MONA: The Munich Motion Dataset of Natural Driving

Luis Gressenbuch[1], Klemens Esterle[2], Tobias Kessler[2], and Matthias Althoff[1]

***Abstract*— Real-world datasets facilitate the development of autonomous vehicles, especially when they are accessible, diverse, and provide a measure of accuracy. While existing datasets have been accessible and diverse, they cannot provide any measure of accuracy. To estimate the accuracy of the detection of traffic participants in our setup, we repetitively drove through our observation area with a measurement vehicle with highly accurate localization and LiDAR sensors. Our experiments showed an average overall position accuracy of $0.51\,\mathrm{m}$. The combined data of the autonomous vehicle and the elevated camera setup yield a unique dataset. The elevated view acts as a super sensor of the autonomous vehicle with extended range and reduced occlusions. We employ an auto-labeling system on the stationary camera data to extract trajectories with bounding boxes for each traffic participant. These extracted trajectories are smoothed for kinematic feasibility, and corresponding maps for each location are provided. The Munich Motion Dataset of Natural Driving (MONA) shall empower new research in prediction and planning. Making raw data and code available to the public without license restrictions allows the dataset to be further improved using more advanced algorithms.**

I. Introduction

A. Motivation

Research and development of autonomous driving systems benefit from real-world datasets for various purposes, such as perception, prediction, and planning [1]. While more and more data from on-board vehicle sensors become available due to the growing fleet sizes of prototype vehicles, this type of sensor setup suffers from shortcomings, like occlusions, short observation periods of other vehicles, or inaccurate estimations of vehicle dimensions [2], [3]. Stationary sensor setups at an elevated position are thus a popular way to record traffic data for longer durations, reduce occlusions, and improve accuracy [4]. For this reason, datasets from stationary sensor setups are especially useful for motion planning and prediction to evaluate the performance of different sensor setups. For instance, one can easily remove vehicles from the dataset outside the sensor range to test limited sensing ranges, technical sensor limitations, or inject sensor faults. They also facilitate studying driver variability in the same situation and identify outliers outside usual human behavior.

B. Related Work

A variety of datasets for autonomous driving have been released. Since our dataset mainly addresses motion prediction and planning, we do not review previous datasets for training perception systems. Table I shows relevant datasets for motion prediction and planning, which are discussed in more detail. We decided to omit the well-known NGSIM [10] dataset from Table I since its quality is no longer state of the art [11].

1) On-board sensor data: Multiple datasets from vehicles with on-board sensors have been published, like the Waymo Open Motion Dataset [2], Argoverse [3], nuPlan [9], or CADC [12]. The advantage of these datasets is their amount of data and the accuracy of the employed sensor setup, which typically consists of cameras, LiDAR, and sometimes RADAR. Vehicles with onboard sensors usually visit many locations, but each only a few times. The temporal horizon length is insufficient to observe complex maneuvers that usually take more than 1 or 2 seconds.

2) Stationary sensor data: In the past, stationary camera systems have been set up on elevated spots to record vehicle trajectories [5]. Their advantage is that the field of view usually reduces occlusions compared to onboard vehicle setups. Similarly, a drone camera also provides a clear top-view recording, e.g., highD [6] of highways, inD [7] of urban intersections, openDD [8] and roundD [13] of roundabouts, or INTERACTION [5] of urban intersections and highways. Due to the limited drone battery capacity it is required to interrupt the recording after some time to recharge or exchange the battery. Consequently, the datasets typically only contain a few hours of data recordings.

3) Maps: The lack of standard map formats, or any map at all, limits the usability and requires manual effort to use the data format within open-source frameworks for motion planning, such as CommonRoad [14] or BARK [15].

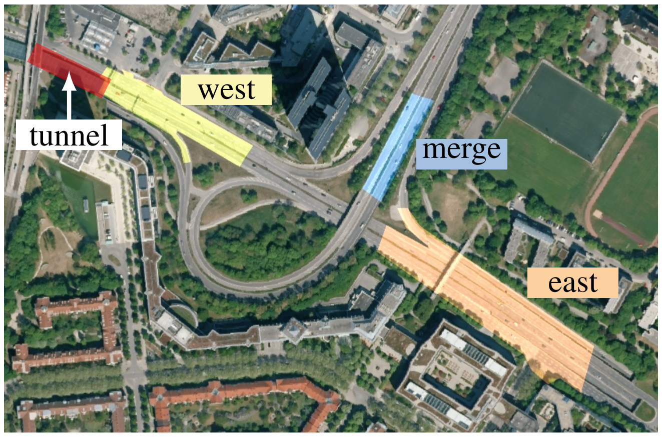


Fig. 1: Detection areas of extracted trajectories. Image source: Bayerische Vermessungsverwaltung (CC BY 3.0 DE).

[1] Department of Informatics, Technical University of Munich, Garching, Germany. {luis.gressenbuch, althoff}@tum.de

[2] fortiss GmbH, Research Institute of the Free State of Bavaria for Software-intensive Systems, Munich, Germany. Klemens Esterle and Tobias Kessler left fortiss GmbH prior to the publication of this work.

TABLE I: Comparison of recent trajectory datasets for road vehicles. For licensing, we distinguish between commercial use (CU) and non-commercial use (NCU). "-" indicates the data not being available or applicable. † processing pipeline for static video records published in [4]. ‡ data taken from [2]. § cameras recorded at different frequencies. †† collected through personal communication.

	INTERACTION [5]	highD [6]	inD [7]	openDD [8]	Waymo [2]	Argoverse [3]	nuPlan [9]	Ours
location	urban intersections + highway	highway	urban intersections	roundabouts	urban	urban	urban	urban
sensors	static drone or camera	static drone	static drone	static drone	on-board	on-board	on-board	static camera
processing	offline	offline	offline	offline	offline	online	offline	offline
No. of locations	11	6	4	7	100k	323k	-	3
No. of tracks	110k	20k	12k	85k	7.6M	11.7M‡	39M††	702k
avg. track length	19.8 s	13.6 s	59.5 s	17.6 s	7.0 s	2.5 s	9.6 s††	16 s
record time	10 h	16.5 h	10 h	62 h	574 h	320 h	1500 h	130 h
record frequency	30 Hz	25 Hz	25 Hz	30 Hz	-	30 Hz/5 Hz§	20 Hz/10 Hz§	25 Hz
track frequency	10 Hz	25 Hz	25 Hz	30 Hz	10 Hz	10 Hz	20 Hz	5 Hz
map format	Lanelet2	none	none	non-standard	non-standard	non-standard	non-standard	opendrive + lanelet
raw data	no	no	no	no	no	no	yes	yes
pipeline	partially†	no	no	no	no	no	no	yes
license	NCU	NCU	NCU	NCU + CU	NCU	NCU	NCU	NCU + CU

4) Data extraction: Due to the amount of data to be processed, all datasets rely on automatic labeling. The recorded data is usually processed offline. However, the lack of complete data, e.g., raw video footage, or reference data, prevents enhancements to the pipeline from persons other than the dataset publishers. NuPlan is the only other dataset that provides processed trajectories together with the raw sensor data (for a subset of their dataset). However, they do not provide the tracking pipeline, prohibiting others from improving the data. No dataset in the literature has provided accuracy measures for the extracted traffic participants.

5) License: Apart from openDD [8], no dataset can be used for commercial use free of charge. To facilitate the transfer from research to industry, we argue that restricting the dataset to non-commercial use limits its impact.

C. Contribution

We created a dataset addressing the above problems by providing the following contributions:

- Dataset covering urban roads with multiple lanes, an inner-city highway stretch, and their transitions cf. Fig. 1;
- Simultaneous data from stationary sensors and onboard vehicle sensors
- Advanced data extraction from video footage at scale
- Open-source tools and data with no license restrictions
- Longer recordings with fewer interruptions compared to other stationary datasets
- Maps in standard formats for all our locations
- Comprehensive evaluation of extracted vehicles length, width, and position accuracy.

We freely provide our data and software as open-source, allowing non-commercial and commercial use. The source code and data are available at `commonroad.in.tum.de`.

The remainder of the paper is structured as follows. Section II gives an overview of our dataset, and Section III describes our trajectory extraction approach from videos. The accuracy and features of the dataset are analyzed in Section IV, and we conclude the paper in Section V.

II. Dataset Overview

In this section, we present the video recordings, trajectory data, maps, and reference data of our dataset.

A. Video Recordings

The video footage was recorded in August 2021. Three cameras were placed on the 28th floor of the Highlight Tower 1, located in north Munich, at the interchange of the Autobahn A9 and the Schenkendorfstraße, a main bypass road in Munich. In total, 130 h of video material from three different perspectives in various weather conditions (e.g., sunshine, overcast and rain) have been recorded. The recording setup consisted of three Sony FDR-AX43 video camcorders with a resolution of 3840x2160 pixels at a frame rate of 25 Hz. The cameras were fitted with a circular polarizer to reduce reflections in the window glass of the building. Moreover, each recorded video was seamlessly divided into segments of 30 min and every video file is accompanied by a meta file containing detailed information about the recording time, resolution, and frame rate.

B. Trajectory Data

We extract trajectories from the video material in the areas highlighted in Fig. 1 by applying the processing pipeline described in Section III. Table II lists the statistics of the extracted trajectory dataset with respect to the recording location. In total, 702k trajectories have been extracted, corresponding to driving a distance of 124 500 km in 3127 h. Our results were saved in the columnar Apache Parquet format[1] for efficient storage and fast reading speeds. Each row provides information about the state and type of the vehicle in every frame. The provided columns are fully compatible with the INTERACTION dataset [5], thereby

[1]`parquet.apache.org`

TABLE II: Dataset statistics by location. † per driving direction

Location	east	west	merge
recorded time in h	45.9	43.1	46.2
driven time in h	1804	1051	272
driven distance in km	71k	41k	13k
No. of cars	276k	235k	113k
No. of trucks	38k	25k	8k
a-priori feasible	81.9%	82.6%	77.5%
feasible after optimization	96.6%	97.8%	97.0%
No. of reference drives†	13 / 12	9 / -	4 / 2

facilitating the usage of our dataset, as existing data loading tools can be reused.

Our toolchain provides command-line interface tools to extract video footage corresponding to individual trajectories, which can be optionally annotated with the pipeline stage results. We also provide a tool to convert extracted trajectories into CommonRoad scenarios [14].

C. Maps

The maps for all three locations were manually created using MathWorks RoadRunner[2] R2021a based on calibrated digital orthographic aerial imagery and information from the Bavarian road information system (BAYSIS)[3]. We provide maps in the standardized OpenDRIVE[4] and CommonRoad format [14]. Additionally, the conversion to Lanelet/Lanelet2 is supported [16]. Fig. 2 displays the three recording locations and the corresponding maps.

D. Reference Data

To provide test data with known ground truth, we used the *fortuna* autonomous driving vehicle demonstrator [17] to measure position, velocity, heading, and yaw rate, while driving through the observation area of all three locations on three separate days. The measurements were conducted using an iMAR iNAT FSSG-1 inertial navigation system that uses information from a fiber optic gyroscope and an RTK GNNS receiver to achieve a position accuracy of up to $0.02\,\mathrm{m}$ at a frequency of $100\,\mathrm{Hz}$. The last row of Table II displays the number of reference drives for the different locations. The separator indicates the number of drives for each of the two structurally separated main carriageways. We also provide the recorded perception data of the Apollo driving stack running on *fortuna* for all reference drives [18]. The perception uses a Velodyne VLP-32C LiDAR sensor to detect surrounding vehicles.

III. Pipeline for Data Processing

We base our trajectory extraction toolchain on the work of Clausse *et al.* [4]. However, multiple modifications described in this section have been made to improve the scalability, accuracy, and occlusion robustness. We apply the following sequential processing stages of the pipeline depicted in Fig. 3, which are described in detail in the remaining section:

[2]`mathworks.com/products/roadrunner.html`
[3]`baysis.bayern.de`
[4]`opendrive.org`

1) During the detection stage, the bounding box of vehicles in pixel coordinates is obtained in video frame (Section III-A).
2) The pose estimation provides the length, width, heading, and position of detected vehicles in world coordinates for each frame (Section III-B).
3) The association stage assigns detections of the same vehicle in consecutive frames to the same vehicle ID (Section III-C).
4) A smoothing stage reduces noise in the detected trajectories (Section III-D).
5) As the smoothing cannot ensure the feasibility of a trajectory with respect to a kinematic vehicle model, feasible trajectories are obtained by solving an optimization problem (Section III-E).

The toolchain is implemented in Python and allows researchers to easily integrate their approaches for detection, pose estimation, tracking, and smoothing. We apply parallelization, vectorization, and just-in-time compilation to improve the processing speed of our pipeline and provide the documentation of our API on the CommonRoad webpage.

A. Detection

For vehicle detection, we use the object detection neural network architecture *Detectron2*[5] with pre-trained model weights from the COCO dataset [19]. The neural network processes each frame independently and outputs the bounding box in image coordinates, an instance segmentation mask, a class label, and a corresponding confidence score for each object detected.

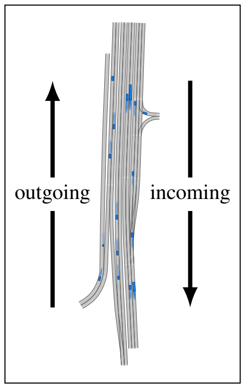
B. Pose Estimation

Vehicle pose estimation from monocular RBG video is an active research topic [20]–[22]. However, its application to our dataset is limited since the ego perspective from a vehicle is often assumed. Likewise, neural networks trained on datasets recorded from a vehicle, like in the KITTI dataset [23], are highly dependent on the perspective. Furthermore, the objects in the image only comprise small patches, making it difficult to extract features. Therefore, geometrical approaches as, e.g., proposed by Clausse *et al.* [4], use the segmentation mask of a detected vehicle to maximize the intersecting area with a projected 3D cuboid in image space. However, this approach assumes a fixed size per vehicle type (e.g., car, truck, and bus), introducing inaccuracies in the occupied area. Moreover, since the optimization problem is highly nonlinear, it often only converges to a local optimum and is computationally expensive.

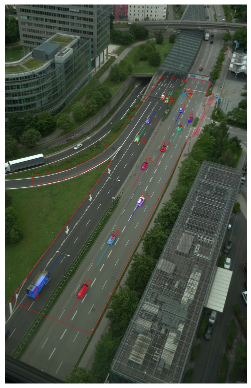
Since the height of a vehicle is usually not of interest for tasks like motion planning or prediction, we limit ourselves to estimating the center position, heading, length, and width of a vehicle. To obtain a partial vehicle contour in world coordinates, we extract contour points from the silhouette of the instance segmentation mask in a known plane, i.e., the underfloor of a vehicle [24], and project them to the

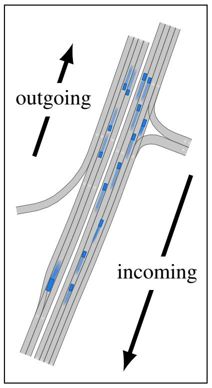
[5]`github.com/facebookresearch/detectron2`

(a) Camera perspective "east".

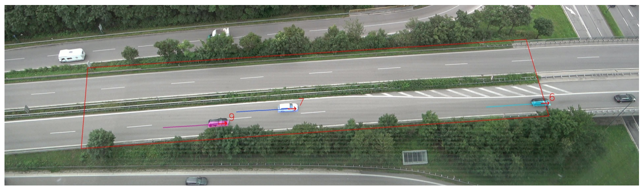


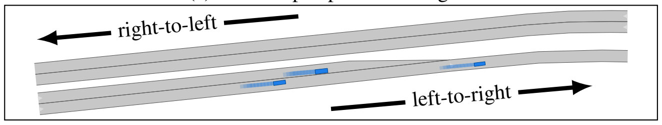
(b) CommonRoad scenario "east".


(c) Camera perspective "west".



(d) CommonRoad scenario "west".


(e) Camera perspective "merge".



(f) CommonRoad scenario "merge".

Fig. 2: Recording and corresponding map in the CommonRoad format for all three locations.

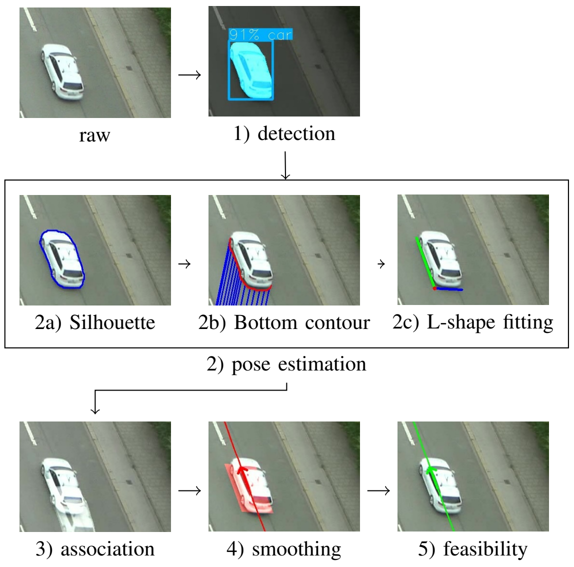


Fig. 3: Processing pipeline stages with corresponding visualized results.

world frame using a calibrated camera model. However, due to the differing perspectives compared to [24], extracting the lower half of the contour does not necessarily correspond to the contour of the underfloor. Therefore, we propose the following approach independent of the perspective: The contour points are projected to the ground plane in world coordinates. The projected points form a polygon with the boundary $\mathcal{B} \subset \mathbb{R}^2$ and vertices $\mathcal{L} \subset \mathcal{B}$. We cast a ray $\mathcal{R}_{\mathbf{l}} = \{\mathbf{c} + s(\mathbf{l} - \mathbf{c}) | s \in \mathbb{R}_{\geq 0}\}$ from the camera position $\mathbf{c}$ to each vertex $\mathbf{l} \in \mathcal{L}$; $\mathbf{l}$ is considered part of the bottom contour, if $\|\mathbf{c} - \mathbf{l}\|_2 = \min_{\mathbf{l}' \in (\mathcal{R}_{\mathbf{l}} \cap \mathcal{B})} \|\mathbf{c} - \mathbf{l}'\|_2$ (see 2b) of Fig. 3). The required intrinsic and extrinsic parameters of the camera model are obtained once during processing pipeline setup by matching pairs of corresponding pixel and world coordinates; see [25] for further details.

We use the approach from Jung *et al.* [26] to efficiently fit an L-shape to the set of bottom contour points via least-squares optimization (see 2c) of Fig. 3). To improve the robustness towards outliers, the fitting is performed using a RANSAC estimator [27]. As the spatial resolution of the camera image decreases with increasing distance and the vehicle dimensions are known to be constant, we average the length and width over the closest 100 poses to the camera position after association, as described in Section III-C. Then, the center point of each pose can be trivially obtained from the corner of the L-shape and the length and width of the vehicle.

C. Association

Clausse *et al.* [4] use a simple multi-object tracking approach derived from Bochinski *et al.* [28]. An optical tracking algorithm bridges gaps in visibility for partial occlusions. However, the approach relies on the assumption that vehicles are not fully occluded. As in our application, vehicles might be fully occluded by bridges, traffic signs, and bushes for several frames, we apply a tracking approach based on the work of Wojke *et al.* [29] to alleviate this limitation. It uses the uncertainty estimate of a Kalman filter to establish a probability measure that a detection belongs to a certain object in addition to the Intersection-over-Union (IoU) metric. We adapt the method by applying the Kalman filter in the world frame since the perspective, and significant distance to the camera makes tracking in the image frame inapplicable. While the original approach also uses a Re-ID network to reduce the number of identity switches, we leave the implementation of this component to future work.

D. Smoothing

As in previous works [4], [30], we process the noisy tracks using the Rauch-Tung-Striebel (RTS) smoother [31]. Schubert *et al.* [32] investigated the performance of different vehicle models for vehicle tracking applications. They found that the model with the lowest position error for an urban scenario is a model assuming constant yaw rate and constant acceleration. In contrast to our application, their experimental setup used vehicle velocity and yaw rate as measurement inputs in addition to position. Since increasing the model order while only measuring the position of the vehicle, increases the sensitivity of the signal-to-noise ratio, we choose the best performing constant velocity model for the underlying filter. The state of our vehicle model is defined as $\mathbf{x} = (s_x, s_y, v, \psi, \omega)^{\mathrm{T}}$, where s_x, s_y denote the x- and y-position of the vehicle, ψ the heading, v the velocity, and ω the yaw rate. The discrete-time dynamic model of our vehicle is [30]

$$\mathbf{x}[k+1] = \mathbf{x}[k] + \int\limits_{k\Delta t}^{(k+1)\Delta t} \begin{pmatrix} v(t)\cos(\theta(t)) \\ v(t)\sin(\theta(t)) \\ 0 \\ \omega(t) \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ w_v \\ 0 \\ w_\omega \end{pmatrix} \mathrm{d}t, \tag{1}$$

where Δt denotes the time step size, and $w_v \sim \mathcal{N}(0, \sigma_v^2)$ and $w_\omega \sim \mathcal{N}(0, \sigma_\omega^2)$ characterize zero-mean Gaussian white noises to account for model errors as well as unknown longitudinal and angular accelerations.

Due to the model's nonlinearity, an extended Kalman filter (EKF) is employed as the underlying filter of the RTS smoother. The filter is updated by position measurements of the vehicle $h = (x, y)^{\mathrm{T}} + (\tilde{v}_x, \tilde{v}_y)^{\mathrm{T}}$, obtained by the pose estimation described in Section III-B. Furthermore, noise in the measurement is captured by the additive zero-mean Gaussian white noises $\tilde{v}_x \sim \mathcal{N}(0, \sigma_x^2)$ and $\tilde{v}_y \sim \mathcal{N}(0, \sigma_y^2)$.

E. Optimization for Drivability

The output trajectories from the track smoothing stage are not necessarily drivable by a standard vehicle model due to measurement noise. Therefore, we implement a nonlinear optimal control approach to post-process the smoothed trajectories and ensure their feasibility.

The CommonRoad drivability checker [33] is used to check the a-priori trajectory feasibility with the kinematic single-track model and corresponding parameters as defined in [14]. We chose the longitudinal acceleration and the steering velocity as input $\mathbf{u}$. Furthermore, $\mathbf{x}$ are the states resulting from smoothing and $\hat{\mathbf{x}}[1], ..., \hat{\mathbf{x}}[N]$ from the optimized input trajectory $\hat{\mathbf{u}}$ and initial state $\hat{\mathbf{x}}[0]$. The state space is chosen as described in the smoothing stage, where the yaw rate is replaced by the steering angle. Finally, the optimization problem is formulated as

$$\min_{\mathbf{u}[0],...,\mathbf{u}[N-1],\mathbf{x}[0]} \sum_{k=0}^{N} \mathbf{e}[k]^{\mathrm{T}} Q_x \mathbf{e}[k] + \sum_{k=0}^{N-1} \hat{\mathbf{u}}[k]^{\mathrm{T}} Q_u \hat{\mathbf{u}}[k]$$
$$\text{subject to } g(\hat{\mathbf{x}}[k], \hat{\mathbf{u}}[k]) \leq \overline{\mathbf{g}}\,,$$

where $\mathbf{e}[k] = \mathbf{x}[k] - \hat{\mathbf{x}}[k]$ is the difference between the smoothed and optimized state, Q_x and Q_u are weight matrices, and g is the vehicle constraint function with upper limit $\overline{\mathbf{g}}$.

Since the main objective is to repair the position and heading from the smoothing stage, we only set weights $Q_x = \mathrm{diag}(0.1, 0.1, 0, 1, 0)$ on position and heading deviations. In order to preserve the characteristics of the original trajectories under the assumption that human drivers primarily maintain an energy-efficient driving style, we use small weights of $Q_u = \mathrm{diag}(10^{-3}, 10^{-4})$ for the control input.

The nonlinear optimization problem is solved by direct multiple shooting [34, Sec. 4.3] using CasADi [35] with its BlockSQP [36] implementation[6]. Before the optimization, trajectories are downsampled to a frequency of $5\,\mathrm{Hz}$ to fit the needs of most application scenarios and reduce unnecessary complexity. Since not all trajectories could be made feasible due to model mismatch or optimizer time out, we remove all trajectories where the optimizer has not converged or where the feasible trajectory is on average more than $0.1\,\mathrm{m}$ away from the original trajectory.

F. Trajectory Filtering

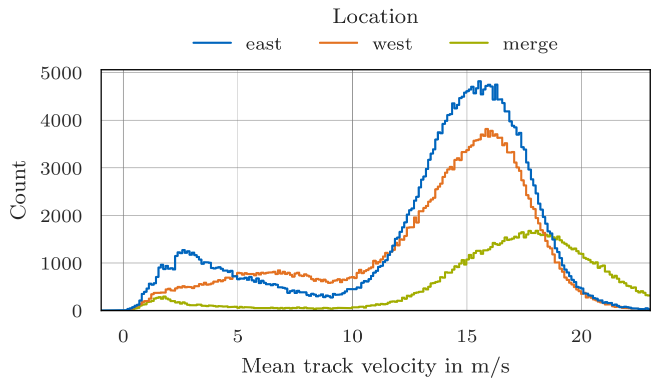
Due to inaccurate or false positive detections, the extraction results are filtered by various plausibility criteria, such as vehicle size, trajectory length, or mean velocity. As the filter parameters are an application-specific trade-off between removing false positives and discarding genuine trajectories, we provide a modular and parameterized filter architecture to comply with the diverse needs of future users.

IV. Dataset Analysis

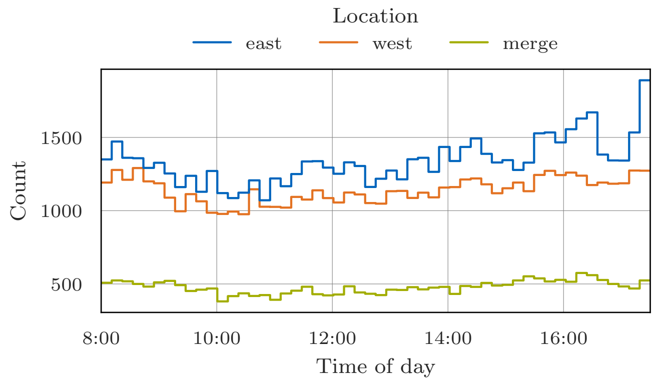
A. Statistical evaluation

We present a statistical analysis since the number of trajectories in the dataset does not allow one to inspect specific features manually. Fig. 4a shows the mean velocity histograms for each location. Two distinct peaks for the most frequent velocities can be observed. The lower velocity peak is around $2.5\,\mathrm{m/s}$, whereas the higher peak is approximately $16\,\mathrm{m/s}$. The latter is not surprising, as it corresponds to the allowed speed of $16.67\,\mathrm{m/s}$. The former corresponds to low velocities during congestion in the morning and afternoon

[6]The implementation uses HSL, a collection of Fortran codes for large-scale scientific computation. See `http://www.hsl.rl.ac.uk`.



(a) Histogram of mean track velocities.



(b) Number of tracks by time of day.

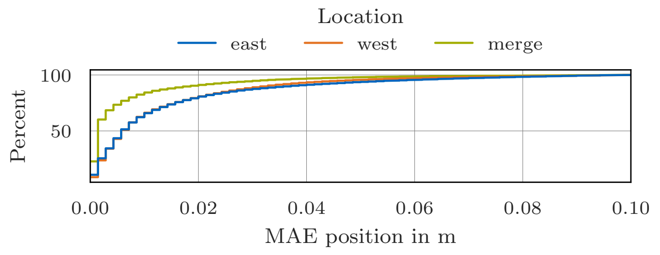
Fig. 4: Statistical evaluation of extracted trajectories.

rush hour. Fig. 4b reveals a peak of up to 18 300 detected vehicles per hour, accumulated over all three locations.

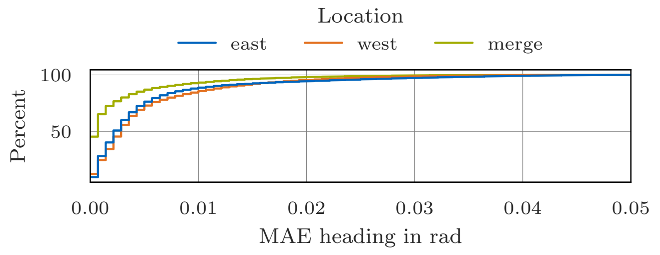
In Table II, we state the percentage of feasible trajectories after smoothing and that could be made feasible with the accepted deviation from the measured trajectory. Of all trajectories, $81.4\,\%$ and $97.1\,\%$ were feasible after smoothing and optimization, respectively. Fig. 5 shows the cumulative histogram of the mean position and heading deviations introduced in the feasibility processing stage. For most cases, only slight deviations from the smoothed trajectories were necessary to obtain a feasible solution.

B. Accuracy Evaluation

1) Time Synchronization: The cameras recording the video footage did not provide an option for accurate time synchronization with the GNSS time. Therefore, we manually determined the frame in the video where the rear axle of the *fortuna* vehicle passes a landmark with a known geographical position, e.g., a lane marking, and found the closest position measurement from the *fortuna* vehicle to the landmark. This time stamp marks the synchronization point between camera and vehicle. Since the duration of each reference drive does not exceed $15\,\mathrm{s}$, the clock drifts during this time are neglected. The error bound of this procedure is given by the sampling frequency of the camera $f_{\text{video}} = 25\,\mathrm{Hz}$ and the frequency of the inertial navigation system $f_{\text{INS}} = 100\,\mathrm{Hz}$: $\frac{1}{2}(f_{\text{video}}^{-1} + f_{\text{INS}}^{-1}) = 25\,\mathrm{ms}$, yielding



(a) Position deviations.



(b) Heading deviations.

Fig. 5: Cumulative histogram of deviations introduced in feasibility optimization of trajectories.

a longitudinal position error bound of $0.417\,\mathrm{m}$ at the road speed limit of $16.67\,\mathrm{m/s}$.

2) Trajectory accuracy: Fig. 6 depicts the error evaluation between the extracted trajectory from our pipeline and the measured trajectory from the *fortuna* vehicle. We present the measurements for each driving direction, as shown in Fig. 2. Here, incoming and outgoing mean driving towards and away from the camera, while merge signifies moving from left-to-right and right-to-left through the camera image. Note, that we could not provide reference drives for “west (incoming)” since the road goes through a tunnel (see Fig. 1), degrading the position accuracy due to losing the GNSS signal. We evaluate the mean absolute error (MAE) of the estimated vehicle extents as well as the lateral and longitudinal positions using curvilinear coordinates [37] with the ground-truth trajectory as the reference path. As expected, the longitudinal error is the lowest for the merge location since the recording perspective is the only one from the side, facilitating the estimation of the longitudinal component. Conversely, the lateral and vehicle width error is larger. Notably, due to the time synchronization procedure described in the previous paragraph, the longitudinal error may also contain an error originating from a time shift. However, the lateral component is less affected, so the average lateral error only ranges from $0.14\,\mathrm{m}$ to $0.28\,\mathrm{m}$. Providing precise lateral position is crucial for several dataset applications, such as traffic rule evaluation, where a misdetected crossing of a lane marking might result in a false positive of a traffic rule violation.

Comparing our results with the ones obtained by Clausse *et al.* [4], our heading root mean squared error (RMSE) is $82.7\,\%$ lower, and the velocity RMSE is $89.8\,\%$ lower, but the position RMSE is $12.5\,\%$ higher. Note that the evaluation data in [4] was synthesized and only covered a small intersection area. An experimental setup similar to ours, where trajectories were recorded from a gantry bridge,

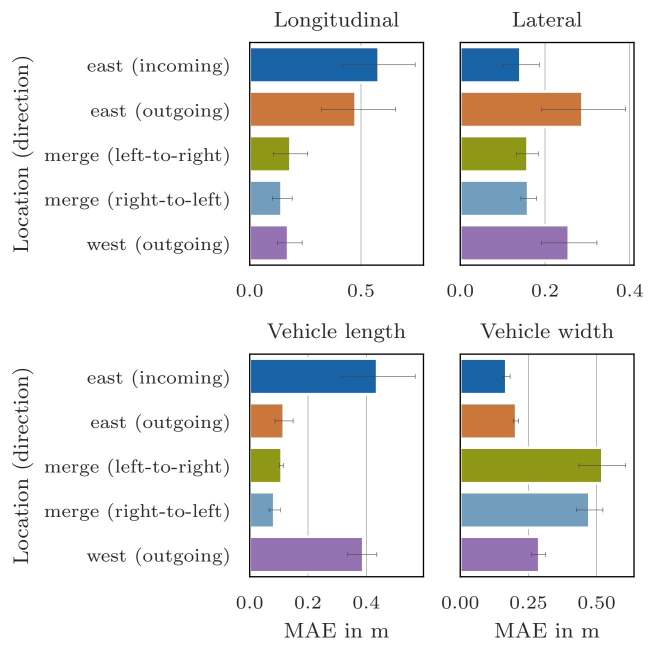


Fig. 6: Analysis of errors between extracted trajectory and reference measurement of the *fortuna* vehicle.

is described by Krämmer *et al.* [38]. Compared to their work, the RMSE values are lower by 63.5 % for the overall position, 66.5 % for the longitudinal position, and 35.4 % for the lateral position. However, their system is intended to operate in real-time, trading off accuracy for processing speed.

C. *Limitations*

We found that black vehicles and motorcycles have a lower detection rate than other vehicles. A possible explanation is that black vehicles have minor contrast to the road, and motorcycles only cover a few pixels. Future work could improve the detection by fine-tuning the model weights on a small, manually labeled subset of our dataset.

During congestions, vehicles may be occluded by static objects for extended periods, e.g., bushes, bridges, gantry bridges, and trucks, which cannot be handled by the association approach described in Section III-C. While for static objects, it could be exploited that vehicles driving into occlusions at one point eventually reappear on the other side. This may lead to a highly location-specific toolchain. More generally, approaches for re-identification and prediction could be used in future work to associate partial trajectories of the same object [39], [40].

Multiple instances of vehicles transporting other vehicles on trailers have been found, e.g., a truck with a semi-trailer transporting multiple cars. As the neural network correctly detects the vehicles, they are mistakenly considered driving vehicles. This issue is difficult to resolve since either the detection has to be improved or heuristics, such as distance or relative velocity to the front vehicle, must be used to identify such situations. On the other hand, the heuristics may lead to false positives, e.g., during congestions.

V. Conclusions

We presented the MONA dataset, a large dataset of naturalistic driving data in Munich, Germany, including videos, maps, ground-truth data from a measurement vehicle, recordings from the perception pipeline of the measurement vehicle, and an improved toolchain to extract trajectories from video. The trajectory data has been statistically analyzed, and its accuracy validated. We provide all data, maps, and software used to create the dataset for full reproducibility and verifiability. All this distinguishes our work from other datasets. Our package is deliberately released under a liberal license making non-commercial and commercial use possible to foster academic and industrial research. Due to the large dataset size, it is particularly suitable for machine learning. Another potential use case is validating planning and prediction methods using only the perception data from the measurement vehicle and simulating the extracted trajectories as ground truth to analyze the impact of partial or erroneous information about the environment. Furthermore, research topics, such as improving computer vision algorithms, can be addressed. Our dataset is the first freely available dataset to combine a stationary sensor setup with onboard sensing in an instrumented vehicle. As our toolchain was developed for general usage and no specific adaptions to the recording locations have been made, more locations can be easily added to obtain a more diverse set of traffic situations.

Acknowledgements

This research was partly funded by the German Research Foundation (DFG) (grant AL 1185/7-1), the German Federal Ministry for Education and Research (BMBF) in the MCube project (grant 03ZU1105KA), the EU Horizon R&I project i4Driving, and the German Federal Ministry for Digital and Transport (BMDV) in the project KoSi. The authors thank Leon Blumenthal for his support in the pose estimation and the IBM Deutschland GmbH for granting access to the Highlight Tower 1, which the authors used to record the traffic from.

References

[1] S. Grigorescu, B. Trasnea, T. Cocias, *et al.*, "A survey of deep learning techniques for autonomous driving," *J. Field Robot.*, vol. 37, no. 3, pp. 362–386, 2020.

[2] S. Ettinger, S. Cheng, B. Caine, *et al.*, "Large scale interactive motion forecasting for autonomous driving : The waymo open motion dataset," 2021. arXiv: `2104.10133`.

[3] M. F. Chang, J. Lambert, P. Sangkloy, *et al.*, "Argoverse: 3D tracking and forecasting with rich maps," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8740–8749.

[4] A. Clausse, S. Benslimane, and A. De La Fortelle, "Large-scale extraction of accurate vehicle trajectories for driving behavior learning," in *Proc. IEEE Intell. Veh. Symp.*, 2019, pp. 2391–2396.

[5] W. Zhan, L. Sun, D. Wang, *et al.*, "Constructing a highly interactive vehicle motion dataset," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2019, pp. 6415–6420.

[6] R. Krajewski, J. Bock, L. Kloeker, *et al.*, "The highD dataset: A drone dataset of naturalistic vehicle trajectories on German highways for validation of highly automated driving systems," in *Proc. IEEE Conf. Intell. Transp. Syst.*, 2018, pp. 2118–2125.

[7] J. Bock, R. Krajewski, T. Moers, *et al.*, "The inD dataset: A drone dataset of naturalistic road user trajectories at german intersections," in *Proc. IEEE Intell. Veh. Symp.*, 2020, pp. 1929–1934.

[8] A. Breuer, J.-A. Termöhlen, S. Homoceanu, *et al.*, "openDD: A large-scale roundabout drone dataset," 2020. arXiv: 2007.08463.

[9] H. Caesar, J. Kabzan, K. S. Tan, *et al.*, "nuPlan: A closed-loop ML-based planning benchmark for autonomous vehicles," in *CVPR ADP3 Work.*, 2021.

[10] V. Alexiadis, J. Colyar, J. Halkias, *et al.*, "The next generation simulation program," *Inst. Transp. Eng. ITE J.*, vol. 74, no. 8, p. 22, 2004.

[11] B. Coifman and Lizhe, Li, "A critical evaluation of the next generation simulation (NGSIM) vehicle trajectory dataset," *Transp. Res. Part B Methodol.*, vol. 105, pp. 362–377, 2017.

[12] M. Pitropov, D. E. Garcia, J. Rebello, *et al.*, "Canadian adverse driving conditions dataset," vol. 40, no. 4-5, pp. 681–690, 2021.

[13] R. Krajewski, T. Moers, J. Bock, *et al.*, "The rounD dataset: A drone dataset of road user trajectories at roundabouts in Germany," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, 2020.

[14] M. Althoff, M. Koschi, and S. Manzinger, "CommonRoad: Composable benchmarks for motion planning on roads," in *Proc. IEEE Intell. Veh. Symp.*, 2017, pp. 719–726.

[15] J. Bernhard, K. Esterle, P. Hart, *et al.*, "BARK: Open behavior benchmarking in multi-agent environments," in *Proc. IEEE Int. Conf. Intell. Robot. Syst.*, 2020, pp. 6201–6208.

[16] S. Maierhofer, M. Klischat, and M. Althoff, "CommonRoad scenario designer: An open-source toolbox for map conversion and scenario creation for autonomous vehicles," in *Proc. IEEE Conf. Intell. Transp. Syst.*, 2021, pp. 3176–3182.

[17] T. Kessler, J. Bernhard, M. Buechel, *et al.*, "Bridging the gap between open source software and vehicle hardware for autonomous driving," in *Proc. IEEE Intell. Veh. Symp.*, 2019, pp. 1430–1437.

[18] T. Kessler, K. Esterle, and A. Knoll, "Mixed-integer motion planning on German roads within the Apollo driving stack," *IEEE Transactions on Intelligent Vehicles*, pp. 1–1, 2022.

[19] T. Y. Lin, M. Maire, S. Belongie, *et al.*, "Microsoft COCO: Common objects in context," *Lect. Notes Comput. Sci.*, no. 5, pp. 740–755, 2014.

[20] F. Chabot, M. Chaouch, J. Rabarisoa, *et al.*, "Deep MANTA: A coarse-to-fine many-task network for joint 2D and 3D vehicle analysis from monocular image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1827–1836.

[21] A. Mousavian, D. Anguelov, J. Košecká, *et al.*, "3D bounding box estimation using deep learning and geometry," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5632–5640.

[22] S. Li, Z. Yan, H. Li, *et al.*, "Exploring intermediate representation for monocular vehicle pose estimation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 1873–1883.

[23] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.

[24] E. Guo, Z. Chen, S. Rahardja, *et al.*, "3D detection and pose estimation of vehicle in cooperative vehicle infrastructure system," *IEEE Sens. J.*, vol. 21, no. 19, pp. 21 759–21 771, 2021.

[25] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, 2000.

[26] H. G. Jung, Y. H. Cho, P. J. Yoon, *et al.*, "Scanning laser radar-based target position designation for parking aid system," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 3, pp. 406–424, 2008.

[27] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[28] E. Bochinski, V. Eiselein, and T. Sikora, "High-Speed tracking-by-detection without using image information," in *IEEE Int. Conf. Adv. Video Signal Based Surveill.*, 2017.

[29] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *Proc. Int. Conf. Image Process.*, 2018, pp. 3645–3649.

[30] J. E. Stellet, L. Walkling, and J. Marius Zöllner, "Post processing of laser scanner measurements for testing advanced driver assistance systems," in *Int. Conf. Inf. Fusion*, 2016, pp. 1999–2006.

[31] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic systems," *AIAA J.*, vol. 3, no. 8, pp. 1445–1450, 1965.

[32] R. Schubert, E. Richter, and G. Wanielik, "Comparison and evaluation of advanced motion models for vehicle tracking," in *Int. Conf. Inf. Fusion*, 2008.

[33] C. Pek, V. Rusinov, S. Manzinger, *et al.*, "CommonRoad drivability checker: Simplifying the development and validation of motion planning algorithms," in *Proc. IEEE Intell. Veh. Symp.*, 2020, pp. 1013–1020.

[34] S. Gros, "Implicit non-convex model predictive control," in *Handbook of Model Predictive Control*, Springer, 2019, pp. 305–333.

[35] J. A. E. Andersson, J. Gillis, G. Horn, *et al.*, "CasADi – A software framework for nonlinear optimization and optimal control," vol. 11, no. 1, pp. 1–36, 2019.

[36] D. Janka, C. Kirches, S. Sager, *et al.*, "An SR1/BFGS SQP algorithm for nonconvex nonlinear programs with block-diagonal Hessian matrix," vol. 8, no. 4, pp. 435–459, 2016.

[37] E. Héry, S. Masi, P. Xu, *et al.*, "Map-based curvilinear coordinates for autonomous vehicles," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, 2017.

[38] A. Krämmer, C. Schöller, D. Gulati, *et al.*, "Providentia–A large scale sensing system for the assistance of autonomous vehicles and its evaluation," *J. Field Robot.*, no. 2, pp. 1156–1176, 2022.

[39] H. Wang, J. Hou, and N. Chen, "A survey of vehicle re-identification based on deep learning," *IEEE Access*, vol. 7, pp. 172 443–172 469, 2019.

[40] F. Leon and M. Gavrilescu, "A review of tracking and trajectory prediction methods for autonomous driving," *Mathematics*, vol. 9, no. 6, p. 660, 2021.