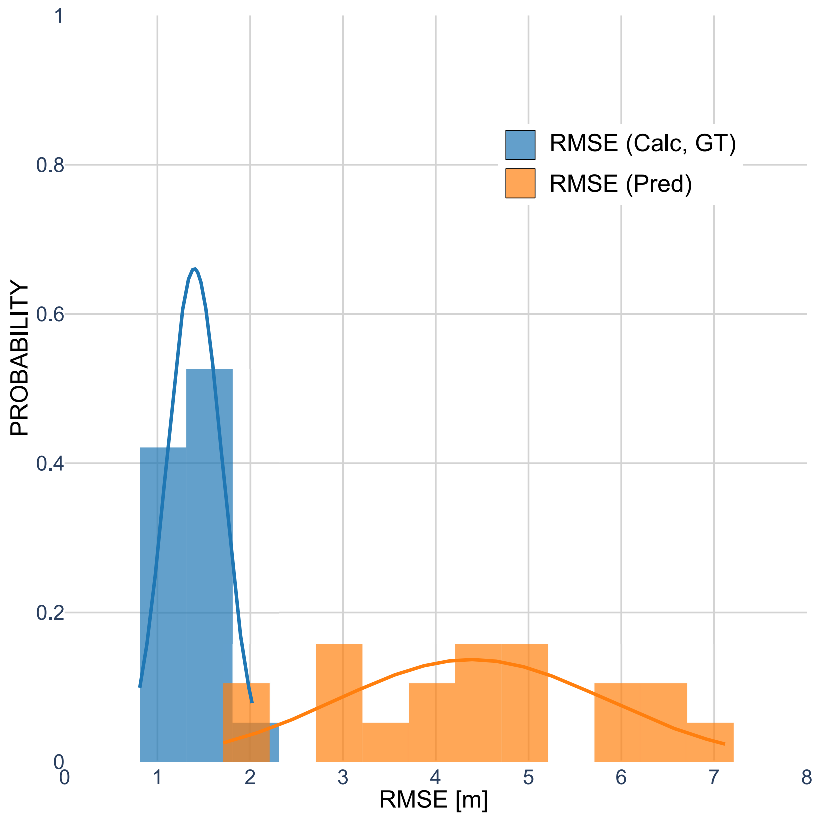


Figure 7: RMSE distribution calculated between the ground truth and proposed methodology.

In order to mitigate the errors associated with the similarity triangle method, advanced algorithms based on deep learning can be integrated Lee et al. (2022), or additional data sources such as LiDAR/RADAR sensors can be utilized de Ponte Müller (2017). Moreover, our observations align with previous research Nienaber et al. (2015), which also noted that errors tend to increase as vehicles move farther away from the camera. In their study, the authors proposed mitigation approaches to reduce such errors. These enhancements would enable more accurate distance calculations.

4.4 Time-space Diagram Results

Finally, a comparison is conducted between the time-space diagrams generated using ground truth data and the proposed methodology, focusing on a specific street-view video sequence. Video ID 0004 from the *KITTI 2D box tracking* dataset is employed for this evaluation. The distance of each detected vehicle on the network link is calculated using the raw GPS values, following the equation described in Eq. 1.

Fig. 8 showcases the time-space diagram for video ID 0004, displaying the trajectories generated by both the actual data and our proposed methodology. It should be noted that the trajectories plotted on the time-space diagram are raw inferences from the proposed methodology and have not undergone any smoothing procedures. The RMSE value between the true trajectories and the trajectories generated by our proposed methodology exhibits a mean of 2.97 [m] and a standard deviation of 1.91 [m].

Upon observing the plot, it becomes apparent that some of the trajectories are not accurately generated using our proposed methodology. Certain errors are attributed to flickering in the bounding boxes when tracking with StrongSORT. This flickering leads to irregular bounding boxes, which subsequently generate erroneous distance values when calculated using the similarity triangle approach. Additionally, other errors discussed in the previous sections, arising from both detection and distance calculation, can also contribute to inaccuracies in trajectory inference. Some of the errors in trajectory extraction because of flickering can be reduced using the trajectory smoothing techniques highlighted in Zhang et al. (2022a).

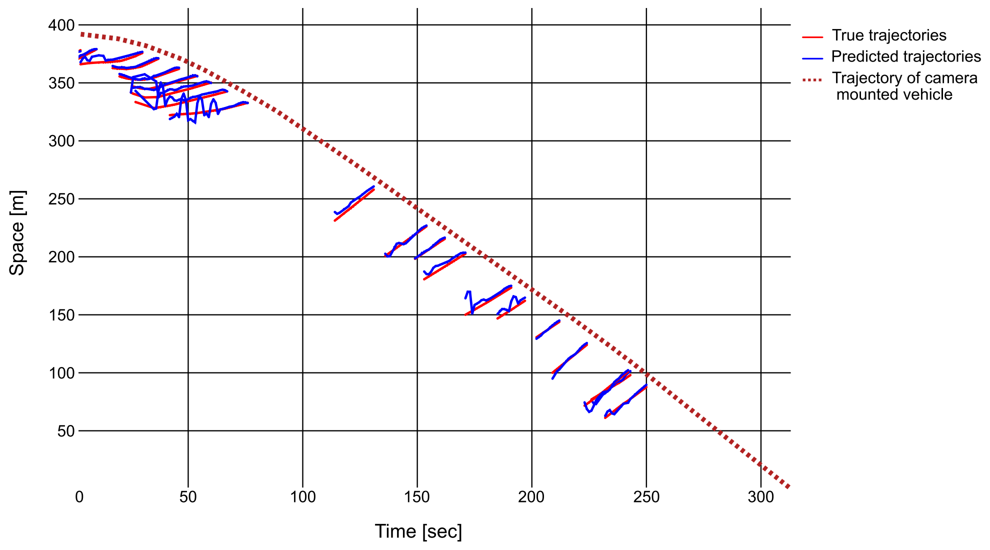


Figure 8: Time-space diagram generated for video 0004.mp4 from KITTI dataset with true trajectories (in RED) and predicted trajectories (in BLUE).

5 Conclusion

This paper introduces an innovative approach for extracting time-space diagrams of road network links from street-view video sequences captured by a camera mounted on a moving vehicle. Our proposed methodology incorporates the YOLOv5 object detector, the StrongSORT multi-object tracker, and the similarity triangle approach for distance calculation.

The evaluation results conducted on the KITTI dataset provide compelling indications that our proposed methodology is capable of generating time-space diagrams that closely resemble the actual ground truth, despite the presence of some inherent errors. These errors arise mainly from the performance of the detector and tracker utilized within the methodology. However, these issues can be mitigated through model re-training using relevant data to enhance their accuracy and robustness. Furthermore, while the simplistic similarity triangle method used for lane distance calculation yields satisfactory results, there is an opportunity for improvement by incorporating more advanced algorithms or integrating additional data sources, such as LIDAR, to achieve even more precise distance calculations.

Advancing the research, the trajectories extracted from the time-space diagram have the potential to be utilized for estimating traffic states throughout the complete road network as well as within specific road segments. Furthermore, an evaluation of the performance of the proposed approach can encompass a range of traffic scenarios, taking into account variables such as speed, density, and volume in the opposing lane. Alternatively, there is an opportunity to explore data fusion techniques, integrating data from stationary sensors like loop detectors to enhance the analytical process.

References

Allström, A., Bayen, A. M., Fransson, M., Gundlegård, D., Patire, A. D., Rydergren, C., and Sandin, M. (2014). Calibration framework based on bluetooth sensors for traffic state estimation using a velocity based cell transmission model. In *Transportation Research Procedia*, volume 3, pages 972–981. Elsevier.

Broström, M. (2022). Real-time multi-camera multi-object tracker using YOLOv5 and StrongSORT with OSNet.

Cao, M., Zhu, W., and Barth, M. (2011). Mobile traffic surveillance system for dynamic roadway and vehicle traffic data integration. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 771–776. IEEE.

de Ponte Müller, F. (2017). Survey on Ranging Sensors and Cooperative Techniques for Relative Positioning of Vehicles. *Sensors*, 17(2):271.

Diamantas, S., Astaras, S., and Pnevmatikakis, A. (2016). Depth estimation in still images and videos using a motionless monocular camera. In *IST 2016 - 2016 IEEE International Conference on Imaging Systems and Techniques, Proceedings*, pages 129–134. Institute of Electrical and Electronics Engineers Inc.

Du, Y., Zhao, Z., Song, Y., Zhao, Y., Su, F., Gong, T., and Meng, H. (2023).

StrongSORT: Make DeepSORT Great Again. *IEEE Transactions on Multimedia*, 25:8725–8737.

Elfwing, S., Uchibe, E., and Doya, K. (2018). Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11.

Essa, M. and Sayed, T. (2018). Traffic conflict models to evaluate the safety of signalized intersections at the cycle level. *Transportation Research Part C: Emerging Technologies*, 89:289–302.

Ge, Z., Liu, S., Wang, F., Li, Z., and Sun, J. (2021). YOLOX: Exceeding YOLO Series in 2021. *arXiv preprint arXiv:2107.08430v2*, 5:12.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11):1231–1237.

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3354–3361.

Glenn Jocher (2022). YOLOv5.

Guerrieri, M. and Parla, G. (2021). Deep Learning and YOLOv3 Systems for Automatic Traffic Data Measurement by Moving Car Observer Technique. *Infrastructures*, 6(9):134.

Herrera, J. C. and Bayen, A. M. (2010). Incorporation of Lagrangian measurements in freeway traffic state estimation. *Transportation Research Part B: Methodological*, 44(4):460–481.

Karney, C. F. F. (2011). Algorithms for geodesics. *Journal of Geodesy*, 87(1):43–55.

Kashinath, S. A., Mostafa, S. A., Mustapha, A., Mahdin, H., Lim, D., Mahmoud, M. A., Mohammed, M. A., Al-Rimy, B. A. S., Fudzee, M. F. M., and Yang, T. J. (2021). Review of data fusion methods for real-time and multi-sensor traffic flow analysis.

Kendal, D. (2007). Measuring Distances Using Digital Cameras. *Australian Senior Mathematics Journal*, 21(2):24–28.

Khajeh Hosseini, M. and Talebpour, A. (2023). Towards Predicting Traffic Shockwave Formation and Propagation: A Convolutional Encoder–Decoder Network. *Journal of Transportation Engineering, Part A: Systems*, 149(4).

Kumar, A., Kashiyama, T., Maeda, H., Omata, H., and Sekimoto, Y. (2022). Citywide reconstruction of traffic flow using the vehicle-mounted moving camera in the CARLA driving simulator. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2022-October:2292–2299.

Kumar, A., Kashiyama, T., Maeda, H., and Sekimoto, Y. (2021). Citywide reconstruction of cross-sectional traffic flow from moving camera videos. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 1670–1678. IEEE.

Lee, S., Han, K., Park, S., and Yang, X. (2022). Vehicle Distance Estimation from a Monocular Camera for Advanced Driver Assistance Systems. *Symmetry*, 14(12):2657.

Li, C.-Y., Huang, H.-J., and Tang, T.-Q. (2017). Analysis of user equilibrium for staggered shifts in a single-entry traffic corridor with no late arrivals. *Physica A: Statistical Mechanics and its Applications*, 474:8–18.

Li, X., She, Y., Luo, D., and Yu, Z. (2013). A Traffic State Detection Tool for Freeway Video Surveillance System. *Procedia - Social and Behavioral Sciences*, 96:2453–2461.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: Common Objects in Context. *Lecture Notes in Computer Science*, 8693:740–755.

Luiten, J. and Hoffhues, A. (2020). TrackEval.

Luiten, J., Osep, A., Dendorfer, P., Torr, P., Geiger, A., Leal-Taixé, L., and Leibe, B. (2021). HOTA: A Higher Order Metric for Evaluating Multi-object Tracking. *International Journal of Computer Vision*, 129(2):548–578.

Nantes, A., Ngoduy, D., Bhaskar, A., Miska, M., and Chung, E. (2016). Real-time traffic state estimation in urban corridors from heterogeneous data. *Transportation Research Part C: Emerging Technologies*, 66:99–118.

Nepal, U. and Eslamiat, H. (2022). Comparing YOLOv3, YOLOv4 and YOLOv5 for Autonomous Landing Spot Detection in Faulty UAVs. *Sensors (Basel, Switzerland)*, 22(2).

Nienaber, S., Kroon, R., and Booysen, M. (2015). A Comparison of Low-Cost Monocular Vision Techniques for Pothole Distance Estimation. In *2015 IEEE Symposium Series on Computational Intelligence*, pages 419–426. IEEE.

Osep, A., Mehner, W., Mathias, M., and Leibe, B. (2017). Combined image- and world-space tracking in traffic scenes. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1988–1995. IEEE.

Padilla, R., Netto, S. L., and da Silva, E. A. B. (2020). A Survey on Performance Metrics for Object-Detection Algorithms. In *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, pages 237–242. IEEE.

Peng, X. and Wang, H. (2023). Network-Wide Coordinated Control Based on Space-Time Trajectories. *IEEE Intelligent Transportation Systems Magazine*, pages 2–16.

Pletzer, F., Tusch, R., Boszormenyi, L., and Rinner, B. (2012). Robust Traffic State Estimation on Smart Cameras. In *2012 IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance*, pages 434–439. IEEE.

Qing-Jie Kong, Zhipeng Li, Yikai Chen, and Yuncai Liu (2009). An Approach to Urban Traffic State Estimation by Fusing Multisource Information. *IEEE Transactions on Intelligent Transportation Systems*, 10(3):499–511.

Rahmani, M., Koutsopoulos, H. N., and Ranganathan, A. (2010). Requirements and potential of GPS-based floating car data for traffic management: Stockholm case study. In *13th International IEEE Conference on Intelligent Transportation Systems*, pages 730–735. IEEE.

Seo, T., Kusakabe, T., and Asakura, Y. (2015). Estimation of flow and density using probe vehicles with spacing measurement equipment. *Transportation Research Part C: Emerging Technologies*, 53:134–150.

Sunderrajan, A., Viswanathan, V., Cai, W., and Knoll, A. (2016). Traffic State Estimation Using Floating Car Data. *Procedia Computer Science*, 80:2008–2018.

Thodi, B. T., Khan, Z. S., Jabari, S. E., and Menendez, M. (2021). Learning

Traffic Speed Dynamics from Visualizations. *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021-September:1239–1244.

Treiber, M. and Kesting, A. (2013). *Traffic Flow Dynamics.* Springer Berlin Heidelberg, Berlin, Heidelberg.

Ua-areemitr, E., Sumalee, A., and Lam, W. H. (2019). Low-Cost Road Traffic State Estimation System Using Time-Spatial Image Processing. *IEEE Intelligent Transportation Systems Magazine*, 11(3):69–79.

Wang, P., Li, P., Chowdhury, F., Zhang, L., and Zhou, X. (2020). A mixed integer programming formulation and scalable solution algorithms for traffic control coordination across multiple intersections based on vehicle space-time trajectories. *Transportation Research Part B: Methodological*, 134:266–304.

Xing, L. and Liu, W. (2022). A Data Fusion Powered Bi-Directional Long Short Term Memory Model for Predicting Multi-Lane Short Term Traffic Flow. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):16810–16819.

Yuan, Y., Zhang, W., Yang, X., Liu, Y., Liu, Z., and Wang, W. (2021). Traffic state classification and prediction based on trajectory data. *Journal of Intelligent Transportation Systems*, pages 1–15.

Zang, J.-r., Song, G.-h., E, R.-t., Sun, J.-p., Zhang, X., and Yu, L. (2019). Experimental Findings about Wide Moving Jams: Case Study in Beijing. *Journal of Transportation Engineering, Part A: Systems*, 145(7).

Zhang, X., Feng, Y., Angeloudis, P., and Demiris, Y. (2022a). Monocular Visual Traffic Surveillance: A Review. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):14148–14165.

Zhang, Y., Sun, P., Jiang, Y., Yu, D., Weng, F., Yuan, Z., Luo, P., Liu, W., and Wang, X. (2022b). ByteTrack: Multi-object Tracking by Associating Every Detection Box. In Avidan, S., Brostow, G., Cissé, M., Farinella, G. M., and Hassner, T., editors, *Computer Vision – ECCV 2022*, volume 13682 of *Lecture Notes in Computer Science*, pages 1–21. Springer Nature Switzerland, Cham.

Zheng, J., Liu, H. X., Misgen, S., Schwartz, K., Green, B., and Anderson, M. (2014). Use of Event-Based Traffic Data in Generating Time–Space Diagrams for Evaluation of Signal Coordination. *Transportation Research Record: Journal of the Transportation Research Board*, 2439(1):94–104.