Deep Traffic Benchmark: Aerial Perception and Driven Behavior Dataset

Guoxing Zhang 2172331925@EMAIL.SZU.EDU.CN
Qiuping Li 2172331941@EMAIL.SZU.EDU.CN
Yiming Liu 2020091010@EMAIL.SZU.EDU.CN
Zhanpeng Wang 2210474142@EMAIL.SZU.EDU.CN
Yuanqi Chen 2210474130@EMAIL.SZU.EDU.CN
Wenrui Cai 2110474146@EMAIL.SZU.EDU.CN
Weiye Zhang 2110474177@EMAIL.SZU.EDU.CN
Bingting Guo 2110474139@EMAIL.SZU.EDU.CN
Zhi Zeng 2100325027@EMAIL.SZU.EDU.CN
Jiasong Zhu ZJSONG@SZU.EDU.CN
Shenzhen University, Shenzhen, Guangdong, China

Editors: Berrin Yanıkoğlu and Wray Buntine

Abstract

Predicting human driving behavior has always been an important area of autonomous driving research. Existing data on autonomous driving in this area is limited in both perspective and duration. For example, vehicles may block each other on the road, although data from vehicles behind them is useful for research. In addition, driving in this area is constrained by the road environment, and the host vehicle cannot observe the designated area for an extended period of time. To investigate the potential relationship between human driving behavior and traffic conditions, we provide a drone-collected video dataset, Deep Traffic, that includes: (1) aerial footage from a vertical perspective, (2) image and annotation capture for training vehicle destination detection and semantic segmentation model, (3) high-definition map data of the captured area, (4) development scripts for various features. Deep Traffic is the largest and most comprehensive dataset to date, covering both urban and high-speed areas. We believe that this benchmark dataset will greatly facilitate the development of drones to monitor traffic flow and study human driver behavior, and that the capacity of the traffic system is of great importance. All datasets and pre-training results can be downloaded from github project.

Keywords: Drone, Driving Behavior, End-To-End Planning, Prediction trajectory of vehicle

1. Introduction

The study of human driving behavior is a key issue for autonomous driving and intelligent transportation systems. A necessary step in validating models of human driving behavior and optimizing the efficiency of transportation systems is the collection of empirical data. Camera equipment is inexpensive and can be analyzed using a variety of computer vision methods, providing an important means for data collection.

Currently, most camera data sets can be divided into mobile observation data sets and static observation data sets. The mobile observation dataset observes the movement of objects in the environment from the perspective of a vehicle equipped with a sensor. The stationary observation dataset observes objects on the road in the field of view of various sensors installed above the road.
Mobile observation datasets refer to observing the behavior of road users in the environment from the perspective of the host vehicle, and many well-known autonomous driving research datasets are mobile observation datasets. Examples include the KITTI dataset from Geiger et al. (2013), the Oxford RoboCar dataset from Maddern et al. (2017), the Waymo Open dataset from Sun et al. (2020), and the Waymo Open Motion dataset from Ettinger et al. (2021). However, since the host vehicle is constantly moving, it is impossible to continuously observe the designated area, and the field of view of the on-board sensor is easily obscured by the surrounding larger vehicles, making it difficult to study the behavior of other road users.
Static observation datasets are more suitable for observing traffic within a fixed spatial area. To facilitate recording the status of vehicles and pedestrians, cameras are typically placed directly over the center of the intersection and remain there for an extended period of time. In this data category, we have also found a number of groundbreaking datasets, including the NGSIM dataset from NGSIM (2016), ARED dataset from Wu et al. (2019), Stanford Drone dataset from Pei et al. (2019), HighD dataset from Krajewski et al. (2018). and the B3D dataset from Wu et al. (2022).
Most existing static observation datasets, including those mentioned above, have several major problems: (1) they rarely provide lane data during traffic operations, (2) they rarely provide the physical world coordinates of vehicles, and (3) they do not fully cover various traffic scenarios. These problems hinder the development of research on human driving behavior and intelligent transportation systems, although these data have great potential. To address the above data problem and compensate for the lack of empirical data, we propose a static ground observation dataset from the perspective of drones: the Deep-Traffic (DT) dataset. The creation process of the DT dataset is shown in Fig. 1.

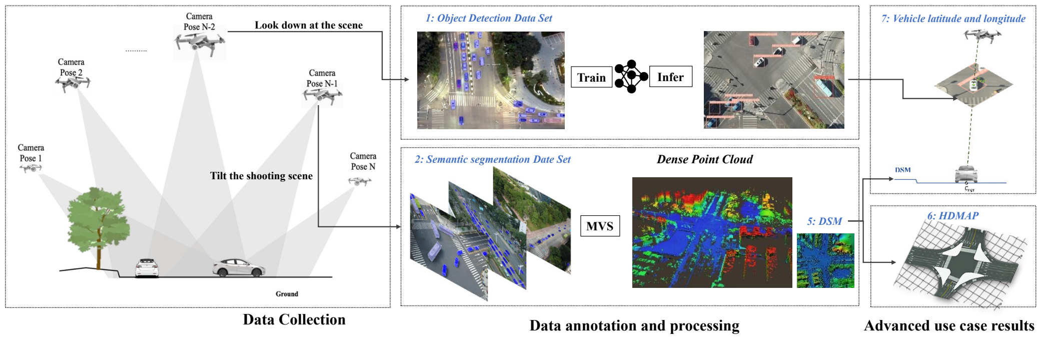


Figure 1: Deep Traffic dataset (DF) workflow.

This dataset has far-reaching implications for solving the problem of airborne detection vehicles, building models for human-controlled vehicle cooperation, and studying the complex interaction between vehicles and the driving environment. It facilitates the design and validation of algorithms for airborne detection vehicles. It can help develop algorithms for autonomous driving that are more human-like, safer, user-friendly, and comply with legal and ethical standards. It can also stimulate the development of new algorithms for traffic system planning and management, further improving traffic management and crash risk reduction strategies. The main contribution to this work is this open source dataset. The rest of this paper is organized as follows. We describe the contents of the dataset in the Dataset section. The details of data collection are discussed in the Data Generation section. Two use cases are presented in the Use Cases section. Finally, we summarize our work in this section and in future work.

2. Dataset

The dataset consists of high-speed and urban areas and includes 30 intersections and road segments. We assign a separate sequence number to each observed intersection as a scene number and use this scene number to be named as head-to-head video, labeled image, high-definition map (HD-MAP) data, and digital surface models (DSM).

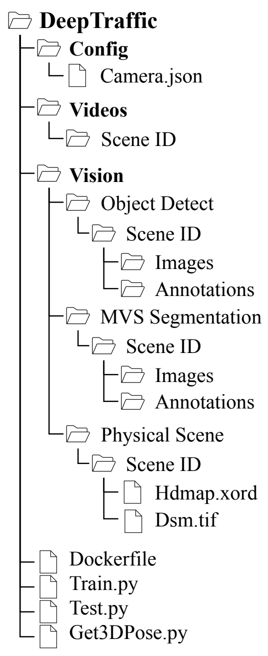
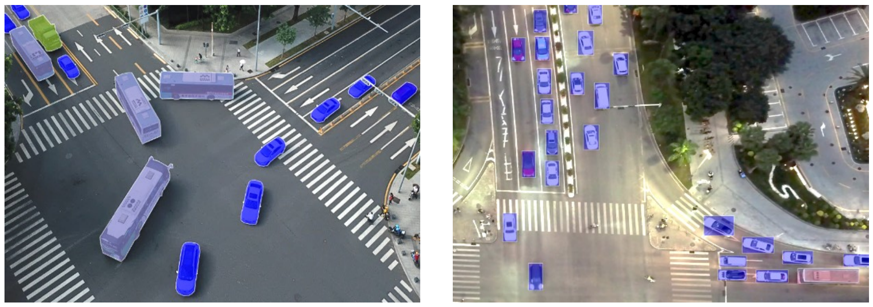


Figure 2: File structure of the DeepTraffic Dataset.

The dataset contains 8000 images with object detection labels, 1000 images with semantic segmentation labels for neural network training, 30 HD-MAP files and DSM files. Anyone can access and download them from our github project. The video, object detection training image, semantic segmentation training image, HDMAP files and DSM files each have a size of 26.5G,10G,5G,255M. Fig. 2 shows the structure of the dataset.

3. Annotated Images

The dataset provides two labeling results for two common computer vision tasks: Object detection and Semantic Segmentation. The training data for object detection is used to train the algorithm to determine the position of each vehicle in the video. The training data for semantic segmentation is used to train the algorithm and determine the pixel position of each vehicle on the image to avoid the interference of vehicles on the road during the creation of the Multi View Stereo (MVS) point cloud. In both object detection and semantic segmentation, vehicles are classified into passenger (e.g., cars and vans), truck (e.g., semi-trucks, trucks with buckets), and bus (e.g., public busses and social busses). To review or edit the annotations provided, we recommend using the open-source computer vision annotation tool Labelme, developed by Torralba et al. (2010).



(a) **(b)**

Figure 3: An example annotated image for semantic segmentation task(a) and object detection task (b).

3.1. Annotated Object Detection

In order to analyze the behavior of the vehicles in the video, the trajectory of each vehicle in the scene must first be captured in the video. When trajectories are captured by a multi-object tracking algorithm, the position of the vehicle is usually captured in key frames and then the tracking algorithm is used to correlate each captured position across frames. The final output is the trajectory of all vehicles in the video on the image. For this purpose, we create a dataset of object detection images and train the vehicle detection model YOLO introduced by Kim et al. (2023). The dataset consists of 8,000 annotated images, 80% of which are used for training, 10% for validation, and 10% for testing. An example of annotation is shown in Fig. 3. The training data covers different intersections and street segments to ensure the diversity of the training patterns.

3.2. Annotated Semantic Segmentation

Driving behavior must be constrained by road conditions, and it is necessary to manually label high-definition map based on dense point clouds for characterization. MVS is a common method in the field of surveying and mapping to generate dense point clouds and DSM, and indicates the pixels on the map that can be ignored before the point cloud is generated to avoid the generation of road marking point clouds by driving vehicles. For this purpose, we create an image semantic segmentation dataset to train a semantic segmentation model UnetPlusplus introduced by Zhou et al. (2018) for vehicles. The dataset consists of 1000 annotated images, 80% of which are used for training, 10% for validation, and 10% for testing. An annotation example is shown in Fig. 3. Based on these annotated semantically segmented images, we generate a dense point cloud of the scene using the MVS method and import it into the high-definition map labeling tool for further processing.

4. Annotated HD-MAP

Based on the point cloud and DSM obtained by MVS method, we manually draw 30 HD-MAP of 30 scenes. Among the 30 scenarios in this dataset, scenes 1-25 are urban intersections and scenes 25-30 are high-speed areas. Fig. 4 shows the scene labeling results. Note that the lane elevation information in the HD-MAP data we provided. We exports the drawing results as a xodr file that conforms to the Open-Driver standard. To review or edit the annotations provided, we recommend using the high-precision map labeling tool (RoadRunner) or using the open-source imap project for visualization.

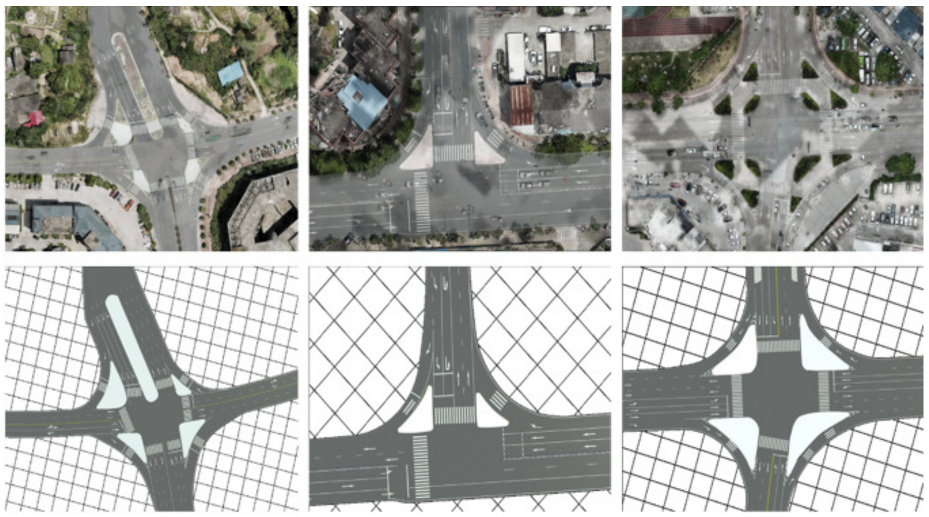


Figure 4: An example of annotated HDMAP by RoadRunner.

5. Development Kit

In addition to the above data, we provide a user-friendly development kit with several sample scripts and a docer image file. It is recommended to use the Docker file we provide to create the container and run the rest of the scripts. The train.py script shows how annotated image data is used to train a neural network model for vehicle detection and semantic segmentation. The test.py script applies the trained model, trained with the train.py script, to the input image. This script serves as an example for evaluation and inference after training. For convenience, we provide optional pre-trained models that can be used directly for inference of object detection and semantic segmentation. This allows users to directly use effective detection models without having to perform the computationally intensive training steps locally. An example of detection results using a pre-trained model is shown in Fig. 5.

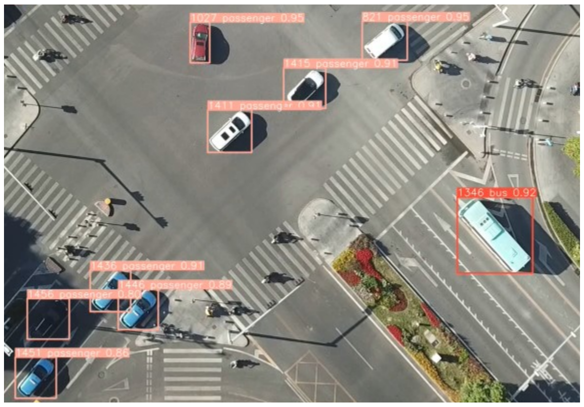


Figure 5: An example image with tracking results. Directly above the rectangle is the object's serial number, category, and confidence scores.

Since most of the current public datasets of drones can only obtain the coordinates on the image of the vehicle, converting the coordinates on the image into world coordinates can avoid the influence of the observation altitude on the research. For this purpose, we provide the scripts Get3DPose.py, which show how to convert the coordinates of a vehicle on the image into latitude and longitude using the principle of minimum projection error.

$$K = \begin{bmatrix} 3042.87009 & 0 & 1543.37909 \\ 0 & 3050.84615 & 1803.67350 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

$$\begin{cases} P_{vgt} = K \cdot (R \cdot C_{vgt} + T) \\ P_{ve} = K \cdot (R \cdot C_{ve} + T) \\ \min_{C_{ve} \in G} \|P_{ve} - P_{vgt}\| \end{cases} \tag{2}$$

The calculation process in the Get3DPose.py script is show in Fig. 6, the rotation matrix R and the translation vector T of the external parameters of the camera are first recovered by the PNP algorithm developed by Schweighofer and Pinz (2008) based on the internal parameters K as Eq.(1) of the UAV camera, the world coordinates of the manually defined ground reference point C_{gn} and the projection coordinates P_{gn} on the image. Finally, the solution of the real coordinates of the vehicle C_{vgt} is transformed to find an estimate of the coordinates of the vehicle $C_{ve} \in G$ such that the spatial distance between the projected point P_{ve} on the image and the central point P_{vgt} of the detected vehicle D_P is minimized, where G is a collection of all points on the DSM data. The mathematical expression is as

Eq.(2). Users can directly use the script to convert the vehicle coordinates detected on the image in the video to latitude and longitude coordinates. Fig. 7 shows an example of the converted result.

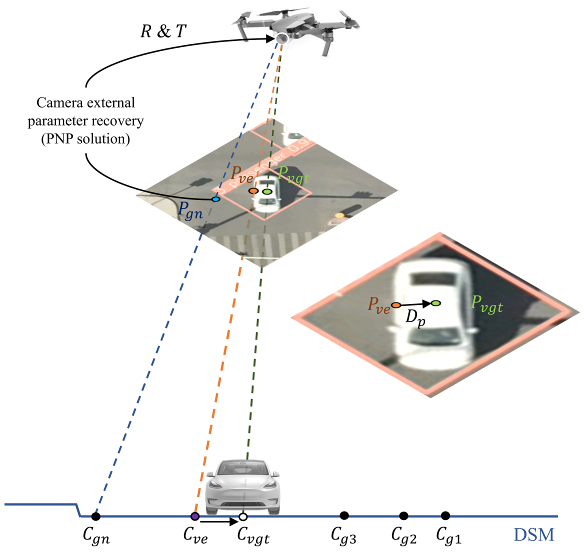


Figure 6: The Convert3d.py tool converts coordinates on the vehicle map to 3D coordinates based on the principle of minimal reprojection error.

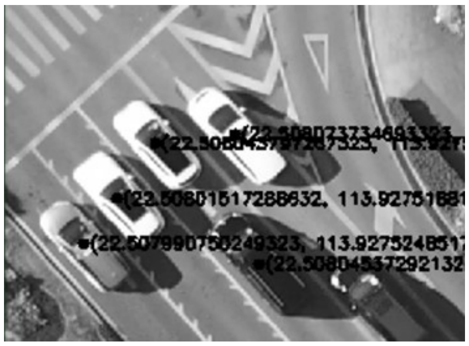


Figure 7: An example image with convert tracking results to Latitude and longitude positioning information.

6. Data Generation

In this section, we describe how the data were generated, including (1) how the videos were collected and annotated and (2) how the MVS images were collected and annotated and (3) how the HDMAP annotations were generated.

6.1. Video Collection and Annotation

The 92 aerial videos were captured between July 2021 and January 2022 in Shenzhen, China, in the Fujian region using the DJI Mavic Pro quadcopter. We numbered each scene based on the location where the video was taken and the key information collected. The date, length, and location of each scene and corresponding video are listed in the 1.

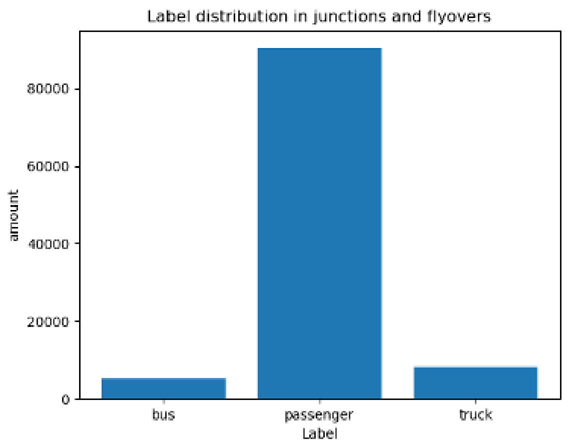


Figure 8: Label distributions in DT.

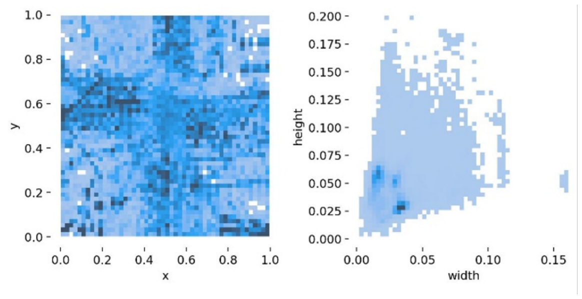


Figure 9: Location and size distributions of labels.

For each video capture, we place the quadcopter's built-in camera vertically above the area to be filmed and hover as long as the battery allows. Most of the data is collected during the day in good weather, but a small amount is collected at night or in poor weather conditions. Since there is an altitude limit, the altitude in urban areas is different from high-speed areas. The altitude is 120 meters or 160 meters above the ground. After data

acquisition, the original video undergoes video compression and image extraction during post-processing. In this process, the original raw video is converted to mp4 format and the 30FPS is reduced to 24FPS.
For labeling the image used for vehicle detection neural network training, we extracted a total of 8000 frames from the processed video by automatically extracting one frame every 72 frames. Then we manually labeled each car in the selected image with the bounding box by category. The distribution of the labeling categories of the final object detection training data is shown in Fig. 8. The distribution range of the labeling results on the image and the size on the image are shown in Fig. 9.

6.2. MVS Images Collection and Annotation

The input to the MVS method is the acquisition of a multiview image of the area of interest by the drone, as shown in Fig. 10. We first collected several sets of data from a fixed height and tilt camera over the observed area, and then manually reshot some areas to complete the MVS image acquisition of the scene. Fig. 10 Dense point cloud (a) is acquired based on the MVS method and converted into DSM(b).

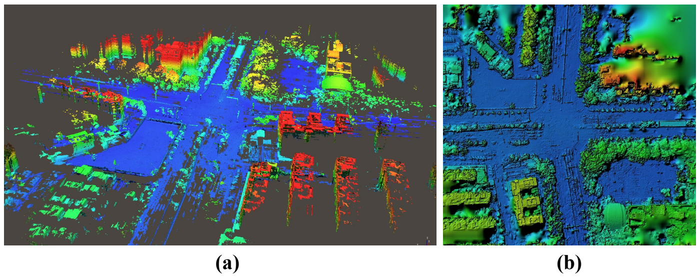


Figure 10: Dense point cloud (a) is acquired based on the MVS method and converted into DSM(b).

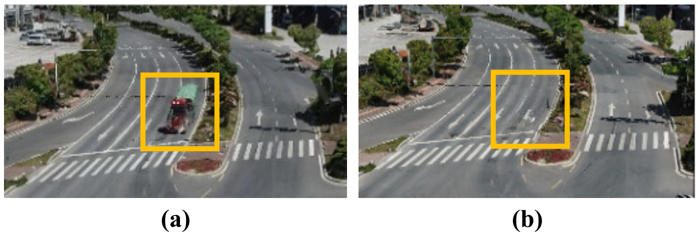
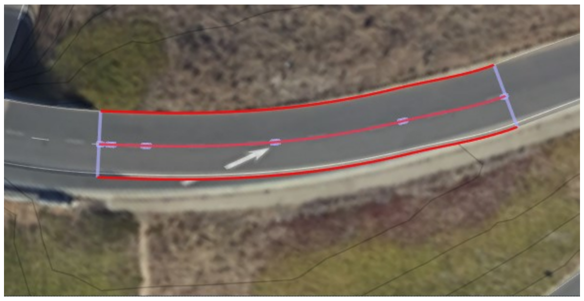


Figure 11: The reconstructed point cloud (b) after using a mask to indicate the pavement vehicle is able to clearly see the markings of the pavement compared to using the original image directly (a).

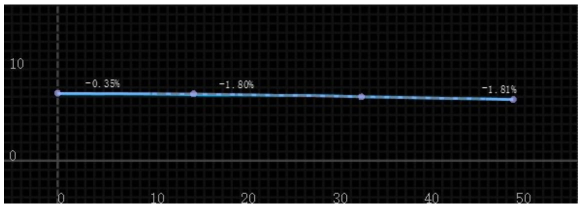
When creating the point cloud, we found that moving vehicles in the MVS image obscured the ground markers, causing interference with the final point cloud data. A common solution is to manually label the mask to specify which pixels in the image input to MVS can be ignored. We extracted high-quality images from the MVS image collection and manually labeled the vehicle contours on the images to create a mask. Masking unrelated vehicles on the image clearly facilitates matching and labeling, as seen in Fig. 11 where the crosshairs are not obscured by vehicles.

7. HDMAP Annotation

We use RoadRunner, the most popular tool for HDMAP labeling, to manually add lane information to the results output by MVS and output them as a map file that conforms to the OpenDriver standard. When mapping, first make sure that the lane and intersection information (such as lane type, number of lanes, lane edges, signage, topological relationships between lanes, etc.) exactly match the MVS output results, as shown (a) in Fig. 12. Finally, the elevations of each lane are manually corrected to match the elevations in the DSM data, as shown (b) in Fig. 12. After the labeling is complete, additional workers review the labeling results and import them into SUMO for simulation to ensure that the topological relationship between the lanes is normal.



(a)



(b)

Figure 12: The horizontal direction of the lane is aligned with the point cloud result (a) and the elevation of the lane is aligned with the DSM (b).

8. Areas of Applications

Specifically, we propose two major use cases: first, predicting vehicle trajectories for driving behavior in structured scenarios, and second, an algorithm for planning autonomous driving end-to-end.
The main application areas are the simulation of human driving behavior in a structured environment and the prediction of vehicle trajectories. A typical modeling process could be as follows: (1) applying detection models and tracking algorithms, such as SORT introduced by Bewley et al. (2016), to obtain vehicle traces and convert them to the world coordinate system; (2) learning human driving models based on trajectories and high-definition map; (3) verifying the correctness of the prediction behavior and the trajectory of the model. Accurate models of human driving behavior can be used in autonomous vehicles to predict the behavior and trajectory of other vehicles on the road.
A second use case is the development of algorithms to plan autonomous driving end-to-end. Using the trajectories of motion of vehicles driven by humans, the most anthropomorphic algorithms for trajectory planning under different road conditions can be studied and the performance of algorithms for end-to-end autonomous driving planning can be improved. As far as we know, using human driving trajectories as learning objects for imitation is still an important part of research on algorithms for autonomous driving. This is mainly due to the lack of data, because it is a difficult task to cover a large enough traffic scene and provide the trajectory of the vehicle in the scenario. Therefore, our dataset can be used as a data source to generate training data and validate candidate algorithms.

9. Summary and Future Work

In this article, we present the Deep-Traffic dataset. In total, the dataset contains 92 processed videos, 9,000 annotated images, 30 HD-MAP files, DSM files, and a development kit. It can be used for a variety of applications including: (1) modeling driver behavior in a structured road environment, (2) predicting multiple types of vehicle motion trajectories, and (3) developing algorithms for intersection prediction and end-to-end autonomous driving. Future work includes enriching the types of vehicles in the video, estimating the trajectories of other participants, and extending the video to capture anomalous traffic phenomena in unstructured road environments.

Acknowledgments

This work was funded in part by the TransUAV team at Shenzhen University. We would like to thank Professor Jiasong Zhu in particular for his generous support in the logistics of data collection.

References

Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE, 2016.

Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9710–9719, 2021.

Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

Jun-Hwa Kim, Namho Kim, and Chee Sun Won. High-speed drone detection based on yolo-v8. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–2. IEEE, 2023.

Robert Krajewski, Julian Bock, Laurent Kloeker, and Lutz Eckstein. The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2118–2125. IEEE, 2018.

Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The oxford robotcar dataset. *The International Journal of Robotics Research*, 36(1):3–15, 2017.

NGSIM. United states federal highway administration (2016) next generation simulation (ngsim) vehicle trajectories and supporting data. 2016.

Zhao Pei, Xiaoning Qi, Yanning Zhang, Miao Ma, and Yee-Hong Yang. Human trajectory prediction in crowded scene using social-affinity long short-term memory. *Pattern Recognition*, 93:273–282, 2019.

Gerald Schweighofer and Axel Pinz. Globally optimal o (n) solution to the pnp problem for general camera models. In *BMVC*, pages 1–10, 2008.

Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020.

Antonio Torralba, Bryan C Russell, and Jenny Yuen. Labelme: Online image annotation and applications. *Proceedings of the IEEE*, 98(8):1467–1484, 2010.

Fangyu Wu, Raphael E Stern, Shumo Cui, Maria Laura Delle Monache, Rahul Bhadani, Matt Bunting, Miles Churchill, Nathaniel Hamilton, Benedetto Piccoli, Benjamin Seibold, et al. Tracking vehicle trajectories and fuel rates in phantom traffic jams: Methodology and data. *Transportation Research Part C: Emerging Technologies*, 99:82–109, 2019.

Fangyu Wu, Dequan Wang, Minjune Hwang, Chenhui Hao, Jiawei Lu, Jiamu Zhang, Christopher Chou, Trevor Darrell, and Alexandre Bayen. Decentralized vehicle coordination: The berkeley deepdrive drone dataset. *arXiv preprint arXiv:2209.08763*, 2022.

Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4*, pages 3–11. Springer, 2018.

Appendix A. Deep Traffic Dataset Scene List

Table 1: Scene sequence ID and video record time

Scene ID	Date	length
N00TCj	2021/10/19 18:06	14:06.0
N03TCf	2021/10/21 8:19	06:30.0
N04TCj	2021/10/22 8:49	08:02.0
N07TCj	2021/10/27 17:09	07:30.0
N08TCj	2021/11/2 18:33	07:33.0
N11TCj	2021/11/11 9:01	08:05.0
N12TCj	2021/11/18 9:07	08:05.0
N14TCj	2021/11/24 11:35	07:46.0
N15TCj	2021/11/28 17:08	07:31.0
N16TCj	2021/12/1 15:09	07:51.0
Y00TCj	2022/4/4 12:11	36:09.0
Y01TCj	2022/4/6 17:42	22:38.0
Y02TCj	2022/4/8 12:26	2:54:20
Y03TCj	2022/4/10 17:31	15:07.0
Y04TCj	2022/4/12 10:30	39:28.0
Y05TCj	2022/4/17 10:21	25:29.0
Y06TCj	2022/4/21 10:36	42:50.0
Y07TCj	2022/4/22 17:29	19:11.0
Y08TCj	2022/4/23 11:04	14:50.0
Y09TCj	2022/4/29 18:00	18:27.0
Y10TCj	2022/4/30 16:44	09:30.0
Y11TSa	2022/10/2 16:41	10:43.0
Y12TSa	2022/10/5 16:12	12:17.0
Y13TSa	2022/10/5 16:39	12:03.0
Y14TSa	2022/10/5 16:41	07:12.0
Y15TSa	2022/10/6 17:34	06:35.0
Y16TSa	2022/10/7 10:08	06:02.0
Y17TSa	2022/10/7 11:06	06:01.0
Y18TSa	2022/10/7 11:14	06:05.0
Y19TSa	2022/10/6 11:33	06:32.0